Neiva, 23 de Enero de 2023_____

Señores

CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN

UNIVERSIDAD SURCOLOMBIANA

Ciudad

El (Los) suscrito(s):

Luis Fernando Enciso Caballero_____, con C.C. No. 1075314119_____,

Pedro Luis Duran Losada_____, con C.C. No. 1083916036_____,

Blanca Ligia Peña Escobar_____, con C.C. No. 1022396698_____,

_____, con C.C. No. _____,

Autor(es) de la tesis y/o trabajo de grado o _____

Titulado: Sistema Inteligente de Riego Basado en Internet de las Cosas.

_____

_____

presentado y aprobado en el año __2023_____ como requisito para optar al título de

__Ingeniería de Software_____;

Autorizo (amos) al CENTRO DE INFORMACIÓN Y DOCUMENTACIÓN de la Universidad Surcolombiana para que, con fines académicos, muestre al país y el exterior la producción intelectual de la Universidad Surcolombiana, a través de la visibilidad de su contenido de la siguiente manera:

- Los usuarios puedan consultar el contenido de este trabajo de grado en los sitios web que administra la Universidad, en bases de datos, repositorio digital, catálogos y en otros sitios web, redes y sistemas de información nacionales e internacionales "open access" y en las redes de información con las cuales tenga convenio la Institución.

- Permita la consulta, la reproducción y préstamo a los usuarios interesados en el contenido de este trabajo, para todos los usos que tengan finalidad académica, ya sea en formato Cd-Rom o digital desde internet, intranet, etc., y en general para cualquier formato conocido o por conocer, dentro de los términos establecidos en la Ley 23 de 1982, Ley 44 de 1993, Decisión Andina 351 de 1993, Decreto 460 de 1995 y demás normas generales sobre la materia.

- Continúo conservando los correspondientes derechos sin modificación o restricción alguna; puesto que, de acuerdo con la legislación colombiana aplicable, el presente es un acuerdo jurídico que en ningún caso conlleva la enajenación del derecho de autor y sus conexos.

De conformidad con lo establecido en el artículo 30 de la Ley 23 de 1982 y el artículo 11 de la Decisión Andina 351 de 1993, "Los derechos morales sobre el trabajo son propiedad de los autores" , los cuales son irrenunciables, imprescriptibles, inembargables e inalienables.

EL AUTOR/ESTUDIANTE:

Firma: _____

EL AUTOR/ESTUDIANTE:

Firma: _____

EL AUTOR/ESTUDIANTE:

Firma: Pedro L. Durán

EL AUTOR/ESTUDIANTE:

Firma: _____

TÍTULO COMPLETO DEL TRABAJO: Sistema Inteligente de Riego Basado en Internet de las Cosas

AUTOR O AUTORES:

Primero y Segundo Apellido	Primero y Segundo Nombre
Enciso Caballero	Luis Fernando
Duran Losada	Pedro Luis
Peña Escobar	Blanca Ligia

DIRECTOR Y CODIRECTOR TESIS:

Primero y Segundo Apellido	Primero y Segundo Nombre
Castro Silva	Juan Antonio

ASESOR (ES):

Primero y Segundo Apellido	Primero y Segundo Nombre

PARA OPTAR AL TÍTULO DE: Ingeniería de Software

FACULTAD: Ingeniería

PROGRAMA O POSGRADO: Ingeniería de software

CIUDAD: Neiva **AÑO DE PRESENTACIÓN: 2023** **NÚMERO DE PÁGINAS: 215**

TIPO DE ILUSTRACIONES (Marcar con una **X**):

Diagramas_x__ Fotografías_x__ Grabaciones en discos___ Ilustraciones en general___ Grabados___ Láminas___ Litografías___ Mapas___ Música impresa___ Planos___ Retratos___ Sin ilustraciones___ Tablas o Cuadros_x_

SOFTWARE requerido y/o especializado para la lectura del documento:

MATERIAL ANEXO:

PREMIO O DISTINCIÓN *(En caso de ser LAUREADAS o Meritoria)*:

PALABRAS CLAVES EN ESPAÑOL E INGLÉS:

	Español	Inglés
1.	Aprendizaje profundo (DL)	Deep learning (DL)
2.	Internet de la cosas (IoT)	Internet of things (IoT)
3.	índice de humedad disponible (IHD)	Interval available humidity
4.	riego autónomo	autonomous irrigation
5.	tipos de suelo	soil types

RESUMEN DEL CONTENIDO: (Máximo 250 palabras)

Las desigualdades en el acceso al agua y los fenómenos de sequía cada vez más fuertes ante el cambio climático, convirtieron el riego en una ciencia. El estudio de los aspectos que influyen en él y los avances en sus componentes, han desarrollado un tipo de riego localizado. Este proyecto desarrolla una plataforma web que permite gestionar un prototipo de sistema de riego localizado, en la cual, se puede dividir la zona a regar en sectores, para aplicar un riego diferenciado. Cada uno de ellos contó con componentes independientes, como electroválvulas y emisores (goteo, cinta y microaspersión). Los dispositivos IoT utilizados fueron: sensores de humedad del suelo y módulos de radiofrecuencia. Se definió como estrategia de riego, irrigar

cuando el déficit hídrico fuera igual al nivel de agotamiento permitido. Para determinar cuánto regar, se tuvieron en cuenta las necesidades hídricas del cultivo según su profundidad radicular y el IHD de cada tipo de suelo. Además, se utilizó Deep Learning con la arquitectura U-Net para entrenar tres modelos de segmentación que detecten arena, limo y arcilla en imágenes de prueba de campo, y así clasificar el tipo de suelo.

ABSTRACT: (Máximo 250 palabras)

The inequalities of access to water and the increasingly strong drought phenomena faced by climate change, turned irrigation into a science. The study of the aspects that influence it and the progress in its components, have developed a type of localized irrigation. This project develops a web platform that allows managing a prototype of a localized irrigation system, in which the area to be irrigated can be divided into sectors, in order to apply a differentiated irrigation. Each of them had independent components, such as solenoid valves and emitters (drip, tape and micro-sprinkler). The IoT devices used were: soil moisture sensors and radio frequency modules. Irrigation strategy was defined as irrigation strategy, to irrigate when the water deficit was equal to the allowable depletion level. To determine how much to irrigate, the water needs of the crop according to its root depth and the IHD of each soil type were taken into account. In addition, Deep Learning with the U-Net architecture was used to train three segmentation models to detect sand, silt and clay in field test images to classify soil type.

APROBACION DE LA TESIS

Nombre Presidente Jurado: Fernando Tejas.

Firma: Ftejas.

Nombre Jurado: Luis Gregori Ramírez.

Firma:

Nombre Jurado: Fredy Medina Tejas

Firma:

Sistema Inteligente de Riego Basado en Internet de las Cosas

Blanca Ligia Peña Escobar, Luis Fernando Enciso Caballero y

Pedro Luis Durán Losada

Trabajo de grado para optar al título de Ingeniero de Software

Director: Juan Antonio Castro Silva

Universidad Surcolombiana, Facultad de Ingeniería

Programa de Ingeniería de Software

Neiva, 2022

Resumen

Las desigualdades en el acceso al agua y los fenómenos de sequía cada vez más fuertes ante el cambio climático, convirtieron el riego en una ciencia. El estudio de los aspectos que influyen en él y los avances en sus componentes, han desarrollado un tipo de riego localizado. Este proyecto desarrolla una plataforma web que permite gestionar un prototipo de sistema de riego localizado, en la cual, se puede dividir la zona a regar en sectores, para aplicar un riego diferenciado. Cada uno de ellos contó con componentes independientes, como electroválvulas y emisores (goteo, cinta y microaspersión). Los dispositivos IoT utilizados fueron: sensores de humedad del suelo y módulos de radiofrecuencia. Se definió como estrategia de riego, irrigar cuando el déficit hídrico fuera igual al nivel de agotamiento permitido. Para determinar cuánto regar, se tuvieron en cuenta las necesidades hídricas del cultivo según su profundidad radicular y el IHD de cada tipo de suelo. Además, se utilizó Deep Learning con la arquitectura U-Net para entrenar tres modelos de segmentación que detecten arena, limo y arcilla en imágenes de prueba de campo, y así clasificar el tipo de suelo.

Palabras clave: Aprendizaje profundo (DL), Internet de la cosas (IoT), índice de humedad disponible (IHD), riego autónomo, tipos de suelo.

Abstract

The inequalities of access to water and the increasingly strong drought phenomena faced by climate change, turned irrigation into a science. The study of the aspects that influence it and the progress in its components, have developed a type of localized irrigation. This project develops a web platform that allows managing a prototype of a localized irrigation system, in which the area to be irrigated can be divided into sectors, in order to apply a differentiated irrigation. Each of them had independent components, such as solenoid valves and emitters (drip, tape and micro-sprinkler). The IoT devices used were: soil moisture sensors and radio frequency modules. Irrigation strategy was defined as irrigation strategy, to irrigate when the water deficit was equal to the allowable depletion level. To determine how much to irrigate, the water needs of the crop according to its root depth and the IHD of each soil type were taken into account. In addition, Deep Learning with the U-Net architecture was used to train three segmentation models to detect sand, silt and clay in field test images to classify soil type.

Keywords: Deep learning (DL), Internet of things (IoT), Interval available humidity, autonomous irrigation, soil types.

Contenido

Lista de Figuras

Lista de Tablas

Lista de Ecuaciones

Director

Juan Antonio Castro Silva.

Líneas de investigación

Inteligencia artificial.

Internet de las cosas.

Ingeniería de software.

1 Introducción

La agricultura se enfrenta a múltiples retos, tales como producir más alimentos a fin de alimentar a una población creciente con una mano de obra menor, además de adoptar métodos de producción más eficaces y sostenibles y adaptarse al cambio climático. La Organización de las Naciones Unidas para la Alimentación y la Agricultura (FAO) advierte del incremento de la población mundial para el 2050, lo cual, genera un problema para producir alimentos a cerca de 9.100 millones de personas. Además, las recientes tendencias migratorias de la población rural a sitios urbanos, provocando menor mano de obra para la producción de los alimentos. (FAO, 2009)

La disponibilidad de los recursos de agua dulce es suficiente, pero está distribuida desigualmente, y cada vez hay más países o regiones dentro de estos cuya escasez de agua alcanza niveles alarmantes. Esta escasez podría agravarse por los cambios en el régimen de lluvias resultantes del cambio climático. El consumo de agua para la agricultura de regadío está previsto crezca a un ritmo menor debido a la disminución de la demanda y un uso más eficiente del agua, pero aun así se incrementará en cerca del 11 por ciento para 2050. (FAO, 2009)

Los diferentes tipos de suelos y condiciones climatológicas son factores decisivos al momento de realizar un riego sin desperdiciar el agua. Con las nuevas tecnologías como el Internet de las cosas (IoT) o la Inteligencia Artificial, e implementando nuevas tendencias como Smart Farming en los sistemas de riego, complementa su función al aplicar una dosificación e irrigación apropiada para el cultivo, pues, se tienen en cuenta más propiedades asociadas al tipo de suelo y las necesidades de agua según el tipo de cultivo. Además, de reducir los costos de producción en las actividades agrícolas. (Peluffo Ordóñez y otros, 2017)

En la agricultura se debe tener en cuenta el punto de marchitez y la capacidad de campo, lo cual es esencial para generar un riego apropiado del cultivo. Cuando el agua del suelo se agota, el suelo contiene un volumen de agua que se conoce como ‘punto de marchitez’, que, a falta de lluvias, se debe suministrar un riego cada vez que el contenido de humedad del suelo se encuentre aproximadamente a la mitad entre la capacidad de campo y el punto de marchitez. (InfoRiego, s.f.)

Para hacer un uso eficiente del agua debemos responder a las preguntas de cuándo se ha de regar y cuánta agua aplicar, para ello debemos conocer las características del cultivo, las características físicas del suelo y las condiciones climáticas de la zona. La influencia del cultivo es importante puesto que las necesidades de agua serán mayores o menores en función del tipo de planta y de su estado de desarrollo. De la misma forma, las raíces de un cultivo ocupan diferente profundidad del suelo en distintas fases dentro del ciclo por lo que la cantidad de agua disponible en esa zona de suelo varía con el estado del cultivo. La capacidad de cada suelo para retener agua también es diferente, lo que implica que tanto la cantidad de agua a aplicar con el riego como la que pueden extraer las plantas puede variar mucho. A ello hay que añadir que las necesidades de agua serán también dependientes del clima, radiación solar, viento, precipitación, etc., por lo que es preciso conocer las características climáticas de la zona y del cultivo para programar adecuadamente los riegos. (Fernadez y otros, 2010)

2 Antecedentes

2.1 Estado del Arte

2.1.1 A Smart IoT-Based irrigation system with automated plant recognition using deep learning

Este trabajo desarrolla un sistema de riego, con el uso de Deep Learning, que es capaz de ajustar las cantidades de agua para cada tipo de planta a través del reconocimiento de plantas.

El sistema pretende realizar un riego diferenciado teniendo en cuenta la humedad del suelo y la humedad que requiere determinadas platas. Para lo cual realiza una aplicación móvil mediante Android Studio el cual le permitirá determinar mediante una foto que planta es y dependiendo del resultado establecer la humedad óptima para la planta.

El sistema de clasificación de planta lo realiza mediante Deep Learning usando Tensor Flow por lo que utiliza 10 tipos de planta, utilizando 30 imágenes por cada tipo de planta para el entrenamiento del modelo. Además, incorpora un sensor de humedad a un Arduino que le proporciona la humedad del suelo en tiempo real lo cual le permitirá automatizar mediante IoT el riego. (Kwok & Sun, 2018)

2.1.2 Implementación de un sistema automático de riego por goteo autocompensado para la producción de árboles ornamentales y nativos en el vivero forestal de la U.EIA.

Este proyecto busca mejorar la producción de árboles ornamentales y nativos para el proceso de repoblación forestal, por medio de un sistema de riego por goteo auto compensado y automático que suministrará la cantidad de agua necesaria requerida por las plantas teniendo en

cuenta la variabilidad del ambiente, para así mejorar la producción de estas usando la mínima cantidad del recurso hídrico.

Para este sistema de riego por goteo se decide implementar una página web y aplicación móvil en el módulo de comunicación que permitan un monitoreo remoto de las diferentes variables que involucra el proceso de riego automático de las especies forestales y nativas gracias a las dinámicas del IoT (Internet of things).

El sistema de IoT presente en proyecto pretende medir las variables de temperatura y humedad del suelo con el objetivo de controlar el riego. Además, de calcular la evapotranspiración mediante el método de Penman-Monteith propuesta por la FAO. Con la evapotranspiración determinada se procede a calcular la Necesidad Hídrica, Lamina de Riego y Volumen de Riego las cuales genera la salida de agua del sistema. (Duque Correa & Villegas Santamaría, 2018)

2.1.3 Sistema de riego autónomo basado en la Internet de las Cosas

Este proyecto busca construir un sistema de riego autónomo basado en el IoT, el cual mediante un sistema de sensores proporciona las variables agroclimatológicas necesarias para determinar un calendario de riego apropiado para el cultivo, con el fin de hacer un uso eficiente del agua. Los datos agroclimatológicos son esenciales para la determinación de la evapotranspiración mediante el método de Penman-Monteith. El cual es primordial para calcular las necesidades de agua de un cultivo y poder generar un modelo predictivo mediante Machine Learning.

Para el modelo predictivo se basaron en los datos recopilados por las estaciones meteorológicas de España, provenientes de los sitios web de la Agencia Estatal de Meteorología (AEmet) y la Junta de Andalucía. Los datos de la AEmet corresponden a 30 años, con una periodicidad mensual y para cerca de 87 estaciones.

Además, el sistema se administra a través de una página web cuya función principal del sistema es realizar automáticamente las tareas de riego programadas. Sin embargo, también se debe permitir el control manual, que se puede realizar mediante la página web.

La recomendación para trabajos futuros propuesta por el Autor es el uso de Apache Spark para Big Data y Machine Learning con el fin de probar con datos en tiempo real de una estación agroclimatológica desde la fecha de siembra hasta la cosecha. Además, el uso de un servidor local ayuda a reducir los costos operativos, así como la capacidad de controlar el sistema de riego a través de una aplicación móvil conectada a través de Bluetooth o Wifi. (Castro Silva, 2016)

2.1.4 Predicting Soil Texture Using Image Analysis

En este artículo se plantea un método analítico para predecir y determinar la textura del suelo mediante el procesamiento de imágenes digitales. Para ello, utilizaron 65 muestras de suelo que fueron secadas, trituradas y tamizadas inferior a los 2 mm. Los tipos de suelo del estudio estuvieron representados con un 73% arcilla, 5% de Arena-limoso, 2%de arena, 2% Arcilla-arenosa y 19% franco arcilloso arenoso. La obtención de las imágenes implicó integrar una cámara digital con un microscopio estereoscópico Leica EZ4 y poner la muestra de suelo en una caja Petri. Este método les permitió recolectar un total de 189 imágenes digitales (2048x1536 píxeles).

Las imágenes obtenidas se procesaron utilizando los sistemas de color RGB, HSV y escala de grises, a partir de ellos se crearon cuatro matrices de histograma medio (RGB+HSV, RGB+escala de grises, HSV+ escala de grises y RGB+HSV+ escala de grises) para generar modelos estadísticos multivariantes que relacionaban la imagen con la distribución de partículas ya conocida. Los datos centrados en la media se organizaron con dos enfoques: para el algoritmo de kennard-Stone; calibración y validación, y para un bootstrapp no paramétrico. Los modelos MIA entrenados se evaluaron con los parámetros RMSE, calibración y predicción, coeficiente de correlación de Pearson, REP y RPD (entre mayor sea, mejor será el modelo).

Al termino, los autores concluyeron que los modelos MIA con PSL pueden emplearse para clasificar la textura del suelo mediante visión por computador, sin embargo, aún debe probarse con muestras de suelo con mayor concentración de limo. Además, el método propuesto es respetuoso con el medio ambiente; ya que no emplea productos químicos, igualmente, se puede considerar más barato y rápido que el método de la pipeta (estándar). Incluso abre la posibilidad de emplear teléfonos móviles para la adquisición de fotos en campo y eliminar el uso del microscopio. (Morais y otros, 2019)

3 Marco Teórico

3.1 Activation Function:

Una función de activación es una función que se añade a una red neuronal artificial para ayudar a la red a aprender patrones complejos en los datos. La función de activación también debe tener la capacidad de diferenciarse, lo cual es una característica muy significativa, ya que permite utilizar la retro propagación de errores para entrenar la red

Las capas de activación no lineal se emplean después de todas las capas con pesos (las llamadas capas aprendibles, como las capas FC y las capas convolucionales) en la arquitectura de la CNN. Esta función no lineal de las capas de activación significa que el mapeo de la entrada a la salida será no lineal; además, estas capas dan a la CNN la capacidad de aprender cosas extra complicadas.

3.1.1 Sigmoide:

La entrada de esta función de activación son números reales, mientras que la salida está restringida entre cero y uno. La curva de la función sigmoide tiene forma de S y puede representarse matemáticamente mediante la Ecuación 1. (Alzubaidi y otros, 2021)

Ecuación 1

Función Sigmoide

$$f_{(x)sigm}=\frac{1}{1+e^{-x}}$$

3.2 Agricultura de Precisión

La agricultura de precisión es un conjunto de técnicas que se establecen para optimizar los recursos (agua, semillas, etc.) empleando herramientas (sensores) que determinan las variables del cultivo, logrando la distribución correcta de los recursos dependiendo de la necesidad de cada punto. Además de adoptar prácticas que puedan identificar y cuantificar las variables del cultivo.

3.3 Capacidad de Campo - CC

La capacidad del suelo para almacenar agua depende de su estructura y textura, es decir, que, en un suelo con todos los poros saturados después de un evento de lluvia o riego, el agua tiende a moverse descendentemente por efecto de la gravedad hacia el subsuelo, pero llega un determinado momento, en el que el drenaje se hace insignificante, a este estado de relativamente constancia de agua en el suelo se le denomina Capacidad de campo o también Límite superior. (Zotarelli y otros, 2019)

Sin embargo, el concepto de capacidad de campo se aplica únicamente en suelos bien estructurados en los que después de 48 horas de drenaje es relativamente rápido, pues en suelo pobremente estructurados rara vez se le puede definir claramente su CC. (Shaxson & Barber, 2005)

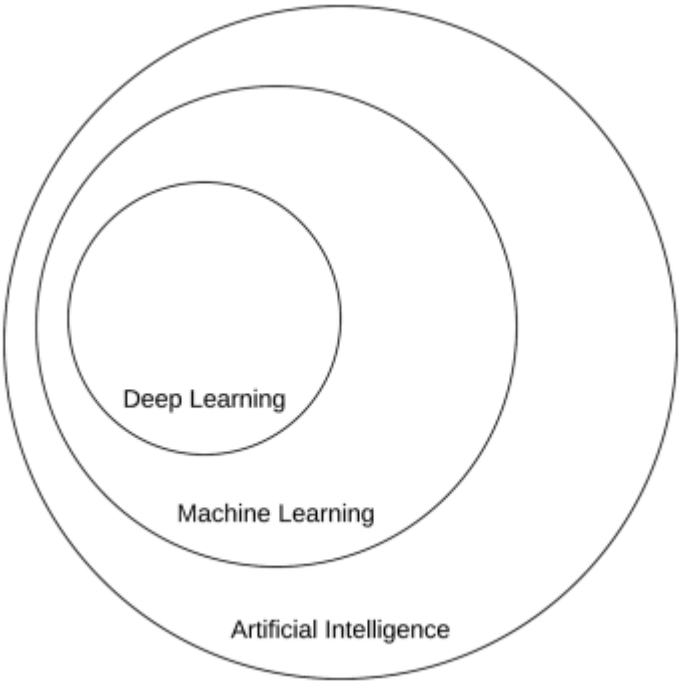
3.4 Deep Learning-DL

El DL es un subconjunto del ML, se basa en los patrones de procesamiento de la información que se encuentran en el cerebro humano. El DL no requiere ninguna regla diseñada por el ser humano para funcionar, sino que utiliza una gran cantidad de datos para asignar la entrada dada a etiquetas específicas. El DL se diseña utilizando numerosas capas de algoritmos,

cada una de las cuales proporciona una interpretación diferente de los datos que se les han suministrado.

Figura 1

Diagrama Venn Subcampo Inteligencia Artificial



Nota. Adaptado de Deep Learning For Computer Vision (pag. 22), Adrian Rosebrock, 2017

El DL se divide en tres grandes categorías: El primero es el aprendizaje profundo supervisado, el cual trabaja con datos etiquetados. El segundo es el aprendizaje semi-supervisado el cual se basa en conjuntos de datos semi-etiquetados lo que genera minimizar la cantidad de datos etiquetados necesitados, pero puede que los datos de entrenamiento presentes en las características de entrada irrelevantes podrían proporcionar decisiones incorrectas. Y el tercero es el aprendizaje no supervisado (Rosebrock, 2017)

3.5 Evapotranspiración

La evaporación como el proceso por el cual el agua de una superficie en estado líquido se convierte en vapor para pasar a la atmosfera, para ello, se requiere de energía, provista por la radiación solar y la temperatura ambiente del aire. Por otro lado, la transpiración es la vaporización del agua líquida contenida en los tejidos de la planta para su posterior remoción hacia la atmósfera. Aunque, la mayoría del agua absorbida por la planta se pierde por la transpiración, una pequeña parte se convierte en parte del tejido de la planta. (Allen y otros, Evapotranspiración del cultivo, 2006)

A partir de lo anterior, se tiene que la evapotranspiración es el resultado de dos procesos que ocurren simultáneamente: la evaporación directamente desde la superficie del suelo y la transpiración de la cubierta vegetal, pero no hay forma sencilla de distinguir ambos procesos, sin embargo, es importante tener en cuenta que la proporción de evaporación y transpiración en un cultivo cambia de acuerdo con las diferentes fases de desarrollo del cultivo, como también, de las condiciones del clima. (Mañas y otros, 2005)

La injerencia de la evapotranspiración en los balances de agua y energía convierte este proceso en interés de diversas disciplinas y así mismo, objeto de estudio para obtener la mayor precisión de su estimación, en consecuencia, se han formulado diferentes conceptos, algunos de ellos son: la evapotranspiración potencial (ETP), evapotranspiración de referencia o del cultivo de referencia (ETo), evapotranspiración máxima (ETm) y evapotranspiración real (ETR). Sin embargo, la mayor precisión alcanzada se obtiene con la ETR, ya que esta tiene en cuenta factores meteorológicos, características de vegetación, el tipo de suelo y la condición real de humedad. (Sanchez, 2000)

3.6 Historia de Usuario

Las historias de usuarios son la forma recomendada en entornos agiles para escribir los requisitos del producto expresados por los usuarios. Dado que son afirmación breves, simples y fáciles entender por el quipo y resultan una mejor comunicación entre los Stakeholders y el equipo.

Al redactar una historia de usuario se debe tener en cuenta describir el rol, la funcionalidad y el resultado esperado en una frase corta y debe ser acompañada por los criterios de aceptación que debe cumplir la historia de usuario. La historia debe cumplir con la siguiente estructura:

- Como: describe el rol del Stakeholders que solicita la funcionalidad o requerimiento.
- Quiero: describe la necesidad o requerimiento del usuario por lo general es una frase corta.
- Para: describe el beneficio esperado por el Stakeholders una vez se desarrolla el requerimiento.

Aunque se puede desarrollar una historia de usuario con esa estructura, existen otros componentes adicionales que permiten mejorar su entendimiento tanto para el usuario como para los desarrolladores.

- Criterios de aceptación: se define con un conjunto de sentencias redactadas de tal manera que conduzca a una respuesta clara de aceptado o rechazado.

- Estimaciones: hace referencia a la dificultad, duración o costo requerido para ejecutar determinado trabajo.

- Prototipos, diagramas, etc.: nos sirve para mostrar un acercamiento al resultado final que se espera obtener.

3.7 Inteligencia Artificial – IA

La IA es una rama computacional que mediante algoritmos tiene como objetivo lograr que los ordenadores desarrollen la misma clase de cosas que puede hacer el ser humano. Tanto para los seres humanos como para la IA, la inteligencia no es una dimensión única, sino un espacio abundantemente estructurado de capacidades para procesar la información. Donde se entrañan competencias como la percepción, la asociación, la predicción la planificación, el control motor, etc. (Boden, 2016)

Las tecnologías basadas en IA están siendo ampliamente utilizadas por los seres humanos, ya que, esta puede analizar grandes cantidades de información con un margen de error significativamente menor sin requerir descanso a diferencia de los humanos. Algunas de las aplicaciones que están creciendo exponencialmente son:

- Reconocimiento de imágenes estáticas, clasificación y etiquetado.

- Mejoras del desempeño de la estrategia de algoritmo comercial.

- Procesamiento eficiente y escalable de datos de pacientes.

- Detección y clasificación de objetos.

- Distribución de contenido en las redes sociales
- Protección contra amenazas de seguridad cibernética.

(Rouhiainen, 2018)

3.8 Internet de las cosas (IoT)

El internet de las cosas (IoT por sus siglas en inglés “Internet of things”) también llamado internet de los objetos se refiere a una arquitectura emergente que permite la interconexión de objetos cotidianos por medio de redes alámbricas o inalámbricas a Internet. Estos objetos capacitados con algún tipo de inteligencia y autonomía recopilan información de su entorno para enviarla a centros de procesamiento, por medio de la red, con la finalidad de traspasar las barreras estructurales entre los objetos del mundo físico y su importancia en los sistemas de información (Barrio Andrés, 2018, págs. 19-22) emplea para determinar una tecnología que interacción “máquina-máquina” (Machine-to Machine,M2M). (Salazar & Silvestre, 2016)

3.9 Machine Learning – ML

El ML o aprendizaje automático en español, es un área de la inteligencia artificial que abarca un grupo de técnicas o métodos algorítmicos para que las maquinas tengan la capacidad de aprender a hacer o mejorar predicciones o comportamientos basados en datos (Rouhiainen, 2018). Existen diversas tareas en las que se aplica el ML y aunque la función varia, el procedimiento se puede sintetizar en los siguientes pasos:

Recolectar datos, estos deben reunir todas las características de lo que se quiere predecir, por eso entre más datos se tenga mejor.

Introducir esta información en un algoritmo de machine learning para generar un modelo, este último es básicamente un programa que aprendió a relacionar las entradas con las predicciones.

Usar el modelo de ML con nuevos datos, integrándolo en un proceso o producto, como por ejemplo en un carro autónomo.

El aprendizaje automático se aplica con gran éxito en diversos campos como el reconocimiento de patrones, visión por computador, la ingeniería espacial, agricultura de precisión, aplicaciones médicas y biomédicas, etc. Pues las maquinas superan a los humanos en muchas tareas, si bien es cierto que pueden hacerlo ligeramente peor sigue prevaleciendo una ventaja en cuanto a velocidad, reproducibilidad y escala. (Molnar, 2020)

3.10 Manifiesto Agile

- La metodología Agile nos permite abordar proyectos con enfoque en la generación de espacios de colaboración, confianza e innovación. el termino hace referencia a poder moverse o responder de forma rápida y fácil.

- La metodología agile plantea un manifiesto en el cual expresa los 12 principios con el fin de formalizar las practicas agiles para el desarrollo de software convirtiéndose en la base de muchos marcos de trabajo agiles como Scrum. (Herrera Uribe & Valencia Ayala, 2007)

- Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor.

- Aceptamos que los requisitos cambien, incluso en etapas tardías del desarrollo. Los procesos agiles aprovechan el cambio para proporcionar ventaja competitiva.

- Entregar productos funcionales frecuentemente, entre dos semanas y dos meses, con preferencia al periodo de tiempo más corto.

- Los responsables del negocio y los desarrolladores trabajan juntos de forma cotidiana durante todo el proyecto.

- Los proyectos se desarrollan en torno a individuos motivados. Hay que dar el entorno y el apoyo que necesitan y confiarles la ejecución del trabajo.

- El método más eficiente y efectivo de comunicar información al equipo de desarrollo y entre sus miembros es la conversación cara a cara.

- El software funcionando es la medida principal de progreso.

- Los procesos agiles promueven el desarrollo sostenible.

- La atención continua a la excelencia técnica y al buen diseño mejora la agilidad.

- La simplicidad o el arte de maximizar la cantidad de trabajo no realizado, es esencial.

- La mejores arquitecturas, requisitos y diseños emergen de equipos autoorganizados y multifuncionales.

- A intervalos regulares el equipó reflexiona sobre como ser más efectivo para a continuación ajustar y perfeccionar sus comportamientos en consecuencia.

3.11 Métricas de evaluación:

Las métricas de evaluación o rendimiento son un proceso normal que ocurre dentro de la etapa de prueba del modelo, y cumplen un papel crucial, pues se encargan de estimar la precisión generalizada de un modelo sobre datos no vistos por el mismo en la etapa de entrenamiento (Alzubaidi y otros, 2021). Las métricas de rendimiento evalúan el modelo con diferentes objetivos y por esto pueden clasificarse en tres tipos: la probabilidad, el umbral y la métrica de clasificación, siendo los dos últimos en la práctica los más utilizados por los investigadores, además, en su mayoría de casos se emplean para tres aplicaciones de evaluación diferentes:

Se emplean para evaluar la capacidad de generalización del modelo o clasificador, es decir, que se utilizan para medir y resumir la calidad del modelo cuando se prueba con datos no vistos, un ejemplo es la precisión o Accuracy.

También se emplean las métricas de evaluación para seleccionar el mejor clasificador o modelo entrenado entre los diferentes modelos que se hayan entrenado.

Finalmente, se emplean las métricas de evaluación para discriminar y seleccionar la mejor solución óptima entre todas las soluciones generadas durante el entrenamiento del modelo.

Aunque, para las dos primeras aplicaciones de métricas de evaluación se puede aplicar casi todos los tipos de métricas para evaluar el rendimiento y eficacia del modelo, solo algunos tipos de métricas se pueden emplear para el tercero, es decir, como discriminadores durante el entrenamiento.(Hossin & Sulaiman, 2015)

Es importante tener en cuenta que muchos problemas reales se enfrentan con volúmenes de datos desbalanceados, por lo cual, es importante seleccionar un algoritmo apropiado mediante una o varias métricas de evaluación del rendimiento en varios algoritmos candidatos (Molnar, 2020). En la tabla 1 se describen algunas métricas de evaluación.

Tabla 1

Métricas para la Evaluación del Rendimiento en Modelos con Deep Learning

Métrica	Formula	Foco de evaluación
Accurracy (acc)	$\frac{tp + tn}{tp + fp + tn + fn}$	En general, mide la proporción de predicciones correctas sobre el número total de instancias evaluadas.
Error Rate(err)	$\frac{fp + fn}{tp + fp + tn + fn}$	Mide la proporción de predicciones incorrectas sobre el número total de instancias evaluadas.
Sensitivity(sn)	$\frac{tp}{tp + fn}$	Esta métrica se utiliza para medir la fracción de patrones positivos que se clasifican correctamente.
Specificity(sp)	$\frac{tn}{tn + fp}$	Esta métrica se utiliza para medir la fracción de patrones negativos que se clasifican correctamente.
Precisión(p)	$\frac{tp}{tp + fp}$	La precisión se utiliza para medir los patrones positivos que se predicen

		correctamente del total de patrones predichos en una clase positiva.
F1-Score	$2 * \frac{accurracy * \text{Recall}}{accurracy + \text{Recall}}$	Media armónica entre los índices de recall y Accurracy.
Area Under the ROC Curve (AUC)	$\frac{S_p - n_p(n_n + 1)/2}{n_p n_n}$	Métrica común del tipo de clasificación. Se utiliza para realizar comparaciones entre algoritmos de aprendizaje, así como para construir un modelo de aprendizaje óptimo. A diferencia de las métricas de probabilidad y umbral, el valor AUC expone el rendimiento completo del clasificador. La fórmula se utiliza para calcular el valor AUC para un problema de dos clases.
False Positive Rate (FPR)	$FPR = 1 - Specificity$	Esta métrica se refiere a la posibilidad de una proporción de falsas alarmas según el cálculo.
Recall (r)	$\frac{tp}{tp + tn}$	La recuperación se utiliza para medir la fracción de patrones positivos que se clasifican correctamente.
F-Measure (FM)	$\frac{2 * p * r}{p + r}$	Esta métrica representa la media armónica entre los valores de recall y precisión.

Nota. tp: verdadero positivo; fp: falso positivo; fn: Falso negativo; tn: Verdadero negativo.

3.12 Pooling Layer

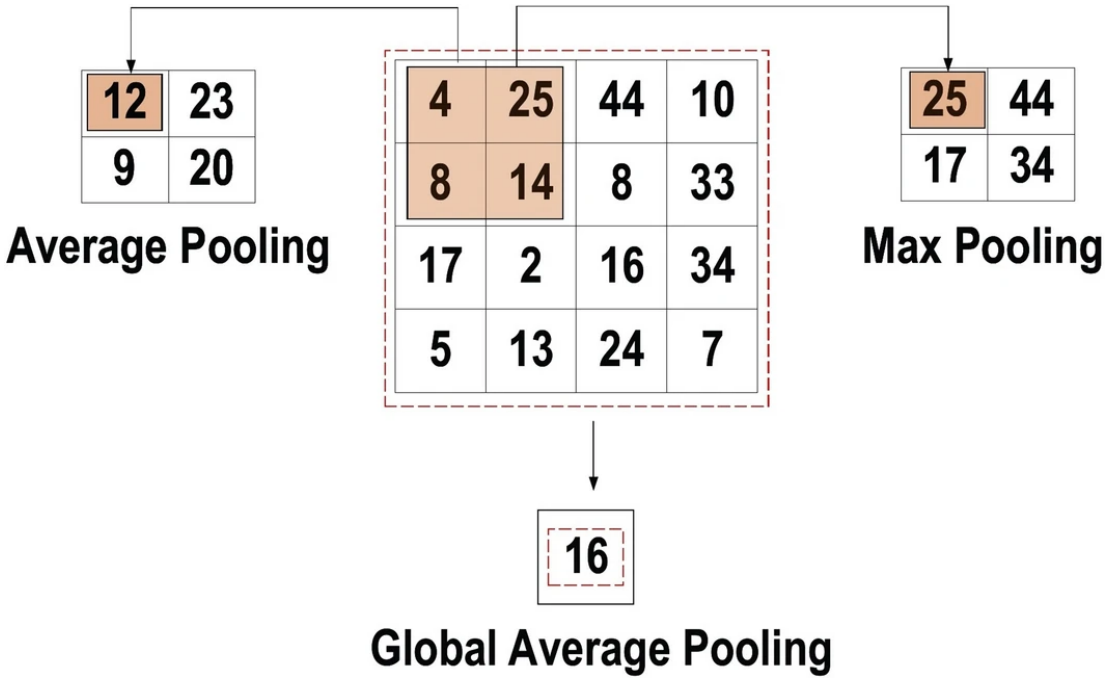
El pooling layer o capa de agrupación se encarga del submuestreo de los mapas de características. Es decir que los mapas de características de gran tamaño se reducirán generando

mapas más pequeños sin perder la información dominante (característica) en cada paso de agrupación.

Existen varios tipos de métodos de pooling para utilizar en diferentes capas. Estos métodos incluyen el tree pooling, gated pooling, average pooling, min pooling, max pooling, global average pooling (GAP), y global max pooling. Los métodos de pooling más conocidos y utilizados son el max, min y GAP. (Alzubaidi y otros, 2021)

Figura 2

Tres tipos de operaciones de Pooling



Nota. Adaptado de Review of deep learning: concepts, CNN architectures, challenges, applications, future directions (p. 17), por Alzubaidi, L., Zhang, J., Humaidi, A.J. et al, 2021, Journal of Big Data.

3.13 Punto de Marchitez Permanente - PMP

El punto de marchitez permanente es el punto donde la planta no tiene la capacidad de absorber el agua. Con la evaporación y extracción de agua de por las plantas se genera pérdida de agua en el suelo, y en ausencia de una fuente agua, las raíces de las plantas no podrán absorber el agua que está más profunda del suelo provocando un estrés hídrico. Lo cual genera el punto de marchitez permanente donde la planta no podrá recuperarse (Fernández, 2010).

3.14 Product Backlog

El product backlog es una lista ordenada de todos los elementos que podrían ser necesarios para el producto y es la única fuente de requisitos para cualquier cambio a realizarse en el producto y también es la única fuente de trabajo para el equipo Scrum.

El product backlog siempre está en constante actualización por lo que evoluciona a medida que el producto y el entorno en el que se usara también lo hacen. Es decir, que el product backlog cambia constantemente para identificar lo que el producto necesita para ser adecuado, competitivo y útil.

3.15 Red neuronal convolucional-CNN

Una red neuronal convolucional (ConvNet/CNN) es un algoritmo de aprendizaje profundo que puede tomar una imagen de entrada, asignar importancia (pesos y sesgos aprendibles) a varios aspectos/objetos de la imagen y ser capaz de diferenciar unos de otros. El preprocesamiento requerido en una ConvNet es mucho menor en comparación con otros algoritmos de clasificación. Mientras que en los métodos primitivos los filtros se diseñan a mano,

con suficiente entrenamiento, las ConvNets tienen la capacidad de aprender estos filtros/características.

La arquitectura de una ConvNet es análoga a la del patrón de conectividad de las neuronas en el cerebro humano y está inspirada en la organización de la corteza visual. Las neuronas individuales responden a los estímulos sólo en una región restringida del campo visual conocida como Campo Receptivo. Un conjunto de estos campos se superpone para cubrir toda el área visual.

3.16 Segmentación:

La segmentación hace parte del procesamiento de imágenes. Cuya tarea es clasificar cada píxel de una imagen entre un conjunto predefinidos de clases, Este proceso incluye desde tareas sencillas como la eliminación de ruido hasta tareas comunes como la identificación de objetos, personas, textos, hasta tareas más complicadas como la clasificación de imágenes, la detección de emociones, la detección de anomalías, la segmentación entre otros. En particular la segmentación de imágenes es el proceso por el cual una imagen es divida en varios subgrupos de pixeles, a los cuales se le asigna una etiqueta que representa una categoría. Haciendo uso de esas etiquetas podemos especificar límites, dibujar líneas, separar objetos entre muchas más aplicaciones. Esta tarea ayuda a reducir la complejidad de la imagen y simplificar su análisis.

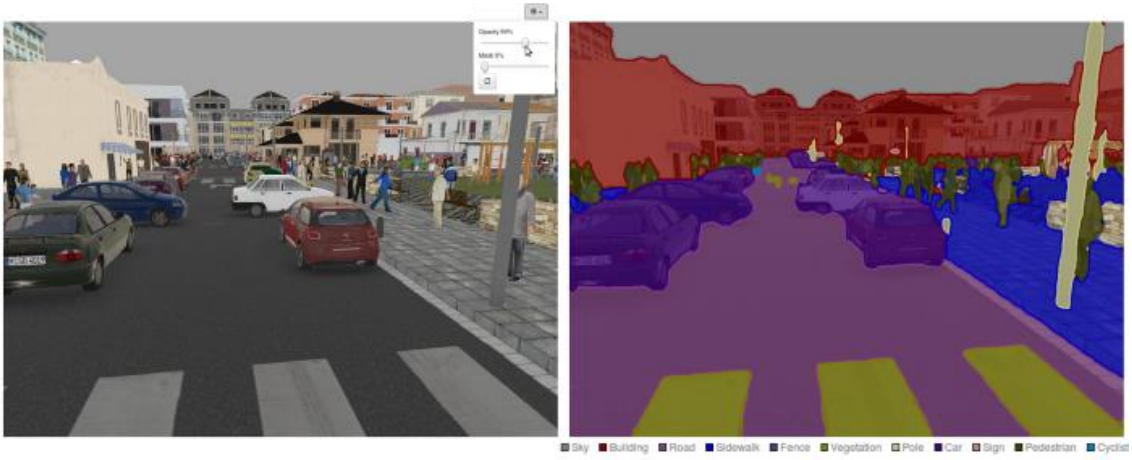
3.17 Segmentación Semántica

La segmentación semántica es una técnica del procesamiento de imágenes que ha presentado un notable progreso gracias a las ricas características jerárquicas y un marco

entrenable de principio a fin (Panqu y otros, 2018). Esta técnica pretende asignar una etiqueta categórica a cada píxel en una imagen, de esta forma, se obtiene como resultado una imagen en donde cada píxel se ha clasificado en una clase determinada.

Figura 3

Ejemplo de Segmentación Semántica



Nota. Adaptado de Image Segmentation Using DIGITS 5, por Greg Heinrich, 2016, NVIDIA Developer Blog (Nvidia dev blog).

La segmentación semántica moderna consta de tres componentes claves:

1. Una red totalmente convolucional (FCN), para hacer eficiente el aprendizaje y la inferencia de extremo a extremo.

2. Campos aleatorios condicionales, capta las dependencias locales y de largo alcance para refinar el mapa de predicción dentro de una imagen.

3. Convolución dilatada o convolución Atrous, se emplea para aumentar la resolución de los mapas de características intermedias y generar predicciones más precisas, pero manteniendo el mismo coste computacional. (Panqu y otros, 2018)

3.18 Sistema Inteligente

Se denomina sistema inteligente aquel sistema autónomo que posee un comportamiento similar a la inteligencia humana, es decir, que cuenta con la capacidad de decidir por sí mismo su comportamiento para alcanzar su propósito, basándose en sus conocimientos o experiencias acumuladas y la interacción con su entorno. Para (D´Aquila, 2007) los sistemas deben poseer tres capacidades básicas: la primera es la capacidad de razonar, para obtener conclusiones y, de ahí, tomar sus propias decisiones. La segunda es aprender, para adquirir nuevos conocimientos, a partir de sus experiencias. Y, por último, interactuar con otros sistemas inteligentes, mediante la comunicación y el entendimiento.

3.19 Sistema de Riego

Un sistema de riego es un conjunto de elementos que interactúan colectivamente para llevar agua a los cultivos de forma racionalizada con el fin de no desperdiciarla y de satisfacer las necesidades de agua de los cultivos. Todo sistema de riego debe ser estudiado previamente para determinar si es el más adecuado, teniendo en cuenta el tipo de vegetación, características físicas del suelo o la forma en que se distribuye el agua para un rendimiento óptimo. (Ambientum, s.f.)

3.20 Trimap

Es una técnica de reducción de la dimensionalidad basada en restricciones de tripletas, que preserva la estructura global de los datos, para cuantificar la precisión global de la incrustación, introduce una puntuación que refleja aproximadamente la ubicación relativa de los clusters en lugar de los puntos individuales. (Warmuth & Amid, 2019)

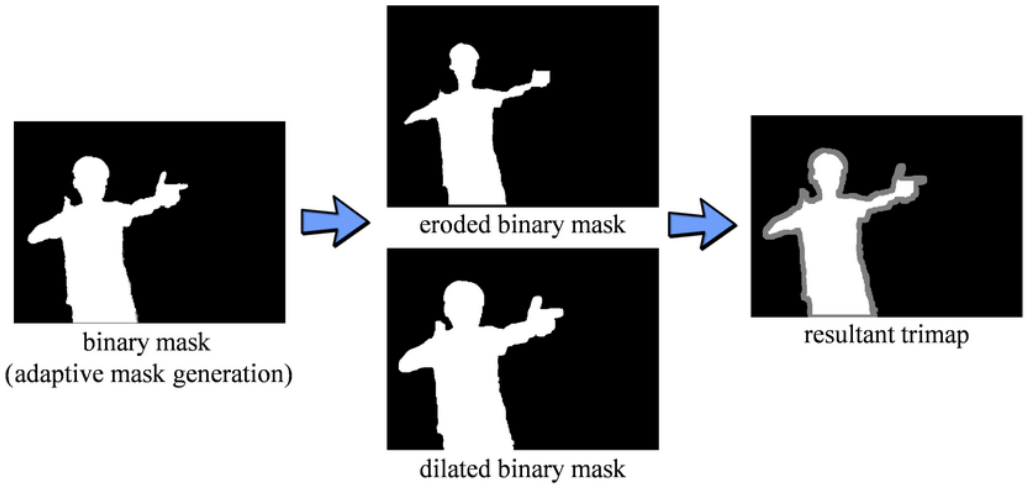
Para generar un Trimap se necesita una imagen binarizada erosionada y dilatada para restarle a la primera la segunda y así obtener la región desconocida (unknow) del trimap.

Esta técnica de matización de imágenes consiste en determinar la combinación convexa de la intensidad del primer plano (foregroung) y del fondo (background) para cada píxel parcial (mixed pixels). (Gupta & Raman, 2016)

Hasta la fecha TriMap, es el único modelo de tripletas que se acerca a los niveles de rendimiento de UMAP y t-SNE, maneja bien la estructura global, pero a veces tiene problemas con la estructura local. Curiosamente, ninguno de t-SNE, UMAP o TriMap puede ajustarse sin problemas de la preservación de la estructura local a la global a través de cualquier ajuste obvio de los parámetros. Incluso las comparaciones básicas de estos algoritmos pueden ser complicadas: cada uno tiene una función de pérdida diferente con muchos parámetros, y no está claro qué parámetros importan, y cómo se corresponden los parámetros entre los algoritmos. (Wang y otros, 2021)

Figura 4

Operaciones Morfologías para Generar Trimap



Nota. Adaptado de Real-time and Temporal-coherent Foreground Extraction with Commodity RGB Camera, por (Zhao y otros, 2015)

4 Justificación

El aumento creciente de la población mundial, la escasez de agua generada por el cambio climático y la necesidad de producir más alimentos requiere realizar un uso eficiente del agua en la agricultura. los métodos de riego tradicional son poco eficientes, causando erosiones en la tierra y deterioro de los nutrientes del suelo, lo cual implica que se debe automatizar y modernizar los sistemas riego de los cultivos, teniendo en cuenta las características del suelo las cuales son un factor importante a la hora de realizar un riego eficiente.

El aumento de personas que migran de sectores rurales al sector urbano implica la disminución de mano de obra para producción de alimentos. Esto requiere de la implementación de un sistema de riego inteligente teniendo en cuenta el tipo de suelo, la capacidad de campo y punto de marchitez. Que genere un sistema de dosificación óptimo para garantizar la producción de alimentos para el futuro.

5 Formulación del Problema

Las diferentes condiciones climatológicas, han generado graves problemas en el sector agrícola, uno de ellos, es el insuficiente aporte de agua para las actividades agrícolas que conllevan pérdidas económicas y afectaciones en el estado físico de los cultivos. Aunque existen diversos métodos de riego que han evolucionado notablemente con el tiempo para adaptarse a las necesidades, estos, no suelen integrar diferentes aspectos a tener en cuenta para regar de manera eficiente y precisa, factores importantes como: el tipo de suelo y la capacidad de campo, las necesidades hídricas en cada etapa del cultivo, los índices de evapotranspiración, las condiciones climatológicas y la humedad del suelo en tiempo real al momento de regar.

5.1 Problema Planteado

Para este proyecto de investigación se ha planteado el siguiente problema: “Cómo construir un sistema inteligente de riego utilizando Deep Learning e internet de las cosas que permita hacer un uso eficiente del agua teniendo en cuenta las características del suelo y las necesidades del cultivo.”

5.2 Preguntas Guía:

las siguientes preguntas guiarán el proceso de investigación:

* ¿Cómo construir un modelo de clasificación utilizando Deep Learning para determinar el tipo de suelo?

* ¿Cómo dosificar la aplicación del riego empleando Internet de las Cosas teniendo en cuenta las características del suelo y las necesidades del cultivo?

* ¿Cómo determinar los diferentes sectores de riego de un área del cultivo utilizando Deep Learning que permita un riego diferenciado según las propiedades del suelo?

* ¿Cómo construir un sistema de información web, empleando metodologías agiles, Deep Learning e Internet de las Cosas para hacer un uso eficiente del agua?

6 Objetivos

6.1 General

Construir un sistema inteligente de riego utilizando Deep Learning e internet de las cosas que permita hacer un uso eficiente del agua teniendo en cuenta las características del suelo y las necesidades del cultivo.

6.2 Específicos

Construir un modelo de clasificación utilizando Deep Learning para determinar el tipo de suelo.

Implementar un componente para dosificar la aplicación del riego empleando Internet de las Cosas teniendo en cuenta las características del suelo y las necesidades del cultivo.

Determinar los diferentes sectores de riego de un área del cultivo utilizando Deep Learning que permita un riego diferenciado según las propiedades del suelo.

Construir un sistema de información web, empleando metodologías agiles, Deep Learning e Internet de las Cosas para hacer un uso eficiente del agua.

7 Alcance y Limitaciones

7.1 Alcance

Este proyecto planteará la implementación de un sistema de riego inteligente y autónomo, que combinará métodos de análisis de tipo de suelo caseros con técnicas de visión por computador para determinar tres tipos de suelo (arenoso, arcilloso y franco), con este parámetro se identificaran los valores de referencia asociados a cada tipo de suelo en capacidad de campo e infiltración de agua en el suelo. Al mismo tiempo, el sistema tendrá interacción con un servicio web que determinará el calendario de riego, este, planificará cuando se deberá regar basándose en la evapotranspiración pronosticada para un intervalo determinado de tiempo y cuánto se debería humedecer el suelo según la etapa en la que se encuentra el cultivo. De esta manera, se unificarán las características del suelo identificadas (capacidad de campo e infiltración) con las indicaciones establecidas por el calendario de riego para determinar una dosificación adecuada de agua. Además, el sistema contará con sensores que permitirán determinar el porcentaje de humedad del suelo en tiempo real, dato con el cual, se contrastará la necesidad real de agua para el cultivo con el planificado por el calendario.

En los casos que el sistema note la necesidad de regar, tendrá interacción con una estación meteorológica mediante un servicio web para identificar las condiciones de precipitaciones y evapotranspiración en la zona de interés, teniendo cuenta la evapotranspiración, las características del suelo identificadas y la más reciente necesidad de humedad de suelo establecida por el calendario de riego para el cultivo. El sistema determinará una dosificación adecuada de agua a regar y deberá notificar el suceso al calendario de riego para reajustar su

programación de riego. La ejecución del riego se llevará a cabo de forma automática por el sistema, pues este, contará con el control operativo de las electroválvulas de riego.

Para el despliegue de este proyecto se construirá una estructura física con componentes automáticos de riego e IoT en un ambiente controlado a baja escala, la cual, contará con una muestra por cada tipo de suelo establecido en este proyecto. Además, cada recipiente tendrá un sensor de humedad, un sistema de mangueras con aspersor de goteo y una electroválvula independiente para cada uno.

7.2 Limitaciones

En este proyecto se trabajarán con tres tipos de suelo en específico: arenoso, arcilloso y franco, ya que por falta de recursos económicos no se puede obtener todos los componentes de riego y de IoT necesarios para desplegar y probar el sistema, al mismo tiempo en los doce tipos de suelo conocidos.

La exactitud de variables como el porcentaje de humedad del suelo están sujetas a la capacidad de precisión de medida establecidas por el fabricante del dispositivo.

El proyecto plantea un sistema de riego adecuado para evitar encharcamientos y escorrentías, pero solo los que pueden ser causados por la ejecución del mismo riego y no de otros factores climáticos o catástrofes ambientales, como, por ejemplo, fuertes inviernos.

7.2.1 Económicas

Durante las fases de pruebas y despliegue del proyecto las variables de evapotranspiración y estado del clima se limita a las brindadas por el servicio web que brindará

la estación meteorológica, pues, no se cuenta con recursos suficientes para adquirir o construir una estación meteorológica propia que determine estas variables.

7.2.2 Tiempo

El despliegue del producto final de este proyecto no se probará durante todo el periodo de vida del cultivo, debido a que el tiempo estimado para el desarrollo del proyecto es limitado y las etapas previas consumen la mayor parte del tiempo.

8 Análisis y Diseño

Para el desarrollo de este proyecto se abordaron tres líneas de investigación, Internet de las cosas (IoT), Inteligencia artificial e ingeniería de software, a continuación, se describen como se implementaron los componentes en cada una de las áreas de investigación:

8.1 Internet de las cosas

El componente de Internet de las cosas cumple cuatro funciones principales: recolectar información de campo en tiempo real mediante sensores, almacenamiento, envío de información al sistema y control operativo de los componentes del sistema de riego.

8.1.1 Análisis

Los sistemas de riego se diseñan teniendo en cuenta diversos factores que pueden incidir para que su desempeño sea eficiente. En general, existen tres métodos de riego (Superficie, aspersión y localizado) que han evolucionado con el paso del tiempo, cada uno contempla diversas variables al momento de regar, algunas de ellas son la topografía, las características del suelo, el tipo de cultivo, el coste de instalación, entre otras. En el módulo 1 del libro Manual de Riego para Agricultores de Rafael Fernández, el calendario de riego es una estrategia de planificación del riego que integra información de diversas fuentes. Una de ellas proviene de las estaciones meteorológicas, que brinda al calendario datos que permiten estimar la evapotranspiración, el estado del clima, entre otras. Otra proviene de las necesidades teóricamente parametrizadas de un cultivo según la edad o etapa de desarrollo en la que se encuentre. Sin embargo, en riegos automáticos esta programación no garantiza ser cien por ciento asertiva o eficaz, pues en ocasiones sin antelación las condiciones del clima pueden

cambiar bruscamente, implicando que se pueda regar sin ser necesario o por el contrario se requiera hacerlo antes de lo programado.

Otra forma de programar los calendarios de riego es contar con datos diarios o en tiempo real, aunque esta no es asiduamente empleada por los costos y la dificultad de contar con este tipo de información. Los avances en el área de IoT abre cada vez más la posibilidad de mejorar las expectativas en la agricultura de precisión, pues integrar estas tecnologías permitirán medir más variables que orienten una programación en tiempo real, para aplicar un riego riguroso y eficiente.

Como se planteó en la propuesta inicial de este proyecto, el objetivo era diseñar un sistema de riego inteligente y autónomo que integrara servicios web de una estación meteorológica y calendario de riego. Aunque estos servicios no están disponibles a la fecha de realización del trabajo, es factible desarrollar un prototipo de riego en tiempo real y autónomo, que integre dispositivos de IoT combinado con estrategias de programación de riego y propiedades del cultivo.

Para medir la humedad del suelo existen métodos directos, como el gravimétrico o con dispositivos como los tensiómetros, la sonda de neutrones (requieren una buena calibración, ya que, funcionan enviando una cierta cantidad de neutrones del tamaño de las moléculas del átomo de hidrogeno son bastante costosas y requieren de una licencia para su operación debido al material radioactivo que manejan). Dispositivos que miden la resistencia eléctrica o la tensión del suelo. (Martin & Muñoz, 2017)Sin embargo, el avance del IoT permite realizar esta medida mediante sensores de humedad de suelo capacitivos, estos pueden realizar esta medida en cuestión de segundos y ser operados mediante un microcontrolador. Dado que

este proyecto desarrolla un prototipo de sistema de riego, se considera asequible emplear el sensor de humedad capacitivo higrómetro modelo v1.2, ya que es de bajo costo, posee una capa de protección anticorrosiva que aumenta considerablemente su vida útil en comparación con sensores resistivos como el FC-28, y además es compatible con la arquitectura planteada.

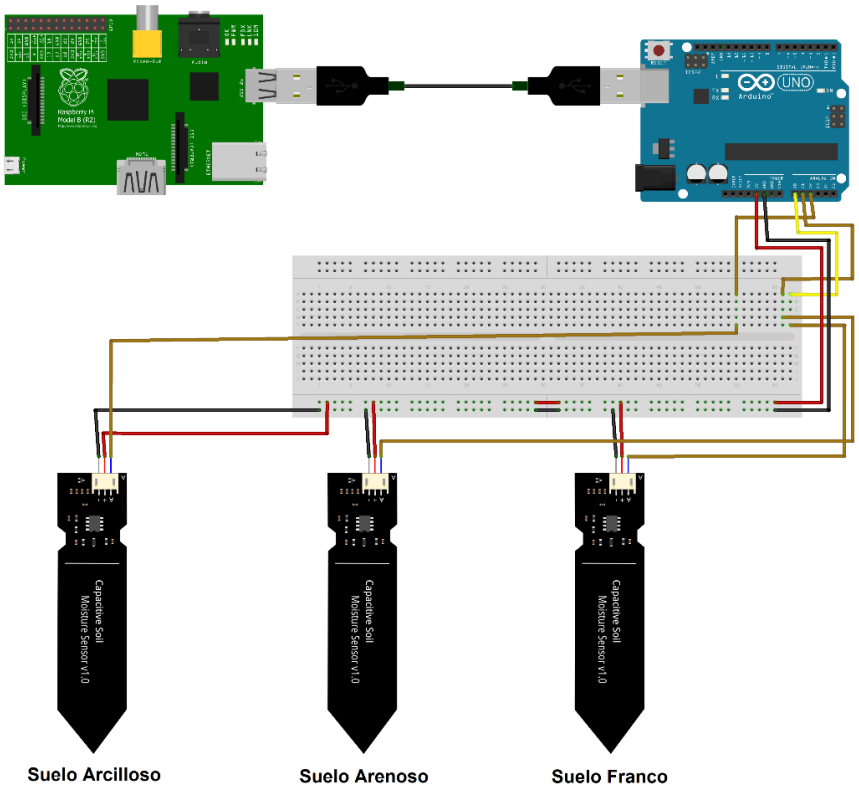
Los dispositivos de radiofrecuencia son componentes electrónicos que permiten transmitir y recibir señales de radio entre dos aparatos, son facilitadores de las comunicaciones inalámbricas ya que no requiere una línea de visión. Los módulos de radiofrecuencia permiten diseñar una red de comunicación en forma de árbol, donde hay un nodo principal (padre) y todos los demás son secundarios(hijos). En consecuencia, usar como canal de comunicación la radiofrecuencia, como el NRF24L01+ es una opción viable al ser de bajo costo, compatible con la arquitectura propuesta y permite una comunicación inalámbrica bidireccional entre la placa de desarrollo y el módulo controlador.

Los microcontroladores son en definitiva un circuito integrado que incluye componentes de un ordenador como procesador, memoria (RAM, ROM/PROM/EPROM), módulos para el control de periféricos y líneas de E/S para comunicarse con el exterior. Su uso trae algunos beneficios, como la diminución del tamaño en el producto y mayor flexibilidad para realizar modificaciones; pues solo necesita cambios en el programa de instrucciones. El uso de estos dispositivos es empleado en multitud de sistemas de nuestra vida cotidiana, como en los televisores, impresoras, instrumentación electrónica; placas de desarrollo, etc. (Zapata & Zapata, 2011). En la actualidad, Arduino es una de las placas de desarrollo más populares del mercado, ya que esta plataforma integra hardware y software Open Source que facilita el desarrollo de prototipos multidisciplinarios electrónicos de forma económica y rápida (Peña, 2020). Por lo

anterior, se decide usar las placas de Arduino nano y Uno, implementado con el uso de protoboards (placa de pruebas) para recibir y enviar la información sobre el funcionamiento y estado de los dispositivos de IoT (sensor de humedad, módulo de radiofrecuencia y electroválvula).

Figura 5

Conexión de los sensores con el Arduino por cada tipo de suelo



Nota. Elaboración propia.

Un sector de riego es una fracción de un área abarcada por un sistema de riego, que es individualizada por sus características; la textura del suelo, el emisor de riego, el tipo de cultivo o la disponibilidad del caudal. Cada bloque requiere elementos de control independientes

(válvulas, mangueras, cableado, etc.). Por ejemplo, en un lote con monocultivo se pueden identificar más de un tipo de suelo, lo ideal, seria definir un sector de riego por cada textura identificada. Aunque las necesidades del cultivo son las mismas en los sectores, la variación en el suelo (capacidad de campo e infiltración) requiere aplicar de diferente forma. Otro caso, se puede presentar que un lote presente el mismo tipo de suelo, pero tengan sembradíos de cultivos diferentes, aunque las capacidades del suelo son las mismas, las necesidades del cultivo son diferentes, lo que implicaría administrar un riego diferenciado. Este proyecto no tendrá a cargo la delimitación de sectores de riego, pero si contempla esta distribución en un sistema de riego, por lo cual, se debe relacionar en un sector los dispositivos de IoT y riego que tenga a disposición.

La infraestructura con los componentes del sistema de riego será de la siguiente manera: cada muestra de suelo a trabajar tendrá un sistema independiente de mangueras plásticas de media pulgada o mayor diámetro de acuerdo con la necesidad, desde el punto de acceso al agua hasta las plantas, implementado el riego por goteo. Así mismo, para obtener un sistema de riego que pueda ser controlado de forma autónoma, se empleará en la infraestructura componentes de riego que puedan ser monitorizados y operados con el Arduino, dentro de estos, se emplearán de forma independiente para cada sector de riego: electroválvulas para que puedan activar o detener el flujo del agua en las tuberías, sensores de flujo en cada sistema de tuberías para medir el caudal con el que circulará el agua en la tubería, pues, para regar de forma eficiente se tendrá en cuenta características como la capacidad de infiltración que posee cada tipo de suelo, por ejemplo, esta determina que suelos como el arenoso se requieren regar con mayor frecuencia, pues su capacidad de infiltración es alta, pero también, en suelos como el arcilloso es correcto regar con una frecuencia más baja de agua, ya que en este la infiltración del líquido es más lenta.

8.1.2 Requisitos funcionales.

Del análisis anterior se identifican los siguientes requisitos funcionales:

- Medir el porcentaje de humedad del suelo en tiempo real.
- Establecer un medio de comunicación inalámbrico y de largo alcance.
- Establecer sectores de riego según el tipo de suelo, de cultivo o emisor.
- Medir el flujo de agua en los sectores de riego.
- Implementar componentes de riego autónomos (electroválvulas).
- Aplicar un riego diferenciado según las necesidades de cada sector por goteo.
- Programar y controlar el sistema de riego automáticamente.

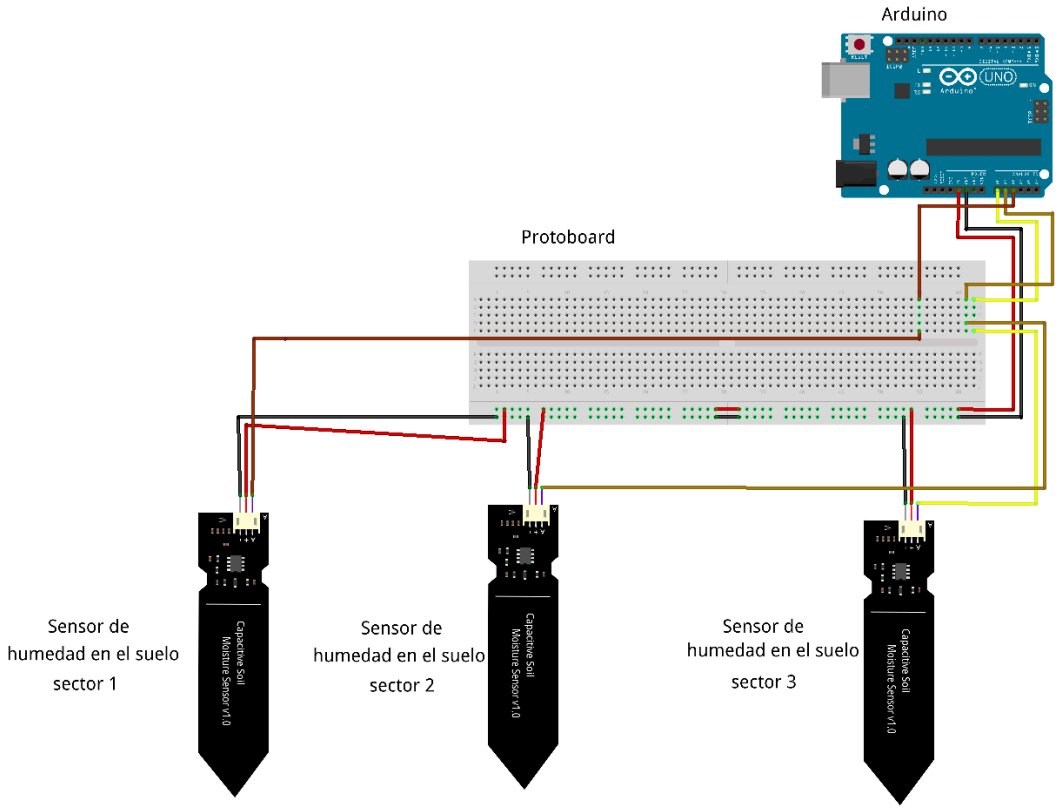
8.1.3 Diseño

Teniendo en cuenta el análisis y los requisitos funcionales, se realiza el diseño del sistema con los componentes principales de riego autónomo e IoT, para cumplir con los requisitos establecidos.

En la Figura 6 se muestra de forma específica la conexión de los componentes de IoT que requiere el sistema para cada sector de riego.

Figura 6

Diseño de la Conexión de los Componentes de IoT

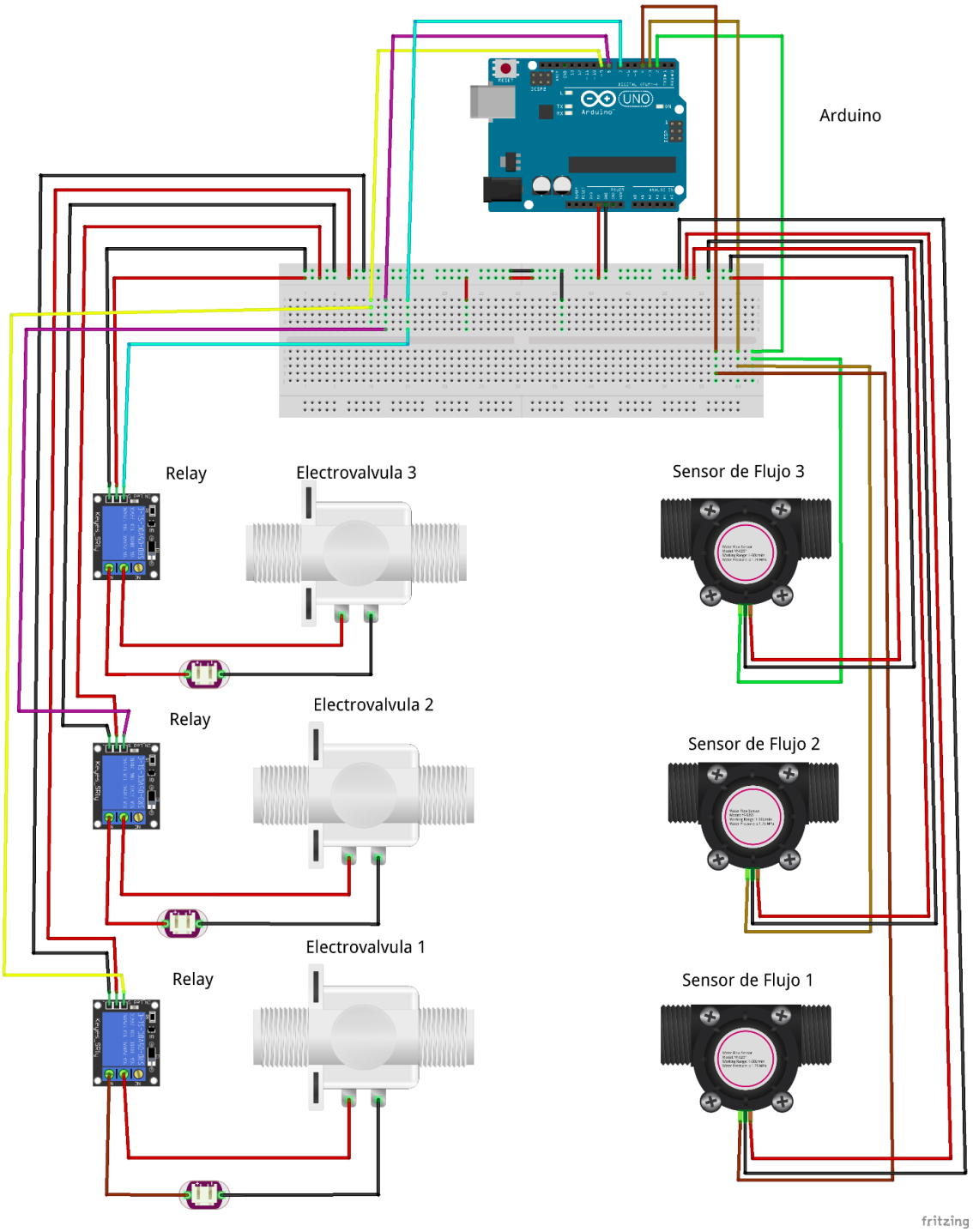


Nota. Elaboración propia.

En la Figura 7 se muestra particularmente la conexión de los componentes de riego autónomo con el Arduino.

Figura 7

Conexión de Componentes de Riego e IoT

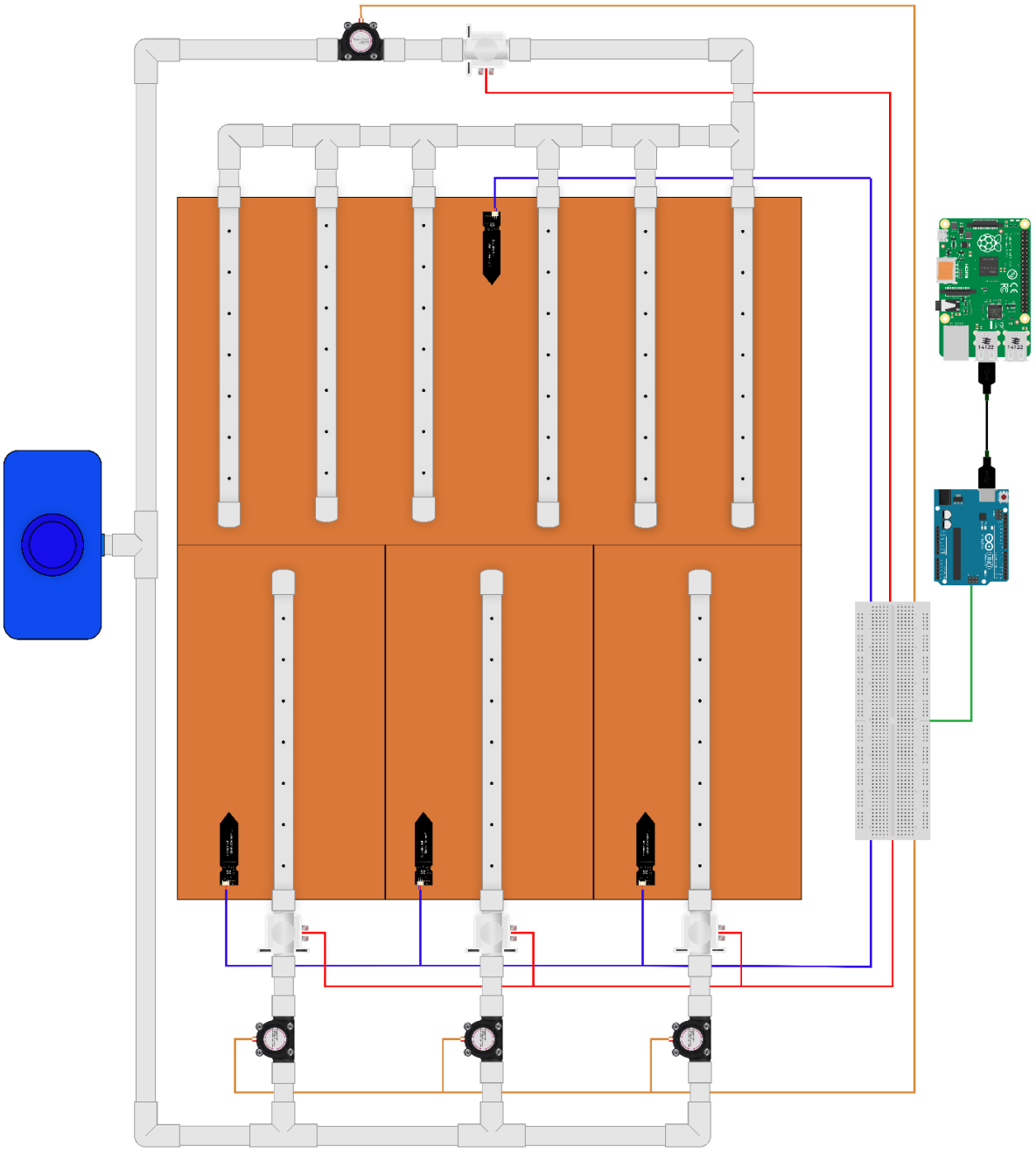


Nota. Elaboración propia.

En la Figura 8 se muestra en forma general las conexiones de los componentes de riego autónomo e IoT para cada sector de riego.

Figura 8

Diseño de Conexión del Sistema de Mangueras para Diferentes Sectores de Riego



Nota. Elaboración propia.

8.1.4 Montaje del sistema

Diseñar y ensamblar una infraestructura sencilla en un entorno controlado que siga la lógica o distribución de la figura 8, donde se emulen sectores de riego con emisores de riego localizado (microaspersión, goteo y cinta), con sus componentes de IoT y riego. Esto permite integrar funcionalmente todos los componentes de hardware, software y demás materiales del sistema de riego, para probar su comportamiento individual y en interacción con el sistema en general.

De igual forma, esta infraestructura permite probar el comportamiento del sistema simulando diferentes condiciones, por ejemplo, ingresar manualmente datos emulando un cultivo en etapas más avanzadas donde sus necesidades de agua puedan variar, y así comprobar que el sistema ajusta su dosificación según la necesidad.

8.2 Ingeniería de software

En este componente se define el marco de trabajo apropiado para el desarrollo del proyecto, las directrices que tendrá en cuenta el módulo controlador para decidir cuándo, cuanto y como regar. También, desarrolla una herramienta web, para que el usuario pueda vigilar y operar el sistema fácilmente.

La agilidad es una mejor manera de resolver los problemas complejos, con enfoque en la generación de espacios de confianza e innovación. La habilidad para aplicar estas técnicas de trabajo está determinada por el hecho de que el equipo de trabajo sea multifuncional, enfocado y autogestionado, lo cual contribuye a generar estrategias enfocadas al cambio,

permitiendo identificar rápidamente la mejora en los procesos. Con lo anterior se puede usar la metodología en los siguientes escenarios:

Proyectos de innovación, donde la investigación es clave

Proyectos con alto grado de complejidad que debe ser abordados iterativamente

Proyectos con alta probabilidad de que los requisitos puedan cambiar con el tiempo.

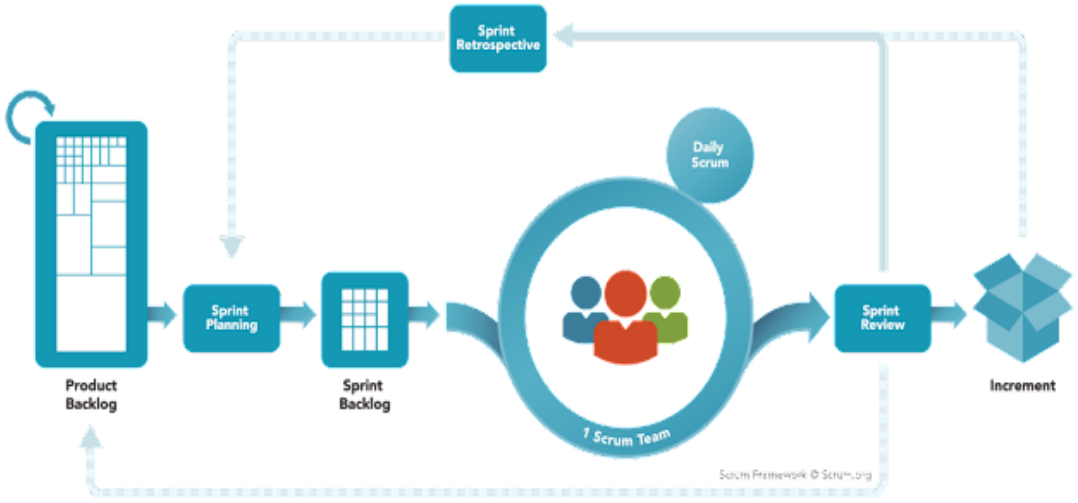
Entornos complicados donde mantener la motivación es fundamental.

8.2.1 Análisis

El marco de trabajo Scrum es uno de las más usados actualmente por su eficiencia en el desarrollo de proyectos de software, pues permite descomponer el proyecto en requerimientos y estos a su vez en pequeñas tareas, para ayudar a realizarlo de manera flexible, adaptable y eficiente. Esta plantea desarrollar las tareas en ciclos cortos y regulares de trabajo (también llamados iteraciones), cada ciclo consta de 3 fases: planificación, ejecución y optimización. En la primera fase se planifica cada iteración, su duración, definición de tareas y el encargado de desarrollarla. En la fase de ejecución o desarrollo se abordan las actividades propuestas para cada Sprint, que por lo general incluyen actividades de diseño, desarrollo y pruebas. En la fase de optimización se realizan revisiones de los avances del proyecto con el equipo; permitiendo hacer cambios o correcciones oportunamente, se analizan las fortalezas y dificultades del grupo de trabajo. Al finalizar todas las iteraciones se realiza la entrega final del producto.

Figura 9

Scrum Framework



Nota. Adaptado de Scrum Framework, por J. F. Huambachano, 2017, Scrum.org (scrum.org).

La Figura 10 muestra un formato utilizado para realizar las historias de usuarios.

Figura 10

Formato de Historias de Usuario

Identidades	Historia de usuario	Descripción	Criterios de aceptación	Responsable	Prioridad	Complejidad

Nota. Elaboración propia.

Las historias de usuarios permiten definir los requerimientos en el lenguaje del usuario, a partir de estos se forma el Product backlog, el cual, es un listado de los requisitos identificados del sistema, para priorizarlos según el grado de importancia o urgencia que tenga

para el proyecto. Estos a su vez se desglosan en tareas que describen en leguaje más técnico de lo que deberá desarrollar los programadores, cada tarea será asignada a una persona en específico. Algo característico de las tareas es que deben ser específicas, medibles, alcanzables y realizables en periodos muy cortos de tiempo. Una vez identificadas las tareas, se agrupan para determinar los conocidos Sprints, teniendo en cuenta que su realización por lo general no debe exceder las 4 semanas.

En el desarrollo del proyecto se considera oportuno implementar metodologías agiles como SCRUM, ya que han demostrado su eficacia en el desarrollo de proyectos de alto grado de complejidad, de enfoque investigativo o comercial.

Por otro lado, la adopción masiva del internet a nivel mundial en las últimas décadas ha transformado la cotidianidad de los seres humanos en todas las áreas, como el trabajo, el entretenimiento, solicitud y oferta de servicios, incluso en relacionarnos con amigos o parejas sentimentales. Todos estos servicios se han integrado masivamente mediante las aplicaciones web y móviles que son consumidas desde un computador o un celular. Por tal razón, las empresas o servicios que no se adapten o implementen con este paradigma se arriesgan a perder competitividad o incluso desaparecer del mercado.

Sin embargo, el ámbito de estos servicios web no se desarrolla para exclusividad de actividades comerciales de consumo masivo, pues también se implementan en la operación u optimización de procesos internos de un producto u organización. Por lo anterior, es factible desarrollar una plataforma web que le permita a los usuarios interactuar con el sistema de riego desde cualquier dispositivo móvil con acceso a internet. Esto abre la posibilidad de aprovechar la versatilidad de este servicio para monitorear o ajustar parámetros de riego desde cualquier lugar

donde se encuentre el usuario, permitirle registrar y operar los componentes en los sectores de riego, modificar variables establecidas inicialmente por el sistema para tener un sistema más parametrizable a las consideraciones del usuario y ser una herramienta informativa del funcionamiento en general del riego.

El patrón arquitectural modelo vista controlador (MVC) es muy utilizado para el desarrollo de aplicaciones web, ya que, separa el sistema en tres componentes (Modelo-Vista-Controlador). Esta separación permite trabajar simultáneamente en el desarrollo del software, hacer cambios sin que interfieran entre sí, y obtener una aplicación escalable, mantenible y fácil de expandir.

Existen diversos lenguajes de programación especializados para el desarrollo de cada componente (Frontend o Backend), como también los hay para desarrollar todos los componentes de la arquitectura. Para trabajar estos lenguajes existen diferentes entornos, uno de ellos son los Framewoks, estos, son un marco de trabajo que estandariza conceptos y prácticas que agilizan el desarrollo de una tarea especifica. Dentro de las más populares tenemos Angular, Laravel, Django, ApacheSpark, Spring, entre otras. Además, existen los Microframeworks, son igualmente un marco de trabajo para aplicaciones web, pero más minimalista. No se refiere a ser limitado en sus funcionalidades, sino que es flexible para crecer de apoco, acorde a las necesidades del proyecto. Entre los más conocidos está Flask, Express, Lumen, Sinatra, etc. Escoger el más apropiado depende directamente del objetivo del proyecto y las funciones que cada uno pueda ofrecerle al mismo, aunque también influye la interoperabilidad con otros componentes, las preferencias del programador y el dominio que tenga del mismo (Cristancho, 2022).

Por otro lado, las librerías en programación son un conjunto de archivos (métodos o funciones puntuales) que se utilizan para desarrollar una necesidad en específico. Este archivo importable, ayuda a resolver una acción de manera fácil, ahorrando gran parte del trabajo y facilitando la integración de otros servicios o tecnologías. De las más populares están React.js, Jquery, Dojo Toolkit, CreateJs, etc.

En cuanto a lenguajes de programación, Python es uno de los más utilizados actualmente, ya que, su sintaxis es muy sencilla, fácil de aprender, es multiparadigma, permite crear todo tipo de programas (web, móvil, estadísticos, etc.) y su interoperabilidad permite trabajar con otros lenguajes (Santos, 2020).

Para el desarrollo de la plataforma web, se considera implementar Flask como Microframework para el Backend y la librería Reajct.js para el Frontend. Esto permite aprovechar las ventajas que cada uno posee como también manejar una interoperabilidad en la plataforma. Como gestor de base de datos se utilizará PostgresSQL mediante su servicio de almacenamiento en la nube gratuito y de código abierto, Elephant. En ella se almacena información personal del cliente, los componentes de riego e IoT de cada sector, información del cultivo, y las lecturas tomadas por los sensores.

El sistema cuenta en su aplicativo web un entorno de registro y login para los usuarios, en este módulo, el cliente podrá: visualizar el estado actual del riego, tener informe del funcionamiento de los componentes, accionar o detener el riego de forma manual y la actualización de algún evento en un sector de riego. La creación de microservicios ayudará a la comunicación con otros sistemas.

8.2.1.1 Consideraciones para regar.

Conocer la textura del suelo permite identificar algunas características que se deben considerar para aplicar un riego eficiente que evite encharcamientos o escorrentías en el cultivo. Como se muestra en la Tabla 2, la velocidad de infiltración varía según el tipo de suelo, por lo tanto, la lámina de riego debe aplicarse de forma diferenciada durante un determinado tiempo, es decir, la cantidad de líquido que se riega no siempre será la misma. Por ejemplo, en suelos arcillosos la velocidad de infiltración es la más lenta (2-5 L/h), por lo cual, se debe considerar no aplicar una cantidad de agua que exceda los litros que se pueden filtrar realmente en el terreno. Esto implica ajustar el caudal de riego en los emisores o regar en periodos muy cortos de tiempo. Por el contrario, en terrenos arenosos el volumen o tiempo de irrigación será mayor en referencia con los otros perfiles de suelo, ya que el agua se filtra con mayor velocidad.

Tabla 2

Velocidad de Infiltración

Textura del Suelo	Litros de agua infiltrados por hora
Arenoso	12 - 25
Arenoso - franco	8 – 12
Franco	7 - 12
Franco - limoso	7 - 10
Franco - Arcilloso	6 - 8
Arcilloso	2 - 5

Nota. Tomado de Agronomía del riego, en inforiego.com

La capacidad de campo (CC) y el punto de marchitez permanente (PMP), como se muestra en la Tabla 3 son propiedades que también varían según el tipo de suelo.

Tabla 3

Capacidad de Campo y Punto de Marchitez Permanente

#	***Textura del Suelo***	***Capacidad de Campo***	***Punto de Marchitez Permanente***
1	Arenoso	6 - 12 % 0,07 - 0,17 m^3/m^{-3}	2 - 6 % 0,02 - 0,07 m^3/m^{-3}
2	Franco	18 - 26 % 0,20 - 0,30 m^3/m^{-3}	8 - 12 % 0,07 - 0,17 m^3/m^{-3}
3	Arcilloso	31 - 40 % 0,30 - 0,40 m^3/m^{-3}	15 - 19 % 0,20-0,24 m^3/m^{-3}

Nota. Adaptado de Evapotranspiración del cultivo (pág. 144), por (Allen y otros, Evapotranspiración del cultivo, 2006), en fao.org

La capacidad que tiene el suelo para retener agua después del drenaje causado por la fuerza gravitatoria se conoce como capacidad campo, esta propiedad es importante tenerla en cuenta al momento de regar para evitar irrigar más agua de la que un suelo puede retener. Por otro lado, el punto de marchitez permanente permite prever el porcentaje de humedad en el cual las plantas ya no pueden absorber el agua del suelo, a tal punto que ya no podrán recuperarse, por lo cual, el sistema de riego debe evitar dejar llegar al cultivo.

Otro aspecto para tener en cuenta por el sistema de riego es la cantidad de agua que puede aplicar el emisor de riego en un determinado tiempo, más conocido como caudal. Los fabricantes o distribuidores de estos productos normalmente disponen al usuario la capacidad de distribución unitaria en litros. Sin embargo, para calcular la cantidad de agua que se distribuirá en litros por hora para una determinada área de cultivo, se debe tener en cuenta la cantidad de emisores que hay en la zona y la distancia que hay entre los laterales.

8.2.2 Requisitos funcionales:

Entendiendo las temas o consideraciones que se analizaron anteriormente para desarrollar el sistema de riego, se identificaron los siguientes requisitos funcionales:

El sistema requiere de una aplicación web que le muestre al usuario la información relacionada con el cultivo y el riego. Además, gestionar manualmente algunas funciones del sistema de riego.

La página web estará alojada en el servidor.

La página web será desarrollada basada en la arquitectura MVC, en la cual, la capa de control será desarrolla en Python, la capa vista será desarrollada con React.js y para la capa modelo se empleará el gestor de base de datos Postgres.

La Base de Datos Almacena información correspondiente del usuario, el cultivo, y el riego.

La plataforma requiere de microservicios que permitan la integración e interoperabilidad entre los diferentes módulos del sistema.

El sistema debe tener un módulo que le permita al usuario cargar imágenes de las pruebas de campo, para que el sistema clasifique el tipo de suelo.

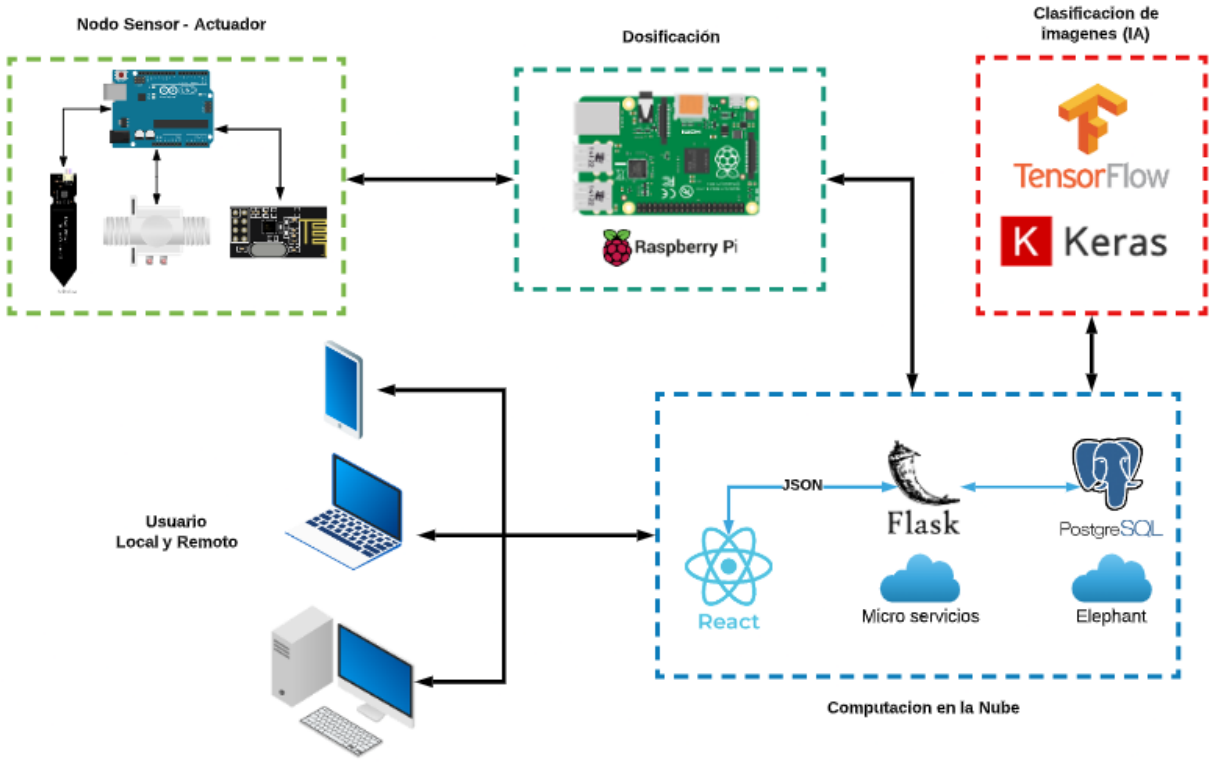
El sistema debe dosificar la cantidad de agua a regar durante un tiempo determinado, teniendo en cuenta las necesidades del cultivo y las características reales del suelo.

8.2.3 Diseño

En la Figura 11 se muestra la arquitectura que se implementó para el desarrollo de este proyecto, en ella se describen la interacción de los diferentes módulos del sistema, los lenguajes de programación implementados y los protocolos de comunicación entre ellos.

Figura 11

Arquitectura Del Sistema de Riego

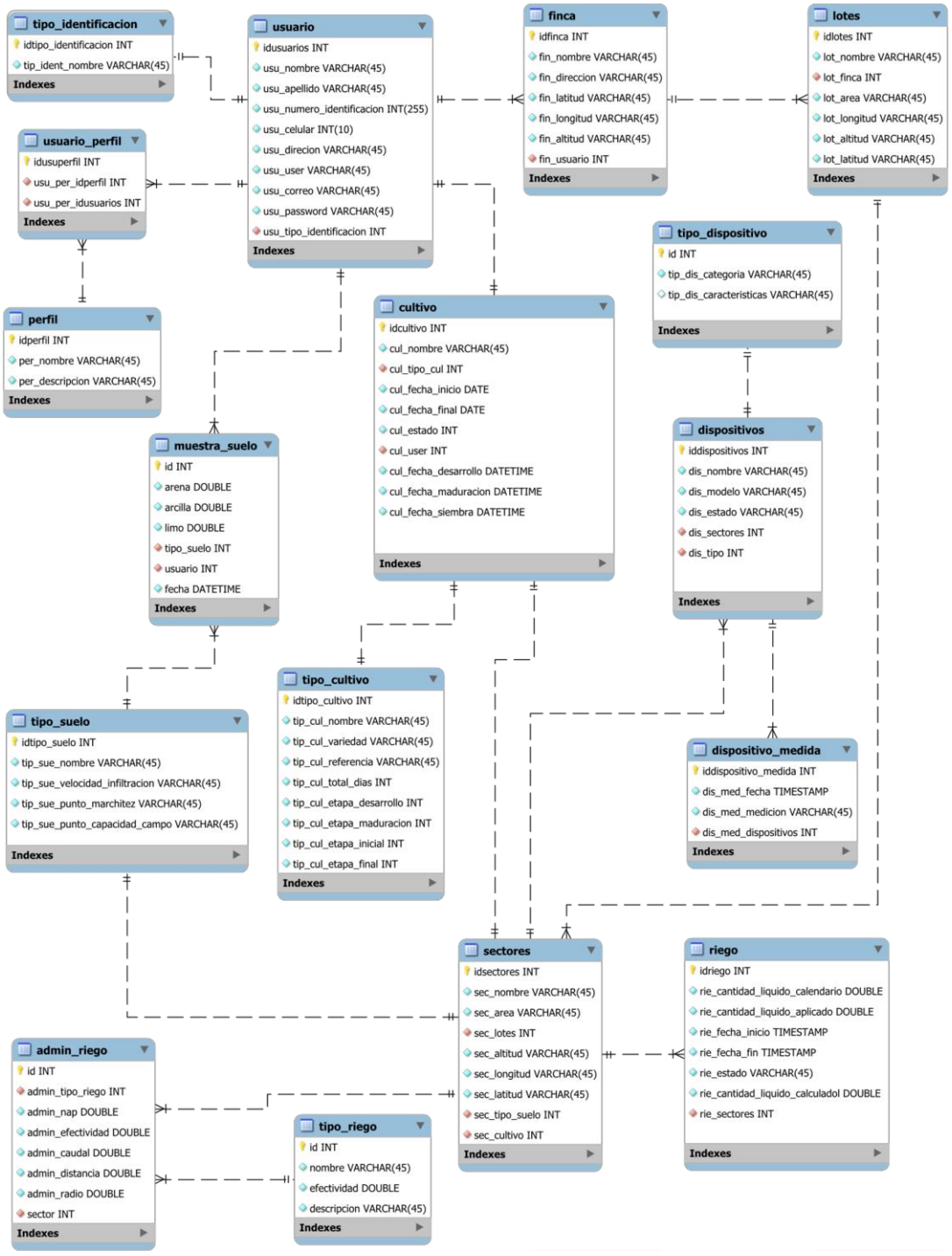


Nota. Elaboración propia.

En la Figura 12 se muestra el modelo entidad-relación, es decir, la estructura de la base de datos que se necesita para el desarrollo y funcionamiento del sistema.

Figura 12

Diseño de la Base de Datos del Sistema de Riego



Nota. Elaboración propia.

8.2.4 Desarrollo

En el desarrollo del proyecto se implementan diferentes leguajes de programación en cada uno de sus módulos, a continuación, se describen los lenguajes a implementar en la arquitectura planteada de la Figura 12.

8.2.4.1 Nodo sensor-actuador.

Este módulo está compuesto por los componentes de IoT y de sistema de riego, como se mencionó el Arduino es el encargado de interactuar con los componentes, para programar este microcontrolador se empleó el lenguaje de programación C++ mediante su IDE Arduino. Este nodo se comunicará con el controlador mediante sistema cableado.

8.2.4.2 Controlador.

El módulo controlador está compuesto principalmente por la Raspberry pi con sistema operativo Raspbian, este a su vez, emplea como lenguaje de programación base Python 3, para establecer comunicación con el Arduino. Este módulo se compone de los siguientes submódulos:

Modulo que recibe y envía la información al Arduino.

Modulo que establezca la comunicación con el gestor de base de datos.

Modulo principal se encarga de integrar la información requerida por los otros módulos para establecer la dosificación del riego.

8.2.4.3 Computación en la nube.

El desarrollo de la página web está comprendida con Flask para el Backend y React.js para el frontend, los cuales implementaron los siguientes módulos principales:

Modulo 1: Se desarrollo el módulo de login para mantener consistencia de datos de los usuarios con relación a sus riegos.

Modulo 2: Se desarrollo un módulo de estadísticas, el cual muestre las gráficas relacionadas con humedad del suelo, capacidad de campo y cantidad de agua aplicada en los sectores de riego.

Modulo 3: El desarrollo del módulo para la gestión de los sectores de riego.

8.2.5 Pruebas

La metodología SCRUM es iterativa, cada ciclo tiene como objetivo desarrollar una funcionalidad entregable ante el cliente, es decir, que al final de cada Sprint se realizaran pruebas de validación con el equipo, pero anterior a esto, cada Sprint deberá establecer dentro de sus actividades o tareas el diseño y ejecución de pruebas unitarias de lo desarrollado y pruebas funcionales según los criterios de aceptación establecidos para cada actividad.

8.2.6 Despliegue

El desarrollo tradicional de un producto software implicaba preocupaciones, por la infraestructura del servidor que alojaría la aplicación, crear entornos de programación, mantener la seguridad dentro de la instalación, etc. En general, resultaba siendo un desgaste de tiempo y económico, tanto para desarrolladores como para el cliente, además, las aplicaciones

quedaban obsoletas rápidamente. Sin embargo, el desarrollo de las herramientas tecnológicas ha permitido desarrollar servicios que reduzcan los costos y agilicen el proceso de despliegue en aplicaciones sencillas o empresariales sofisticadas. Con la consolidación del Cloud Computing como una herramienta que se encarga de ejecutar parte del trabajo en la nube, surgieron diferentes tipos servicios como las Paas (plataforma como servicio), IaaS (Infraestructura como servicio) y SaaS (Software como servicio). (¿Qué es PaaS? - Descripción de plataforma como servicio, 2022)

Las Plataformas como Servicio (PaaS) es un entorno de desarrollo e implementación herramientas informáticas completo en la nube, que le permite a desarrolladores y usuarios alojar las aplicaciones con mayor rapidez y a bajo costo, acceder fácilmente a herramientas de desarrollo, servidores y entornos de programación. Todo esto libera del diseño y el mantenimiento de una infraestructura que suelen estar relacionadas para el funcionamiento de una aplicación (¿Qué es una PaaS?, 2022). En el mercado existen diversas PaaS, cada una con sus diferentes enfoques, servicios y costos. Sin embargo, algunas de ellas ofrecen un plan con servicios limitados de forma gratuita; entre las más populares se encuentra Amazon web Services, IBM Cloud y Google Cloud. Pero existen otras alternativas de fácil acceso como Heroku, Vercel, Netlify, Render, Railway, entre otras.

Teniendo en cuenta lo anterior, se escogen las plataformas de servicio para desplegar las capas o nodos desarrollados en el sistema de riego, teniendo en cuenta que los planes gratuitos que estas ofrecen no limiten el despliegue de este por su tamaño final. De esta manera, la capa Vista (frontend) de la plataforma web que se desarrolla para que el usuario interactúe con el sistema de riego, se despliega en Netlify. Ya que su plan gratuito maneja los

entornos de programación compatibles con la arquitectura y lenguajes empleados en su desarrollo.

El módulo encargado de clasificar el tipo de suelo a partir de las imágenes tomadas por el usuario de las pruebas de campo es alojado en la misma PaaS establecido para el backend.

8.3 Inteligencia Artificial

En este componente se define los procesos que en las pruebas de campo se puedan abarcar con el procesamiento de imágenes mediante un modelo de inteligencia artificial. Se describe los procedimientos que se deben realizar al formar un conjunto de datos apropiado para emplear una arquitectura de Deep Learning en el entrenamiento del modelo. De igual forma, se establece la integración de esta herramienta en la plataforma web del sistema de riego.

8.3.1 Análisis

Los suelos están compuestos por partículas minerales de arcilla, limo y arena, que varían de tamaño y forma como lo muestra la Figura 13. Estas características físicas del suelo ayudan a determinar la infiltración, la capacidad de almacenar el agua (C.C) y la capacidad de ceder el agua a las plantas. El departamento de agricultura de Estados Unidos (USDA) emplea una metodología de clasificación de suelos en cuatro grupos texturales principales: arena, franco, limo y arcilla. De estos grupos se pueden determinar doce clases texturales de suelo, las cuales son determinadas según la proporción cuantitativa de partículas minerales que se encuentre en el suelo como se puede ver en la figura 14.

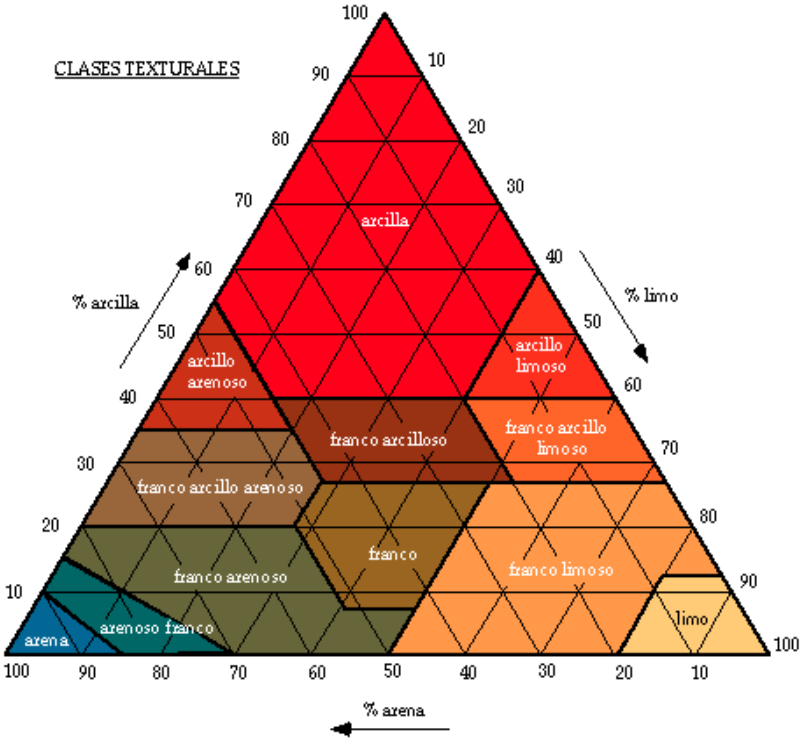
Figura 13

Partículas Minerales del Suelo

Nota. Adaptado de Manual de riego para agricultores módulo1Fundamento de Riego (p.36) por Fernández, Ávila, López, Gavilán, & Oyonarte, 2010.

Figura 14

Clases de Texturas



Nota. Adaptado de Introducción a la edafología (tema 4), por Dorronsoro en (edafologia.net)

La Tabla 4 muestra la proporción de partículas minerales para cada tipo de suelo según el triángulo de la Figura 14. Esta característica física del suelo es relevante, ya que, en un suelo arenoso su infiltración es más rápida, pues sus poros son grandes por lo tanto tiene poca capacidad de retención de agua, mientras que los suelos arcillosos los poros son más pequeños lo que provoca que la infiltración sea más lenta, debido a que, pueden retener mayor cantidad de agua.

Tabla 4

Texturas de Suelo Según la Clasificación Americana (USDA)

#	Textura del Suelo	% Arena	% Limo	% Arcilla
1	Arenoso	86-100	0-14	0-10
2	Arenoso franco	70-86	0-30	0-15
3	Franco arenoso	50-70	0-50	0-20
4	Franco	23-52	28-50	7-27
5	Franco limoso	20-50	74-88	0-27
6	Franco arcilloso arenoso	45-80	0-28	20-35
7	Franco arcilloso	20-45	15-52	27-40
8	Franco arcilloso limoso	0-20	40-73	27-40
9	Limoso	0-20	88-100	0-12

10	Arcilloso arenoso	45-65	0-20	35-55
11	Arcilloso limoso	0-20	40-60	40-60
12	Arcilloso	0-45	0-40	40-100

Nota. Tomado de Textura del suelo, por FAO, 2006**,** en www. fao.org

Con esta variedad de suelos es necesario identificar su textura para hacer una dosificación apropiada a sus capacidades. Determinar el tipo de suelo mediante pruebas de laboratorio implica desplazamientos desde el lugar donde se encuentra ubicada la finca hasta donde haya disponibilidad de laboratorios especializados en este tipo de análisis. Como también, el pago por cada una de las muestras que se desea analizar, lo cual, se refleja en inversiones de tiempo y dinero que pueden resultar siendo costosas según las condiciones de accesibilidad del usuario. Sin embargo, existen pruebas de campo (artesanales-caseras) que permiten determinar el tipo de suelo de una forma eficiente y sencilla. Además, resulta siendo más económica y ecológica, pues evitarían muchos procesos y sustancias químicas, implícitos al hacerlo mediante un laboratorio.

Las pruebas de campo o caseras para analizar y determinar la textura consisten en los siguientes pasos:

1. A una profundidad de mínimo 20 centímetros, tomar una muestra de tierra de al menos 0,5 kilos.

2. Aplicar un proceso de tamización a las muestra o muestras recolectadas para separar la tierra fina de partículas más grandes.

3. En un recipiente transparente se introduce la muestra de suelo previamente tamizada, hasta ocupar aproximadamente un tercio de la botella. Seguido, se agrega agua limpia hasta cubrir casi la totalidad de la botella, posteriormente se tapa.

4. Se agita el recipiente por aproximadamente 1 o 2 minutos con el objetivo de mezclar todas las partículas de la muestra de suelo con el agua. Después se deja en reposo durante un tiempo mínimo de doce horas.

5. Después del reposo se observa como la sedimentación separa en capas los tipos de texturas de suelo en la muestra, en las tres primeras capas se observa la arena, el limo y la arcilla respectivamente, en algunos casos hay una cuarta capa que corresponde a compuestos orgánicos.

6. Con un instrumento de medición como regla o metro, se procede a medir la altura de cada una de las capas. Con estas, se calcular su proporción en porcentaje.

7. Se compara los porcentajes de cada capa con los valores de la tabla 4 para determinar el tipo de suelo.

De lo anterior, es viable abordar mediante Deep Learning y visión por computador, los dos últimos pasos, es decir, que el usuario mida la altura de la capa de cada mineral y desarrolle los cálculos porcentuales necesarios para determinar el tipo de suelo en la muestra. Esto se reemplaza, solicitando que el usuario capture una fotografía clara o enfocada, donde se logre ver el recipiente y la muestra de suelo; indispensable que se observe claramente separadas las capas de la muestra. La imagen se carga al sistema para determinar automáticamente el tipo de suelo. Sin embargo, las imágenes que tome el usuario deberán

cumplir con ciertas características: el recipiente debe ser transparente, de forma cilíndrica regular, de base plana; con tapa. La cantidad de muestra de suelo y agua deben ser proporcionales. La fotografía se toma en posición vertical, con fondo blanco y a una distancia máxima de 25 cm entre el recipiente y la cámara.

Para que el sistema realice el procesamiento de imágenes descrito anteriormente, es necesario entrenar un modelo de inteligencia artificial. Para ello, es necesario emplear técnicas de visión por computador; como la segmentación de imágenes, y una arquitectura de Deep Learning. Esto implica recolectar una gran cantidad de imágenes de pruebas de campo, que requieren ser etiquetadas con las capas de cada mineral que componen la muestra.

8.3.2 Requisitos funcionales

Establecer la arquitectura apropiada para el entrenamiento del modelo.

Preparación de los datos(fotografías) de entrenamiento.

Delimitar y etiquetar en las imágenes de las pruebas de campo, cada una de las capas de los minerales que se formen en la muestra.

Preprocesamiento de los datos.

Entrenamiento de los modelos de segmentación de las imágenes que identifiquen el limo, la arena y la arcilla en fotografías de pruebas de campo.

Evaluación de modelo mediante métricas de segmentación.

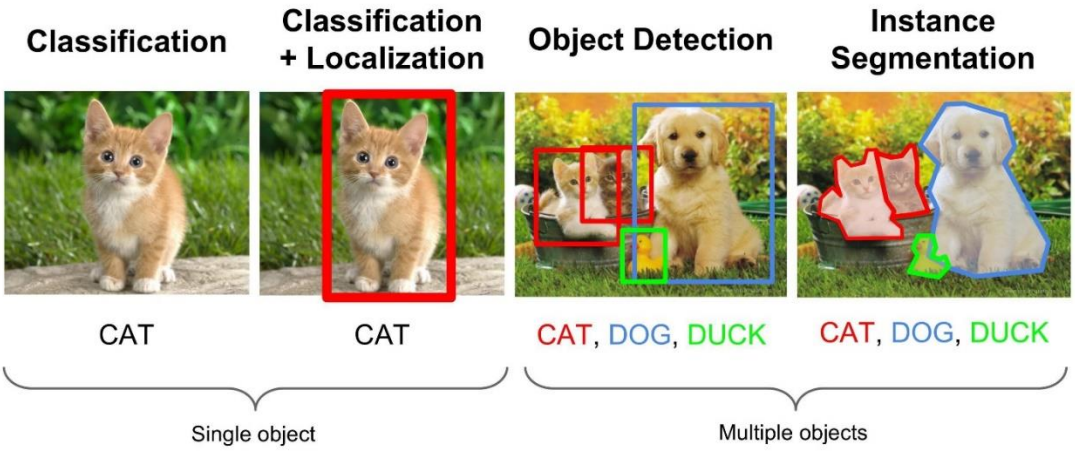
8.3.3 Arquitectura del modelo

En el área de la inteligencia artificial se encuentran varias arquitecturas para entrenar modelos mediante Deep Learning como los son RNN, CNN, RVNN. Siendo la CNN la más utilizada en las investigaciones realizadas. De igual forma, en el área de visión por computador existen diversas técnicas para el procesamiento de imágenes como son: la clasificación de imágenes, clasificación con localización, detección de objetos, segmentación semántica, segmentación por instancia, entre otras.

La segmentación de imágenes permite clasificar cada píxel de la imagen con la clase correspondiente que está representando. A diferencia de las técnicas de clasificación el resultado esperado no es solamente las etiquetas, sino que, es una imagen de alta resolución con los pixeles clasificados en las clases correspondiente.

Figura 15

Detección de objetos



Nota. Adaptado de Deep Learning for Image segmentation, por Jibin Mathew, 2018, DataDrivenInvestor (datadriveninvestor.com).

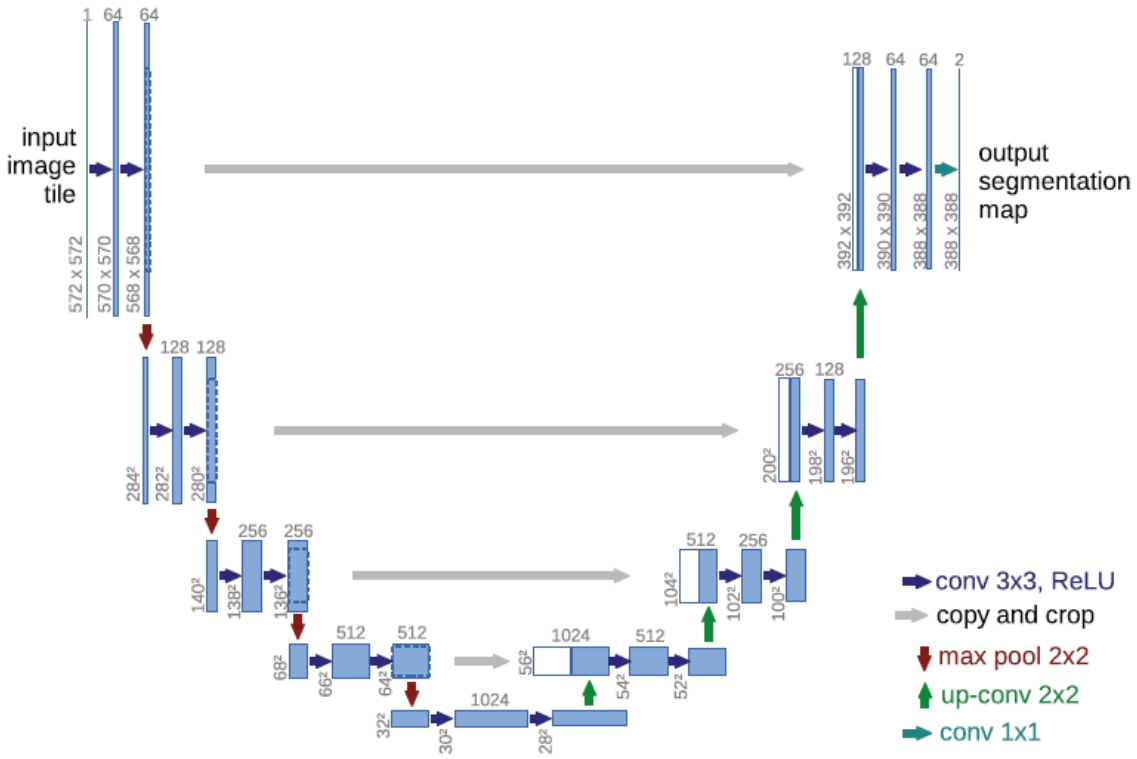
8.3.3.1 Segmentación

La arquitectura U-net que está basada en redes neuronales convolucionales, es una de la más utilizadas en el campo de la segmentación. Con las imágenes enfocadas en el área de la muestra de suelo, se aplicó la técnica de segmentación empleando esta arquitectura para identificar y delimitar las capas de los minerales visibles en la muestra, a cada capa etiquetada se le calcula su porcentaje respecto al total de la muestra, para finalmente asociar estos porcentajes con la clasificación que plantea el triángulo de texturas de suelo.

Esta arquitectura cuenta con dos caminos como lo muestra la Figura 16, donde el primer camino es la contracción o codificador (encoder) y el segunda camino es de expansión. La contracción se utiliza para captar el contexto de las imágenes donde aplica la arquitectura típica de CNN(redes neuronales convolucionales) , consiste en la aplicación de repetidas convoluciones de 3x3 . (Ronneberger y otros, 2015)

Figura 16

Arquitectura U-NET



Nota. Adaptado de U-net: Convolutional networks for biomedical image segmentation (p. 235), por O. Ronneberger, P. Fischer & T. Brox, 2015, Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015.

8.3.4 Recolección de datos

Para entrenar un modelo que permita identificar el tipo de suelo con imágenes de pruebas de campo, es necesario formar y normalizar un conjunto de datos (dataset) de fotografías obtenidas de dichas muestras. La cantidad de imágenes que se recolectan varía según el objetivo del modelo, por eso, es importante que en el conjunto de imágenes recolectadas se logre abarcar la mayoría de las características y variantes del objeto en estudio. Además, se requiere que las

imágenes a recolectar cumplan ciertas características: el recipiente donde se procesará la muestra debe ser transparente, de forma cilíndrica regular, base plana y con tapa. De igual forma, las imágenes deben ser tomadas con un fondo blanco o negro y a una exposición lumínica adecuada en la que se pueda ver claramente el recipiente y cada una de las capas de la muestra.

Para determinar el tipo de suelo combinando la metodología de pruebas de campo y técnicas de visión por computador, se realiza ensayos que permiten estandarizar las condiciones en las cuales se deben tomar las imágenes necesarias para formar el dataset. Las pruebas consisten en formar diferentes tipos de suelo mezclando los tres minerales que componen la tierra, es decir, por cada muestra de tierra formada se agrega un porcentaje de arena, limo y arcilla. También, se considera tomar muestras de suelo reales, siguiendo los pasos estipulados en el análisis.

Las muestras de suelo que se formen manualmente deben estar compuestas con porcentajes variados de arena, limo y arcilla. Cada muestra es etiquetada según los porcentajes de mineral que la compongan, se agita durante aproximadamente un minuto y se deja en reposo durante mínimo doce horas.

Es necesario disponer de un espacio adecuado para capturar las imágenes de las muestras, donde se cuente con una mesa sin desnivel, en la cual, se adapte un fondo blanco o negro para ubicar el recipiente con la muestra y el celular que tomará la fotografía.

8.3.5 Preprocesamiento de datos

La anotación de imágenes también conocida como el etiquetado de imagen, es un subconjunto o proceso que involucra a humanos para que etiqueten las fotografías con

información de metadatos y atributos, que ayudaran a las maquinas a identificar mejores objetos según el propósito y conjunto de datos. Esto requiere un software que ofrezca las funcionalidades y herramientas necesarias para anotar imágenes. Si un proyecto lo requiere, se puede obtener una herramienta personalizada, obviamente, esto demanda más dinero y tiempo. Sin embargo, existe muchas herramientas comerciales o de código abierto, con muchas técnicas de anotación de imágenes disponibles. Entre las OpenSorce más conocidas se encuentra Cvat, Labelme, etc. Existen diversas técnicas universales en el área de etiquetado: las cajas delimitadoras, puntos de referencia, polígonos, líneas, entre otras. Para la segmentación es usual emplear la anotación con polígonos, permite delimitar objetos que no son simétricos o regulares, esta implica colocar puntos en el perímetro del objeto que al unirlos con una la línea forman la figura a etiquetar (Anotación y etiquetado de imágenes para visión artificial., 2022).

Cambiar el tamaño de una imagen se puede hacer mediante un recorte o una redimensión, son procesos diferentes, pero usuales en el preprocesamiento de imágenes. La primera se pierde información ya que se corta o eliminan pixeles en la imagen, y consecuentemente el tamaño es afectado. En la segunda, se mantiene la imagen completa, pero las dimensiones de la imagen se modifican.

La técnica Data Augmentation se usa para generar más datos de entrenamiento a partir de los disponibles. Es muy útil en dataset de tipo imagen, ya que, aplica transformaciones como la rotación, trasladar, acercar, cambiar el contraste del color, etc. Sin embargo, se debe tener cuidado con las técnicas a utilizar para no generar exageraciones en los datos y ocasionar imágenes de otra clase o nunca podrían presentarse en realidad. El objetivo de conseguir esta

nueva información es aumentar la cantidad de datos de entrenamiento, para mejorar la precisión y controlar el sobreentrenamiento (overfitting) del modelo (Torres, 2019).

Con base en lo anterior, para este proyecto se considera factible emplear la plataforma cvat para etiquetar las imágenes de las pruebas de campo, rotulando las capas que se formen en la muestra con las categorías de “arena”, “arcilla” y “limo”. De igual forma, se establece que recortar parte del fondo en las imágenes, concentra el procesamiento con los pixeles de la región de mayor interés (muestra de suelo). Además, ajustar el tamaño de la imagen es básico para ahorrar el coste computacional que implica entrenar modelos con imágenes grandes.

8.3.6 Entrenamiento del Modelo

En el entrenamiento del modelo mediante Deep Learning, es necesario organizar el dataset según la distribución Train-Valid-Test Split. En esta se divide el conjunto de datos en tres subconjuntos, el primero abarca el 70% de los datos para entrenamiento (Train), el segundo, alberga el 15% de las imágenes para la validación (Valid) y el restante 15% para el proceso de pruebas (Test).

Así mismo, para entrenar un modelo con Deep Learning, es indispensable contar con ciertos componentes de hardware, como la CPU, RAM y GPU de gran capacidad de procesamiento. Hay dos grandes opciones para suplir este requerimiento. Por un lado, se pueden utilizar servicios como Amazon Web Services (AWS), Azure o GoogleCloud, que permiten crear un servidor según las necesidades. Sin embargo, estos servicios suponen un costo elevado donde se puede pagar hasta 300 euros por un servidor con GPU por 24 horas. Por otro lado, se necesita

adquirir un PC configurado para conseguir un alto rendimiento utilizando hardware de consumo. Sin embargo, esto podría requerir un costo de hasta 3000 euros.

Teniendo en cuenta las limitaciones económicas y las consideraciones anteriormente expuestas para entrenar el modelo, se opta por emplear el servicio en línea y gratuito de Google Colaboratory. Esta herramienta permite escribir y ejecutar Código en lenguaje Python sin necesidad de hacer configuraciones previas y de fácil posibilidad de compartir para un trabajo colaborativo. (colab.research.google.com, 2018). Además, dispone de un entorno de ejecución con GPU (con ciertas limitaciones) y gratuito para entrenar redes neuronales.

8.3.7 Pruebas del Modelo

Los modelos de Machine Learning requieren ser probados con un conjunto de datos diferente que plantee casos nuevos a los usados en el proceso de entrenamiento. Esta medición permite detectar las falencias o inconsistencias en los resultados de predicción (del modelo) ante nueva información. Si el resultado de esta evaluación es insuficiente, se opta por ajustar parámetros en la red neuronal (decisiones de bajo nivel) como la función de perdida, optimización o el número de épocas. Si esta calibración persiste con un resultado insatisfactorio, se considera mejorar otros aspectos que puede implicar: obtener más datos o realizar otro preprocesamiento de estos, balanceo de clases, cambiar la arquitectura, etc.

Evaluar un modelo de segmentación requiere comparar en una imagen las regiones segmentadas verdaderas (su mascara o etiqueta) con las regiones segmentadas estimadas por el modelo (Shamir y otros, 2018). Algunas de las métricas para evaluar son: Píxel Accuracy, Intersection-Over-Union (Jaccard Index) y Dice Coefficient (F1 Score) (Tiu, 2019).

Las dos últimas son las más usadas para evaluar la segmentación, en ambas la clasificación oscila entre 0 y 1, donde 1 es la mayor similitud entre la predicción y la clase verdadera.

Teniendo en cuenta lo anterior, se decide evaluar cada modelo aplicando las métricas para segmentación en el proceso de pruebas (Testing). Para ello se separa el 15% de las imágenes del dataset, las cuales, son completamente desconocidas en el proceso de entrenamiento. El otro 15% (del 30% de los datos asignado para Valid), se asigna para el proceso de validación durante el entrenamiento.

8.3.8 Despliegue del Modelo

Contar con modelos que detecten las regiones de cada mineral en la imagen de una prueba de suelo, da cabida para crear un microservicio que integre estas predicciones con cálculos matemáticos. El objetivo es determinar el valor porcentual del área identificada, para clasificar la muestra con el tipo de suelo según los porcentajes establecidos por la USDA.

Este microservicio hace parte de los módulos de la capa controlador (backend), sin embargo, existen dos opciones para el despliegue, la primera consiste en alojarlo sencillamente como una ruta más dentro del backend. Se acepta este planteamiento, si el desempeño del sistema no resulta afectado por este módulo. De lo contario, se despliega con la segunda opción. La cual, consiste en crear un proyecto exclusivo para alojar los modelos y consumir el microservicio mediante un servicio web, ya sea en el mismo u otro hosting cloud gratuito.

Sin embargo, hay que tener en cuenta que los servicios de hosting gratuitos limitan los recursos del sistema, por lo tanto, hay que evaluar el rendimiento del sistema de acuerdo con la funcionalidad.

9 Metodología

9.1 Inteligencia artificial

9.1.1 Dataset

Para entrenar los modelos que detectaron las capas de cada uno de los minerales en las muestras de suelo, se necesitó recolectar imágenes de pruebas de campo. Estas lograron abarcar diferentes características, pues, en cada muestra se varió la concentración de cada mineral y el color de la tierra. Basado en lo anterior, se formó un dataset de 600 imágenes, las cuales, se dividieron en 3 grupos, el primero estaba compuesto por el 70% de las imágenes que se asignaron para el entrenamiento (Train), el segundo con un porcentaje de 15% para validación y el restante 15% se asignó para pruebas (test). Cada grupo de imágenes era único; es decir, que no se usaban imágenes idénticas o repetidas entre los tres grupos, para cumplir con en cada fase.

9.1.2 Toma de fotografías

En el proceso de recolección de imágenes para el dataset se ejecutaron dos métodos para obtener las pruebas de campo. En primer lugar, se realizaron experimentos controlados, esto implicó conseguir en su estado más puro la arena, el limo y arcilla, recipientes de vidrio transparente con tapa, de forma cilíndrica regular, base plana y con estos se formaron los siguientes:

- En recipientes de 10cm de alto con características descritos anteriormente, se formaron 13 tipos de texturas de suelo.

- En recipientes similares a los anteriores, pero de 30 cm de alto se formaron 5 texturas de suelo con diferentes porcentajes de cada mineral como se observa en la Tabla 5.

Por otro lado, de tres municipios del Huila se recolectaron muestras de tierra en las fincas: El Porvenir, La Esperanza y La Argelia; ubicadas en la vereda La Vega de Gigante, el predio El Vergel en Garzón y la finca Dos Quebradas en Pitalito. Estas muestras se recolectaron en diferentes lotes, tomadas a una profundidad mayor de 20 cm. Cada una se analizó de manera individual, como también, se mezclaron entre sí para formar suelos de diferentes texturas y matices.

Es importante mencionar que la tierra recolectada fue secada, pulverizada y cernida con un costal de gasa o colador previo a realizar las pruebas. Además, a estas muestras se les adicionó 16 gramos de sal para acelerar el proceso de sedimentación, esto permitió que en la mayoría de los casos a partir de la tercera hora o incluso menos se pudiese distinguir el perfil del suelo.

Tabla 5

Texturas de Suelo Formadas Experimentalmente

		% mineral			Textura	Visibilidad
		arena	limo	arcilla		
Experimento 1	E1	76	14	10	Franco-Arenoso	Buena
	E2	86	14	0	Arenoso-franco	Buena
	E3	90	0	10		Mala
	E4	30	30	40	Arcilla	Regular
	E5	45	0	55		Regular
	E6	0	40	60		Mala
	E7	20	68	12	Franco-Limoso	Regular

	E8	20	80	0	Limo	Regular
	E9	0	88	12	Franco-Limoso	Regular
	E10	33	34	33	Franco-Arcilloso	Mala
	E11	50	25	25	Franco-Arcillo-arenoso	Mala
	E12	25	25	50	Arcilla	Mala
	E13	25	50	25	Franco-limoso	Mala
Experimento 2	E1B	76	14	10	Franco-Arenoso	Buena
	E2B	86	14	0	Arenoso-Franco	Buena
	E5B	45	0	55	Arcilla	Mala
	E8B	20	80	0	Limo	Regular
	E10B	33	34	33	Franco-Arcilloso	Regular

Nota. Elaboración propia

Para capturar las fotografías se tuvieron en cuenta como parámetros: la inclinación, distancia, luminosidad, y el ángulo. Estos se variaron en tres condiciones, como se describe en la Tabla 6 .

Tabla 6

Parámetros de Fotografías

Inclinación	165° – 150°	180°	195° - 210°
	I_1	I_2	I_3
Distancia	≅ 25 cm	≅ 40 cm	≅ 50 cm
	D_1	D_2	D_3
Luminosidad	Baja	Media	Alta
	L_1	L_2	L_3
Angulo	≅ 105°	≅ 90°	≅ 75°
	A_1	A_2	A_3

Nota. Elaboración propia.

Con lo anterior, se ejecutaron los parámetros propuestos para tomar las fotografías de las pruebas de campo obtenidas. Sin embargo, en un proceso de retroalimentación se concertó

que las fotos debían tener mayor concentración de pixeles por cada mineral, para lo cual, se ajustaron los siguientes parámetros:

- Fue conveniente aumentar el tamaño de los recipientes, con el objetivo de ampliar las muestras y con ello obtener capas de mayor altura en las imágenes.

- Se descartó tomar fotografías a distancias superiores a los 25 cm, pues estas implicaban mayor ruido y menos píxeles aprovechables para la región de interés (muestra).

- Se prescindió capturar fotos con inclinaciones diferentes a 180°, ya que, esto afectaba la altura de las capas en su escala real, generando una perdida considerable de píxeles.

En conclusión, se tomaron las imágenes a una inclinación de 180° con distancias inferiores a los 25 cm con luminosidad media y alta, en ángulos de 105°, 90° y 75°.

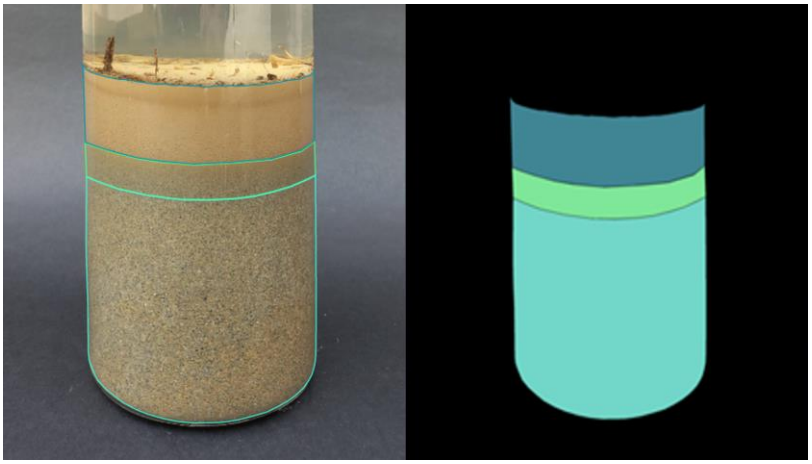
9.1.3 Etiquetado de Imágenes

El proceso de anotación de las imágenes (Image annotation en inglés) se realizó manualmente con la plataforma CVAT (Computer Vision Annotation Tool). Esta nos permitió cargar las fotos y dibujar en ellas polígonos que delimitaban cada una de las capas que se observaban en la muestra, para asociarlos a la etiqueta que le correspondía; es decir, si era arena, limo o arcilla. De igual forma, la plataforma permitió exportar el dataset con sus respectivas anotaciones en un formato denominado Segmentation Mask 1.1. Este, generaba dos tipos de mascara en formato PNG y escala RGB: Segmentation Object y Segmentation Class. Se trabajó

con este último, ya que, nos permitía identificar y asociar cada etiqueta de color con una textura, como se observa en la Figura 17.

Figura 17

Imagen y máscara en formato SegmentationMask1.1



Nota. Elaboración propia.

9.1.4 Preprocesamiento de imágenes

Como producto de los pasos anteriores, el dataset formado estuvo compuesto, por un lado, con imágenes a color de alta calidad (8000 X 6000 píxeles) y por otro, sus respectivas máscaras de iguales dimensiones en formato PNG y escala RGB. Con ayuda del lenguaje Python y librerías como PIL, Matplotlib, Numpy, cv2; se realizó un preprocesamiento. Esto consistió en disminuir las dimensiones de las imágenes y ajustarlas a ciertos parámetros establecidos por la arquitectura del modelo.

En primer lugar, el entrenamiento del modelo requería máscaras en formato Trimap y sus respectivas etiquetas asociadas con valores enteros. Para lo cual, se ejecutaron los siguientes procesos.

- Se identificó en las máscaras el color hexadecimal que representaba la arena, el limo y la arcilla.

- Se recortaron las máscaras, esto implicó identificar los bordes de las etiquetas en la imagen y a partir de estos, dejar un espacio de 30 píxeles para delimitar el área de corte.

- Con la imagen anterior, se generaron 3 nuevas imágenes, una por capa de mineral. Cada una de ellas se transformaba a escala de grises, y se cambiaba el valor decimal que identifica la etiqueta por un entero. Además, se les realizaba una reducción de tamaño a 400 por 600 píxeles antes de guardar la máscara en su respectiva carpeta.

- A partir de las máscaras anteriores se generaban unas nuevas en formato Trimap. Para ello, en el algoritmo se establecieron como parámetros: el color de fondo (unknow) que corresponde píxel de valor 1, el contorno (foreground) al valor 2 y el área del mineral (groundtruth) al valor 3. Se tuvo en cuenta que algunas imágenes tenían capas muy delgadas y establecer un contorno amplio terminaría afectando el área de interés en la máscara, por lo cual, se determinó un erode y dilate de 4 y 1 respectivamente. Finalmente, cada Trimap generado se guardó en carpetas individuales según correspondían arena, limo y la arcilla.

Por otro lado, las imágenes originales se recortaron utilizando las mismas coordenadas que se establecieron para el recorte de las máscaras. De igual forma, se realizó una

reducción en las dimensiones de las imágenes a 400 X 600 píxeles. Esto permitió conservar sus características, al mismo tiempo, evitó un costo computacional innecesario en su procesamiento.

En conclusión, el dataset obtenido en este preprocesamiento se organizó de tal manera, que se guardó en carpetas separadas las imágenes y los trimaps en formato jpg y png respectivamente.

9.1.5 Entrenamiento del Modelo

Se entrenaron tres modelos para detectar las tres capas de minerales en la muestra de suelo: uno para la arena, otro para el limo y finalmente uno para la arcilla. La red neuronal que se empleó para entrenar los modelos se basó en la arquitectura U-net utilizando Keras y tensorflow.

Algunos parámetros establecidos en la red neuronal fueron:

* Se aplicó un bloque inicial de tres capas: la primera es una convolución 2D, seguida por una normalización y la otra es una función de activación “relu”

* Se aplicaron tres filtros descendientes (downsampling) de [64, 128, 256] definidos por dos grupos de capas idénticas definidas por: función de activación relu, una convolución separable y una normalización. Seguido se aplicó un maxpooling con strides y padding 2.

* Se establecieron 4 filtros ascendentes (upsampling) de [256, 128, 64, 32] con dos grupos de capas iguales definidas así: función de activación relu, convolución transpuesta y normalización. Seguido se aplicó una capa de upsampling.

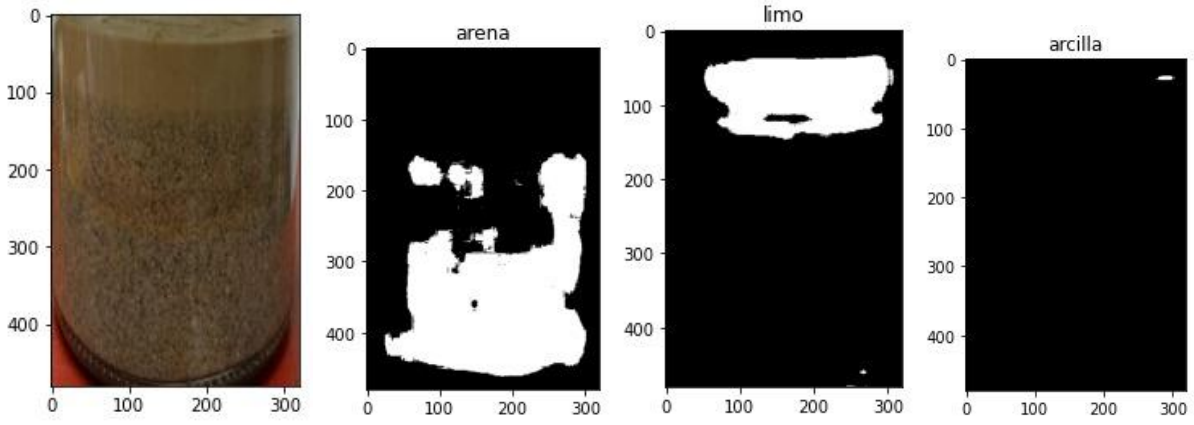
* Para el entrenamiento de cada modelo se empleó el dataset obtenido del preprocesamiento. Algunos parámetros establecidos en este proceso fueron: Tamaño de la imagen: de 480 X 320, número de clases 3, tamaño de bloque (Batch size) es 2. 15 épocas, como Optimizador el “rmsprop” y como función de perdida: “sparse-categorical-crossentropy” y métricas dice coefficient, jaccard o mean IOU.

9.2 Inferencia

Cada modelo procesa la imagen y retorna una máscara donde muestra la región que detectó del mineral para el que fue entrenado.

Figura 18

Imagen de prueba evaluada por los modelos



Nota. Elaboración propia.

Para enfatizar en una sección de la máscara y continuar el procesamiento solo con el área detectada por el modelo, se hallaron dos coordenadas $((X_1,Y_1)(X_2,Y_2))$ que delimitaban el área de interés. Para ello, se tuvo en cuenta que en las máscaras la información es binaria,

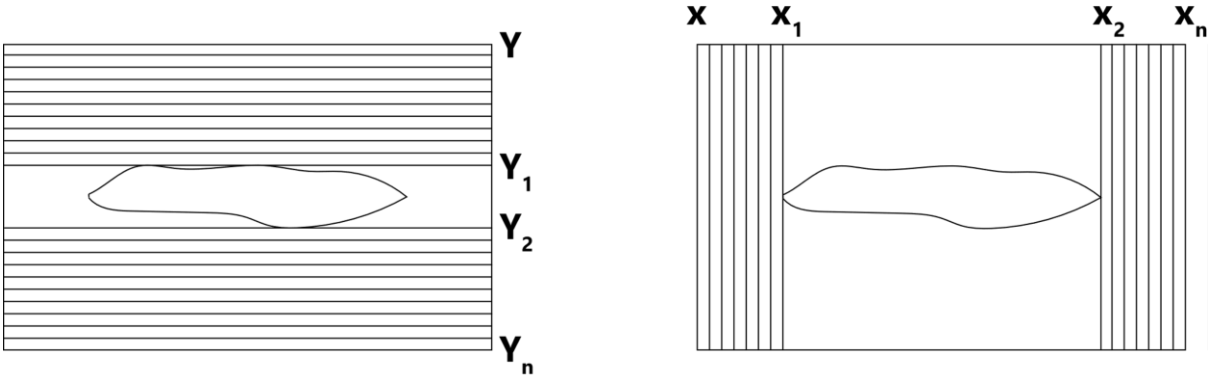
donde el background de color negro el píxel vale 0 y el ground truth de color blanco vale 1. Por lo cual, se realizó lo siguiente:

Se recorrió la máscara desde el borde superior izquierdo hasta el límite de esta en sentido del eje X, a medida que va pasando cada píxel de la fila lo suma y posterior valida, si el acumulador que registra la suma es mayor de cero, termina el ciclo, pues esto indica que se halló el límite superior del área detectada y asigna el píxel como Y_1. Para el Y_2 se hizo el mismo proceso anteriormente descrito, pero la máscara se inicia a leer desde la esquina inferior derecha cuando se halla el píxel mayor que 0 se asigna como Y_2.

De igual forma, se recorrió la máscara desde el borde superior izquierdo hasta el límite de la misma en sentido del eje Y, a medida que va pasando cada píxel de la columna lo suma y posteriormente valida, si el acumulador que registra la suma es mayor de cero, termina el ciclo, pues esto indica que se halló el límite derecho del área detectada y asigna el píxel como X_1. Para el X_2 se hizo el mismo proceso anteriormente descrito, pero la máscara se inicia a leer desde la esquina inferior derecha cuando se halla el píxel mayor que 0 se asigna como X_2. En la Figura 19 se ilustra la detección de los bordes en las imágenes de predicción y en la Figura 20 se representa el algoritmo del proceso anteriormente descrito.

Figura 19

Delimitación del área detectada por el modelo



Nota. Elaboración propia.

Figura 20

Seudocódigo para Detectar Bordes

Algorithm 1 Obtener Bordes

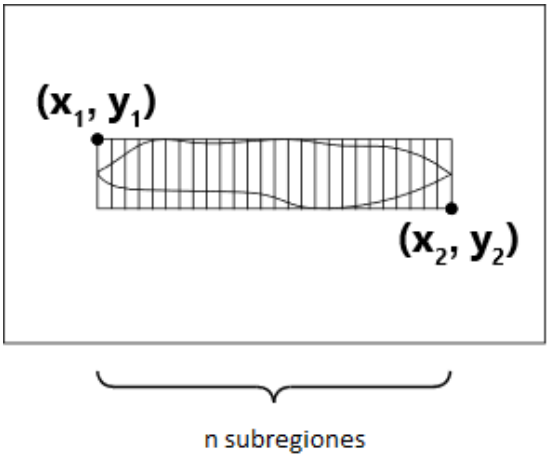
```
Input: (imagen como Numpy array)
Output: (x1,x2,y1,y2)
  x1 ← 0
  y1 ← 0
  x2 ← maximo pixel de la fila
  y2 ← maximo pixel de columna
  for y ← 0 maximo pixel de columna do
     linea ←  to lista de pixeles de la columna
     suma ← sumatoria de linea
     if suma > 0 then
        y1 ← y
        break
     end if
  end for
  for maximo pixel de columna to y ← 0 do
     linea ← lista de pixeles de la columna
     suma ← sumatoria de linea
     if suma > 0 then
        y2 ← y
        break
     end if
  end for
  for x ← 0 to maximo pixel de la fila do
     linea ← lista de pixeles de la fila
     suma ← sumatoria de linea
     if suma > 0 then
        x1 ← x
        break
     end if
  end for
  for maximo pixel de la fila to x ← 0 do
     linea ← lista de pixeles de la fila
     suma ← sumatoria de linea
     if suma > 0 then
        x2 ← x
        break
     end if
  end for
  return x1,x2,y1,y2
```

Nota. Elaboración propia.

Se observó que el área que los modelos detectaban no eran figuras rectangulares uniformes, por lo cual, se decidió hallar de la zona detectada la altura en pixeles más representativa. Para esto, la región de interés delimitada se dividió en 21 partes en sentido vertical como se observa en la Figura 21, cada columna se recorrió para contar y almacenar en una lista el total de pixeles blancos (ground truth) que contenía cada una. Dicha lista se organizó de mayor a menor y se le calculó el promedio, para asociarlo como la altura de la capa. Finalmente, se identificó en la misma los valores máximos y mínimos para depurarlos, ya que estos correspondían a pixeles dispersos que se detectaban en zonas que no correspondían a la textura o que sí, pero no lograban abarcar la altura real de la capa.

Figura 21

División del área detectada en subregiones



Nota. Elaboración propia.

Posteriormente, determinada la altura de cada capa detectada, se calculó la altura total de la muestra (H_t) mediante la Ecuación 2 y el porcentaje de cada capa de mineral

detectada aplicando la Ecuación 3. En la Figura 22 se describe como se manejó las alturas detectadas por los modelos para determinar la altura total.

Ecuación 2

Altura total de la muestra detectada por los modelo

$$H_t = H_1 + H_2 + H_3$$

Donde H_t es la altura total. H_1, H_2, H_3 son las alturas de la arena, limo y arcilla respectivamente.

Ecuación 3

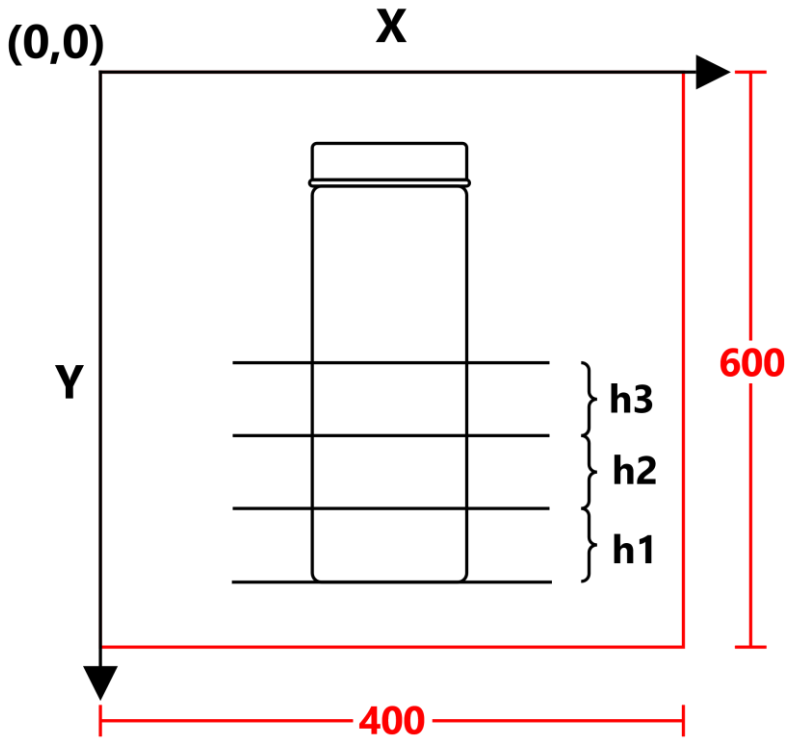
Porcentaje del mineral

$$\%X = \frac{H_x}{H_{total}} x\ 100$$

Donde X es la capa de mineral (arena, limo, arcilla), y H_x es la altura de la capa de mineral que corresponda.

Figura 22

Altura detectadas por los modelos



Nota. Elaboración propia.

9.2.1 Métricas de evaluación

Cada uno de los tres modelos entrenados para detectar los minerales de la muestra en las imágenes, fueron evaluados con la función de pérdida (Loss Function) durante el proceso de entrenamiento y validación. Además, en la fase de Testing se midió el rendimiento de cada modelo, para ello, se emplearon las métricas de evaluación Mean IOU (Jaccard Score) y Dice Coefficient (F1 Score).

Dado que ambas métricas están correlacionadas, se emplearon los mismos parámetros para ejecutar las funciones de cada una. Para ello, se acudió al grupo de imágenes y sus respectivas mascaras que se habían separado para el proceso de Testing (15% del dataset). Las fotografías se cargaron al modelo, este generó su respectiva predicción o segmentación en una nueva imagen. De esta manera, se pasó como segmentación verdadera las mascara de las imágenes y como predicción la imagen generada por el modelo.

Mean IOU (Jaccare Score). Esta métrica se implementó de dos maneras, por un lado, se utilizó la función para esta métrica de la librería sklearn, a la cual, se le definieron como parámetros: average= ”micro”, las imágenes de predicción y las verdaderas

Por otro lado, manualmente se transcribió y ejecutó la Ecuación 4 en una función de lenguaje Python, para aplicar esta métrica, como se observa en la Figura 23.

Ecuación 4

Intersection Over Union

$$IOU = \frac{|A \cap B|}{|A \cup B|}$$

Figura 23

Función IOU en Python

```python
from keras import backend as K

## IOU in pure numpy
def numpy_iou(y_true, y_pred, n_class=2):
    def iou(y_true, y_pred, n_class):
        # IOU = TP/(TP+FN+FP)
        IOU = []
        for c in range(n_class):
            TP = np.sum((y_true == c) & (y_pred == c))
            FP = np.sum((y_true != c) & (y_pred == c))
            FN = np.sum((y_true == c) & (y_pred != c))

            n = TP
            d = float(TP + FP + FN + 1e-12)

            iou = np.divide(n, d)
            IOU.append(iou)

        return np.mean(IOU)

    batch = y_true.shape[0]
    y_true = np.reshape(y_true, (batch, -1))
    y_pred = np.reshape(y_pred, (batch, -1))

    score = []
    for idx in range(batch):
        iou_value = iou(y_true[idx], y_pred[idx], n_class)
        score.append(iou_value)
    return np.mean(score)
```

Nota. Elaboración propia.

Dice Coefficient (F1 Score). Esta métrica se implementó después del entrenamiento en cada uno de los tres modelos. Se utilizó el lenguaje Python para transcribir y ejecutar la ecuación 5, esto se puede ver en la Figura 24.

Ecuación 5

Dice Coefficient (F1 Score).

$$F1 = \frac{2\,|A \cap B|}{|A| + |B|}$$

Donde A es la verdadera y B es la predicción que se obtiene del modelo. (Shamir y otros, 2018)

Figura 24

Función Dice Coefficient en Python

```python
#metrica de dice_coef

smooth=1
dice_list = []

for i in range(len(test_preds)):
    y_true = get_image_seg(i)
    y_pred = get_image_pred(i)

    intersection = np.logical_and(y_true, y_pred)
    dice_coef = (2. * intersection.sum())  / (y_true.sum() + y_pred.sum())
    dice_list.append(dice_coef)


mdice_mean = np.mean(dice_list)
mdice_stdv = np.std(dice_list)
mdice_n = len(dice_list)
print(mdice_mean,mdice_stdv,mdice_n)
```

Nota. Elaboración propia.

9.3 Riego

Se realizó un montaje en un entorno controlado para simular sectores de riego. Esto permitió verificar el correcto funcionamiento de los componentes del sistema ante diferentes contextos o situaciones. Las directrices empleadas para definir la estrategia, programación, lámina y tiempo de riego se basaron en el libro Manual de riego para agricultores modulo 1 y 4.

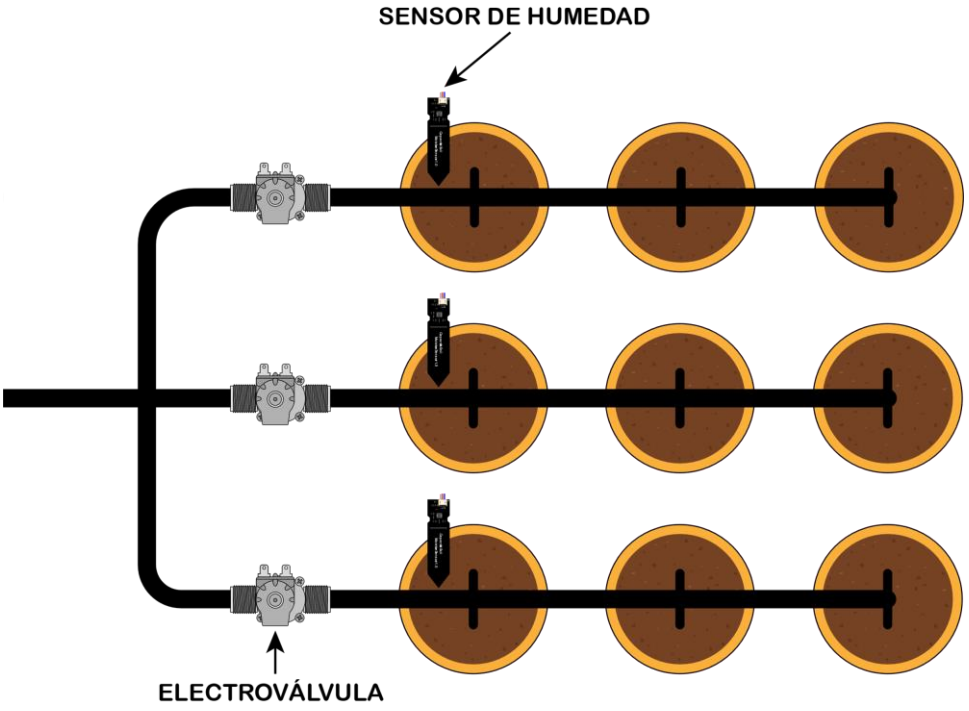
9.3.1 Sectores de riego

En el montaje realizado se manejaron cuatro sectores de riego. Para definirlos se tuvo en cuenta variaciones en el tipo de suelo, el cultivo o el emisor de riego en un lote o finca.

Cada uno contó con componentes independientes como sensor de humedad y electroválvulas. (Ver Figura 25)

Figura 25

Sectores de Riego



Nota. Elaboración propia.

El primero estuvo determinado por el tipo de cultivo y su emisor de riego. En este caso, se utilizó una manguera de media pulgada y cuatro microaspersores en árboles leñosos.

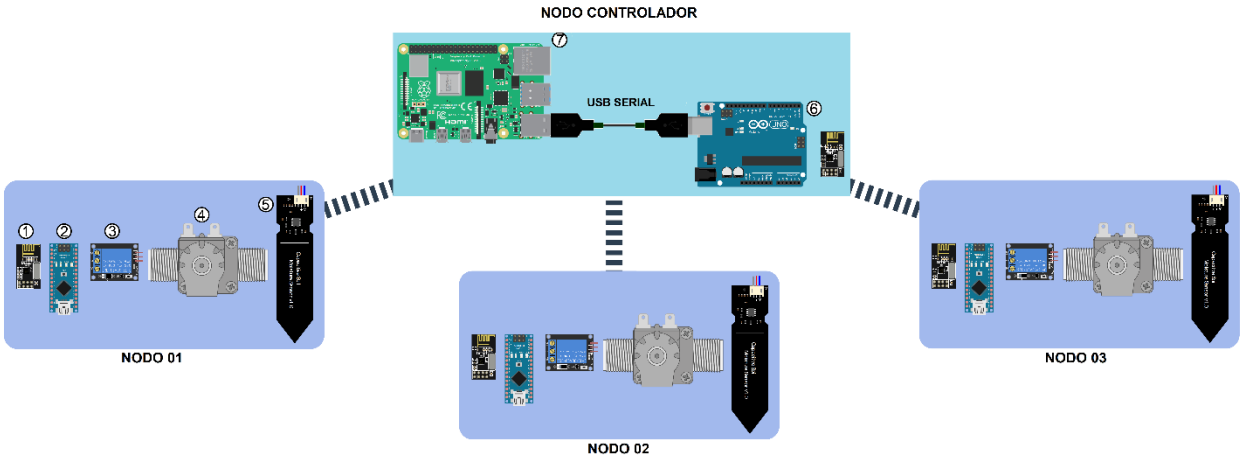
El segundo y tercer sector de riego tenían el mismo tipo de suelo y emisores de goteo autocompensado, pero se diferenciaban por el tipo de cultivo. Uno de ellos contó con plantas de frijol; con 7 días de germinación, y el otro con plantas de mango de aproximadamente cuatro meses de edad.

El cuarto sector de riego tenía plantas de limón de cerca (limón swinglea) y tuvo como emisor de riego una cinta de 6 metros perforada cada 30 cm. Además, este tenía el mismo tipo de suelo que el primer sector.

Los módulos de radiofrecuencia permitieron establecer una comunicación bidireccional entre el módulo controlador y los componentes de los sectores de riego. Se estableció un nodo principal, encargado de recibir y enviar lecturas de los sensores de humedad al controlador. De igual forma, recibía desde la raspberry pi las instrucciones para las electroválvulas de todos los sectores.

Figura 26

Nodos Radiofrecuencia en Sectores de Riego



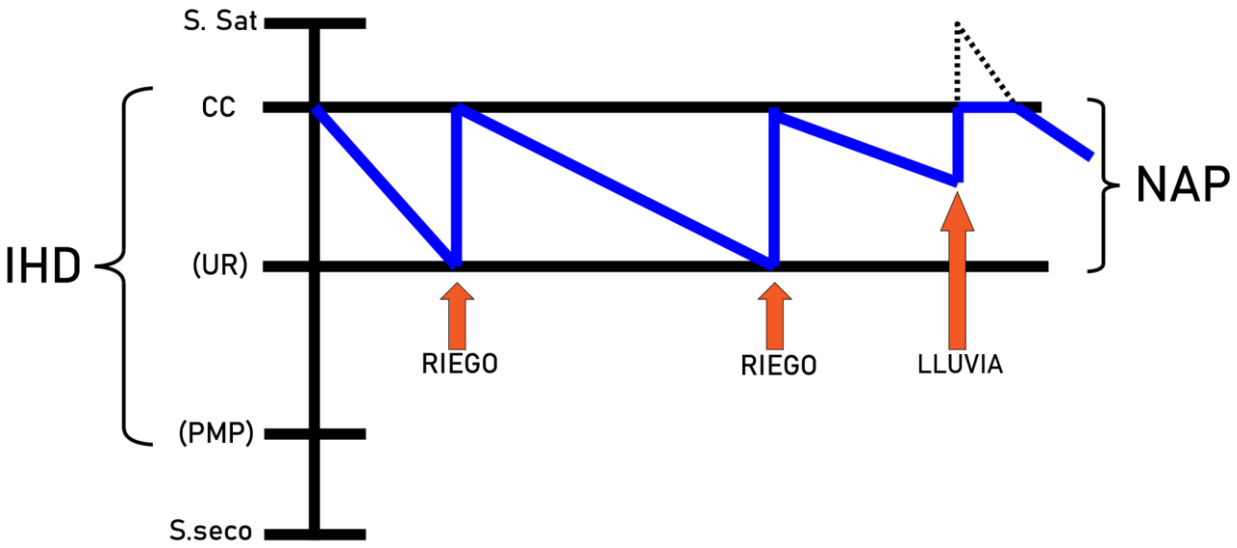
Nota. 1. Modulo radiofrecuencia nrf24L01, 2. Arduino Nano, 3. Modulo Relé, 4. Electroválvula, 5. Sensor de humedad, 6. Arduino UNO rev, 7. Raspberry Pi 4 B.

9.3.2 Estrategias y programación del riego

La estrategia de riego implementada determinó aplicar la lámina de agua neta cuando el Déficit de agua en el suelo (DAS) fuera igual al Nivel de Agotamiento permisible (NAP). Es una de las estrategias más recomendadas, ya que con una adecuada implementación su balance de agua evita que la planta tenga problemas de extracción de agua.

Figura 27

Estrategia de Riego



Nota. Adaptado de Modulo 1: Cálculo de la lámina de riego primer parte [Vídeo], por canal MGAP, 2015, (Canal MGAP, 2015).

Se estableció el sensor de humedad como el componente determinante para efectuar el riego. Este dispositivo permitió determinar en tiempo real si el porcentaje de humedad en el suelo se encontraba entre el rango del NAP establecido para cada sector de riego.

A la fecha de realización del proyecto, las Apis de calendario de riego y estación meteorológica no estaban disponibles para implementarlas en este sistema como se había

planteado en la propuesta, por lo cual, el método anteriormente descrito fue la opción más viable a implementar para programar y aplicar el riego.

9.3.3 Lámina de riego

La lamina de riego permitió expresar en milímetros (mm) la cantidad de agua a regar en un sector. Para determinar su volumen, se tuvo en cuenta la profundidad de la raíz de la planta, el IHD y el NAP del cultivo.

Ecuación 6

Lámina de riego neta

$$LR_n = IHD \times Pr \times NAP$$

Donde: LR_n es la lamina de riego neta, IHD es el Índice de Humedad Disponible, Pr es la profundidad de la raíz y el NAP es el Nivel de agotamiento permisible.

La siguiente tabla encontramos las profundidades de la raíz de diferentes cultivos

Tabla 7

Profundidad de Raíz en Cultivos

Cultivo	Profundidad (metros)
Aguacate	0.8 – 1.2
Albaricoque	0.6 – 1.4
Alcachofa	0.6 – 0.9
Alfalfa	1.2 – 1.8
Algodón	0.6 – 1.8
Almendro	0.6 – 1.2
Avena	0.6 – 1.1

Berenjena	0.5 – 0.6
Cebada	0.9 – 1.1
Cebolla	0.3 – 0.6
Cerezo	0.8 – 1.2
Ciruelo	0.8 – 1.2
Cítricos	0.9 – 1.5
Col y coliflor	0.6
Espárrago	1.2 – 1.8
Espinaca	0.4 – 0.6
Fresa	0.3 – 0.5
Girasol	1.5 – 2.5
Guisantes	0.4 – 0.8
Lechuga	0.2 – 0.5
Leguminosas grano	0.5 – 1.0
Maíz grano	0.6 – 1.2
Manzano	0.8 – 1.4

Nota. Adaptado de Manual de riego para agricultores modulo 1 (p. 62), por Fernández G. Rafael, 2010, Sevilla, Consejería de Agricultura y pesca. (Fernández, 2010)

Para determinar el IHD se implementó la Ecuación 7, se tomaron como capacidad de campo y punto de marchitez permanente los valores medio de los rangos teóricos registrados en la Tabla 8 para cada tipo de suelo.

Ecuación 7

Índice de Humedad Disponible

$$IHD = CC - PMP$$

Donde, IHD es el Índice de Humedad Disponible, CC es la Capacidad de Campo teórica, PMP es el Punto de Marchitez Permanente teórica.

Tabla 8

IHD en Algunas Texturas de Suelo

Tipo de suelo	IHD (mm por m de profundidad del suelo)
Arenoso	70 - 100
Franco	140 - 190
Arcilloso	200 - 250
Franco-Arcilloso	170 -220
Franco-Arenoso	90 - 150

Nota. Tomad de Manual de riego para agricultores modulo 1 (p. 62), por Fernández G. Rafael, 2010, Sevilla, Consejería de Agricultura y pesca.

Sin embargo, la cantidad de agua que determinaba la lámina de agua neta no contemplaba un remanente de agua para suplir la pérdida del líquido. se tuvo en cuenta el porcentaje de eficiencia del método de riego implementado. Para ello se implementó la Ecuación 8.

Ecuación 8

Lámina de Riego Bruta

$$LR_b = \frac{LR_n}{Ea} \times 100$$

Donde, LR_b es lámina de riego bruta, LR_n es lámina de riego neta y Ea es la eficiencia de aplicación del método de riego.

La tabla 9 recopila la eficiencia de riego en los métodos de riego. Basados en estos porcentajes, se asignó como Ea el porcentaje de eficiencia para el riego localizado.

Tabla 9

Métodos de Riego y su Porcentaje de Eficiencia de Aplicación

Método de riego	**Eficiencia de aplicación (%)**
Riego por superficie	55%
Riego por aspersión	65%
Riego localizado	75%

Nota. Tomad de Manual de riego para agricultores modulo 1 (p. 62), por Fernández G. Rafael, 2010, Sevilla, Consejería de Agricultura y pesca.

En los cultivos a medida que se van desarrollando, se pueden asociar a una etapa. En las plantaciones anuales se establecen cuatro etapas de desarrollo después de la siembra, como se observa en la Figura 28. Esta característica se tuvo en cuenta para adicionar un porcentaje adicional a la lámina de riego bruta. Descrito en la tabla 10.

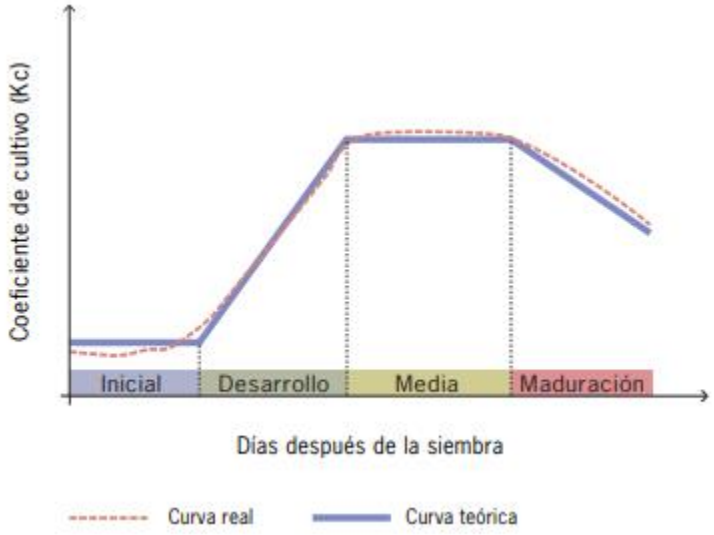
Tabla 10

Volumen Adicional en Lámina de Bruta por Etapa

Fase	Inicial	Desarrollo	Media	Maduración
Porcentaje adicional en LR_b(%)	0	23	53	23

Figura 28

Fases de Desarrollo en Cultivos Anuales



Nota. Adaptado de Manual de riego para agricultores modulo 1 (p. 59), por Fernández G. Rafael, 2010, Sevilla, Consejería de Agricultura y pesca.

Para aplicar la lámina de riego bruta se tuvo en cuenta tres parámetros: el caudal, la cantidad y capacidad de distribución de los emisores en el sector de riego. Por lo cual, se midió en la cabecera del sistema y los emisores, la cantidad agua distribuida en un determinado tiempo. Con lo anterior, se determinó mediante el método volumétrico el caudal disponible para el sector y el caudal unitario del emisor (goteo, microaspersión y cinta) con la ecuación 9.

Ecuación 9

Caudal de Riego

$$Q = \frac{V}{t}$$

Donde, $\boldsymbol{Q}$ es el caudal. $\boldsymbol{V}$ es el volumen y $\boldsymbol{t}$ es el tiempo.

Se tuvo en cuenta el tipo de suelo asociado a cada sector, para controlar que la capacidad de distribución de los emisores no fuese superior a la capacidad de infiltración del suelo. Si esto ocurría, se podrían generar encharcamientos, por lo cual, se planeó que el sistema aplicara la lámina de riego en periodos cortos.

9.3.4 Cálculo de tiempo de riego

Se aplicó las ecuaciones planteadas en el Manual de riego para agricultores: módulo 4 (Fernández, 2010). Para calcular el tiempo de riego necesario para aplicar la lámina de riego bruta se requiere conocer: Lamina de riego bruta, distancia entre los emisores de riego en una tubería lateral, distancia entre los laterales de riego y el caudal de los emisores.

Ecuación 10

Numero de Emisores por Metro Cuadrado

$$Nº\ de\ emisores\ por\ metro\ cuadrado\ = \frac{1}{\mathrm{D_E} \times \mathrm{D_L}}$$

Donde, D_E es la distancia entre emisores de riego y D_L es la distancia entre laterales. Ambos en metros cuadrados (m^2)

Ecuación 11

Cálculo de Tiempo de Riego

$$Tiempo\ de\ reigo\ = \frac{LR_b}{Q_e} x \frac{1}{Nº\ de\ emisores\ por\ metro\ cuadrado} x\ 60$$

Donde, LR_b es la lámina de riego bruta (L/m^2), Q_e es el caudal del emisor (L/h)

Así mismo el tipo suelo es esencial para determinar la cantidad, ya que cada tipo de suelo cuanta con el intervalo de humedad disponible.

La lógica implementada para la activación del riego se ilustra en la Figura 29. Además, los procesos que se usaron para determinar la lámina de riego siguieron la lógica del algoritmo de la Figura 30. Adicionalmente en la Figura 31 se muestra el algoritmo para calcular el tiempo de riego por sector.

Figura 29

Algoritmo del Riego del Sector

Algorithm 2 Activación del riego del sector

Input: NAP, humedad del sensor
Output: Activación del riego

```
riego ← False
if humedaddelsensor ≥ NAP then
    riego ← True
end if
return riego
```

Nota. Elaboración propia.

Figura 30

Algoritmo de Lamina de Riego por Sector

Algorithm 3 Lamina de riego por sector

Input: IHD, profundidad de la raíz, NAD, eficiencia riego, etapa
Output: Lamina de agua

$laminaderiegoneta \leftarrow IHD \cdot ProfundidadRaiz \cdot NAP$
$laminaderiegobruta \leftarrow \frac{laminaderiegoneta}{eficiencia} \cdot 100$
if $etapa == inicial$ **then**
  $laminaderiegobruta$
end if
if $etapa == desarrollo$ **then**
  $laminaderiegobruta \leftarrow laminaderiegobruta \cdot 0{,}23 + laminaderiegobruta$
end if
if $etapa == media$ **then**
  $laminaderiegobruta \leftarrow laminaderiegobruta \cdot 0{,}53 + laminaderiegobruta$
end if
if $etapa == maduración$ **then**
  $laminaderiegobruta \leftarrow laminaderiegobruta \cdot 0{,}23 + laminaderiegobruta$
end if
return lamina de riego bruta

Figura 31

Algoritmo Cálculo de Tiempo de Aplicación del Riego por Sector

Algorithm 4 Cálculo de tiempo de aplicación del riego por sector

Input: caudal del emisor, lamina de riego,distancias emisores, distancia laterales
Output: Tiempo de riego en el sector

$tiempo \leftarrow 0$
$numeroemisores \leftarrow 0$
$numerodeemisores \leftarrow \frac{1}{distancialaterales \cdot distanciaemisores}$
$tiempo \leftarrow \frac{laminaderiego}{caudaldelemisor} \cdot \frac{1}{numerodeemisor} \cdot 60$
return tiempo

9.4 Ingeniería de software

Para el proyecto se desarrolló una plataforma web que nos permitió integrar las soluciones planteadas en las líneas de IoT e inteligencia artificial. Se implementó la metodología SCRUM para desglosar los requerimientos del en actividades más cortas y sencillas de abordar.

9.4.1 Product backlog

Tabla 11

Product Backlog del Proyecto

Identificador	Historia de usuario	Descripción	Criterios de aceptación	Responsable	Prioridad	Complejidad
HU001	Toma de fotos	Se captura las fotos de las muestras suelos para la formación del dataset	La distancia de la cámara con respecto al objecto sea 25, 40 y 50	Blanca, Pedro, Luis	Alta	Media
			La inclinación de la cámara sea de 165 a 150, 180, 195 a 210			
			El Angulo de la cámara sea 105, 90, 75			
			La luminosidad de ambiente sea baja, media, alta			
			Evitar el ruido del fondo, agregando un fondo de color blanco			
HU002	Preparación de la Muestras de campo	Se realiza la preparación de muestras controladas para la captura de las fotos	El frasco debe ser de forma cilíndrica, con base plana y tapa	Blanca, Pedro, Luis	Alta	Media
			Se debe formar diferentes muestras variando la cantidad de los minerales			
			La muestra debe ocupar el 50% del recipiente y el resto en agua			

			la muestra se debe agitar durante 1 minuto La muestra debe estar en reposo por lo mínimo de 24 horas			
HU003	Etiquetar Imágenes del dataset	Se realiza la etiquetación de las imágenes mediante el programa cvat para delimitar cada mineral	Etiquetar de forma individual los diferentes minerales de la nuestra	Blanca, Pedro, Luis	Alta	Baja
HU004	Preprocesamiento de imágenes	La arquitectura requiere que las fotografía y anotaciones se ajusten a los parámetros u-net para el correcto entrenamiento del modelo	Recortar y reducir la dimensión de la imagen Establecer la orientación de las imágenes de forma vertical Agrupar las anotaciones por tipo mineral Generar Imágenes binarias Generar las imágenes Trimap	Blanca, Pedro, Luis	Alta	Alta
HU005	Entrenamiento del modelo	yo como usuario requiero un modelo de Deep Learning que identifique en imágenes de pruebas de campo las capas de cada mineral del suelo	Entrenar un modelo por cada tipo de mineral Aplicar la arquitectura U-net Aplicar métricas de entrenamiento accuracy y mean iou	Blanca, Pedro, Luis	Alta	Alta
HU006	Comunicación entre nodos	El sistema requiere la comunicación inalámbrica para gestionar los nodos	Los módulos se comunican bidireccional mediante radio frecuencia Los nodos hijo enviar datos al principal cada 1 h	Blanca, Pedro, Luis	Alta	Alta
HU007	Sectorizar el sistema de riego	El usuario requiere organizar el sistema de riego por sectores para aplicar un riego diferenciado	Aplicación de diferentes tipos de riegos en cada sector Usar diferentes tipos de cultivo por cada sector	Blanca, Pedro, Luis	Medio	Medio

			Cada sector de riego debe contar con el nodo de comunicación y con los sensores de humedad, electroválvula			
HU008	Módulo de Registro de usuario	Yo como usuario requiero registrar credenciales en la plataforma para poder gestionar el sistema de riego	La información personal del usuario se debe guardar en base de datos: nombre, apellido, numero documento, username, password, celular, email, dirección. La contraseña se guarda de forma encriptada Validar que no exista un usuario con el mismo número de documento y username	Blanca, Pedro, Luis	Medio	Medio
HU009	Módulo de autenticación	Yo como usuario requiero acceder a la plataforma con las credenciales registrada para administrar y gestionar la información del sistema de riego	El sistema de autenticación debe ser por JWT(Json Web Token) la autenticación debe ser valida con usuarios registrados Las credenciales de acceso deben coincidir con las registradas en base de datos	Blanca, Pedro, Luis	Alta	Media
HU010	Módulo de Registro de Finca	Yo como usuario requiero registrar fincas para administrar los cultivos y lotes	La información de la finca se debe guardar en base de datos: nombre, dirección, latitud, longitud, altitud, usuario. El usuario puede registrar más de una finca Las coordenadas son de grados decimales	Blanca, Pedro, Luis	Media	Media

HU011	Módulo de Registro de Lotes	Yo como usuario requiero registrar los lotes que pertenecen a una finca para administrar sectores	La información del lote se debe guardar en base de datos: nombre, finca, área, latitud, longitud, altitud. Las coordenadas son de grados decimales El usuario no puede registrar un lote sin relacionarlo a una finca El usuario puede registrar más de un lote	Blanca, Pedro, Luis	Media	Media
HU012	Módulo de Registro de Sectores	Yo como usuario requiero registrar los sectores que pertenecen a un lote para administrar los diferentes riegos	La información del sector se debe guardar en base de datos: nombre, lote, área, latitud, longitud, altitud, tipo suelo, cultivo. Las coordenadas son de grados decimales El usuario no puede registrar un lote sin relacionarlo a un lote El usuario puede registrar más de un sector	Blanca, Pedro, Luis	Media	Media
HU013	Modulo registro de cultivo	Yo como usuario requiero registrar los cultivos	La información del cultivo debe guardar en base de datos: nombre, tipo de cultivo, fecha de inicio, fecha final, estado y usuario. El usuario puede registrar más de un cultivo	Blanca, Pedro, Luis	Media	Media
HU014	Módulo de fincas	yo como usuario requiero visualizar las fincas para gestionar y administrar los lotes	El usuario puede visualizar las fincas registradas El usuario puede ver el detalle de las fincas	Blanca, Pedro, Luis	Media	Media
HU015	Módulo de Lotes	yo como usuario requiero visualizar los lotes asociados a	El usuario puede visualizar los lotes registrados	Blanca, Pedro, Luis	Media	Media

		una finca para gestionar y administrar los sectores de riego	El usuario puede ver el detalle del lote			
HU016	Módulo de sectores	yo como usuario requiero visualizar los sectores de riego para gestionar y administrar el riego en los cultivos	El usuario puede visualizar los sectores registrados El usuario puede ver el detalle del sector El usuario puede interactuar con los componentes de riego e IoT mediante la plataforma	Blanca, Pedro, Luis	Media	Media
HU017	Módulo de cultivos	yo como usuario requiero visualizar los cultivos para identificar el seguimiento del riego en las diferentes etapas	El usuario puede visualizar los cultivos registrados El usuario puede visualizar el estado del cultivo	Blanca, Pedro, Luis	Media	Media
HU018	Módulo de predicción	yo como usuario requiero determinar el tipo de suelo mediante imágenes de pruebas de campos para asociarla a un sector de riego	El usuario puede subir una imagen de la muestra de suelo en formato png, jpg, jpeg. El sistema debe retornar el tipo de suelo y los porcentajes de minerales El usuario debe contar con una herramienta que le permita cuando sea necesario ajustar las alturas de las capas de la imagen generada por el sistema	Blanca, Pedro, Luis	Media	Media

Nota. Elaboración propia.

10 Resultados

10.1 Ingeniería de software

El proyecto se realizó mediante el marco de trabajo ágil (scrum) el cual nos permite desarrollar las actividades mediante sprint, lo cual nos permite abordar cambios en el transcurso del proyecto. Durante el proyecto se realizaron 12 sprint tomando las historias de usuario del product backlog establecido por el equipo.

10.1.1 Sprints

Sprint 1

* Planificación del sprint:

Tabla 12

Historias de Usuario del Sprint 1

HU001	Toma de fotos	Se captura las fotos de las muestras suelos para la formación del dataset	La distancia de la cámara con respecto al objecto sea 25, 40 y 50	Blanca, Pedro, Luis	Alta	Media
			La inclinación de la cámara sea de 165 a 150, 180, 195 a 210			
			El ángulo de la cámara sea 105, 90, 75			
			La luminosidad de ambiente sea baja, media, alta			
			Evitar el ruido del fondo, agregando un fondo de color blanco			
HU002	Preparación de la Muestras de campo	Se realiza la preparación de muestras controladas para la captura de las fotos	El frasco debe ser de forma cilíndrica, con base plana y tapa	Blanca, Pedro, Luis	Alta	Media
			Se debe formar diferentes muestras variando la cantidad de los minerales			

			La muestra debe ocupar el 50% del recipiente y el resto en agua			
			la muestra se debe agitar durante 1 minuto			
			La muestra debe estar en reposo por lo mínimo de 24 horas			

En la planificación de sprint de definió un lapso de 4 semanas para la realización de las actividades de las historias de usuario.

* Revisión del sprint:

Terminado las actividades del sprint, se continuo con la validación de los criterios de aceptación los cuales fueron aplicado en desarrollo de las tareas. Dando como resultado la aprobación de los criterios de aceptación.

Pero los resultados esperados por el equipo no satisfacen los requerimientos por tal motivo, las historias de usuario fueron agregadas de nuevo al product backlog con nuevos criterios de aceptación que correspondieron a las mejoras para el desarrollo de la historia de usuario.

* Impedimentos:

Las distancias mayores a 25 cm generaban imágenes con mucho ruido de fondo y poca área de aprovechamiento para el entrenamiento.

Algunos ángulos inclinación generaban mayor ruido en las imágenes.

La baja luminosidad generaba, que las fotografía no captaban los componentes de las muestras.

* Retrospectiva del sprint:

En el desarrollo del sprint la comunicación entre los integrantes del proyecto fue participativa. Esto permitió el desarrollo exitoso de las actividades. Además de detectar los impedimentos que resultaron el desarrollo del sprint.

En la etapa de planificación tuvimos deficiencia en el análisis de problema a solucionar. Por tal motivo se generó los impedimentos en la toma y preparación de las muestras.

Sprint 2

* Planificación del sprint:

Tabla 13

Historias de Usuario del Sprint 2

HU001	Toma de fotos	Se captura las fotos de las muestras suelos para la formación del dataset	La distancia de la cámara con respecto al objecto no sea mayor a 25 cm	Blanca, Pedro, Luis	Alta	Media
			La inclinación de la cámara sea de 180			
			El ángulo de la cámara sea 105, 90, 75			
			La luminosidad de ambiente sea alta			
			Evitar el ruido del fondo, agregando un fondo de color blanco o negro			
HU002	Preparación de la Muestras de campo	Se realiza la preparación de muestras controladas para la captura de las fotos	El frasco debe ser de forma cilíndrica, con base plana y tapa	Blanca, Pedro, Luis	Alta	Media
			Se debe formar diferentes muestras variando la cantidad de los minerales, o muestra de suelo			
			La muestra debe ocupar el 50% del recipiente, el resto en agua y adicionarle 16 gramos de sal			

			la muestra se debe agitar durante 1 minuto			
			La muestra debe estar en reposo hasta que se pueda diferenciar las capas de los minerales			

En la planificación de sprint, el equipo definió el lapso de 4 semanas para el desarrollo del sprint tomando los nuevos criterios de aceptación establecidos para dar cumplimiento a los requerimientos funcionales.

* Revisión del sprint:

Terminado las actividades del sprint, se continuo con la validación de los criterios de aceptación los cuales fueron aplicado en desarrollo de las tareas. Dando como resultado la aprobación de los criterios de aceptación y complimiento de los requisitos de equipo.

Como resultados se generaron 600 imágenes las cuales dan formación al dataset.

* Retrospectiva de sprint:

En el desarrollo del sprint la comunicación entre los integrantes del proyecto fue participativa. Esto permitió el desarrollo exitoso de las actividades.

Sprint 3

* Planificación del sprint:

Tabla 14

Historias de Usuario del Sprint 3

HU003	Etiquetar Imágenes del dataset	Se realiza la etiquetación de las imágenes mediante el programa cvat para delimitar cada mineral	Etiquetar de forma individual los diferentes minerales de la muestra	Blanca, Pedro, Luis	Alta	Baja

La planificación de sprint se definió un lapso de 4 semanas para realizar la etiquetación de las imágenes mediante la plataforma cvat.

* Revisión del Sprint:

Al culminar las tareas del Sprint 3 se validó que cumplieran con los criterios de aceptación. La revisión concluyo con la aprobación de los criterios de aceptación y un dataset con 600 imágenes y sus respectivas etiquetas.

* Impedimentos:

- Fallas del internet

- La plataforma tenía fallas del Servio durante varios días, lo cual impedía el desarrollo de las actividades

- La plataforma cvat realizo una migración hacia otro dominio, implicando la migración de las imágenes a la nueva plataforma.

* Retrospectiva del sprint:

Se cumplieron con los criterios de aceptación. Pero en el desarrollo de las actividades la plataforma de cvat tenía falla en el servicio web lo que provocó que la entrega no se terminara a tiempo por lo que se decidió dar continuidad al sprint para terminar totalmente el etiquetado de las imágenes.

Sprint 4

* Planificación de Sprint:

Tabla 15

Historias de Usuario del Sprint 4

HU004	Preprocesamiento de imágenes	La arquitectura requiere que las fotografía y anotaciones se ajusten a los parámetros u-net para el correcto entrenamiento del modelo	Recortar y reducir la dimensión de la imagen	Blanca, Pedro, Luis	Alta	Alta
			Establecer la orientación de las imágenes de forma vertical			
			Agrupar las anotaciones por tipo mineral			
			Generar Imágenes binarias			
			Generar las imágenes Trimap			
HU005	Entrenamiento del modelo	yo como usuario requiero un modelo de Deep Learning que identifique en imágenes de pruebas de campo las capas de cada mineral del suelo	Entrenar un modelo por cada tipo de mineral	Blanca, Pedro, Luis	Alta	Alta
			Aplicar la arquitectura U-net			
			Aplicar métricas de entrenamiento accuracy y mean iou			

El equipo realizo la planificación de las actividades de las historias de usuario teniendo en cuenta el dataset de imágenes con sus respectivas etiquetas. El sprint tiene un estimado de tiempo de 2 semanas en la cuales se realizarán las diferentes actividades de

preprocesamiento de imágenes y entrenamiento del modelo buscando las mejores métricas y optimizadores para una eficiencia mayor en predicciones.

* Revisión del Sprint:

En el preprocesamiento de imágenes se realizó el recorte y reducción de la dimensión de las imágenes, generación de las imágenes binarias con las etiquetas y Trimap. En estas actividades se tuvo en cuenta por la arquitectura u-net para el entrenamiento del modelo. Se realizo el entrenamiento de 3 modelos uno por cada mineral.

* Impedimentos:

Para el entrenamiento del modelo no se contó con equipos con las especificaciones mínimas de hardware para la realización de las actividades.

La plataforma colab de forma gratuita tenía una limitante en el uso de los recursos para el entrenamiento.

La universidad Surcolombiana cambio la política de uso del almacenamiento en el drive lo cual nos generó problemas para guardar y usar herramientas en línea.

* Retrospectiva:

En el desarrollo del sprint se contó con una comunicación participativa entre los integrantes del proyecto. Esto permitió el desarrollar exitoso de las actividades.

Sprint 5

* Planificación del sprint

Tabla 16

Historias de Usuario del Sprint 5

HU006	Comunicación entre nodos	El sistema requiere la comunicación inalámbrica para gestionar los nodos	Los módulos se comunican bidireccional mediante radio frecuencia Los nodos hijo enviar datos al principal cada 1 h	Blanca, Pedro, Luis	Alta	Alta
HU007	Sectorizar el sistema de riego	El usuario requiere organizar el sistema de riego por sectores para aplicar un riego diferenciado	Aplicación de diferentes tipos de riegos en cada sector Usar diferentes tipos de cultivo por cada sector Cada sector de riego debe contar con el nodo de comunicación y con los sensores de humedad, electroválvula	Blanca, Pedro, Luis	Medio	Medio

La planificación de sprint tiene en cuenta la configuración de los radios frecuencia, sensores, electroválvulas y releys, para la gestión del riego y comunicación entre nodos, generando un nodo principal y dos nodos hijos.

Para realizar la sectorización del sistema de riego se realizó la gestión controlados de sectores los cuales se tiene en cuenta los diferentes tipos de suelos, tipos de riego y cultivo.

* Revisión del sprint:

Las actividades realizadas se durante el sprint da como resultado la generación de los nodos con sus respectivos componentes y la comunicación inalámbrica bidireccional.

* Impedimentos:

La comunicación de los nodos se vieron afectos por la desconexión de los cables de los Arduino por tal motivo se decidió crear los nodos con Arduino nano y fijar los componentes con soldadura generando mayor estabilidad.

Se organizaron cuatro sectores de riego, en uno de ellos se implementó emisores con microaspersión para un cultivo de árboles leñosos. En otro se utilizó la cinta o manguera perforada para un cultivo de arbustos. Y los otros dos sectores se manejaron emisores de goteo autocompensado, pero se diferenciaban por el tipo de cultivo ya que uno contó con plantas de frijol y el otro con árboles de mango.

Culminadas las actividades del sprint 5, se realizó la validación de los criterios de aceptación. Dando cumplimiento a los requisitos funcionales del sistema. Además, la implementación de comunicación inalámbrica entre nodos y sectores de riego en el sistema.

* Retrospectiva del sprint:

En el desarrollo del sprint se contó con una comunicación participativa entre los integrantes del proyecto. Esto permitió el desarrollar exitoso de las actividades.

En el análisis de la planificación de sprint no se tuvo en cuenta algunas herramientas necesarias para el ensamblaje y protección del sistema de riego, por tal motivo los integrantes del trabajo se vieron afectados para el desarrollo de las actividades.

Sprint 6

* Planificación del sprint

Tabla 17

Historias de Usuario del Sprint 6

HU008	Módulo de Registro de usuario	Yo como usuario requiero registrar credenciales en la plataforma para poder gestionar el sistema de riego	La información personal del usuario se debe guardar en base de datos: nombre, apellido, numero documento, username, password, celular, email, dirección.	Blanca, Pedro, Luis	Medio	Medio
			La contraseña se guarda de forma encriptada			
			Validar que no exista un usuario con el mismo número de documento y username			
HU009	Módulo de autenticación	Yo como usuario requiero acceder a la plataforma con las credenciales registrada para administrar y gestionar la información del sistema de riego	El sistema de autenticación debe ser por JWT (Json Web Token)	Blanca, Pedro, Luis	Alta	Media
			la autenticación debe ser valida con usuarios registrados			
			Las credenciales de acceso deben coincidir con las registradas en base de datos			

El primer módulo de plataforma web debe ser el registro y autenticación del usuario para mantener la consistencia de los datos y la seguridad se los servicio. Para el desarrollo de este sprint se definió un tiempo de 1 semana.

* Revisión del Sprint:

La autenticación de la plataforma se realiza mediante JWT el cual tiene un tiempo de expiración de 4 horas. Adicionalmente, la contraseña del usuario debe ser guardada de forma encriptada. Al termino de las actividades del sprint 6, se realizó la validación de los criterios. El resultado concluyó con el cumplimiento de los requisitos funcionales. Además, se desarrolló el módulo que permite el registro e inicio de sesión de usuarios en el sistema.

* Impedimentos:

- El almacenamiento de los datos en elephant es limitado por su suscripción gratuita.

* Retrospectiva del sprint:

En el desarrollo del sprint se contó con una comunicación participativa entre los integrantes del proyecto. Por ende, las tareas planteadas se ejecutaron con éxito.

Sprint 7

* Planificación del sprint:

Tabla 18

Historias de Usuario del Sprint 7

HU010	Módulo de Registro de Finca	Yo como usuario requiero registrar fincas para administrar los cultivos y lotes	La información de la finca se debe guardar en base de datos: nombre, dirección, latitud, longitud, altitud, usuario.	Blanca, Pedro, Luis	Media	Media
			El usuario puede registrar más de una finca			
			Las coordenadas son de grados decimales			
HU011	Módulo de Registro de Lotes	Yo como usuario requiero registrar los lotes que pertenecen a una finca para administrar sectores	La información del lote se debe guardar en base de datos: nombre, finca, área, latitud, longitud, altitud.	Blanca, Pedro, Luis	Media	Media
			Las coordenadas son de grados decimales			
			El usuario no puede registrar un lote sin relacionarlo a una finca			
			El usuario puede registrar más de un lote			

La planificación del Sprint tuvo en cuenta la continuidad que tendrá el usuario con la plataforma una vez que se autentique en el sistema. Por lo cual, se desarrollaron los

módulos que permiten el registro de fincas con sus respectivos lotes. El desarrollo de este sprint se realizó en un tiempo de 2 semanas.

* Revisión del sprint:

Terminadas las actividades del sprint 7, se realizó la validación de los criterios. El resultado concluyó con la aprobación de los criterios de aceptación. Además, se desarrolló las vistas y funciones básicas del CRUD para el módulo de registro de fincas y lotes en el sistema, aplicando sus pruebas unitarias.

* Retrospectiva del sprint:

El desarrollo del sprint se caracterizó por tener una comunicación participativa entre el equipo de desarrollo. En consecuencia, el desarrollo de las tareas se realizó satisfactoriamente.

Sprint 8

* Planificación del sprint:

Tabla 19

Historias de Usuario del Sprint 8

HU012	Módulo de Registro de Sectores	Yo como usuario requiero registrar los sectores que pertenecen a un lote para administrar los diferentes riegos	La información del lote se debe guardar en base de datos: nombre, lote, área, latitud, longitud, altitud, tipo suelo, cultivo.	Blanca, Pedro, Luis	Media	Media
			Las coordenadas son de grados decimales			
			El usuario no puede registrar un lote sin relacionarlo a un lote			
			El usuario puede registrar más de un sector			

HU013	Modulo registro de cultivo	Yo como usuario requiero registrar los cultivos	La información del cultivo debe guardar en base de datos: nombre, tipo de cultivo, fecha de inicio, fecha final, estado y usuario. El usuario puede registrar más de un cultivo	Blanca, Pedro, Luis	Media	Media

Se planificó el sprint con actividades de historias de usuario que daban continuidad a los módulos ya desarrollados. La realización de este sprint se tardó un lapso de 2 semanas. en la cuales se realizarán las vistas y las funciones básicas del CRUD para el manejo de sectores y fincas, con sus respectivas pruebas unitarias.

* Revisión del sprint:

Al finalizar las actividades del sprint 8, se realizó la validación de los criterios. El resultado concluyó con la aprobación de los criterios de aceptación, además, se desarrolló el módulo que permite el registrar y asociar sectores y cultivos a finca previamente registradas.

* Retrospectiva del sprint:

El desarrollo del sprint tuvo una comunicación participativa entre los integrantes del equipo de desarrollo. Por lo anterior, las tareas propuestas se desarrollaron satisfactoriamente.

Sprint 9

* Planificación del sprint:

Tabla 20

Historias de Usuario del Sprint 9

HU014	Módulo de fincas	yo como usuario requiero visualizar las fincas para gestionar y administrar los lotes	El usuario puede visualizar las fincas registradas El usuario puede ver el detalle de las fincas	Blanca, Pedro, Luis	Media	Media
HU0015	Módulo de Lotes	yo como usuario requiero visualizar los lotes asociados a una finca para gestionar y administrar los sectores de riego	El usuario puede visualizar los lotes registrados El usuario puede ver el detalle del lote	Blanca, Pedro, Luis	Media	Media

La planificación del sprint se definió un lapso de 2 semanas para la realización de las actividades.

* Revisión del Sprint:

Finalizadas las actividades del sprint 9, se realizó la validación de los criterios. De lo anterior, se concluyó con el cumplimiento de los requisitos funcionales. El desarrollo de vistas que le permiten al usuario visualizar en la plataforma las fincas registradas con sus respectivos lotes, adicionalmente de la vista del detalle de finca y lotes.

* Retrospectiva del Sprint:

En el desarrollo del sprint se contó con una comunicación participativa entre los integrantes del proyecto. Por ende, las tareas planteadas se ejecutaron con éxito.

Sprint 10

* Planificación del sprint:

Tabla 21

Historias de Usuario del Sprint 10

HU016	Módulo de sectores	yo como usuario requiero visualizar los sectores de riego para gestionar y administrar el riego en los cultivos	El usuario puede visualizar los sectores registrados	Blanca, Pedro, Luis	Media	Media
			El usuario puede ver el detalle del sector			
			El usuario puede interactuar con los componentes de riego e IoT mediante la plataforma			

La planificación del sprint se definió un lapso de 3 semana para la realización de las actividades.

* Revisión del Sprint:

Al terminar las actividades del sprint 10, se procedió a realizar la validación de los criterios. Resultado de lo anterior, confirmó el cumplimiento de los requisitos funcionales, en el desarrollo de vistas que le permiten al usuario visualizar en la plataforma los sectores registrados, descripción del cultivo y los componentes de riego e IoT asociado al sector, y la posibilidad de operar manualmente la electroválvula desde la plataforma.

*Retrospectiva del Sprint:

En el desarrollo del sprint se contó con una comunicación participativa entre los integrantes del proyecto. Por ende, las tareas planteadas se ejecutaron con éxito.

Sprint 11

* Planificación del sprint:

Tabla 22

Historias de Usuario del Sprint 11

HU017	Módulo de cultivos	yo como usuario requiero visualizar los cultivos para identificar el seguimiento del riego en las diferentes etapas	El usuario puede visualizar los cultivos registrados El usuario puede visualizar el estado del cultivo	Blanca, Pedro, Luis	Media	Media

La planificación de sprint de definió el lapso de 2 semana para la realización de las actividades.

* Revisión del Sprint:

Al terminar las actividades del sprint 11, se comprobó que lo desarrollado cumplió con los criterios de aceptación y requisitos funcionales. Resultado de lo anterior, se evidenció con el desarrollo de vistas que le permiten al usuario visualizar en la plataforma información descriptiva de los cultivos registrados, y el historial de riego en sus diferentes etapas.

* Retrospectiva del sprint:

En la planificación del sprint se fue mejorando el análisis de las actividades favoreciendo el desarrollo y mejorando los tiempos de ejecución. Además de la reducción de los impedimentos y entrega de los resultados.

Sprint 12

* Planificación del sprint:

Tabla 23

Historias de Usuario del Sprint 12

HU018	Módulo de predicción	yo como usuario requiero determinar el tipo de suelo mediante imágenes de pruebas de campos para asociarla a un sector de riego	El usuario puede subir una imagen de la muestra de suelo en formato png, jpg, jpeg.	Blanca, Pedro, Luis	Media	Media
			El sistema debe retornar el tipo de suelo y los porcentajes de minerales			
			El usuario debe contar con una herramienta que le permita cuando sea necesario ajustar las alturas de las capas de la imagen generada por el sistema			

La planificación del sprint debe tener en cuenta los modelos entrenados los cuales deben determinar los porcentajes de minerales y determinar el tipo de suelo en imágenes de pruebas de campo. Para el desarrollo de estas actividades se definió un lapso de 3 semanas.

* Revisión del Sprint:

Culminadas las actividades del sprint 12, se comprobó que lo desarrollado cumplía con los criterios de aceptación. El Resultado se evidenció con el desarrollo de un módulo en la plataforma que le ofrece al usuario la opción cargar imágenes con muestras de campo, para que el sistema determine el tipo de suelo. De ser necesario, el usuario puede ajustar los niveles de las capas detectadas para mejorar la clasificación y asignar el resultado a un sector

de riego. Además, se puede visualizar en la plataforma información descriptiva de los cultivos registrados, y el historial de riego en sus diferentes etapas.

* Retrospectiva del Sprint:

En el desarrollo del sprint, el manejo de las actividades fue adecuada para la culminación de las actividades. Además, no se presentó impedimentos y la entrega de los resultados de dio en la fecha establecida.

10.1.2 Pruebas unitarias

La siguiente tabla muestra las diferentes pruebas unitarias para verificar el funcionamiento correcto del sistema. sin embargo, no se encuentra todas las pruebas realizadas sino las principales.

Tabla 24

Descripción y resultado de pruebas unitarias.

Clase	Método	Datos de entrada	Descripción de la prueba	Resultado esperado	¿Cumplió? S/N
CultivoDao	seleccionarTodos	No aplica	Se debe obtener el listado de cultivos.	El método obtiene una lista registro de la base de datos correctamente y retorna una lista de tipo cultivo.	Si
	buscarPorUsuario	id_usuario: 25	Se valida la obtención del listado de cultivos para un usuario.	El método obtiene una lista registro de la base de datos correctamente y retorna una lista de tipo cultivo.	Si
	buscarPorId	id:1	Se valida la no obtención de un cultivo por el id especificado	El método no obtiene un registro de la base de datos y retorna un None.	Si
	buscarPorId	id:2	Se valida la obtención de un cultivo por	El método obtiene un registro de la base de datos	Si

			el id especificado.	correctamente y retorna un objeto de tipo cultivo.	
	eliminar	id:1	Se valida la eliminación de un cultivo con el id especificado.	El método elimina el registro de la base de datos correctamente y retorna un el número de línea afectadas.	Si
	insertar	nombre: maiz, tipoCultivo: 1, fechaInicio: 2022-01-01, fechaFinal : 2023-01-01 , estado: 1, user_id:25	Se valida la creación de un registro en base de datos en la tabla cultivo.	El método crea un registro de la base de datos correctamente y retorna un el número de línea afectadas.	Si
	actualizar	nombre: maiz, tipoCultivo: 1, fechaInicio: 2022-01-01, fechaFinal : 2023-01-01, estado: 1, user_id:25, id: 3	Se valida la actualización de un registro en base de datos en la tabla cultivo.	El método actualiza un registro de la base de datos correctamente y retorna un el número de líneas afectadas.	Si
	seleccionarTodos	No aplica	Se debe obtener el listado de fincas.	El método obtiene una lista registro de la base de datos correctamente	Si

FincaDao				y retorna una lista de tipo Finca.	
	buscarFincaPorUsuario	id_usuario: 25	Se valida la obtención del listado de fincas para un usuario.	El método obtiene una lista registro de la base de datos correctamente y retorna una lista de tipo Finca.	Si
	buscarFincaPorId	id:1	Se valida la obtención de una finca por el id especificado.	El método obtiene un registro de la base de datos correctamente y retorna un objeto de tipo Finca.	Si
	eliminar	id:1	Se valida la eliminación de una finca con el id especificado.	El método elimina el registro de la base de datos correctamente y retorna un el número de líneas afectadas.	Si
	insertar	nombre: eden, direccion: cr 31 sur, latitud: 2.883137459182263, longitud:-75.26639088403925, altitud:0,	Se valida la creación de un registro en base de datos	El método crea un registro de la base de datos correctamente y retorna un el	Si

		user_is: 25	en la tabla finca.	número de líneas afectadas.	
	actualizar	nombre : eden 2, direccion: cr 31 sur, latitud: 2.883137459182263, longitud:-75.26639088403925, altitud:0, user_is: 25, id:2	Se valida la actualización de un registro en base de datos en la tabla finca.	El método actualiza un registro de la base de datos correctamente y retorna un el número de líneas afectadas.	Si
LoteDao	seleccionarTodos	No aplica	Se debe obtener el listado de lotes.	El método obtiene una lista registro de la base de datos correctamente y retorna una lista de tipo Lote.	Si
	buscarPorId	id:1	Se valida la obtención de una Lote por el id especificado.	El método obtiene un registro de la base de datos correctamente y retorna un objeto de tipo Lote.	Si
	buscarPorFinca	id_finca:1	Se debe obtener el listado de lotes por id de la finca.	El método obtiene una lista registro de la base de datos correctamente	Si

				y retorna una lista de tipo Lote.	
	eliminar	id:1	Se valida la eliminación de un lote con el id especificado.	El método elimina el registro de la base de datos correctamente y retorna un el número de líneas afectadas.	Si
	insertar	nombre: lot-07, area: 1, latitud: 2.883137459182263, longitud:-75.26639088403925, altitud:0, finca_id:1	Se valida la creación de un registro en base de datos en la tabla lote.	El método crear un registro de la base de datos correctamente y retorna un el número de líneas afectadas.	Si
	actualizar	nombre: lot-07, area: 1, latitud: 2.883137459182263, longitud:-75.26639088403925, altitud:0, finca_id:1, id:2	Se valida la actualización de un registro en base de datos en la tabla lote.	El método actualiza un registro de la base de datos correctamente y retorna un el número de líneas afectadas.	Si
	buscarPorDocumento	num_identificacion:1075314119	Se valida la obtención de una Usuario	El método obtiene un registro de la	Si

			por el id especificado.	base de datos correctamente y retorna un Objecto de tipo User.	
UsuarioDao	buscarUserName	username:luis	Se valida la obtención de una Usuario por el username especificado.	El método obtiene un registro de la base de datos correctamente y retorna un Objecto de tipo User.	Si
	eliminar	id:2	Se valida la eliminación de un usuario con el id especificado.	El método elimina el registro de la base de datos correctamente y retorna un el número de líneas afectadas	Si
	insertar	"username":"fernando", "password":"fernando", "num_identificacion":107531411911, "num_celular":3204455211, "direccion":"cr 31 torre 11 apto 301", nickname:luis fernando, apellido: enciso caballero, correo:encisolf901@gmail.com, tipo_identificacion:1	Se valida la creación de un registro en base de datos en la tabla usuario.	El método crear un registro de la base de datos correctamente y retorna un el número de líneas afectadas.	Si
	actualizar	"username":"fernando", "password":"fernando", "num_identificacion":107531411911,	Se valida la actualización de un registro	El método actualiza un registro de la	Si

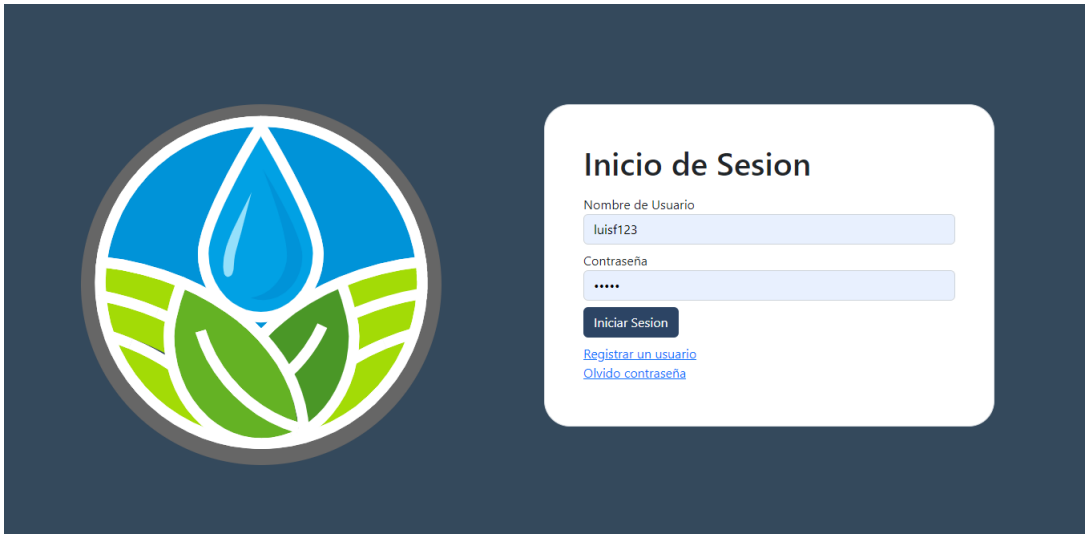
		"num_celular":3204455211, "direccion":"cr 31 torre 11 apto 301", nickname:luis fernando, apellido: enciso caballero, correo:encisolf901@gmail.com, tipo_identificacion:1, id:1	en base de datos en la tabla usuario.	base de datos correctamente y retorna un el número de líneas afectadas.

10.1.3 Plataforma Web

la plataforma permite al usuario poder registrase y acceder a la plataforma de forma segura, guardando la información proporcionada por el usuario en la base de datos. Con la autenticación de los usuarios podemos mantener protegida la información de cada usuario.

Figura 32

Módulos de Registro e Inicio de Sesión en la Plataforma.



Nota. Elaboración propia.

Cuando el usuario inicie sesión podrá acceder al menú de opciones donde podrá realizar sus acciones dependiendo del módulo.

Figura 33

Home de la Plataforma

Nota. Elaboración propia.

En el módulo de propiedades se encuentra información relacionada a las fincas, lotes y sectores de riego que tiene el usuario registrado, además de poder realizar registros de los mismo.

Figura 34

Vista del Módulo para Registrar Lotes

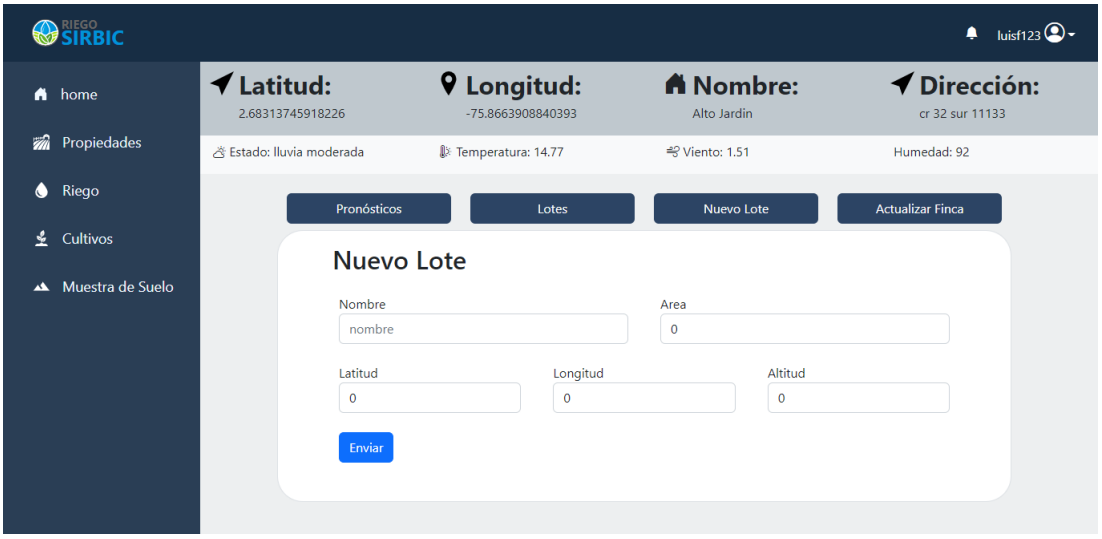


Figura 35

Listar Lotes Registrados en la Plataforma

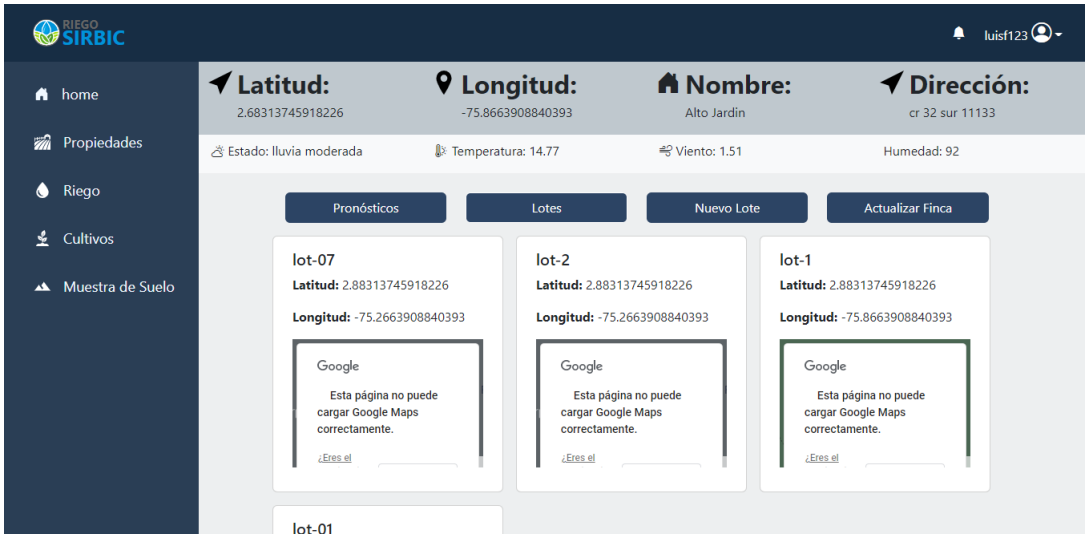


Figura 36

Visualizar Pronósticos Climáticos de la Finca

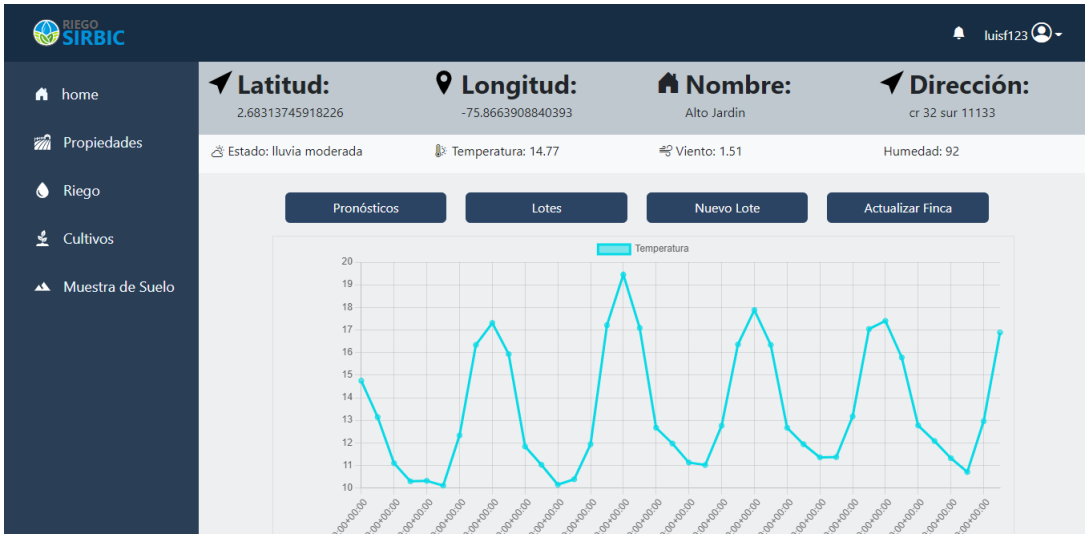


Figura 37

Vista del Módulo para Actualizar Información de una Finca en la Plataforma

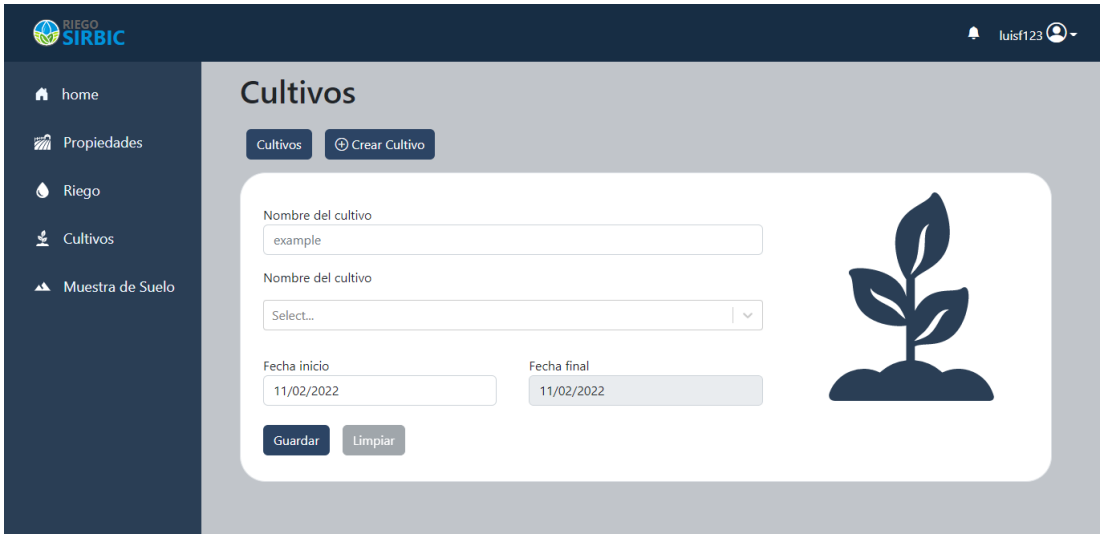
En el módulo de cultivos se encuentra la información de los cultivos registrados por el usuario, además del registro de nuevos cultivos.

Figura 38

Listar Cultivos Registrados en la Plataforma

Figura 39

Vista del Módulo de Registro Cultivos en la Plataforma



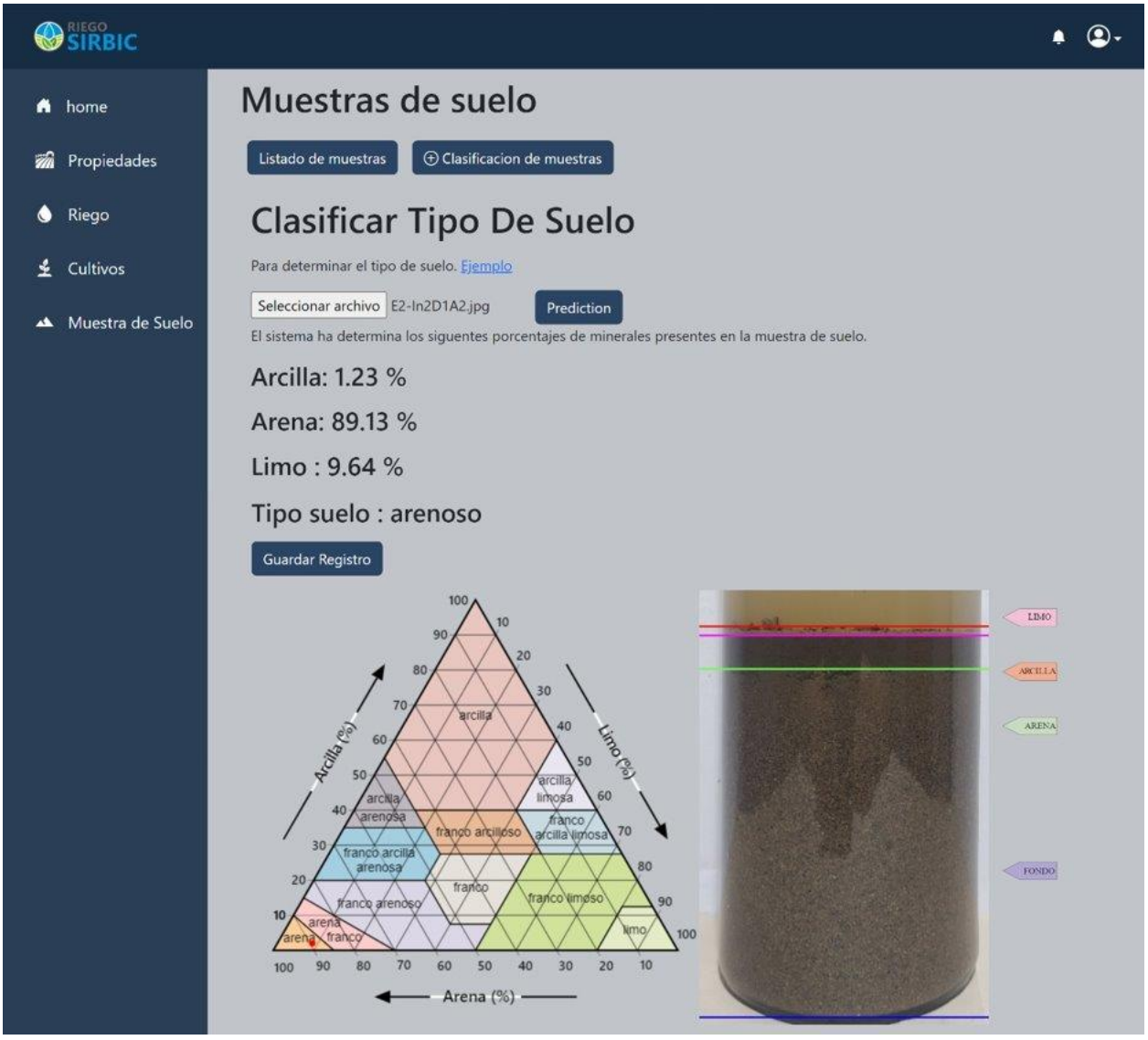
En el módulo de muestras de suelo, el usuario puede realizar la predicción de tipo de suelo presente en los cultivos mediante fotografías de muestra de campo,

Figura 40

Vista del Módulo de Muestras de Suelo

Figura 41

Vista del Módulo de Clasificación de Suelos



El módulo de riego el usuario puede gestionar y administrar el riego de cada sector y los dispositivos de IoT, además de visualizar los riegos aplicados.

10.1.4 Repositorios y Despliegue

GitHub es el repositorio remoto utilizado para manejar el control de versión de la plataforma de forma pública. La cual se dividió en frontend, backend e IoT. A continuación, se lista los enlaces de acceso a los repositorios.

https://github.com/luisf123321/RiegoReact

https://github.com/luisf123321/RiegoFlask

https://github.com/luisf123321/RiegoIoT

En la plataforma Drive de Google se almaceno el dataset de las imágenes, los códigos de preprocesamiento de las imágenes, entrenamiento y aplicación de las métricas de evaluación en cada modelo.

https://cutt.ly/oMBot5x

La plataforma se desplego según lo establecido en netlify, el cual nos proporciona el siguiente dominio para el acceso de los usuario.

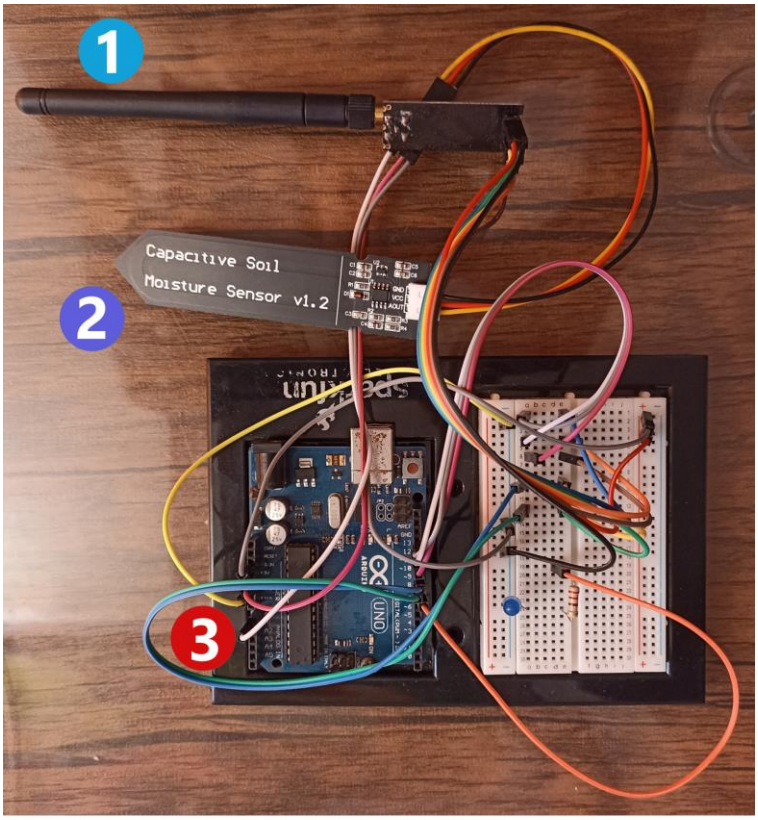
https://sirbic.netlify.app/

10.2 IoT

En la primera fase de ensamblaje se realiza la conexión del sensor de humedad y módulo de radio frecuencia, se configuro la comunicación entre nodos estableciendo un canal de comunicación. Adicionalmente se realizó la calibración del sensor de humedad para las condiciones locales.

Figura 42

Conexión del Nodo de Radiofrecuencia y Sensor de Humedad con el Arduino

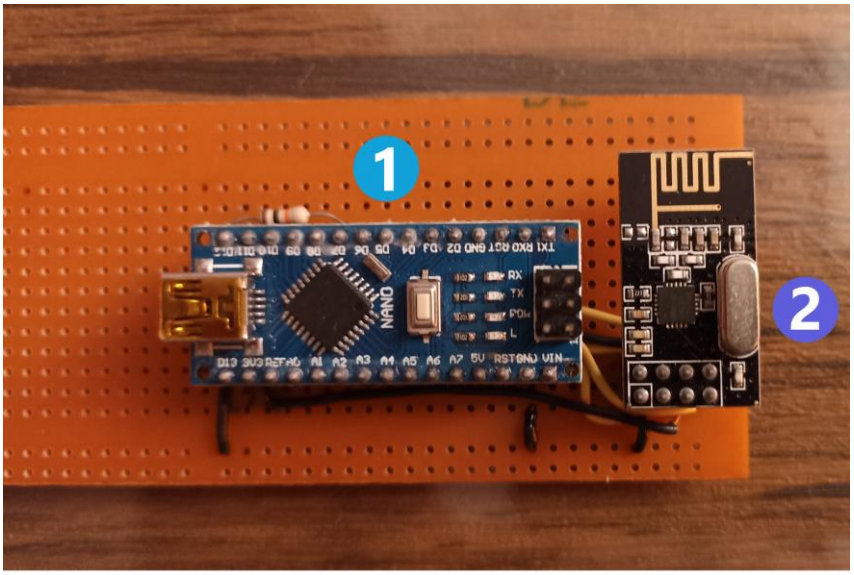


Nota. Elaboración propia.

En la fase 2 se estructura el ensamblaje mediate una placa de pruebas para la conexión de los deferentes componentes del nodo. Para mayor estabilidad se soldaron los componentes reduciendo los problemas de comunicación entre nodos.

Figura 43

Conexión del Arduino Nano con el Módulo de Radiofrecuencia



1 **Arduino Nano**

2 **Módulo de radiofrecuencia nrf24L01**

Nota. Elaboración propia.

En las siguientes imágenes se muestra el ensamblaje del sistema de riego divido en tres tipos de riego que son goteo, microaspersión y cinta.

Para el riego por goteo se usó dos tipos de cultivos, tres plantas de frijol y tres plantas de mango.

Figura 44

Sector de Riego con Cultivo de Frijol y Emisores de Goteo

Figura 45

Sector de Riego Emisor de Goteo con Cultivo de Mango

Nota. Elaboración propia.

En el caso de riego por microaspersión se utilizó tres aspersores que fueron colocados cada en un árbol de mango.

Figura 46

Sector de Riego con Microaspersión

Nota. Elaboración propia.

Y por último en el riego por cinta se empleó una cinta en arbustos de limoncillos.

Figura 47

Sensor de Humedad en Sector de Riego con Cinta

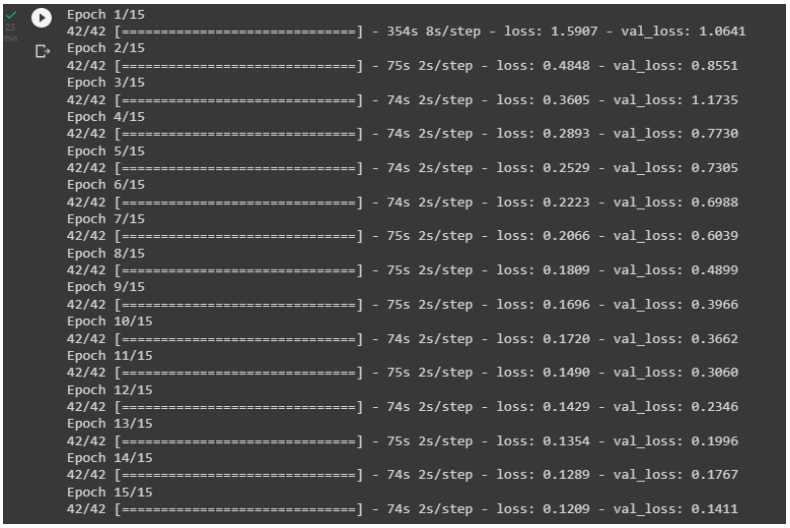
Nota. Elaboración propia.

10.3 Inteligencia Artificial

En la Figura 48 se observa cual fue el comportamiento durante el entrenamiento del modelo de Arena, que tuvo los siguientes parámetros: Bach size 10, época 15, 424 imágenes de entrenamiento (train), 91 de validación (validation) y 91 de prueba (test).

Figura 48

Entrenamiento del Modelo de Arena

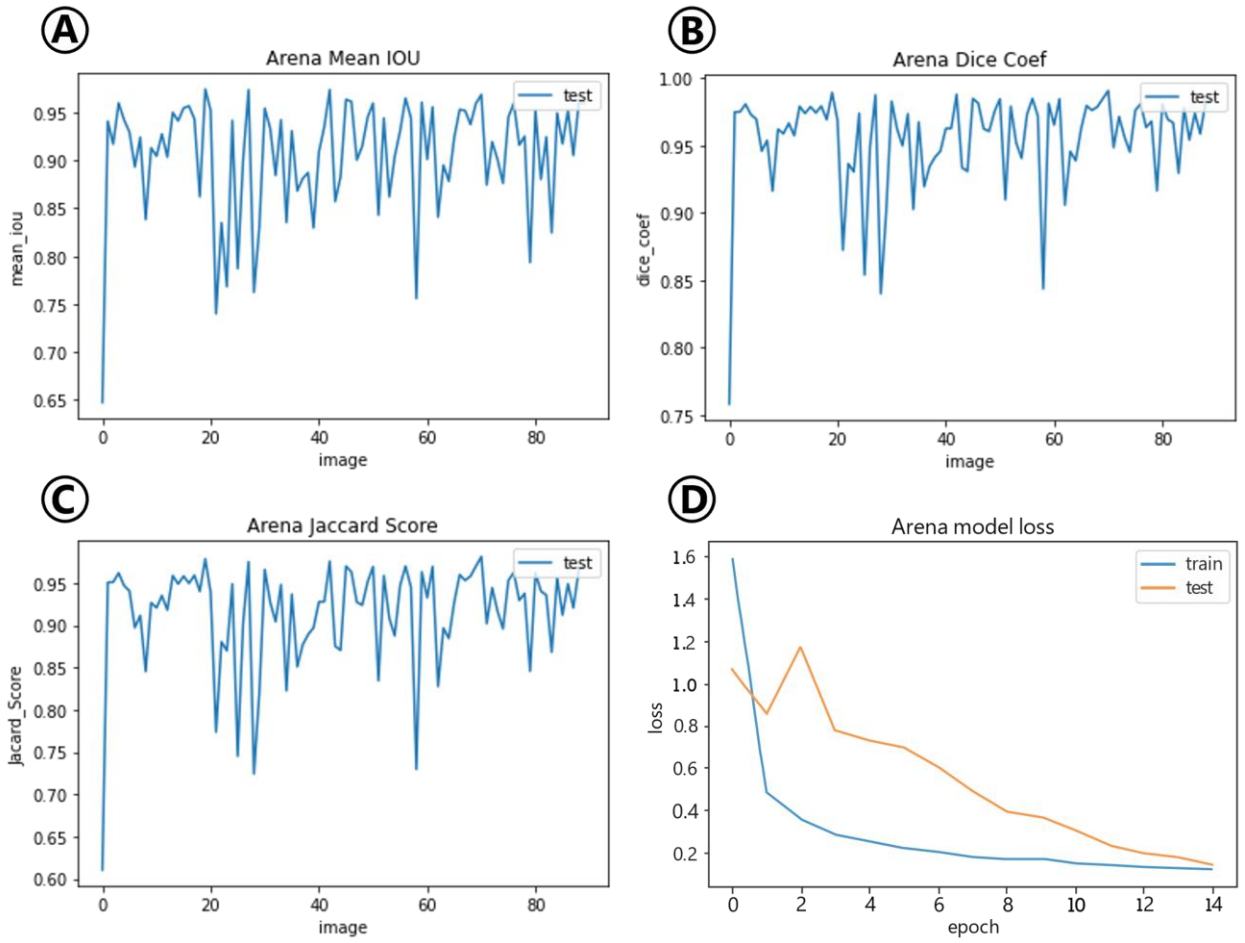


Nota. Elaboración propia.

En la Figura 49 se observa el resultado que se obtuvo al aplicar las métricas de evaluación con las imágenes de Testing, para el modelo de detección de la arena.

Figura 49

Métricas de Evaluación en Modelo Arena



Nota. Comparación de las métricas de evaluación para el modelo de Arena: A métrica Mean IOU, B métrica Dice-Coef, C Jaccard Score y D métrica función de perdida (loss).

Al aplicarlas las métricas de evaluación con el modelo entrenado para la arena, en general se observó que tuvo resultados buenos, especialmente con la métrica Dice Coef, como se ve en la Tabla 25.

Tabla 25

Resultado Métricas de Evaluación para Modelo de Área

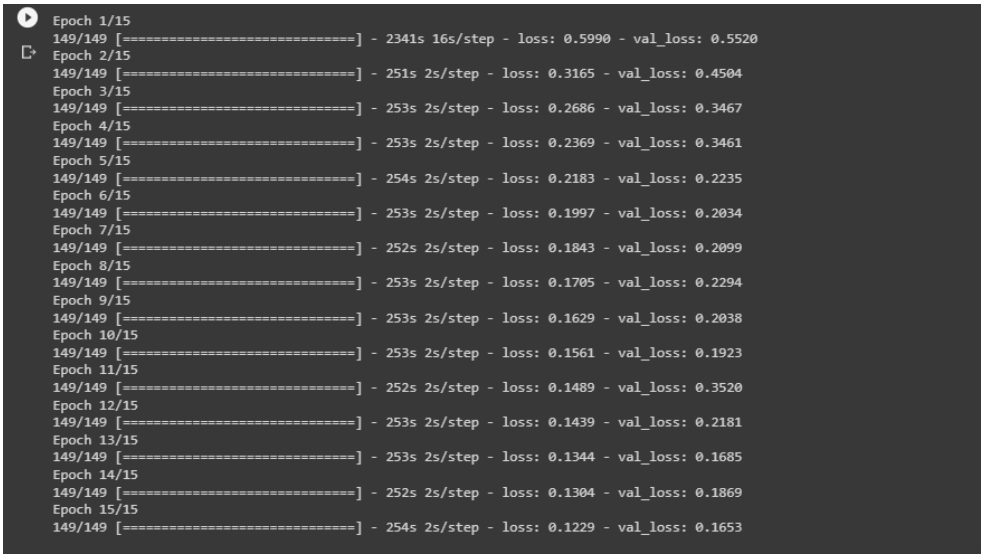
Métrica	Resultado	Desviación estándar	Número de imagines
Mean iou	91%	6%	90
Jaccard Score	91%	6%	90
Dice coef	95%	3%	90

Nota. Elaboración propia.

En la Figura 50 se observa el comportamiento que se tuvo durante el entrenamiento del modelo para la arcilla.

Figura 50

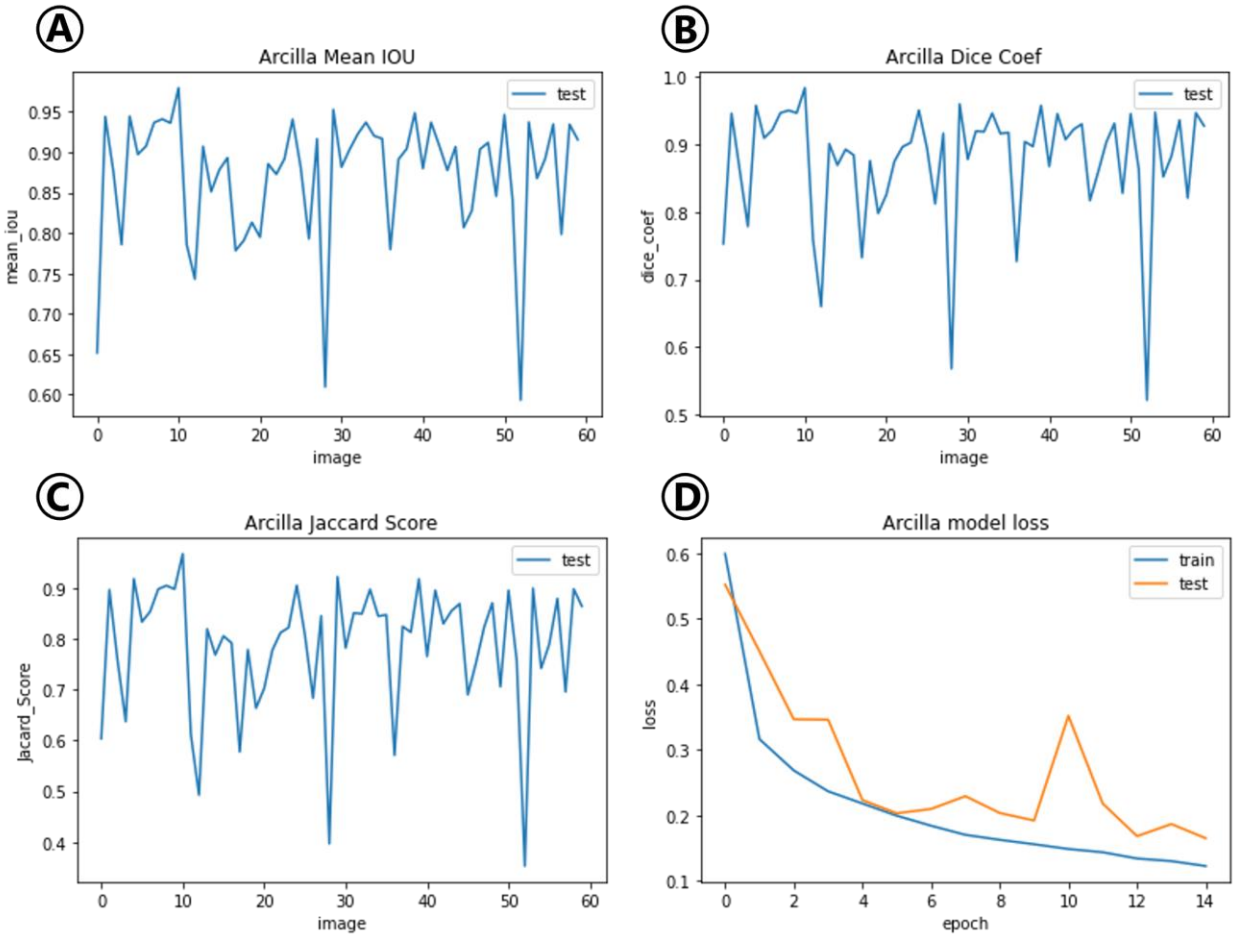
Entrenamiento del Modelo para la Arcilla



En la Figura 51 se observa el resultado que se obtuvo al aplicar las métricas de evaluación con las imágenes de Testing.

Figura 51

Métricas de Evaluación para Modelo Arcilla



Nota. Comparación de las métricas de evaluación para el modelo de Arcilla: A métrica Mean IOU, B métrica Dice-Coef, C Jaccard Score y D métrica función de perdida (loss).

En la Tabla 26 se observa que la evaluación del modelo tuvo resultados aceptables, especialmente con las métricas Mean IOU y Dice Coef.

Tabla 26

Resultado Métricas de Valuación para la Arcilla

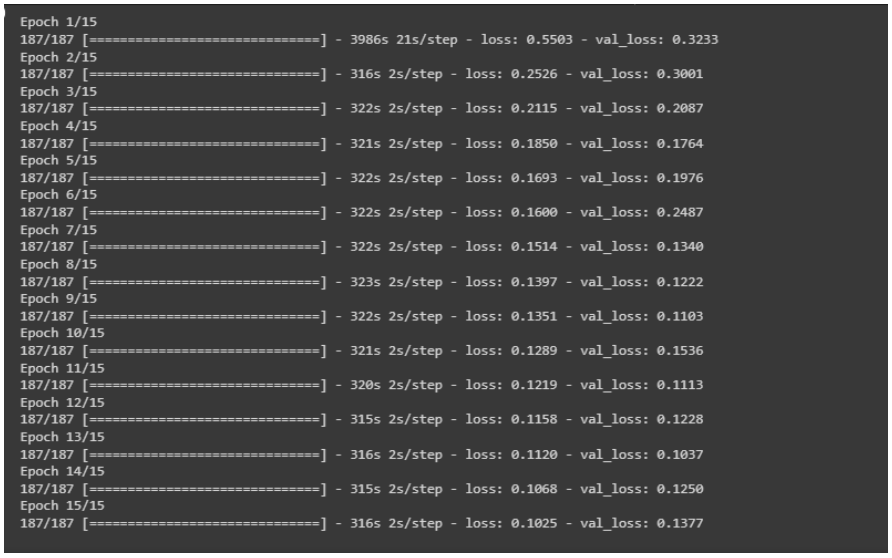
Métrica	Resultado	Desviación estándar	Numero de imágenes
Mean iou	87%	7%	60
Jaccard Score	78%	12%	60
Dice coef	87%	8%	60

Nota. Elaboración propia.

En la Figura 52 se muestra el comportamiento de la red neuronal durante el proceso de entrenamiento, para obtener el modelo del limo.

Figura 52

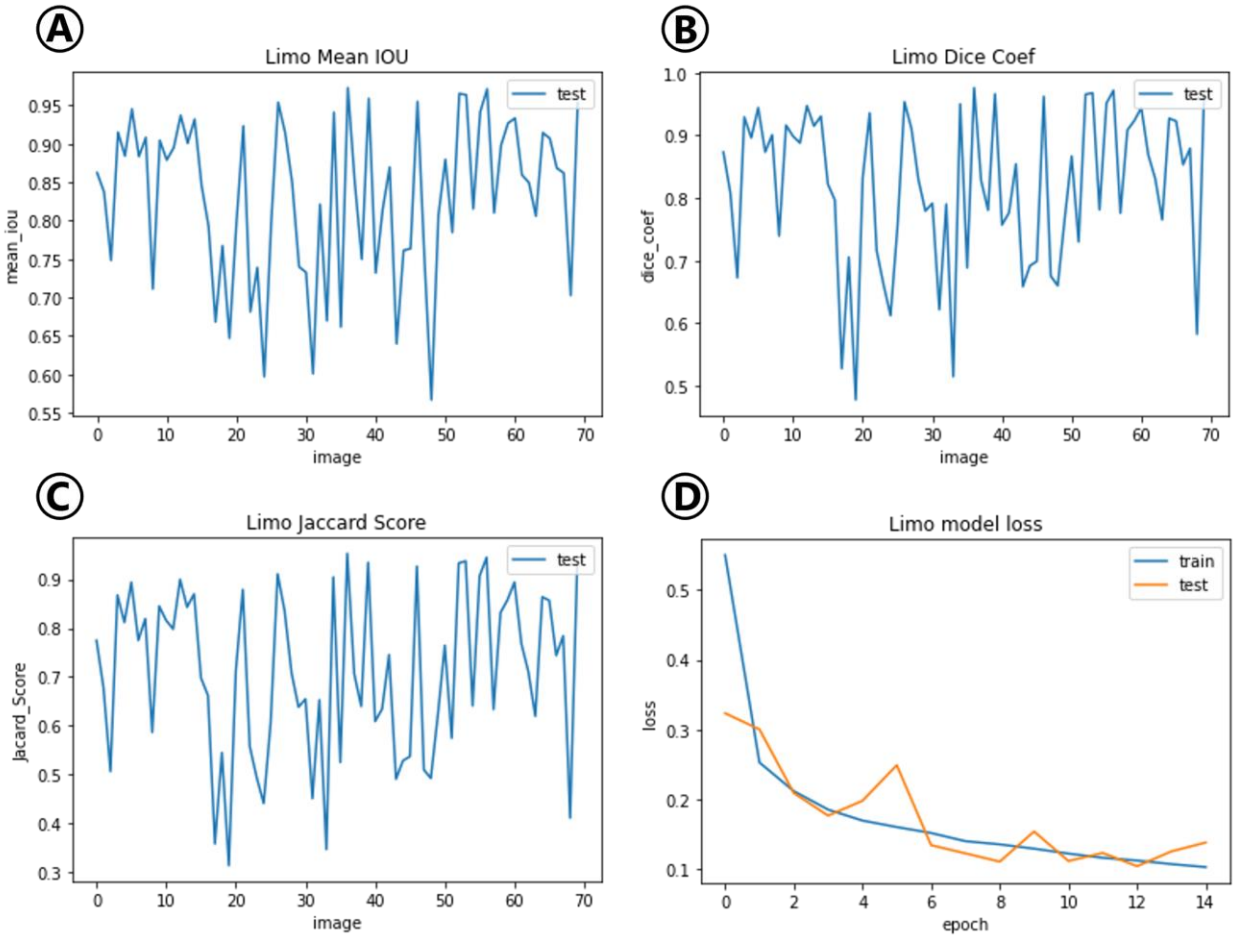
Entrenamiento del Modelo para el Limo.



En la Figura 53 se observa el resultado que se obtuvo al aplicar las métricas de evaluación con las imágenes de Testing, para el modelo de detección del limo.

Figura 53

Métricas de Evaluación para Modelo del Limo



Nota. Comparación de las métricas de evaluación para el modelo de limo: A métrica Mean IOU, B métrica Dice-Coef, C Jaccard Score y D métrica función de perdida (loss).

En la Tabla 27 se observa que la evaluación del modelo tuvo resultados aceptables, especialmente con las métricas Mean IOU y Dice Coefficient.

Tabla 27

Resultado Métricas de Evaluación para el Limo

Métrica	Resultado	Desviación estándar	Numero de imágenes
Mean IOU	82 %	10 %	70
Jaccard Score	70 %	16 %	70
Dice coef	81 %	12 %	70

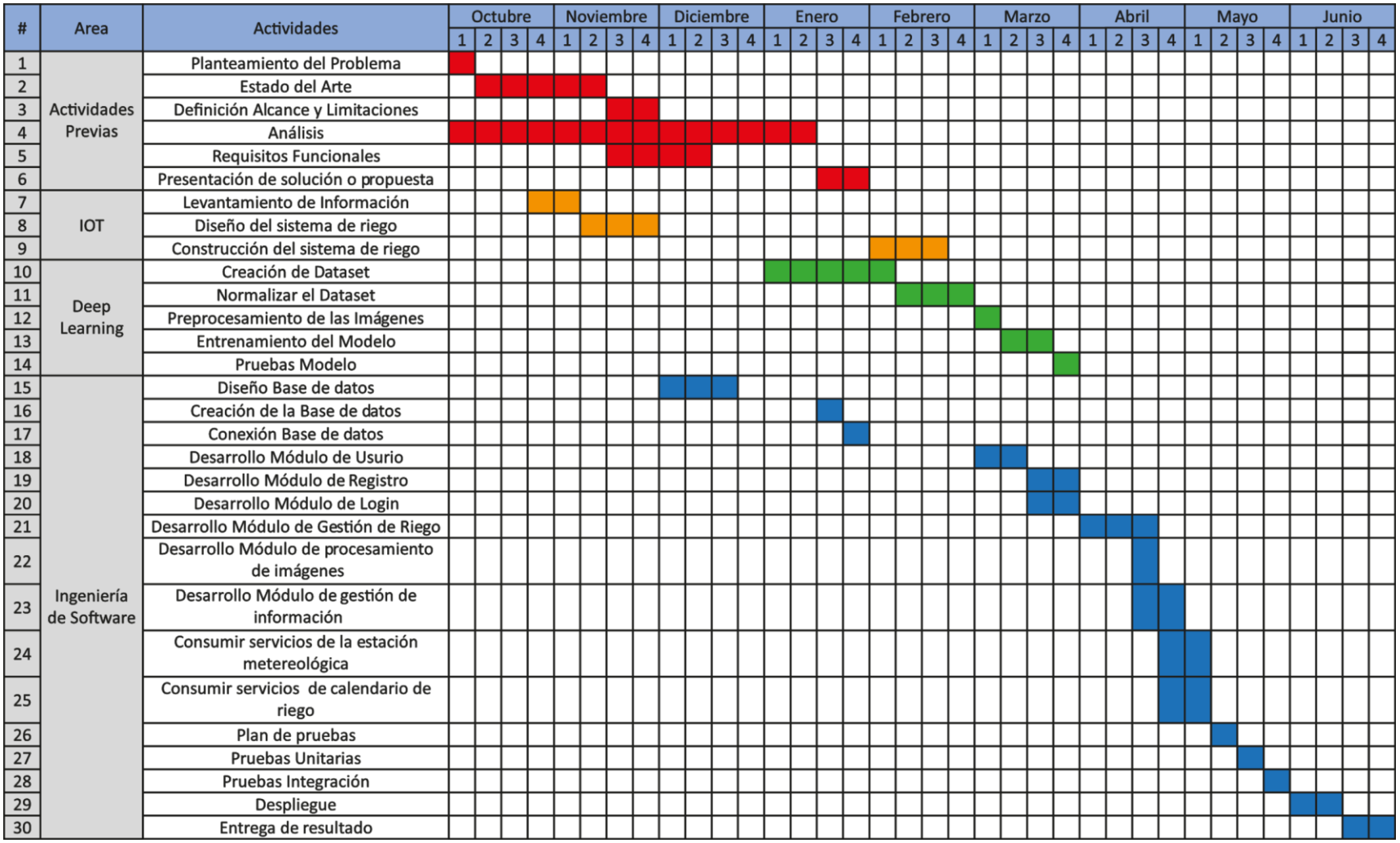
Nota. Elaboración propia.

11 Cronograma

El proyecto se abordó durante 36 semanas en las cuales se desarrolló las actividades expuestas en la metodología, dando como inicio el 1 de octubre del 2021 y la finalización el 30 de junio de 2022.

Figura 54

Cronograma del proyecto



12 Recursos

Para el desarrollo del proyecto se requiere algunos recursos materiales de hardware, herramientas de software y recurso humano. Es importante tener en cuenta que el precio de estos materiales se basa en los costos al momento de realizar el planteamiento de esta propuesta, por lo cual los precios están sujetos a cambios. A continuación, se describen:

12.1 Materiales de Equipos

En la siguiente tabla se describen materiales de hardware e IoT con sus respectivas especificaciones, el costo unitario y la cantidad de unidades necesarios para el desarrollo de este proyecto.

Tabla 28

Materiales de hardware

#	Nombre	Especificaciones	Costo $COP	Cantidad
1	Computador Portátil	Marca: Lenovo Modelo: G40-80 RAM: 6.00 GB Disco Duro: 1 TB Procesador: Intel Core i5	2.000.000	1
2	Computador Portátil #2	Marca: Lenovo Modelo: G40-80 RAM: 8.00 GB Disco Dura: 1 TB Procesador: Intel Core i5 Tarjeta Gráfica: AMD	2.500.000	1

3	Computador Portátil #3	Marca: Apple Modelo : MacBook Pro-RAM : 8.00 GB Disco Sólido : 250 GB Procesador : Intel Core i7	2.500.000	1
4	Raspberry	Modelo : Pi 4 B RAM : 4.00 GB Procesador : Broadcom BCM2711	300.000	1
5	Arduino	Modelo : UNO rev	30.000	1
6	Sensor de Humedad	Modelo : FC-28	6.000	4
7	Electroválvulas	120 voltios 1/2 pulgada corriente nominal 200 mA 0.2 mpa (pascales) Diafragma operado por servo	35.000	4
8	Radio frecuencia	Modelo:NRF24L01+ Voltaje de alimentación: 3-3.6V Potencia máxima de salida: +20 dBm Empresa: Nordic Semiconductor Distancia a Tasa de 2MB (zona abierta): 520m Distancia a Tasa de 250Kb (zona abierta): 1000m	20.000	2
9	Smart Phone	Modelo: Xiaomi redmi note 10S RAM: 6 GB Almacenamiento: 128 GB Cámara: 64 MP	1.100.000	1
10	Smart Phone	Modelo: Xiaomi redmi note 8 RAM: 4 GB Almacenamiento: 64 GB Cámara: 48 MP	700.000	1

11	Smart Phone	Modelo: Samsung Galaxy M32 RAM: 6 GB Almacenamiento: 128 GB Cámara: 64 MP	780.000	1
9	Protoboard		12.000	2
			Total:	**9’983.000**

Nota. Elaboración propia.

12.2 Software

Por otro lado, las herramientas de software seleccionadas para este proyecto fueron seleccionadas teniendo en cuenta no solo su objetivo, sino que también se tuvo en cuenta que las licencias sean Open Source o free, como se muestra en la Tabla 30.

Tabla 29

Recursos de Software

#	***Nombre***	***Versión***	***Licencia***
1	Python	3.8	Opensource
2	Flask		Opensource
3	MongoDB	Atlas	Suscripción gratis
4	React.js	16.8.0	Opensource
5	Node.js		Opensource
6	C++		Opensource
7	Colaboratory		—

Nota. Elaboración propia.

12.3 Materiales de Riego

Finalmente, en la Tabla 31 se listan los componentes básicos de riego necesarios para realizar el sistema.

Tabla 30

Materiales de riego

#	Nombre	Cantidad	Valor ($ COP)
1	Manguera plástica	50 m.	30.000
2	Aspersores de goteo	10 u.	25.000
3	Unión rápida de manguera ½	1 u.	1.000
4	Manguera perforada (Cinta)	10 m.	10.000
5	Microaspersores Con Estacas	3 u.	27.000
6	Estacas de goteros	6 u.	4.000
7	Válvulas rosca hembra manuales 1/2	1	3.000
8	Válvula manguera- manguera	2	8.000
9	Conexión hembra de manguera 16 mm 1/2	2	10.000
10	Conexión macho de manguera 1/2	2	5.000
11	Reducción Macho-Hembra 1/2 Pulg X 3/4 Pulg	2	18.000
12	Cinta teflón	1	2000
13	Tapón de manguera	3	1.000
14	Adaptador recto dentado de 1/2 pulgada	3	12.000

15	Conexión hembra de manguera o poliducto de 16 mm a 1/2 pulgada	3	12.000
16	Reductor de presión	1	20.000
17	Conector cinta-manguera	1	1.000
18	Conector de manguera en cruz 1/2	1	10.000
		Total:	**181.000**

Nota. Elaboración propia.

Tabla 31

Lista de Materiales Extra que se Usaron en el Proyecto

#	Nombre	Cantidad	Valor
1	Frasco de vidrio con base plana, altura 10 cm	14	56.000
2	Frasco de vidrio con base plana, altura 30 cm	6	60.000
3	Marcadores	2	6.000
4	Arena	12 libras	20.000
5	Arcilla	10 libras	10.000
6	Humus de lombriz	4 kg	15.000
7	Bolsas de sellado fácil	1	10.000
8	Reglas	3	5.000
9	Tijeras	1	3.000
10	Cartulina Negra	2	6.000
11	Set de Cucharas medidoras	2	10.000

12	Etiquetas adhesivas	1	5.000
		Total:	**206.000**

Nota. Elaboración propia.

12.4 Costos

En el desarrollo del proyecto se recurrirá a implementar herramientas de licencia Open Source, es decir, que no requiere pago o suscripción por el uso de ellas, sin embargo, como se describe en el capítulo anterior, es indispensable comprar algunos materiales, por lo cual se estima que la realización del proyecto tendrá un valor aproximado de $10.370.000.

12.5 Financiamiento

Este proyecto se desarrollará con fines investigativos sin ánimo de lucro con el objetivo principal de cumplir con el requisito de grado, por lo cual, la única fuente de financiamiento provendrá de los mismos proponentes del proyecto.

13 Conclusiones y consideraciones finales

Se comprobó la viabilidad de implementar herramientas tecnológicas basadas en Deep Learning para detectar en imágenes de pruebas de campo, las regiones formadas por los minerales de suelo (arena, limo y arcilla); para determinar el tipo de suelo presente en los cultivos.

En futuros trabajos o proyectos que deseen implementar el procesamiento de imágenes mediante Deep Learning, se recomienda aumentar en el dataset con fotografías que abarquen más propiedades de suelos naturales. Dado que, los suelos son variados en sus porcentajes de minerales, en el color y otras propiedades fisicoquímicas que influyen en la visibilidad de una muestra. De igual forma, se recomienda probar otras arquitecturas de red neuronal con diferentes parámetros de entrenamiento.

En el desarrollo de este proyecto se observó la viabilidad de integrar en sistemas de riego componentes de IoT, ya que estos aportan en tiempo real información vital para aplicar y gestionar el riego. Además, se puede integrar con las necesidades del cultivo y el tipo de suelo para una dosificación apropiada teniendo en cuenta cuando, como y cuanto regar.

La metodología que se implementó para definir los sectores de riego en el sistema planteó que el usuario tenga en cuenta si en un lote se presenta variaciones en el tipo se suelo, el emisor de riego o el tipo de cultivo. De esta manera, el usuario debe instalar los componentes individuales de IoT y riego en cada sector. Así, el sistema los podrá monitorear y operar automáticamente, para aplicar un riego diferenciado.

La creación de una plataforma en la que el usuario pueda administrar y gestionar de forma integrada todos los componentes de IoT, la determinación del tipo de suelo mediante Deep Learning, y la información de los diferentes cultivos y riegos. Ayuda a generar un sistema de información que el usuario puede usar para controlar las condiciones en la que encuentra los cultivos y los sistemas riego.

Para futuros trabajos se puede tener en cuenta otros factores para la gestión del riego como la evapotranspiración, la calidad de agua, la topografía entre otras propiedades del suelo y del cultivo, para la aplicar el riego de forma más integral las cuales no se tuvieron en cuenta por las limitaciones económicas y de tiempo.

La implementación de scrum como marco de trabajo agile en el proyecto de grado, nos permitió resolver impedimentos de forma adecuada y rápida. Además, de adoptar cambios de mínimo impacto durante el transcurso de la investigación sin afectar los objetivos. Así como afrontar las diferentes condiciones sociales presentadas durante la pandemia.

14 Referencias

¿Qué es PaaS? - Descripción de plataforma como servicio. (2022). salesforce.com: https://www.salesforce.com/es/learning-centre/tech/paas/

Allen, R. G., Pereira, L. S., Raes, D., & Smith, M. (2006). *Evapotranspiración del cultivo Guías para la determinación de los requerimientos de agua de los cultivos.* Roma: FAO FIAT PANIS.

Alzubaidi, L., Zhang, J., & Humaidi, A. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Big Data*, 8-53.

Ambientum. (s.f.). *Sistemas de riego*. Ambientum · Portal de Medio Ambiente: https://www.ambientum.com/enciclopedia_medioambiental/suelos/sistemas_de_riego.asp

Anotación y etiquetado de imágenes para visión artificial. (2022). shaip.com: https://es.shaip.com/blog/image-annotation-for-computer-vision/

Barrio Andrés, M. (2018). *INTERNET DE LAS COSAS* (Vol. 1). Editorial Reus. https://doi.org/10.30462/9788429020380

Barrio Andrés, M. (2018). Introducción al internet de las cosas. En *Internet de las cosas* (págs. 18-22). Reus.

Boden, M. A. (2016). *Inteligencia Artificial.* España: Turner Publicaciones S.L.

Borja, R., Monleón, A., & Rodellar, J. (2020). Estandarización de métricas de rendimiento para clasificadores Machine y Deep Learning. *Revista Ibérica de sistemas y Tecnologias de la información*, 184-196.

Canal MGAP. (6 de Julio de 2015). Modulo 1: Cálculo de la lámina de riego primer parte [Video]. YouTube. https://youtu.be/hs9edqD3eP8?t=1296

Castro Silva, J. A. (15 de enero de 2016). *Sistema de riego autónomo basado en la Internet de las Cosas [Tesis de master, Universidad Internacional de La Rioja]*.

colab.research.google.com. (2018). *colab.research.google.com*. ¿Qué es Colaboratory?: https://colab.research.google.com/notebooks/intro.ipynb#scrollTo=GJBs_flRovLc

Cristancho, F. (02 de 08 de 2022). *¿Qué es un framework en programación?* talently.tech: https://talently.tech/blog/que-es-un-framework-en-programacion/

D´Aquila, R. (2007). Definición de Sistema Inteligente. *5° Seminario del Ciclo en La Facultad de Ingeniería de la Universidad de Palermo y el Capítulo Argentino de la IEEE Computational Intelligence Society (CIS)* (págs. 1-19). Buenos Aires: A. N. Ingeniería, Anales de la Academia Nacional de Ingeniería.

Duque Correa, J. A., & Villegas Santamaría, J. M. (2018). *Implementación de un sistema automático de riego por goteo autocompensado para la producción de árboles ornamentales y nativos en el vivero forestal de la U. EIA.*

FAO. (23 de septiembre de 2009). *2050: un tercio más de bocas que alimentar*. Organización de las Naciones Unidas para la Alimentación y la Agricultura: https://www.fao.org/news/story/es/item/35675/icode/

FAO RLC. (2008). Factores que se deben considerar para seleccionar el sistema de riego más adecuado. En *El Desarrollo del microrriego en américa central. Oportunidades, Limitaciones y Desafíos* (págs. 27-37). Santiago de Chile.

Fernadez, R. G., Avila, R. A., Lopez, M. L., Gavilan, P., & Oyonarte, N. A. (2010). El agua en el suelo y la planta. Pérdidas de agua. En R. F. Gomez, R. A. Alabarces, M. L. Rodriguez, & P. Gavilan, *Fundamento de riego* (pág. 105). Sevilla: Junta de andalucía.

Fernández, R. (2010). *Manual de riego para agricultores: módulo 4. Riego localizado: manual y ejercicios.* Sevilla: Consejería de Agricultura y Pesca.

Gupta, V., & Raman, S. (2016). Automatic Trimap Generation for Image Matting. *International conference on signal and information processing (IConSIP) IEEE*, 1-5.

Herrera Uribe, E., & Valencia Ayala, L. E. (Mayo de 2007). Del manifiesto ágile sus vares y principios. *Scientia Et Technica, XIII*(34), págs (381-385).

Hossin, M., & Sulaiman, M. (2015). A REVIEW ON EVALUATION METRICS FOR DATA CLASSIFICATION EVALUATIONS. *International Journal of Data Mining & Knowledge Management Process*, 2.

InfoRiego. (s.f.). *Agronomía del Riego*. InfoRiego: https://www.inforiego.org/opencms/opencms/info_tecnica/6_agronomia/index.html

Kwok, J., & Sun, Y. (2018). A smart IoT-based irrigation system with automated plant recognition using deep learning. *ACM International Conference Proceeding Series*, 87-91.

Mañas, F. M., Fuster, P. L., & Belmote, A. C. (2005). Agua y agronomia. En A. C. Belmonte, *La evapotranspiración: concepto y metodología de cálculo* (págs. 165-167). Madrid: Mundi-Prensa Libros.

Martin, E. C., & Muñoz, C. (2017). Métodos para Medir la Humedad del Suelo para la Programación del Riego ¿Cuándo? *College of Agriculture, University of Arizona* .

Molnar, C. (2020). Interpretable Machine Learning A Guide for Making Black Box Models Explainable. En C. Molnar, *Interpretable Machine Learning A Guide for Making Black Box Models Explainable* (págs. 13-15). Morrisville: Lulu.com.

Morais, P. A., Souza, D. M., Carvalho, M. T., Madari, B. E., & Oliveira, A. E. (2019). Predicting Soil Texture Using Image Analysis. *Microchemical Journal*, 455-463.

Panqu, W., Chen, P., Yuan, Y., Liu, D., Huang, Z., Hou, X., & Cottrell, G. (2018). Understanding Convolution for Semantic Segmentation. *IEEE Winter Conference on Applications of Computer Vision*, 1451-1460.

Peluffo Ordóñez, D. H., Anaya Isaza, A. J., Rios, J. I., Castro Silva, J. A., Carvajal Ruiz, D. A., & Espinosa Llanos, L. H. (noviembre de 2017). *Sistema de Riego Basado En La Internet De Las Cosas (IoT).* ResearchGate: https://www.researchgate.net/publication/315793360_Sistema_de_Riego_Basado_En_La_Internet_De_Las_Cosas_IoT

Peña, C. M. (2020). *Descubriendo Arduino.* Buenos Aires: RedUSERS.

Redhat. (26 de Agosto de 2022). *¿Qué es una PaaS?* redhat.com: https://www.redhat.com/es/topics/cloud-computing/what-is-paas

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *Lecture Notes in Computer Science, 9351*, págs (234-241). https://doi.org/10.1007/978-3-319-24574-4_28

Rosebrock, A. (2017). *Deep Learning for Computer Vision with Python.* PYIMAGESEARCH.

Rouhiainen, L. P. (2018). *Inteligencia artificial 101 cosas que debes saber hoy sobre nuestro futuro.* Barcelona: Planeta, S.A.

Salazar, J., & Silvestre, S. (2016). Internet de las cosas. *TechPedia,České vysoké učení technické v Praze Fakulta elektrotechnicka*, 7-8.

Salazar, J., & Silvestre, S. (2016). Internet de las cosas. *TechPedia,České vysoké učení technické v Praze Fakulta elektrotechnicka*.

Sanchez, M. I. (2000). Características y apreciaciones generales de los métodos de medida y estimación de la evapotranspiración. *Geografía Norte Grande,* 27-36.

Santos, P. R. (5 de febrero de 2020). *Las 5 razones por las que todo el mundo quiere aprender Python*. https://empresas.blogthinkbig.com/: https://empresas.blogthinkbig.com/las-5-razones-por-las-que-todo-el-mundo-quiere-aprender-python/

Shamir, R. R., Duchin, Y., Kim, J., Sapiro, G., & Harel, N. (2018). Continuous Dice Coefficient: a Method for Evaluating Probabilistic Segmentations. *bioRxiv*.

Shaxson, F., & Barber, R. (2005). Glosario de términos sobre humedd del suelo. En C. d. FAO, *Optimización de la humedad del suelo para la producción vegetal.* Roma: Organización de las Naciones Unidas para la Agricultura y la Alimentación.

Tiu, E. (9 de Agosto de 2019). *Metrics to Evaluate your Semantic Segmentation Model*. towardsdatascience.com: https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2

Torres, J. (8 de Septiembre de 2019). *Data Augmentation y Transfer Learning*. torres.ai: https://torres.ai/data-augmentation-y-transfer-learning-en-keras-tensorflow/

Wang, Y., Huang, H., Rudin, C., & Shaposhnik, Y. (2021). Understanding How Dimension Reduction Tools Work: An Empirical Approach to Deciphering t-SNE, UMAP, TriMap, and PaCMAP for Data Visualization. *Journal of Machine Learning Research* , 1-73.

Warmuth, M. K., & Amid, E. (2019). TriMap: Large-scale Dimensionality Reduction Using Triplets. *arXiv preprint*.

Zapata, O. E., & Zapata, F. B. (2011). *Microcontroladores PIC con programación PBP.* Madrid: RA-MA.

Zhao, M., Fu, C.-W., Cai, J., & Cham, T.-J. (2015). Real-Time and Temporal-Coherent Foreground Extraction With Commodity RGBD Camera. *IEEE Journal of Selected Topics in Signal Processing*, 9.

Zotarelli, L., Dukes, M. D., & Morgan, K. T. (2019). Interpretación del contenido de la humedad del suelo para determinar capacidad de campo y evitar riego excesivo en suelos arenosos utilizando sensores de humedad. *UF-IFAS extension Agricultural and biological engineering*, 1.

15 Anexos

15.1 IOT

Figura 55

Conexión Electroválvula Arduino

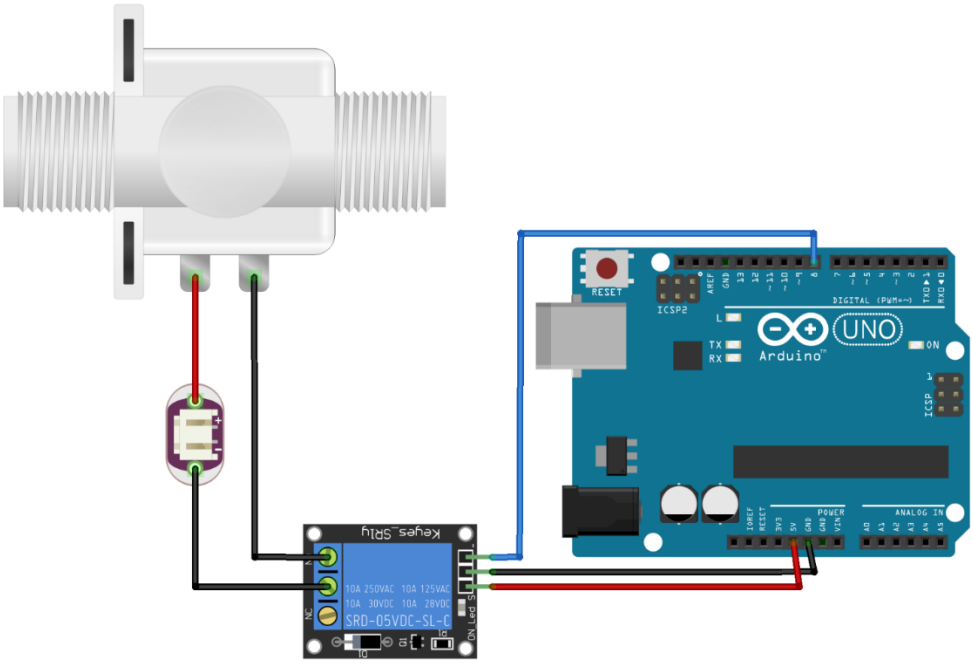


Figura 56

Conexión Modulo Radiofrecuencia Arduino

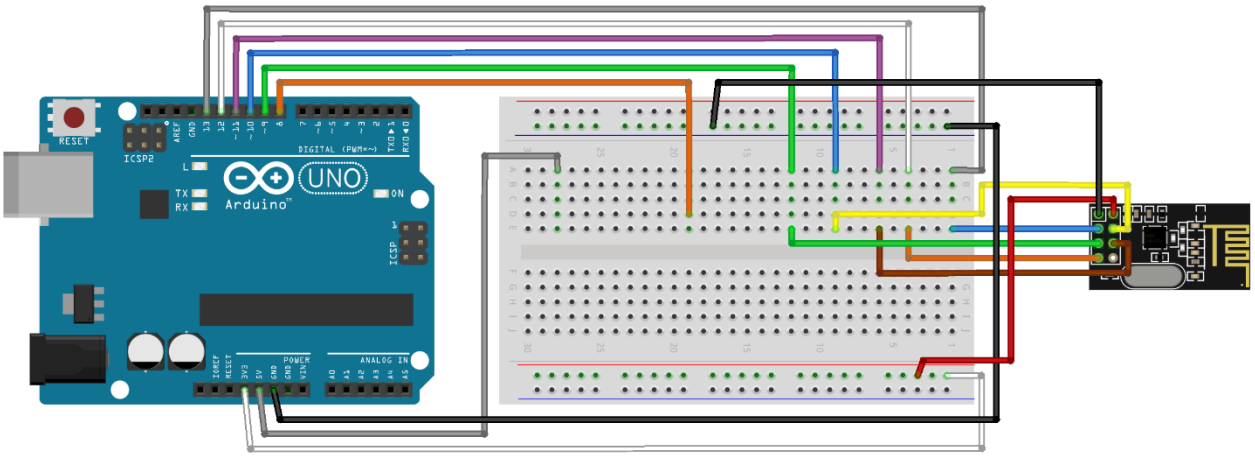
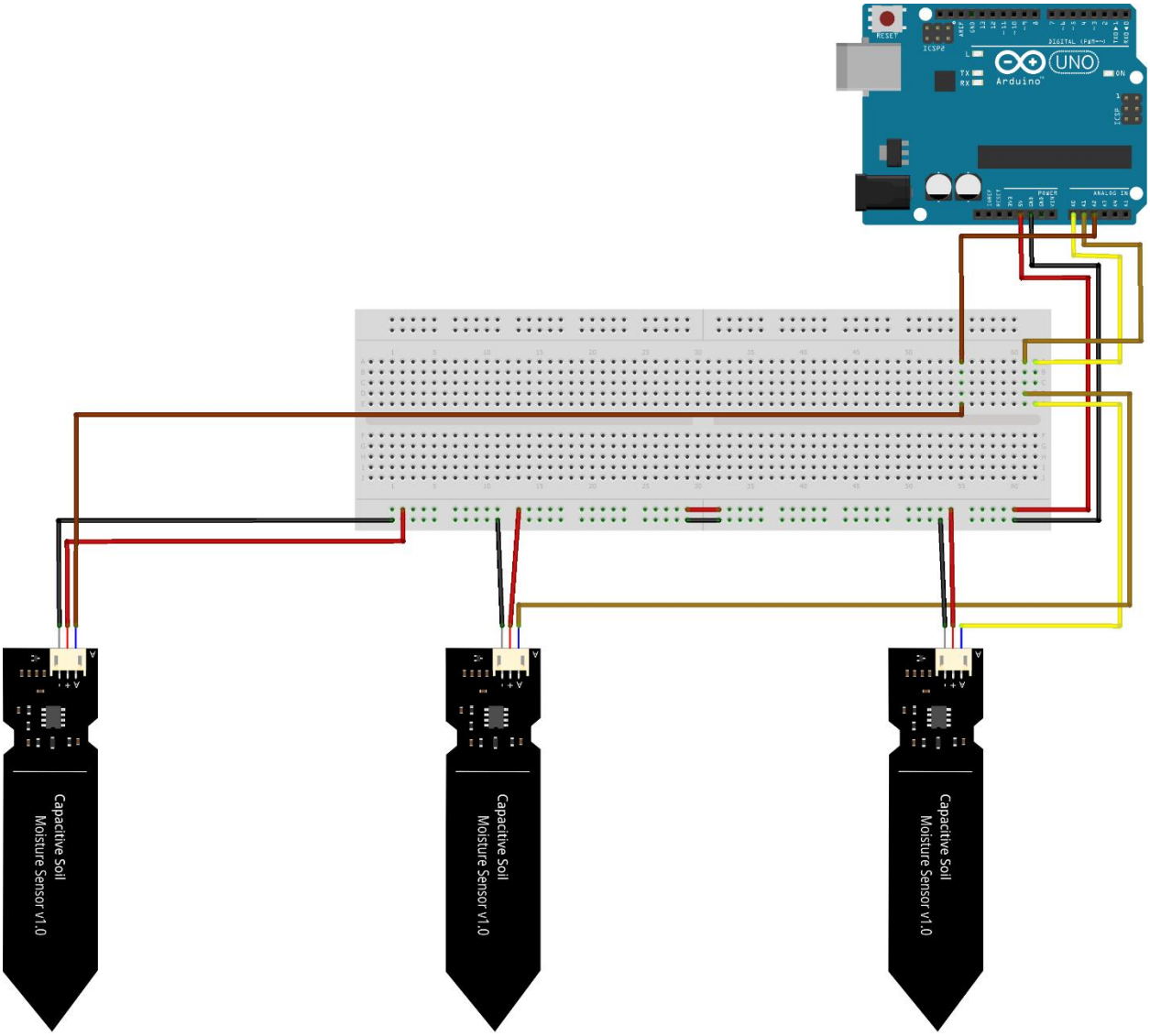


Figura 57

Conexión Sensor Humedad Capacitivo Arduino



15.2 Ingeniería de software

15.2.1 Pruebas Unitarias

```python
class CultivoDaoTest(unittest.TestCase):
    def test_get_cultivos(self):
        self.assertIsNotNone(CultivoDao.seleccionarTodos())
    def test_get_cultivos_by_user(self):
        cultivo = Cultivo(user=25)
        self.assertIsNotNone(CultivoDao.buscarPorUsuario(cultivo))
    def test_get_cultivo_by_id(self):
        cultivo = Cultivo(id=2)
        self.assertIsNotNone(CultivoDao.buscarPorId(cultivo))
    def test_get_cultivo_by_id_not_valid(self):
        cultivo = Cultivo(id=3)
        self.assertIsNone(CultivoDao.buscarPorId(cultivo))
    def test_delete_cultivo_by_id(self):
        cultivo = Cultivo(id=29)
        self.assertEqual(CultivoDao.eliminar(cultivo),1)
    def test_insert_cultivo(self):
        cultivo = Cultivo(cultivoNombre="manzana", tipoCultivo=1,
                          fechaInicio="2022-01-01", fechaFinal="2023-01-01", cultivoEstado=1, user=25)
        self.assertEqual(CultivoDao.insertar(cultivo),1)
    def test_actualizar_cultivo(self):
        cultivo = Cultivo(cultivoNombre="manzana", tipoCultivo=1,
                          fechaInicio="2022-01-01", fechaFinal="2023-01-01", cultivoEstado=1, user=25,id=26)
        self.assertEqual(CultivoDao.actualizar(cultivo),1)
```

```
(riego) PS C:\Users\LUISFERNANDO\Documents\proyecto-code\RiegoFlask> python test\dao\CultivoDaoTest.py -v
test_actualizar_cultivo (__main__.CultivoDaoTest) ... 09:28:13 PM: DEBUG [cultivo_dao.py:89] actualizar cultivo,
            Id cultivo: 26, Nombre cultivo: manzana ,
            tipo cultivo: 1, Fecha inicio: 2022-01-01, fecha final: 2023-01-01, Estado : 1

ok
test_delete_cultivo_by_id (__main__.CultivoDaoTest) ... 09:28:14 PM: DEBUG [cultivo_dao.py:73] eliminar cultivo,
            Id cultivo: 29, Nombre cultivo: None ,
            tipo cultivo: None, Fecha inicio: None, fecha final: None, Estado : None

ok
test_get_cultivo_by_id (__main__.CultivoDaoTest) ... ok
test_get_cultivo_by_id_not_valid (__main__.CultivoDaoTest) ... ok
test_get_cultivos (__main__.CultivoDaoTest) ... ok
test_get_cultivos_by_user (__main__.CultivoDaoTest) ... ok
test_insert_cultivo (__main__.CultivoDaoTest) ... 09:28:16 PM: DEBUG [cultivo_dao.py:81] insertar cultivo,
            Id cultivo: None, Nombre cultivo: manzana ,
            tipo cultivo: 1, Fecha inicio: 2022-01-01, fecha final: 2023-01-01, Estado : 1

ok

----------------------------------------------------------------------
Ran 7 tests in 4.720s

OK
```

```python
class FincaDaoTest(unittest.TestCase):
    def test_get_fincas(self):
        self.assertIsNotNone(FincaDao.seleccionarTodos())
    def test_get_finca_by_user(self):
        finca = Finca(usuario=25)
        self.assertIsNotNone(FincaDao.buscarFincaPorUsuario(finca))
    def test_get_finca_by_id(self):
        finca = Finca(id=2)
        self.assertIsNotNone(FincaDao.buscarFincaPorId(finca))
    def test_get_finca_by_id_not_valid(self):
        finca = Finca(id=90)
        self.assertIsNone(FincaDao.buscarFincaPorId(finca))
    def test_delete_finca_by_id(self):
        finca = Finca(id=11)
        self.assertEqual(FincaDao.eliminar(finca),1)
    def test_insert_finca(self):
        finca = Finca(nombre="eden 2", direccion="cr 31 sur ",
                      latitud=2.883137459182263, longitud=-75.26639088403925, altitud=0, usuario=25)
        self.assertEqual(FincaDao.insertar(finca),1)
    def test_actualizar_finca(self):
        finca = Finca(nombre="eden 2", direccion="cr 31 sur ",
                      latitud=2.883137459182263, longitud=-75.26639088403925, altitud=0, usuario=25, id=4)
        self.assertEqual(FincaDao.actualizar(finca),1)
```

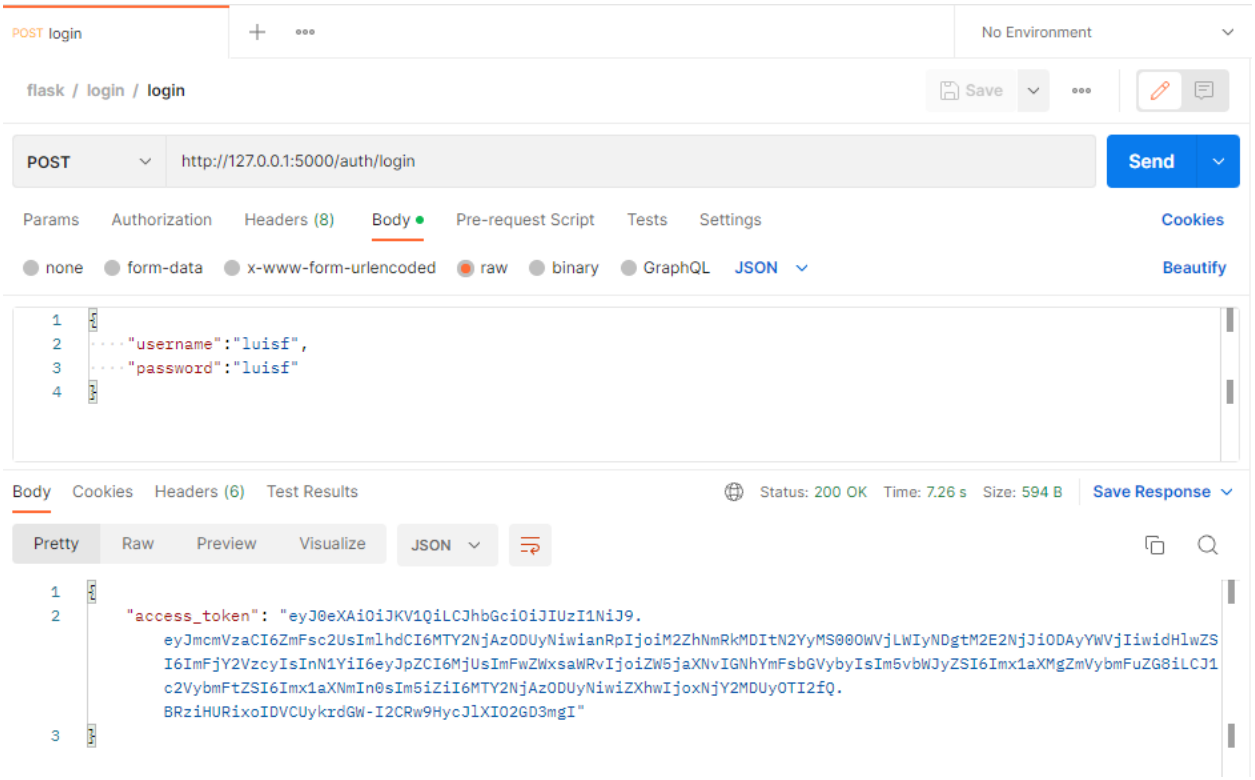
```
(riego) PS C:\Users\LUISFERNANDO\Documents\proyecto-code\RiegoFlask> python test\dao\FincaDaoTest.py -v
test_actualizar_finca (__main__.FincaDaoTest) ... 04:14:24 PM: DEBUG [finca_dao.py:81] actualizar finca,
            Id finca: 4, nombre: eden 2 , direccion : cr 31 sur ,
            altitud: 0, longitud: -75.26639088403925, latitud: 2.883137459182263, usuario: 25

ok
test_delete_finca_by_id (__main__.FincaDaoTest) ... 04:14:24 PM: DEBUG [finca_dao.py:65] eliminar finca,
            Id finca: 11, nombre: None , direccion : None,
            altitud: None, longitud: None, latitud: None, usuario: None

ok
test_get_finca_by_id (__main__.FincaDaoTest) ... ok
test_get_finca_by_id_not_valid (__main__.FincaDaoTest) ... ok
test_get_finca_by_user (__main__.FincaDaoTest) ...
            Id finca: 1, nombre: eden , direccion : cr 31 sur ,
            altitud: 0.0, longitud: -75.2663908840393, latitud: 2.88313745918226, usuario: 25


            Id finca: 2, nombre: eden 2 , direccion : cr 31 sur ,
            altitud: 0.0, longitud: -75.2663908840393, latitud: 2.88313745918226, usuario: 25
```

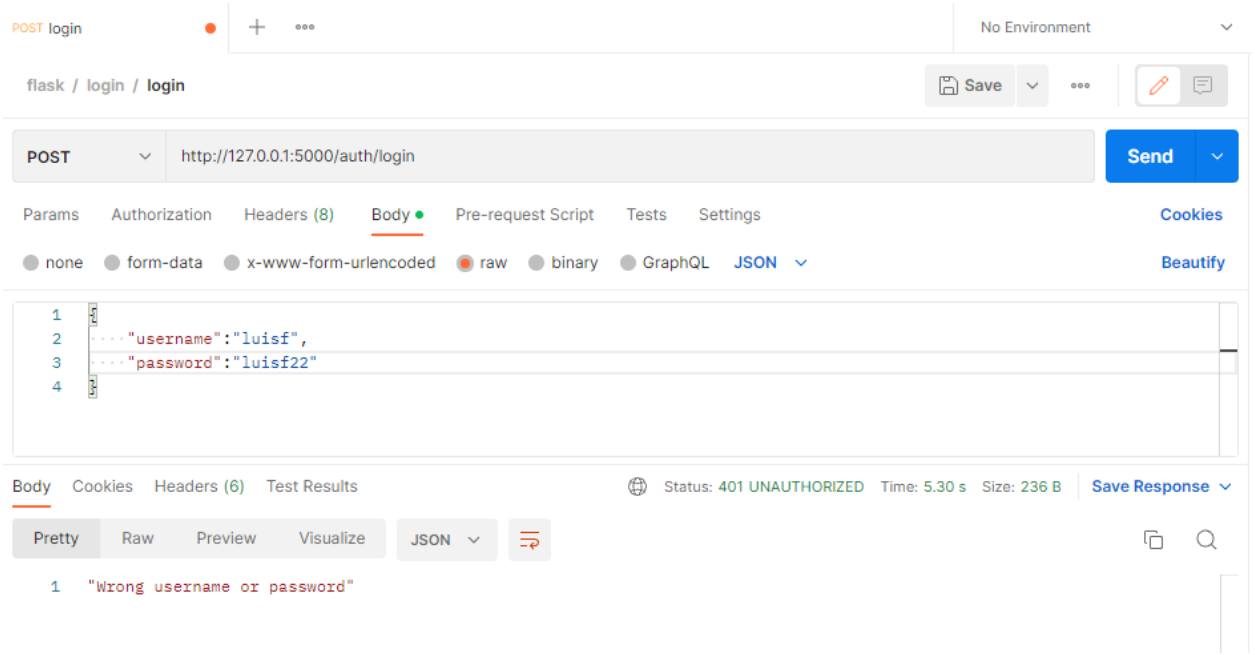
```
            Id finca: 12, nombre: eden 2 , direccion : cr 31 sur ,
            altitud: 0.0, longitud: -75.2663908840393, latitud: 2.88313745918226, usuario: 25

ok
test_insert_finca (__main__.FincaDaoTest) ... 04:14:26 PM: DEBUG [finca_dao.py:73] insertar finca,
            Id finca: None, nombre: eden 2 , direccion : cr 31 sur ,
            altitud: 0, longitud: -75.26639088403925, latitud: 2.883137459182263, usuario: 25

ok

----------------------------------------------------------------------
Ran 7 tests in 2.804s

OK
```

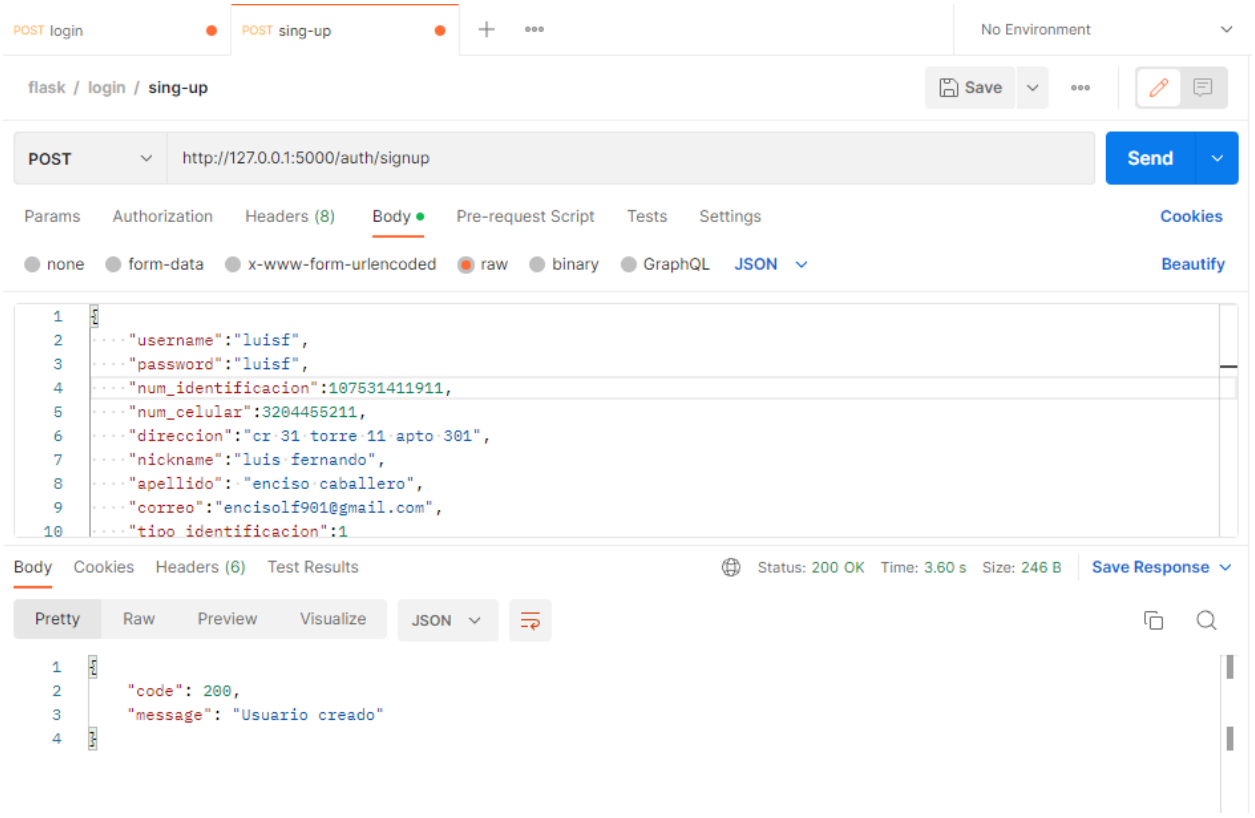
```python
class LoteDaoTest(unittest.TestCase):
    def test_get_lotess(self):
        self.assertIsNotNone(LotesDao.seleccionarTodos())
    def test_get_lotes_by_finca(self):
        lote = Lote(finca=1)
        self.assertIsNotNone(LotesDao.buscarPorFinca(lote))
    def test_get_lote_by_id(self):
        lote = Lote(id=2)
        self.assertIsNotNone(LotesDao.buscarPorId(lote))
    def test_get_lote_by_id_not_valid(self):
        lote = Lote(id=15)
        self.assertIsNone(LotesDao.buscarPorId(lote))
    def test_delete_lote_by_id(self):
        lote = Lote(id=7)
        self.assertEqual(LotesDao.eliminar(lote),1)
    def test_insert_lote(self):
        lote = Lote(nombre="lot-07", area=12,
                    latitud=2.883137459182263, longitud=-75.26639088403925, altitud=0, finca=1)
        self.assertEqual(LotesDao.insertar(lote),1)
    def test_actualizar_lote(self):
        lote = Lote(nombre="lot-07", area=12,
                    latitud=2.883137459182263, longitud=-75.26639088403925, altitud=0, finca=1, id=6)
        self.assertEqual(LotesDao.actualizar(lote),1)
```

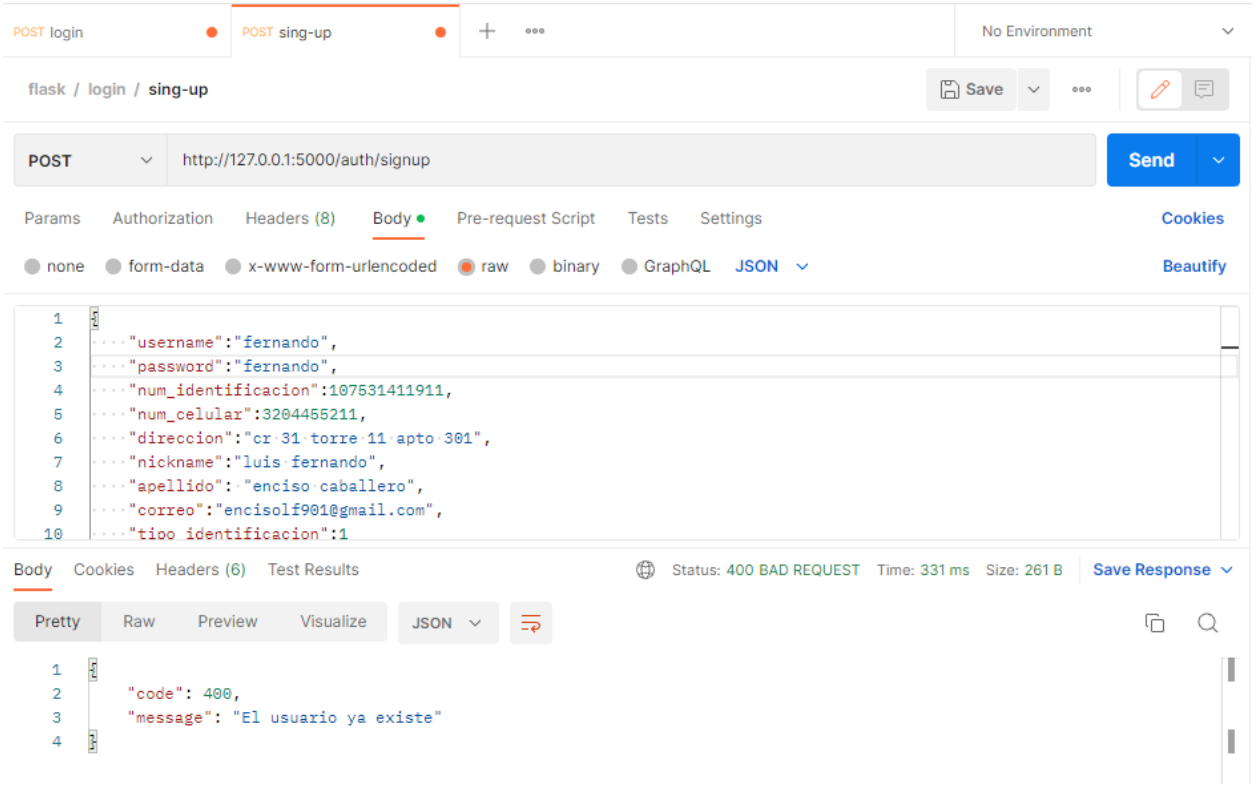
```
(riego) PS C:\Users\LUISFERNANDO\Documents\proyecto-code\RiegoFlask> python test\dao\LoteDaoTest.py -v
test_actualizar_lote (__main__.LoteDaoTest) ... 04:45:11 PM: DEBUG [lotes_dao.py:87] actualizar lote,
            Id lote: 6, nombre: lot-07 , finca : 1,
            altitud: 0, longitud: -75.26639088403925, latitud: 2.883137459182263, area: 12

ok
test_delete_lote_by_id (__main__.LoteDaoTest) ... 04:45:11 PM: DEBUG [lotes_dao.py:69] eliminar lote,
            Id lote: 7, nombre: None , finca : None,
            altitud: None, longitud: None, latitud: None, area: None

ok
test_get_lote_by_id (__main__.LoteDaoTest) ... ok
test_get_lote_by_id_not_valid (__main__.LoteDaoTest) ... ok
test_get_lotes_by_finca (__main__.LoteDaoTest) ...
            Id lote: 3, nombre: lot-02-f1 , finca : 1,
            altitud: 0.0, longitud: -75.2663908840393, latitud: 2.88313745918226, area: 1.0


            Id lote: 1, nombre: lot-01 , finca : 1,
            altitud: 0.0, longitud: -75.2663908840393, latitud: 2.88313745918226, area: 1.0



ok
test_insert_lote (__main__.LoteDaoTest) ... 04:45:13 PM: DEBUG [lotes_dao.py:78] insertar lote,
             Id lote: None, nombre: lot-07 , finca : 1,
             altitud: 0, longitud: -75.26639088403925, latitud: 2.883137459182263, area: 12

ok

----------------------------------------------------------------------
Ran 7 tests in 3.294s

OK
```

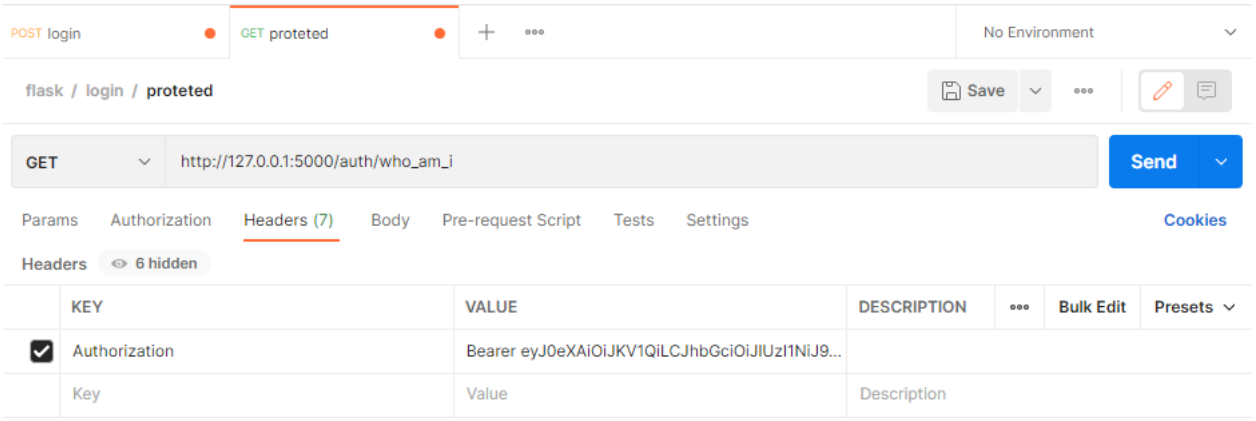
15.2.2 Pruebas API REST

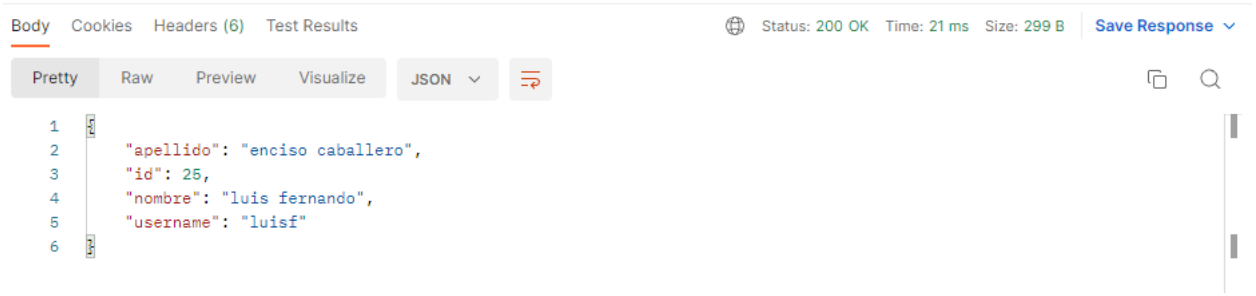

POST login
No Environment
flask / login / login
Save
POST
http://127.0.0.1:5000/auth/login
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"username":"luisf",
"password":"luisf"
}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 7.26 s
Size: 594 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.
eyJmcmVzaCI6ZmFsc2UsImlhdCI6MTY2NjAzODUyNiwianRpIjoiM2ZhNmRkMDItN2YyMS00OWVjLWIyNDgtM2E2NjJiODAyYWVjIiwidHlwZSI6ImFjY2VzcyIsInN1YiI6eyJpZCI6MjUsImFwZWxsaWRvIjoiZW5jaXNvIGNhYmFsbGVybyIsIm5vbWJyZSI6Imx1aXMgZmVybmFuZG8iLCJ1c2VybmFtZSI6Imx1aXNmIn0sIm5iZiI6MTY2NjAzODUyNiwiZXhwIjoxNjY2MDUyOTI2fQ.
BRziHURixoIDVCUykrdGW-I2CRw9HycJlXIO2GD3mgI"
}

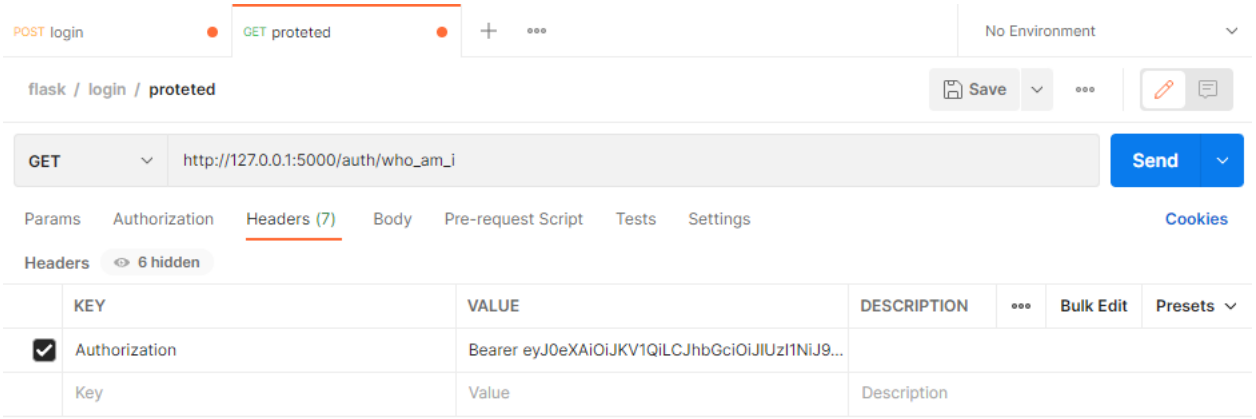

POST login
No Environment
flask / login / login
Save
POST
http://127.0.0.1:5000/auth/login
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"username":"luisf",
"password":"luisf22"
}
Body
Cookies
Headers (6)
Test Results
Status: 401 UNAUTHORIZED
Time: 5.30 s
Size: 236 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
"Wrong username or password"

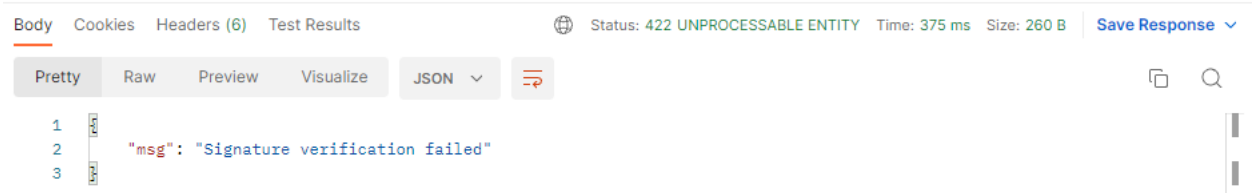

POST login
POST sing-up
No Environment
flask / login / sing-up
Save
POST
http://127.0.0.1:5000/auth/signup
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"username":"luisf",
"password":"luisf",
"num_identificacion":107531411911,
"num_celular":3204455211,
"direccion":"cr 31 torre 11 apto 301",
"nickname":"luis fernando",
"apellido": "enciso caballero",
"correo":"encisolf901@gmail.com",
"tipo_identificacion":1
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 3.60 s
Size: 246 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 200,
"message": "Usuario creado"
}

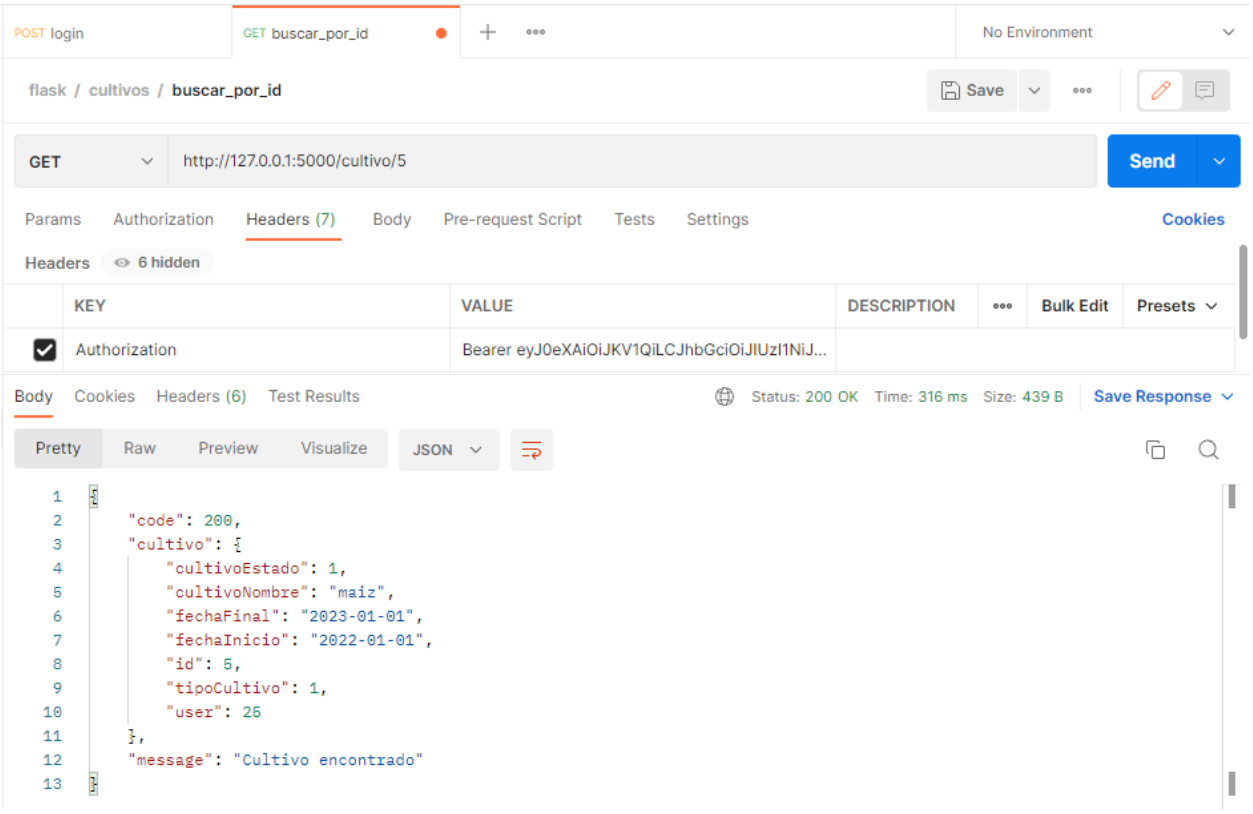

POST login
POST sing-up
No Environment
flask / login / sing-up
Save
POST
http://127.0.0.1:5000/auth/signup
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"username":"fernando",
"password":"fernando",
"num_identificacion":107531411911,
"num_celular":3204455211,
"direccion":"cr 31 torre 11 apto 301",
"nickname":"luis fernando",
"apellido": "enciso caballero",
"correo":"encisolf901@gmail.com",
"tipo_identificacion":1
Body
Cookies
Headers (6)
Test Results
Status: 400 BAD REQUEST
Time: 331 ms
Size: 261 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 400,
"message": "El usuario ya existe"
}

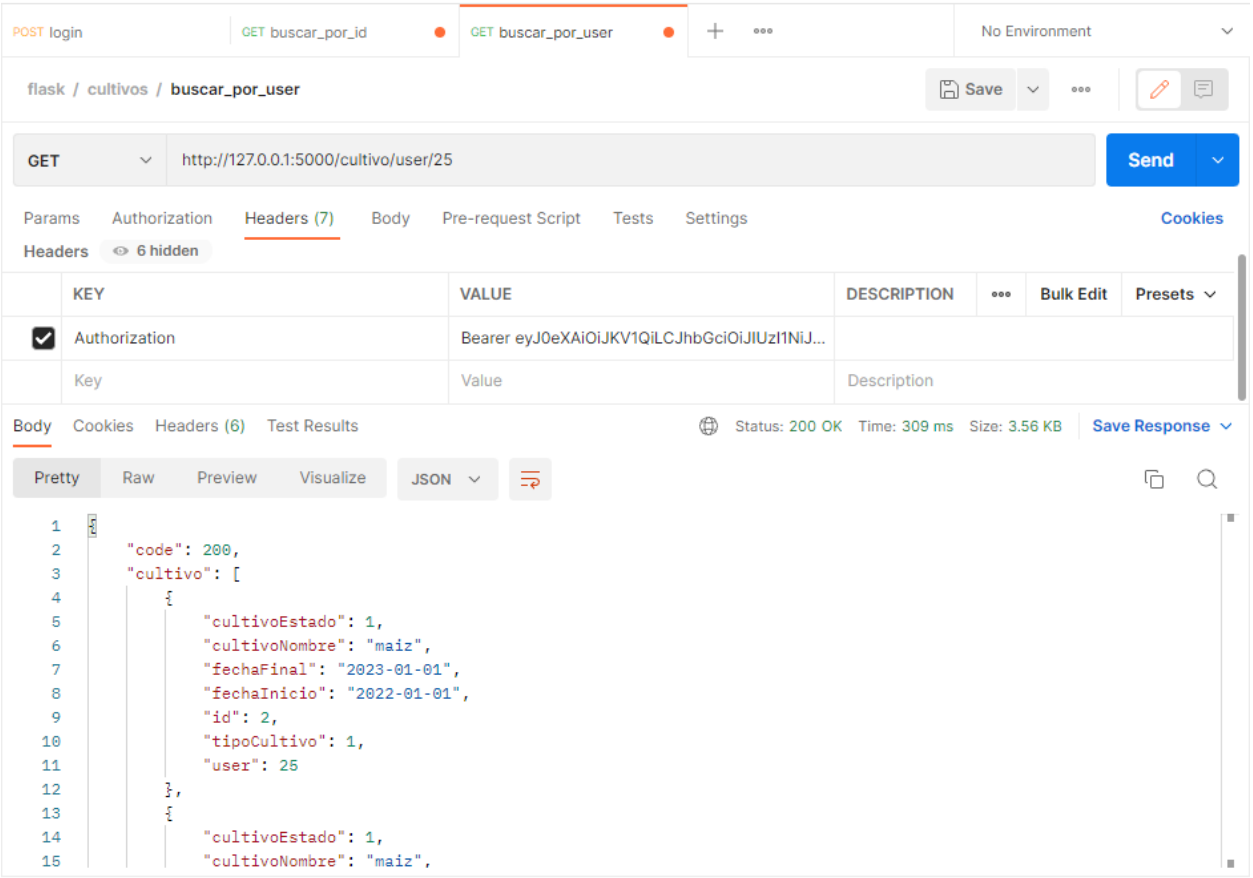

POST login
GET proteted
No Environment
flask / login / proteted
Save
GET
http://127.0.0.1:5000/auth/who_am_i
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
Key
Value
Description

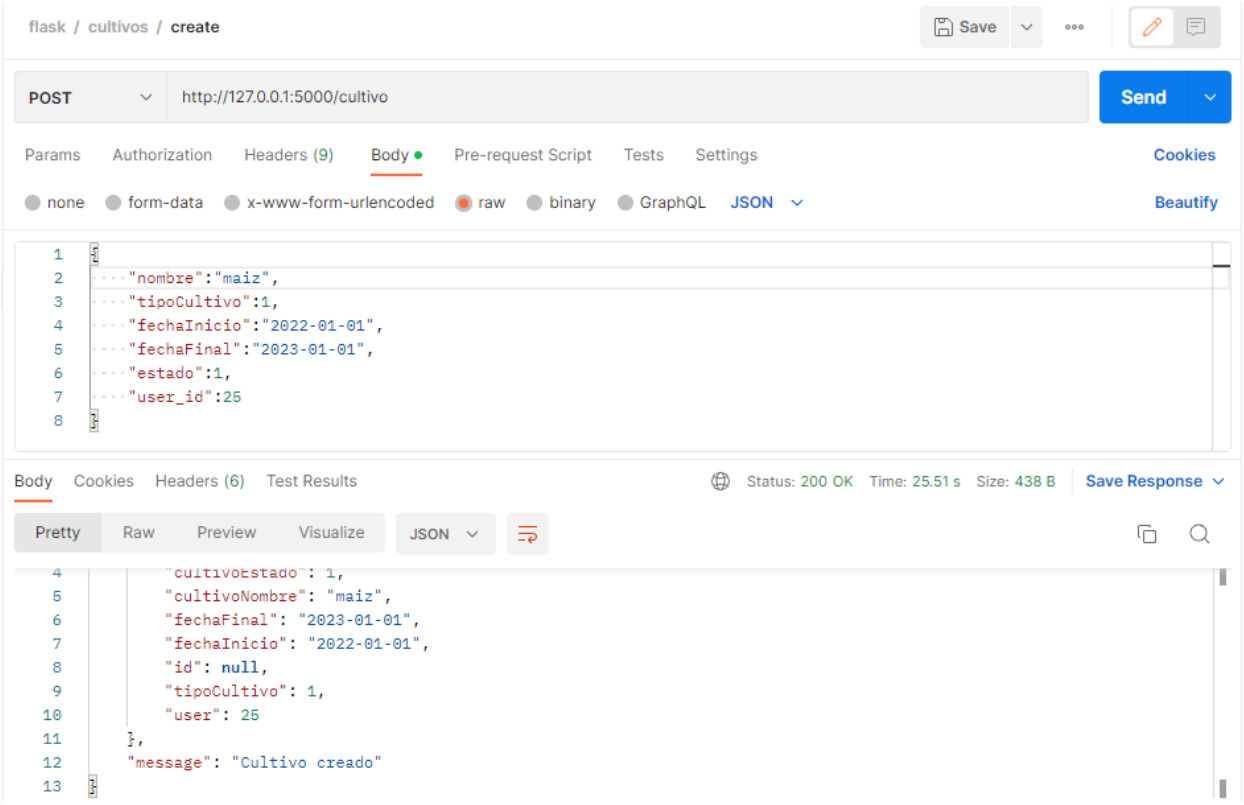

Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 21 ms
Size: 299 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"apellido": "enciso caballero",
"id": 25,
"nombre": "luis fernando",
"username": "luisf"
}

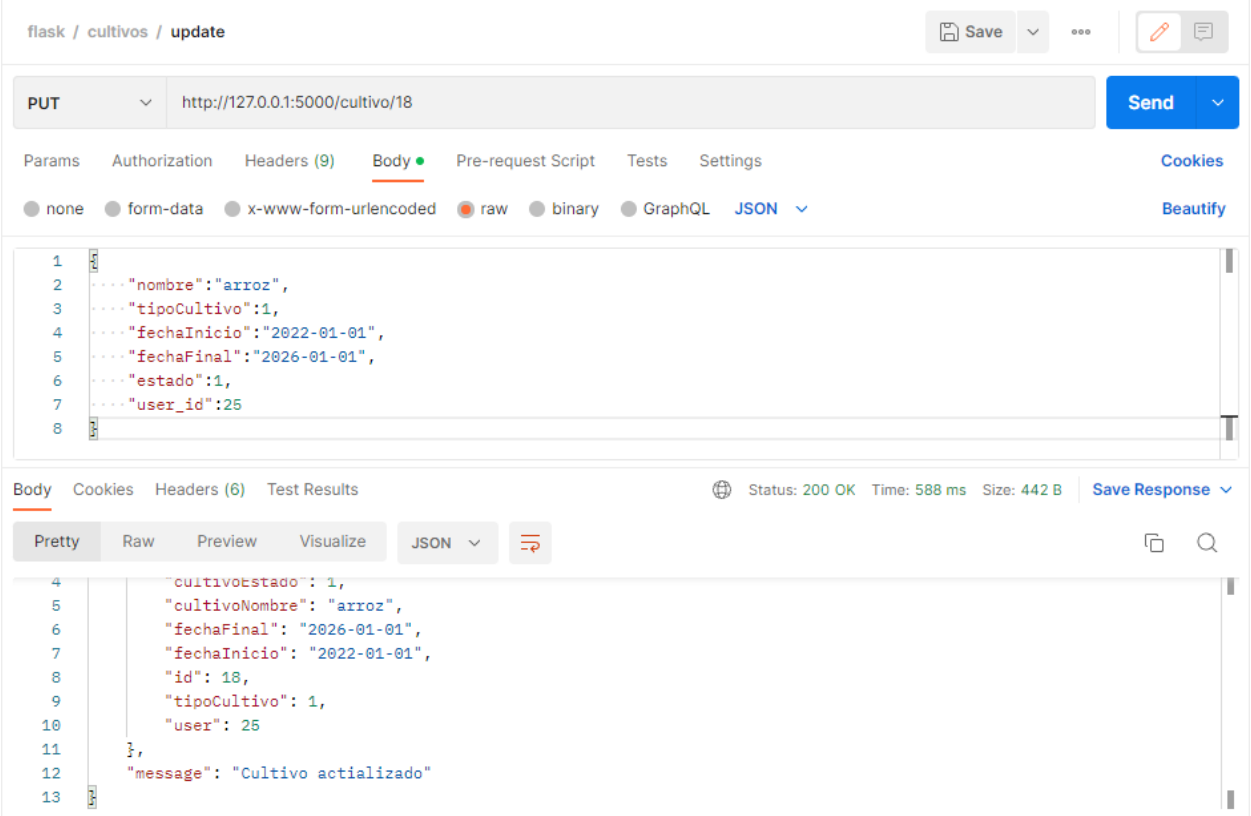

POST login
GET proteted
No Environment
flask / login / proteted
Save
GET
http://127.0.0.1:5000/auth/who_am_i
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
Key
Value
Description

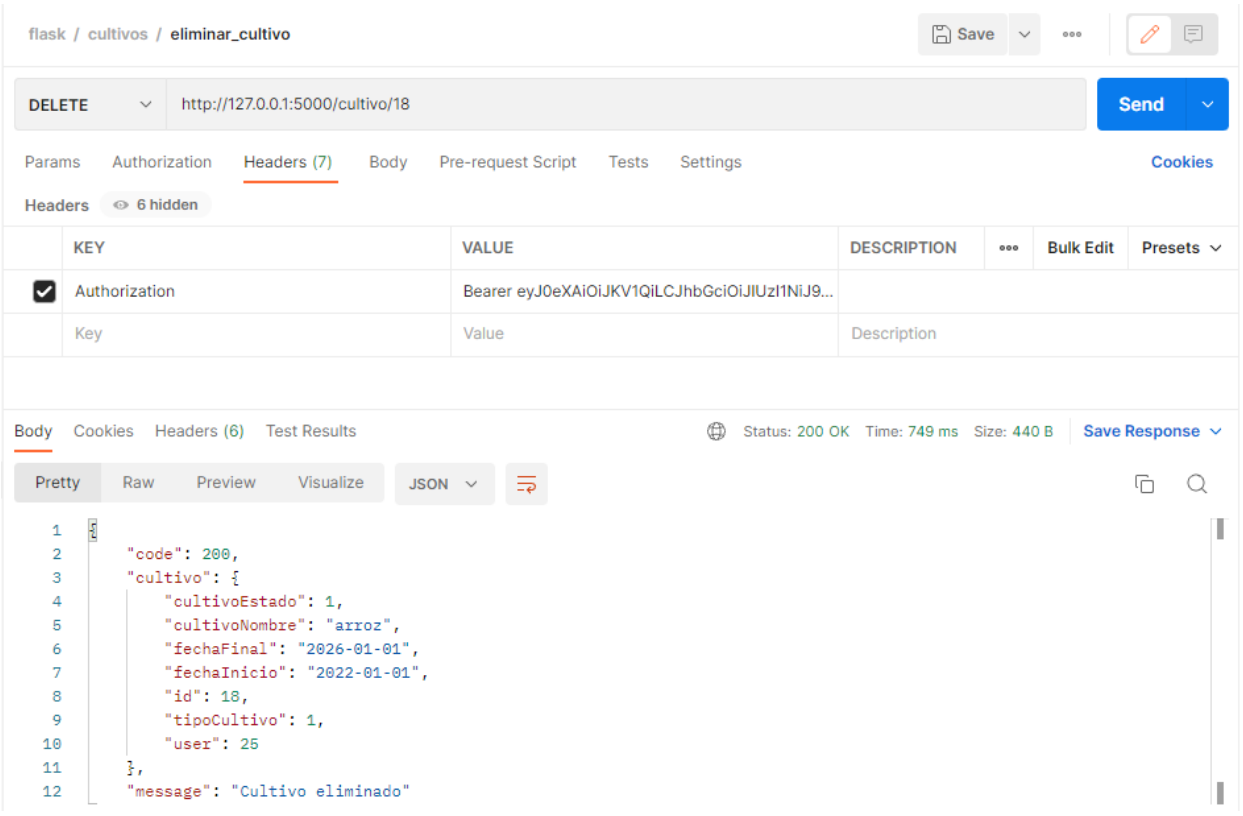

Body
Cookies
Headers (6)
Test Results
Status: 422 UNPROCESSABLE ENTITY
Time: 375 ms
Size: 260 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"msg": "Signature verification failed"
}

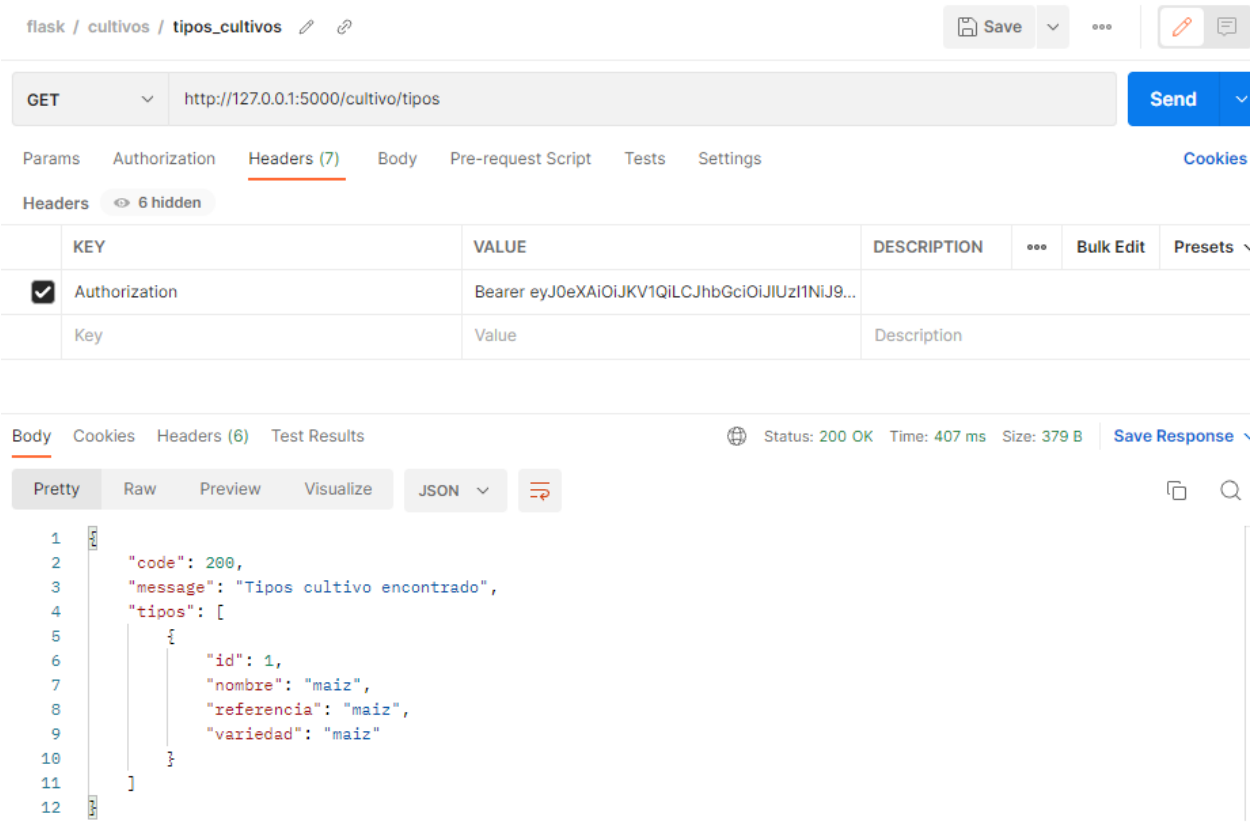

POST login
GET buscar_por_id
No Environment
flask / cultivos / buscar_por_id
Save
GET
http://127.0.0.1:5000/cultivo/5
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ...
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 316 ms
Size: 439 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 200,
"cultivo": {
"cultivoEstado": 1,
"cultivoNombre": "maiz",
"fechaFinal": "2023-01-01",
"fechaInicio": "2022-01-01",
"id": 5,
"tipoCultivo": 1,
"user": 25
},
"message": "Cultivo encontrado"
}

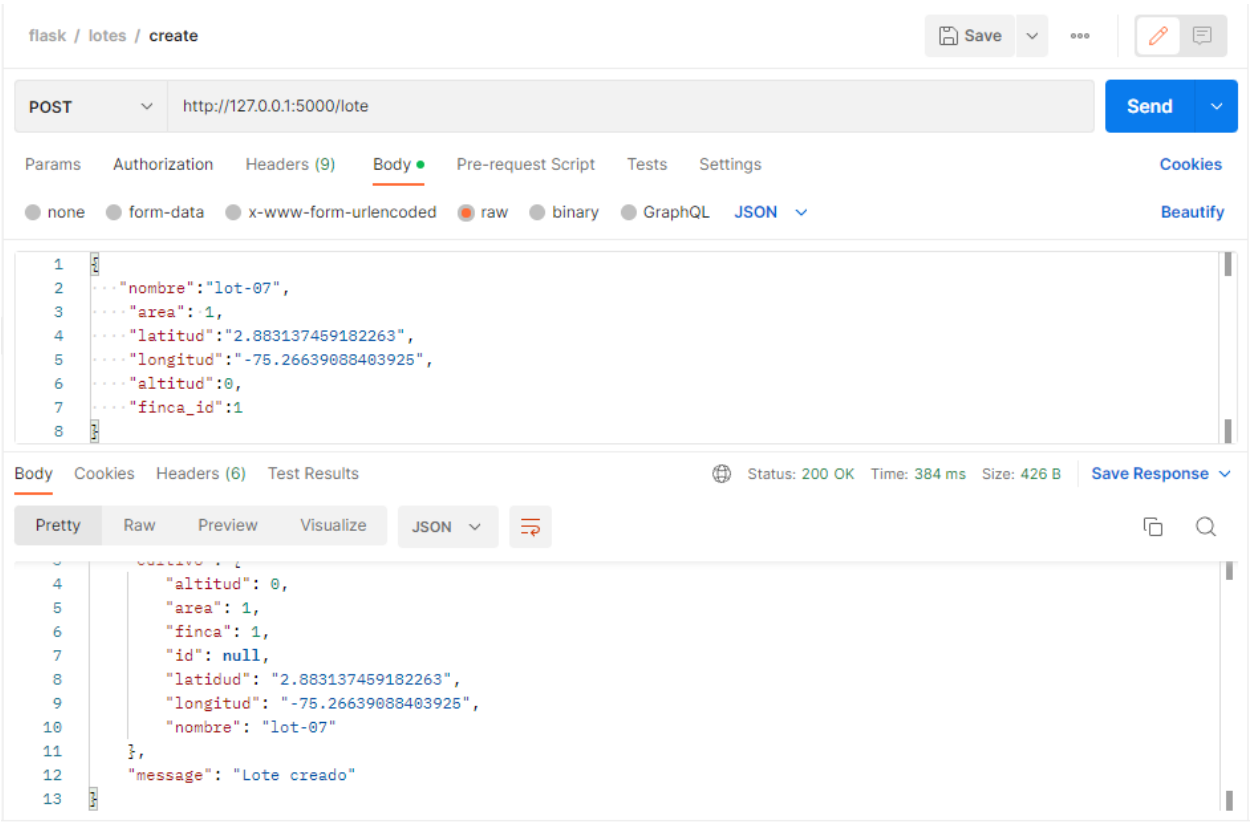

POST login
GET buscar_por_id
GET buscar_por_user
No Environment
flask / cultivos / buscar_por_user
Save
GET
http://127.0.0.1:5000/cultivo/user/25
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ...
Key
Value
Description
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 309 ms
Size: 3.56 KB
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 200,
"cultivo": [
{
"cultivoEstado": 1,
"cultivoNombre": "maiz",
"fechaFinal": "2023-01-01",
"fechaInicio": "2022-01-01",
"id": 2,
"tipoCultivo": 1,
"user": 25
},
{
"cultivoEstado": 1,
"cultivoNombre": "maiz",

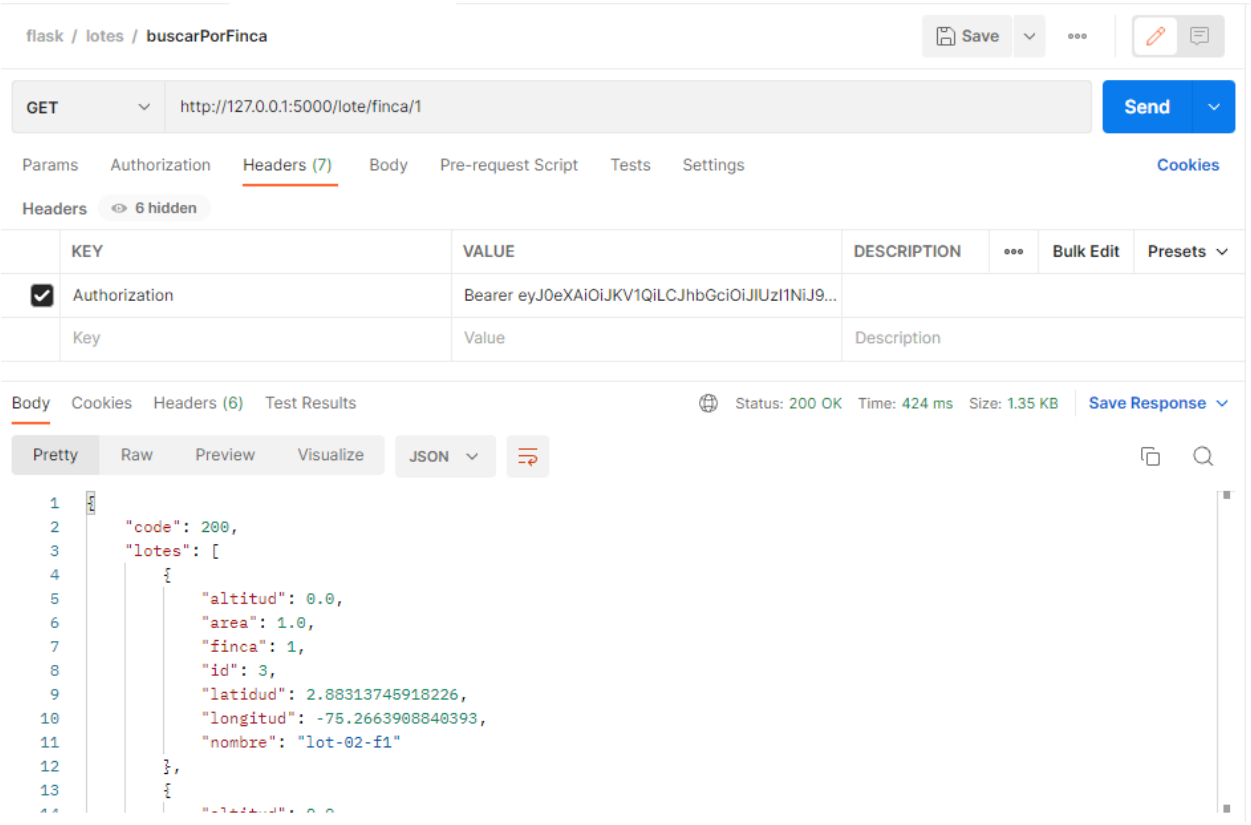

flask / cultivos / create
Save
POST
http://127.0.0.1:5000/cultivo
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"nombre":"maiz",
"tipoCultivo":1,
"fechaInicio":"2022-01-01",
"fechaFinal":"2023-01-01",
"estado":1,
"user_id":25
}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 25.51 s
Size: 438 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
"cultivoEstado": 1,
"cultivoNombre": "maiz",
"fechaFinal": "2023-01-01",
"fechaInicio": "2022-01-01",
"id": null,
"tipoCultivo": 1,
"user": 25
},
"message": "Cultivo creado"
}

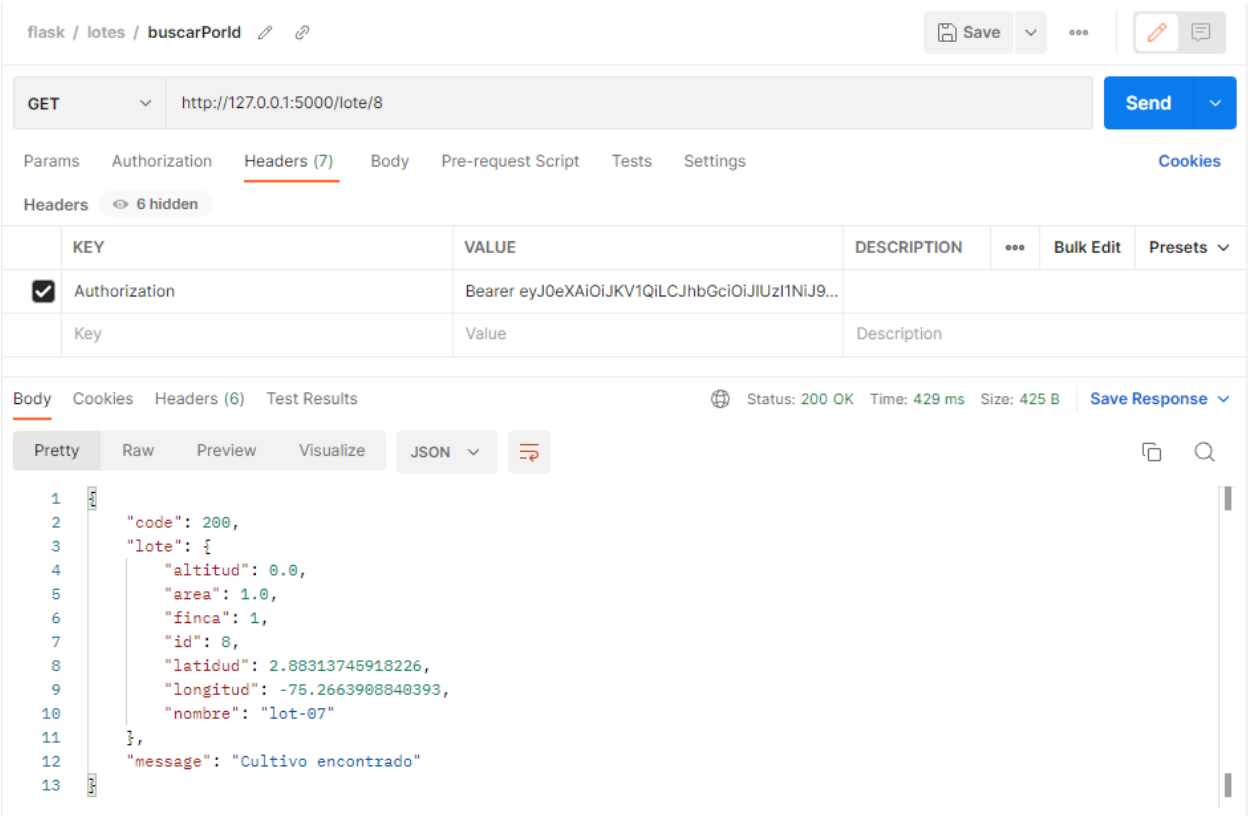

flask / cultivos / update
Save
PUT
http://127.0.0.1:5000/cultivo/18
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"nombre":"arroz",
"tipoCultivo":1,
"fechaInicio":"2022-01-01",
"fechaFinal":"2026-01-01",
"estado":1,
"user_id":25
}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 588 ms
Size: 442 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
"cultivoEstado": 1,
"cultivoNombre": "arroz",
"fechaFinal": "2026-01-01",
"fechaInicio": "2022-01-01",
"id": 18,
"tipoCultivo": 1,
"user": 25
},
"message": "Cultivo actializado"
}

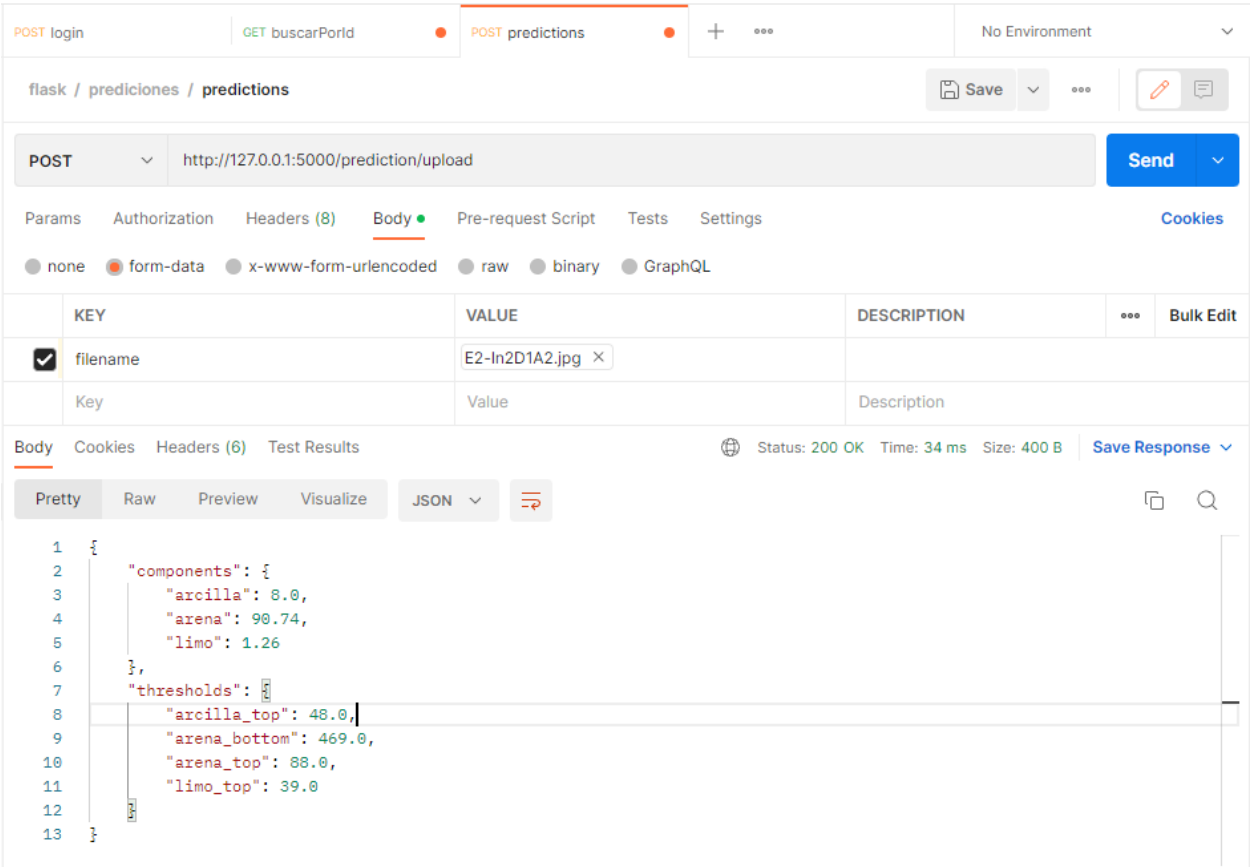

flask / cultivos / eliminar_cultivo
Save
DELETE
http://127.0.0.1:5000/cultivo/18
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
Key
Value
Description
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 749 ms
Size: 440 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 200,
"cultivo": {
"cultivoEstado": 1,
"cultivoNombre": "arroz",
"fechaFinal": "2026-01-01",
"fechaInicio": "2022-01-01",
"id": 18,
"tipoCultivo": 1,
"user": 25
},
"message": "Cultivo eliminado"


flask / cultivos / tipos_cultivos
Save
GET
http://127.0.0.1:5000/cultivo/tipos
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
Key
Value
Description
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 407 ms
Size: 379 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 200,
"message": "Tipos cultivo encontrado",
"tipos": [
{
"id": 1,
"nombre": "maiz",
"referencia": "maiz",
"variedad": "maiz"
}
]
}


flask / lotes / create
Save
POST
http://127.0.0.1:5000/lote
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"nombre":"lot-07",
"area": 1,
"latitud":"2.883137459182263",
"longitud":"-75.26639088403925",
"altitud":0,
"finca_id":1
}
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 384 ms
Size: 426 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
"altitud": 0,
"area": 1,
"finca": 1,
"id": null,
"latidud": "2.883137459182263",
"longitud": "-75.26639088403925",
"nombre": "lot-07"
},
"message": "Lote creado"
}


flask / lotes / buscarPorFinca
Save
GET
http://127.0.0.1:5000/lote/finca/1
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
Key
Value
Description
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 424 ms
Size: 1.35 KB
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 200,
"lotes": [
{
"altitud": 0.0,
"area": 1.0,
"finca": 1,
"id": 3,
"latidud": 2.88313745918226,
"longitud": -75.2663908840393,
"nombre": "lot-02-f1"
},
{


flask / lotes / buscarPorId
Save
GET
http://127.0.0.1:5000/lote/8
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
6 hidden
KEY
VALUE
DESCRIPTION
Bulk Edit
Presets
Authorization
Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9...
Key
Value
Description
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 429 ms
Size: 425 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"code": 200,
"lote": {
"altitud": 0.0,
"area": 1.0,
"finca": 1,
"id": 8,
"latidud": 2.88313745918226,
"longitud": -75.2663908840393,
"nombre": "lot-07"
},
"message": "Cultivo encontrado"
}


POST login
GET buscarPorId
POST predictions
No Environment
flask / prediciones / predictions
Save
POST
http://127.0.0.1:5000/prediction/upload
Send
Params
Authorization
Headers (8)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
KEY
VALUE
DESCRIPTION
Bulk Edit
filename
E2-In2D1A2.jpg
Key
Value
Description
Body
Cookies
Headers (6)
Test Results
Status: 200 OK
Time: 34 ms
Size: 400 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"components": {
"arcilla": 8.0,
"arena": 90.74,
"limo": 1.26
},
"thresholds": {
"arcilla_top": 48.0,
"arena_bottom": 469.0,
"arena_top": 88.0,
"limo_top": 39.0
}
}

15.2.3 Diccionario de datos

Tabla: Usuario		
Descripción: Almacena información personal relacionada con el usuario.		
Nombre	**Tipo de datos**	**Descripción**
id	serial	Llave primaria de identificación única para cada usuario.
usu_nombre	character varying	Nombre de usuario
usu_apellido	character varying	Apellido de usuario
usu_numero_identificacion	bigint	Número de identificación del usuario
usu_celular	bigint	Número del celular del usuario
usu_direccion	character varying	Dirección de usuario
usu_user	character varying	User de usuario
usu_password	character varying	Password de usuario
usu_correo	character varying	Correo de usuario
usu_tipo_identificacion	bigint	Llave foránea a la tabla tipo_identificacion

Tabla: usuario_perfil		
Descripción: Almacena las relaciones de los entre el usuario y los perfiles que puede tener.		
Nombre	**Tipo de datos**	**Descripción**
id	serial	Llave primaria de identificación única para cada relación usuario - perfil.
usu_per_idusuario	bigint	Llave foránea a la tabla usuario
usu_per_idperfil	bigint	Llave foránea a la tabla perfil

Tabla: perfil
Descripción: almacena información perfiles presentes en la plataforma.

Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada perfil
per_nombre	bigint	Nombre del perfil
per_description	character varying	Descripción del perfil

Tabla: finca		
Descripción: Almacena información relacionada con las fincas que el usuario administra.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada finca
fin_nombre	character varying	Nombre de la finca
fin_direccion	character varying	La dirección en la se encuentra la finca
fin_latitud	double	La latitud de la finca en coordenadas decimales
fin_longitud	double	La latitud de la finca en coordenadas decimales
fin_altitud	double	La latitud de la finca en coordenadas decimales
fin_usuario	bigint	Llave foránea a la tabla usuario.

Tabla: Lote		
Descripción: Almacena información relacionada con los lotes que tiene el usuario en una finca.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada lote
lot_nombre	character varying	Nombre del lote
lot_finca	bigint	Llave foránea a la tabla finca.
lot_area	double precision	Área del lote en metros cuadrados
lot_latitud	double precision	La latitud de la finca en coordenadas decimales
lot_logitud	double precision	La latitud de la finca en coordenadas decimales
lot_altitud	double precision	La latitud de la finca en coordenadas decimales

Tabla: sectores		
Descripción: Almacena información relacionada con los sectores de riego que tiene el usuario en un lote.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada sector.
sec_nombre	character varying	Nombre del sector
sec_lotes	bigint	Llave foránea a la tabla sectores.
sec_latitud	double precision	La latitud de la finca en coordenadas decimales
sec_longitud	double precision	La latitud de la finca en coordenadas decimales
sec_altitud	double precision	La latitud de la finca en coordenadas decimales
sec_tipo_suelo	bigint	
sec_cultivo	bigint	Llave foránea a la tabla cultivo.

Tabla: cultivo		
Descripción: almacena información relacionada con cultivos que el usuario tiene.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada cultivo.
cul_nombre	character varying	Nombre del cultivo
cul_tipo_cultivo	bigint	Llave foránea a la tabla tipo_cultivo
cul_fecha_inicio	date	Fecha de la etapa inicial del cultivo
cul_fecha_desarrollo	date	Fecha de la etapa desarrollo del cultivo
cul_fehca_final	date	Fecha de la etapa final del cultivo
cul_estado	bigint	Estado del cultivo
cul_user	bifint	Llave foránea a la tabla usuario

Tabla: muestra_suelo		
Descripción: almacena información relacionada con la muestra realizada por el usuario.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada muestra.
arena	double precision	Porcentaje de arena
limo	double precision	Porcentaje de limo.
arcilla	double precision	Porcentaje de arcilla.
usuario	bigint	Llave foránea a la tabla usuario.
tipo_suelo	bigint	Llave foránea a la tabla tipo_suelo.

Tabla: riego		
Descripción: almacena información relacionada con el riego.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada muestra.
rie_cantidad_liquido_aplicado	double precision	Cantidad liquido aplicado.
rie_fecha_inicio	timestamp	Fecha de inicio del riego.
rie_fecha_fin	timestamp	Fecha del final del riego.
rie_estado	bigint	Estado del riego.
rie_cantidad_liquido_calculado	double precision	Cantidad del líquido calculado.
rie_sectores	bigint	Llave foránea a la tabla sectores.
rie_cultivos	bigint	Llave foránea a la tabla cultivos.

Tabla: tipo_suelo		
Descripción: almacena información relacionada con el tipo de suelo.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada muestra
tip_sue_nombre	character varying	Nombre del tipo de suelo
tip_sue_velocidad_infiltracion	double precision	Velocidad de infiltración
tip_sue_punto_marchitez	double precision	Punto de marchitez permanente
tip_tipo_capacidad_campo	double precision	Punto de capacidad de campo

Tabla: tipo_cultivo		
Descripción: almacena información relacionada con los tipos de cultivo que se manejan en la plataforma.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada muestra
tip_cul_nombre	character varying	Nombre del tipo de cultivo
tip_cul_variedad	character varying	Variedad del cultivo
tip_cul_referencia	character varying	Referencia del cultivo

Tabla: tipo_identificacion		
Descripción: almacena información relacionada con el tipo de identificación de los usuarios.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada muestra
tip_ident_nombre	character varying	Nombre del tipo de identificación
tip_ident_descripcion	character varying	Descripción del documento

Tabla: admin_riego		
Descripción: almacena información relacionada con la administración del riego en los sectores.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada muestra.
admin_tipo_riego	bigint	Llave foránea a la tabla tipo_riego.
admin_nap	double precision	Porcentaje de NAP (Nivel de Agotamineto Permicible).
admin_efectividad	double precision	Porcentaje de la efectividad del riego.
admin_caudal	double precision	Caudal del emisor de riego.
admin_distancia	double precision	Distancia entre emisores.
admin_radio	double precision	Distancia entre laterales de riego.
admin_sector	bigint	Llave foránea a la tabla sectores.

Tabla: dispositivo		
Descripción: almacena información de los dispositivos presente en los sectores de riego.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada dispositivo
dis_nombre	character varying	Nombre del dispositivo
dis_tipo	bigint	Llave foránea a la tabla tipo_dispositivo
dis_modelo	character varying	Modelo del dispositivo.
dis_sectores	bigint	Llave foránea a la tabla sectores
dis_estado	bigint	Estado del dispositivo

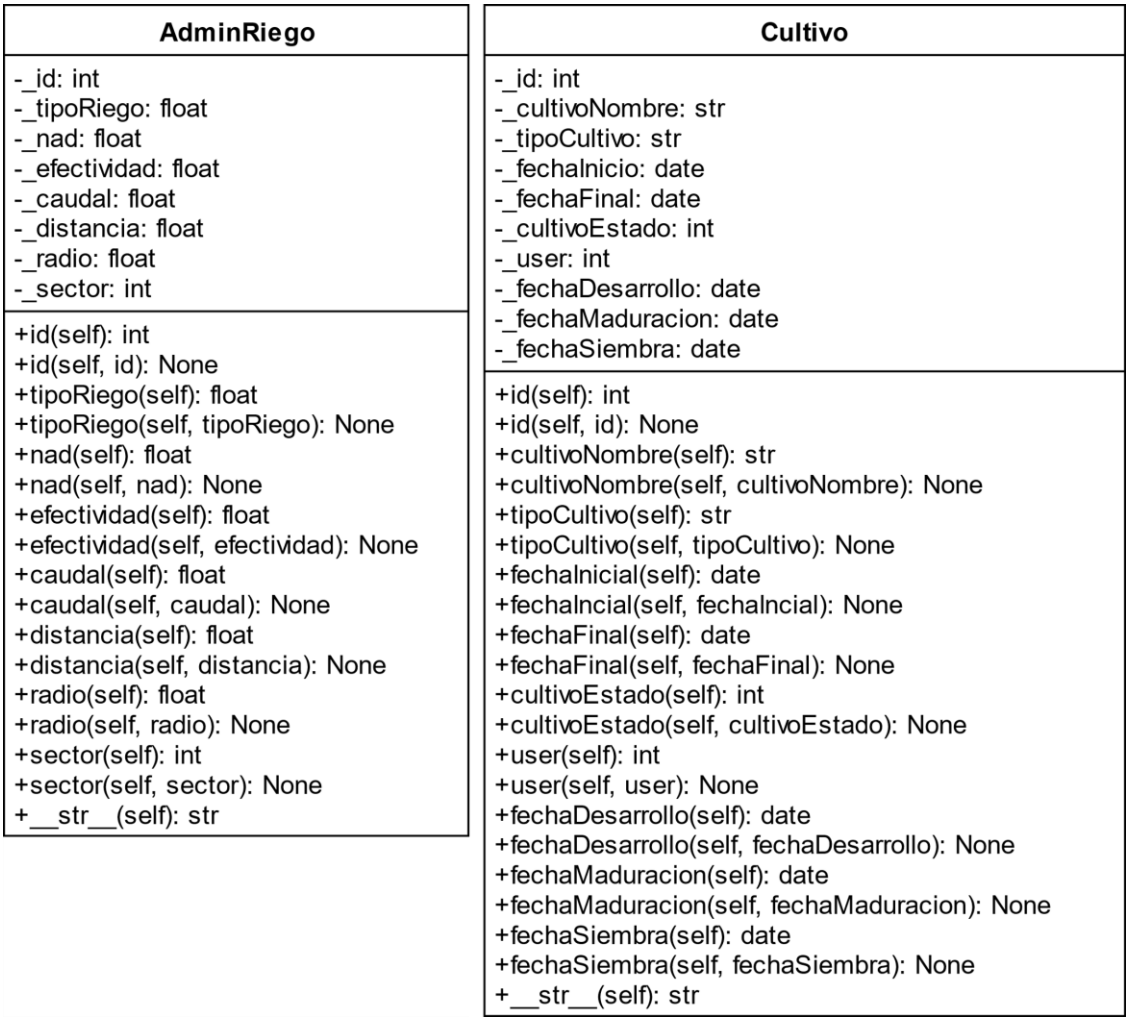
Tabla: dispositivo_medida		
Descripción: almacena información relacionada con la medición de los dispositivos disponible en el sistema de riego.		
Nombre	Tipo de datos	Descripción
id	id	Llave primaria de identificación única para cada medida del dispositivo
dis_med_fecha	timestamp	Fecha en la que se realiza la medición
dis_med_medicion	double precision	Valor de la medición
dis_med_dispositivo	bigint	Llave foránea con la tabla dispositivo

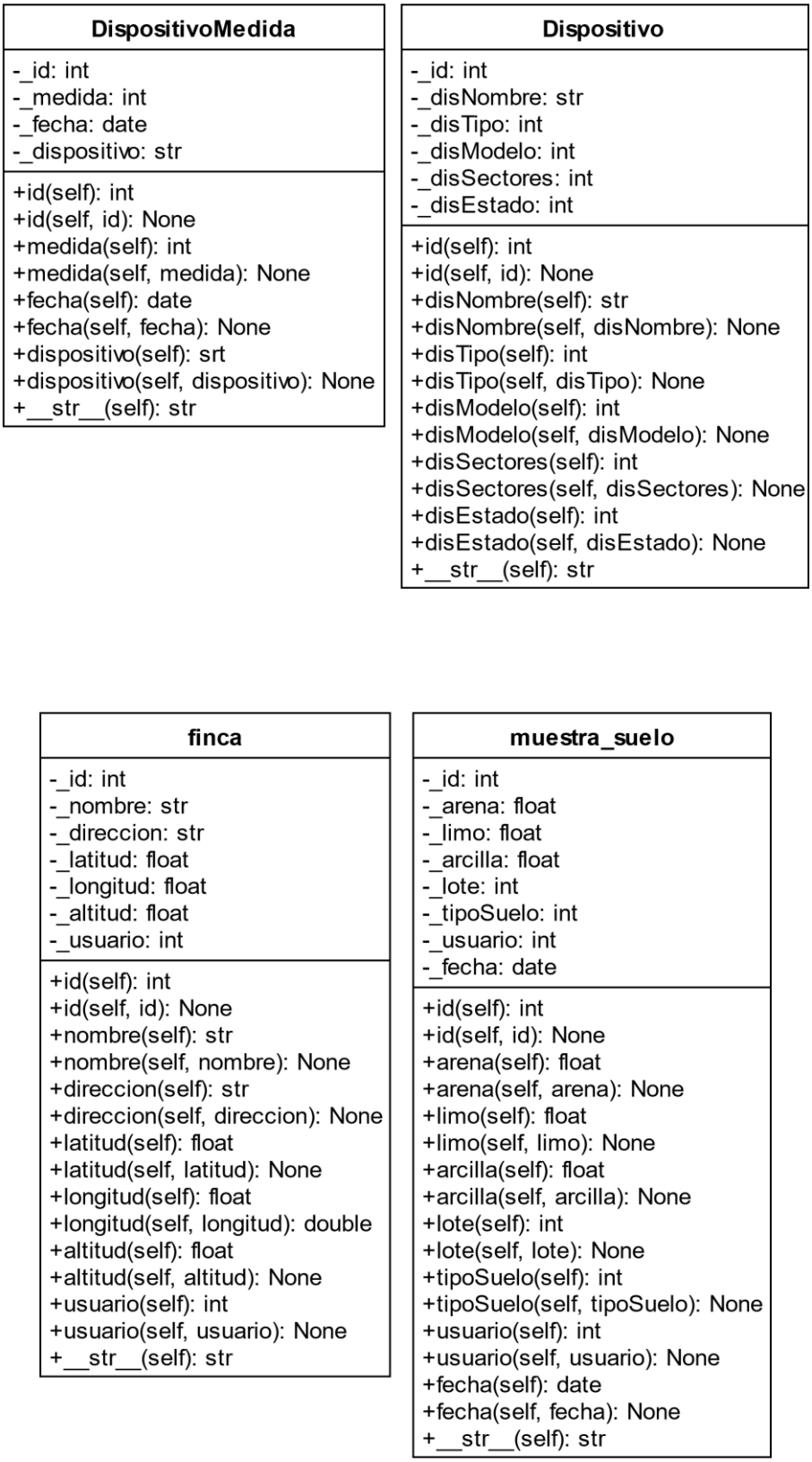
Tabla: tipo_riego		
Descripción: almacena información de los tipos de riego.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada tipo de riego
nombre	character varying	Nombre del tipo de riego
efectividad	double precision	Porcentaje de efectividad del riego
descripcion	character varying	Descripción del sistema de riego

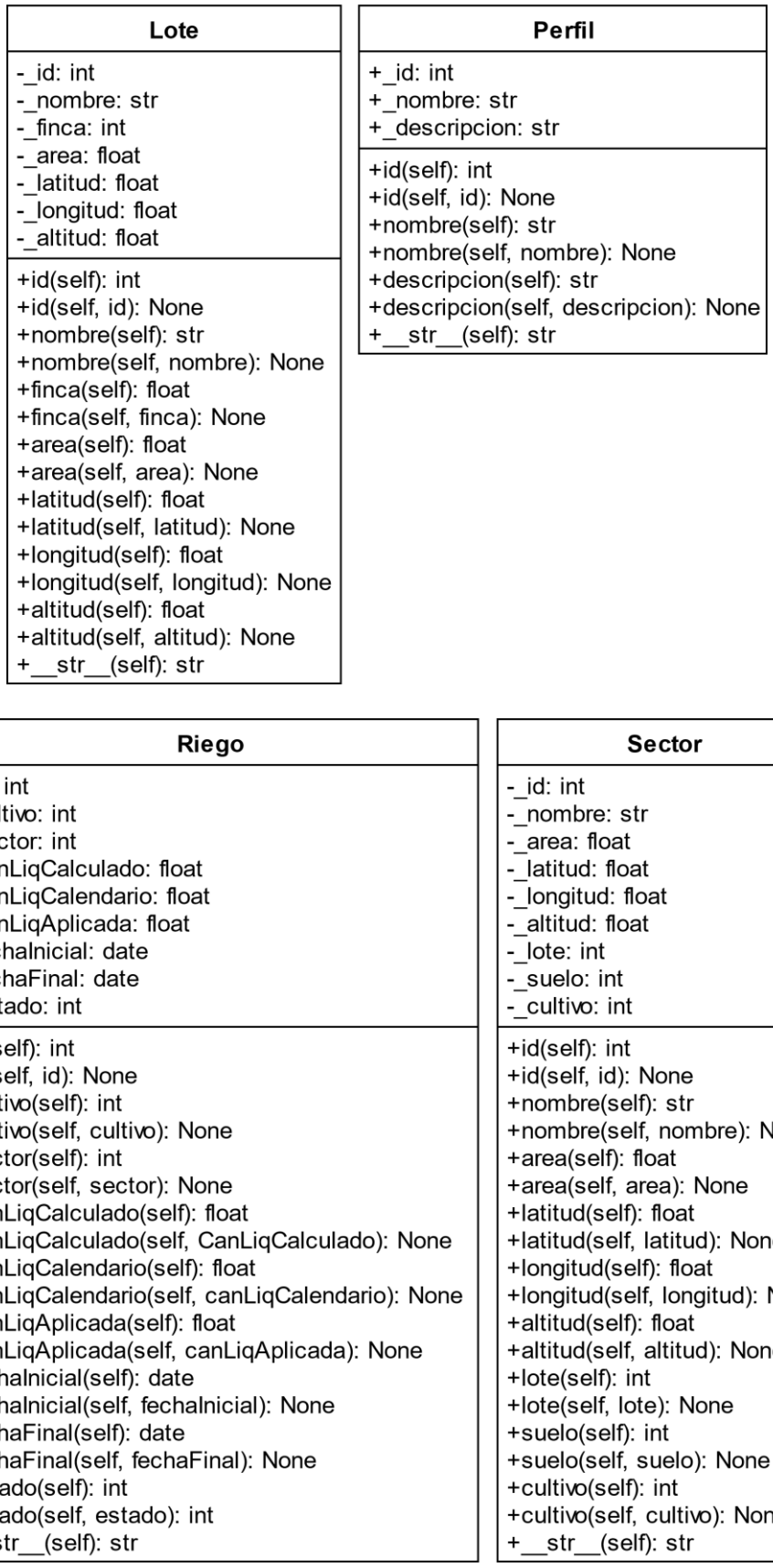
Tabla: tipo_dispositivo		
Descripción: almacena información de los tipos de dispositivos.		
Nombre	Tipo de datos	Descripción
id	serial	Llave primaria de identificación única para cada tipo de dispositivo
tip_dis_categoria	character varying	Categoría del dispositivo
tip_dis_caracteristica	character varying	Característica - Descripción del dispositivo

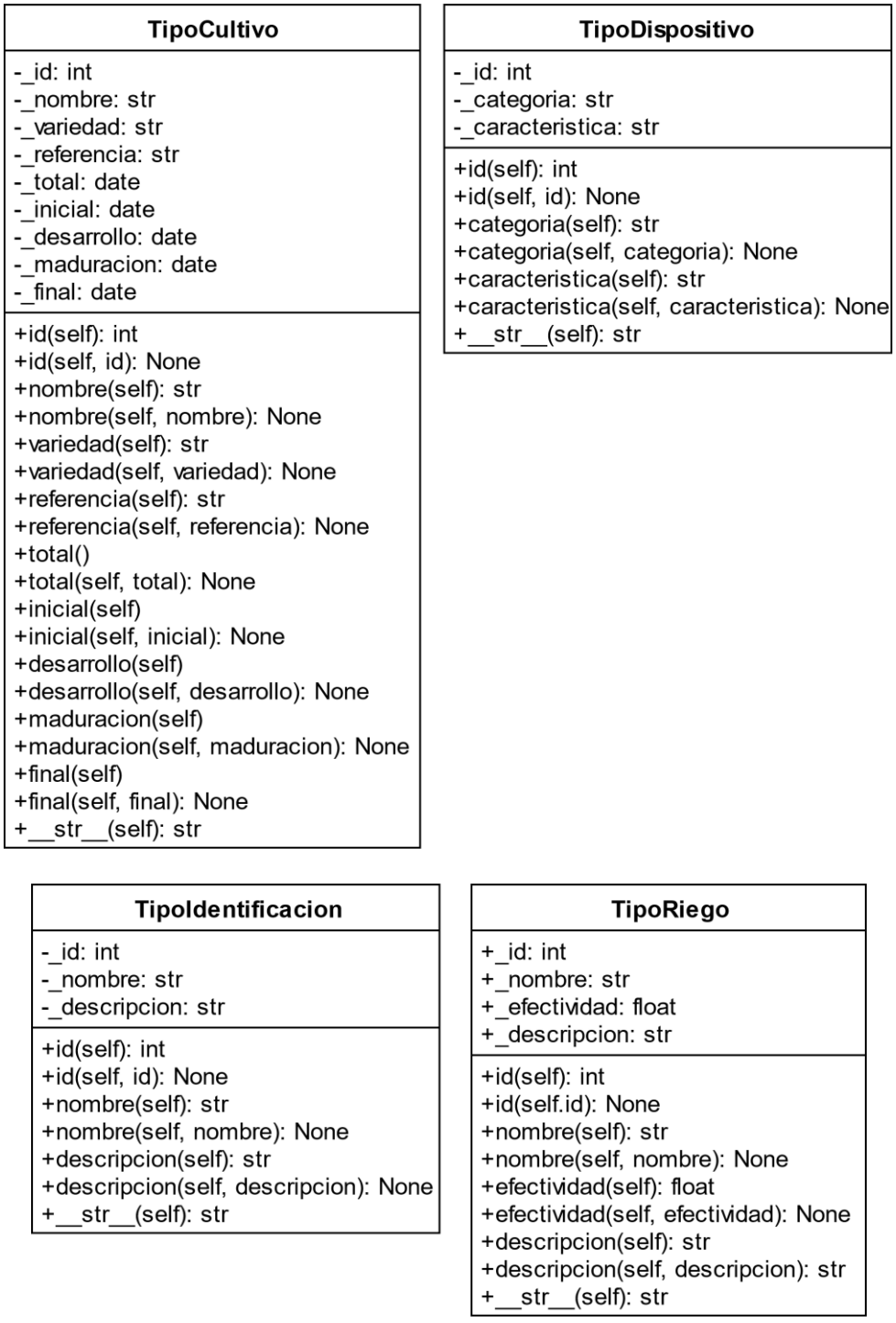
15.2.4 Diagrama de clases

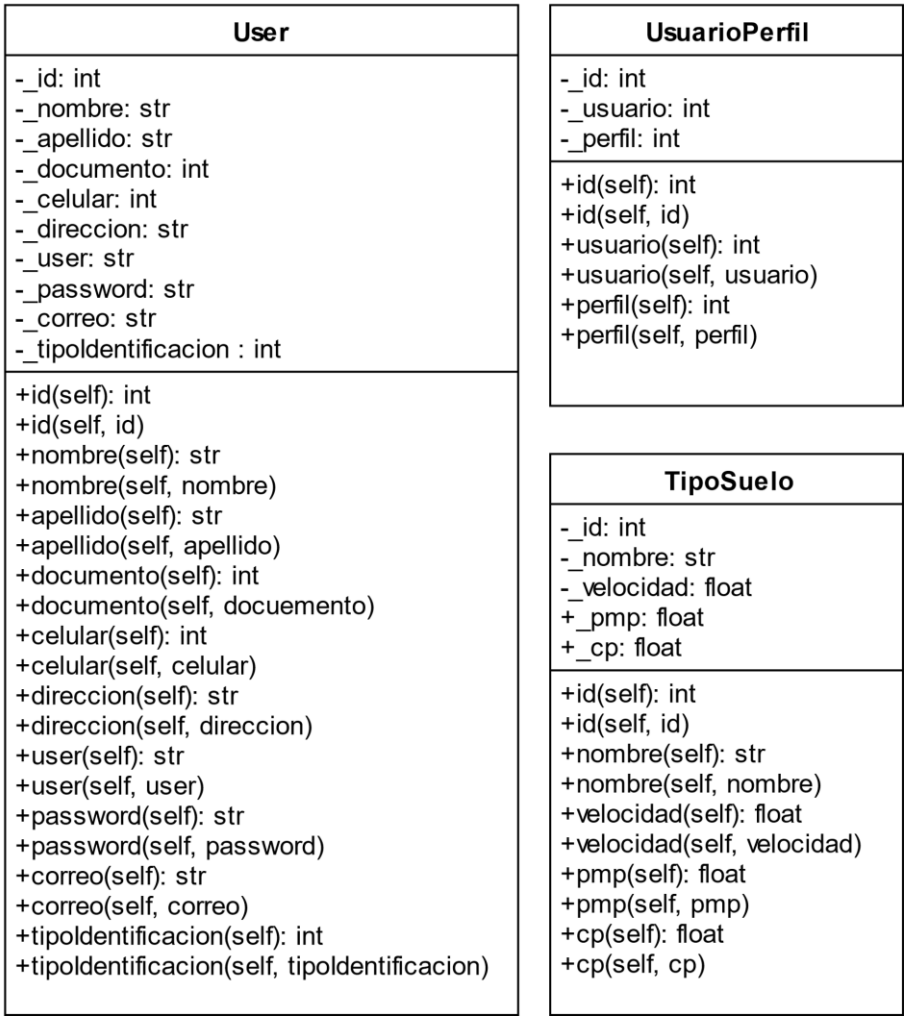
Paquete Models:



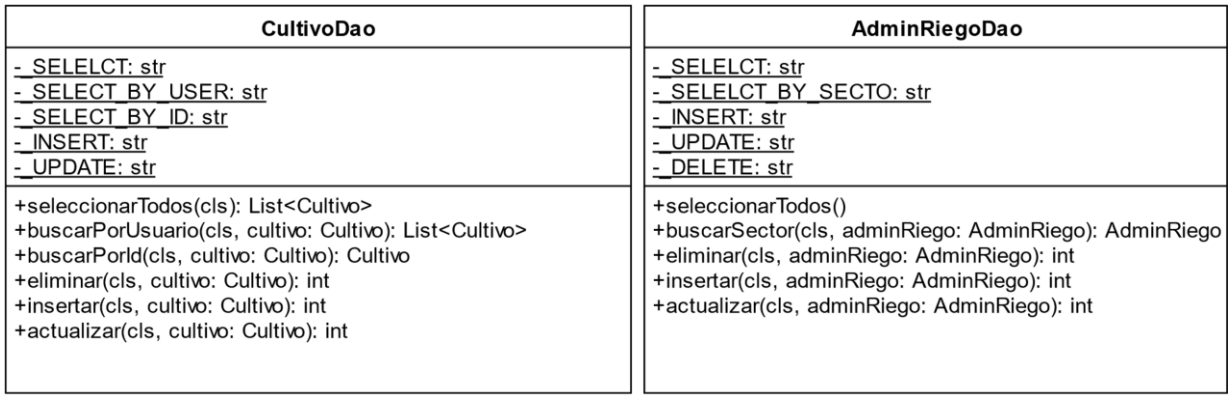

DispositivoMedida
-_id: int
-_medida: int
-_fecha: date
-_dispositivo: str
+id(self): int
+id(self, id): None
+medida(self): int
+medida(self, medida): None
+fecha(self): date
+fecha(self, fecha): None
+dispositivo(self): srt
+dispositivo(self, dispositivo): None
+__str__(self): str
Dispositivo
-_id: int
-_disNombre: str
-_disTipo: int
-_disModelo: int
-_disSectores: int
-_disEstado: int
+id(self): int
+id(self, id): None
+disNombre(self): str
+disNombre(self, disNombre): None
+disTipo(self): int
+disTipo(self, disTipo): None
+disModelo(self): int
+disModelo(self, disModelo): None
+disSectores(self): int
+disSectores(self, disSectores): None
+disEstado(self): int
+disEstado(self, disEstado): None
+__str__(self): str
finca
-_id: int
-_nombre: str
-_direccion: str
-_latitud: float
-_longitud: float
-_altitud: float
-_usuario: int
+id(self): int
+id(self, id): None
+nombre(self): str
+nombre(self, nombre): None
+direccion(self): str
+direccion(self, direccion): None
+latitud(self): float
+latitud(self, latitud): None
+longitud(self): float
+longitud(self, longitud): double
+altitud(self): float
+altitud(self, altitud): None
+usuario(self): int
+usuario(self, usuario): None
+__str__(self): str
muestra_suelo
-_id: int
-_arena: float
-_limo: float
-_arcilla: float
-_lote: int
-_tipoSuelo: int
-_usuario: int
-_fecha: date
+id(self): int
+id(self, id): None
+arena(self): float
+arena(self, arena): None
+limo(self): float
+limo(self, limo): None
+arcilla(self): float
+arcilla(self, arcilla): None
+lote(self): int
+lote(self, lote): None
+tipoSuelo(self): int
+tipoSuelo(self, tipoSuelo): None
+usuario(self): int
+usuario(self, usuario): None
+fecha(self): date
+fecha(self, fecha): None
+__str__(self): str

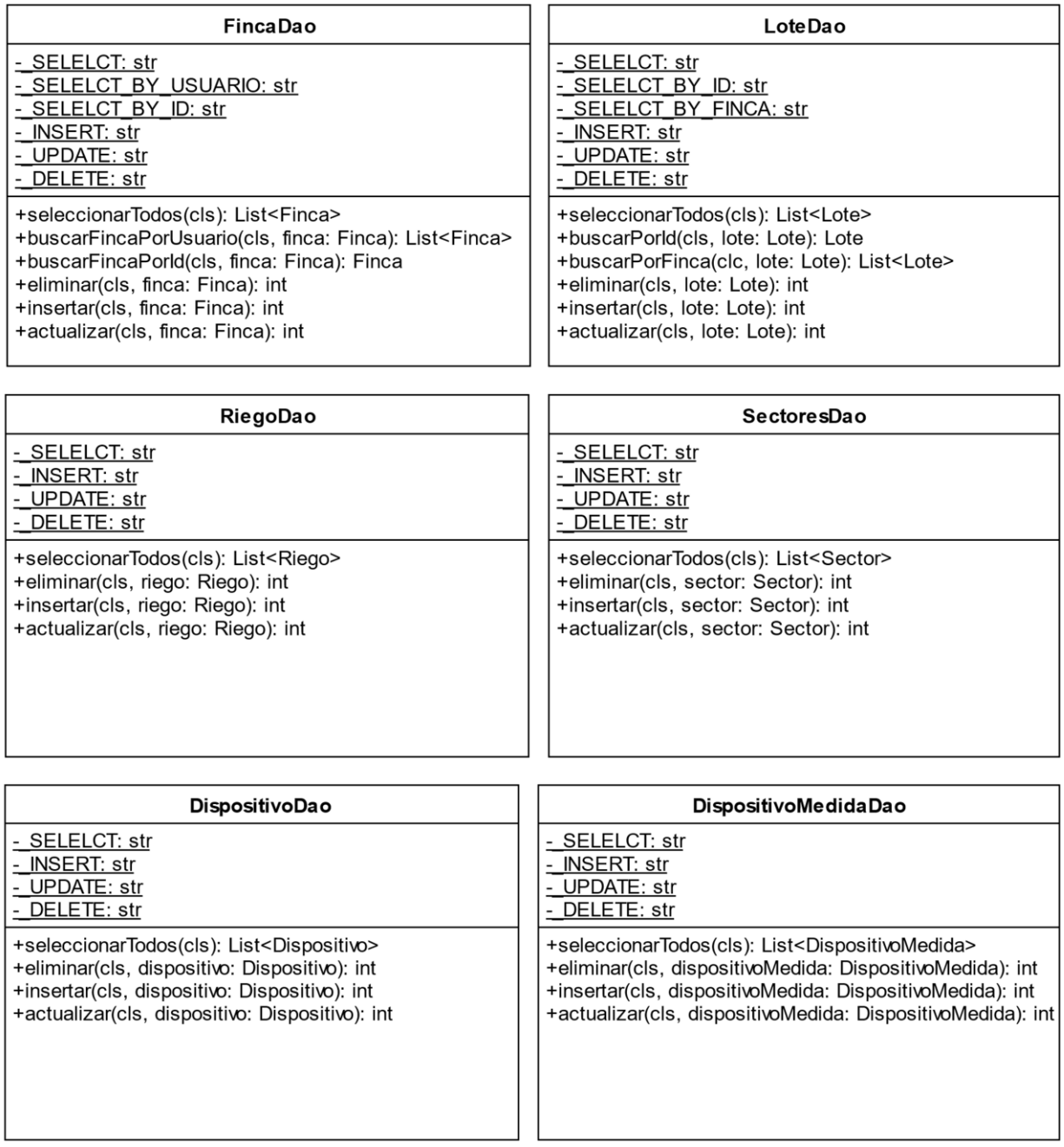

Lote
-_id: int
-_nombre: str
-_finca: int
-_area: float
-_latitud: float
-_longitud: float
-_altitud: float
+id(self): int
+id(self, id): None
+nombre(self): str
+nombre(self, nombre): None
+finca(self): float
+finca(self, finca): None
+area(self): float
+area(self, area): None
+latitud(self): float
+latitud(self, latitud): None
+longitud(self): float
+longitud(self, longitud): None
+altitud(self): float
+altitud(self, altitud): None
+__str__(self): str
Perfil
+_id: int
+_nombre: str
+_descripcion: str
+id(self): int
+id(self, id): None
+nombre(self): str
+nombre(self, nombre): None
+descripcion(self): str
+descripcion(self, descripcion): None
+__str__(self): str
Riego
-_id: int
-_cultivo: int
-_sector: int
-_canLiqCalculado: float
-_canLiqCalendario: float
-_canLiqAplicada: float
-_fechaInicial: date
-_fechaFinal: date
-_estado: int
+id(self): int
+id(self, id): None
+cultivo(self): int
+cultivo(self, cultivo): None
+sector(self): int
+sector(self, sector): None
+canLiqCalculado(self): float
+canLiqCalculado(self, CanLiqCalculado): None
+canLiqCalendario(self): float
+canLiqCalendario(self, canLiqCalendario): None
+canLiqAplicada(self): float
+canLiqAplicada(self, canLiqAplicada): None
+fechaInicial(self): date
+fechaInicial(self, fechaInicial): None
+fechaFinal(self): date
+fechaFinal(self, fechaFinal): None
+estado(self): int
+estado(self, estado): int
+__str__(self): str
Sector
-_id: int
-_nombre: str
-_area: float
-_latitud: float
-_longitud: float
-_altitud: float
-_lote: int
-_suelo: int
-_cultivo: int
+id(self): int
+id(self, id): None
+nombre(self): str
+nombre(self, nombre): None
+area(self): float
+area(self, area): None
+latitud(self): float
+latitud(self, latitud): None
+longitud(self): float
+longitud(self, longitud): None
+altitud(self): float
+altitud(self, altitud): None
+lote(self): int
+lote(self, lote): None
+suelo(self): int
+suelo(self, suelo): None
+cultivo(self): int
+cultivo(self, cultivo): None
+__str__(self): str

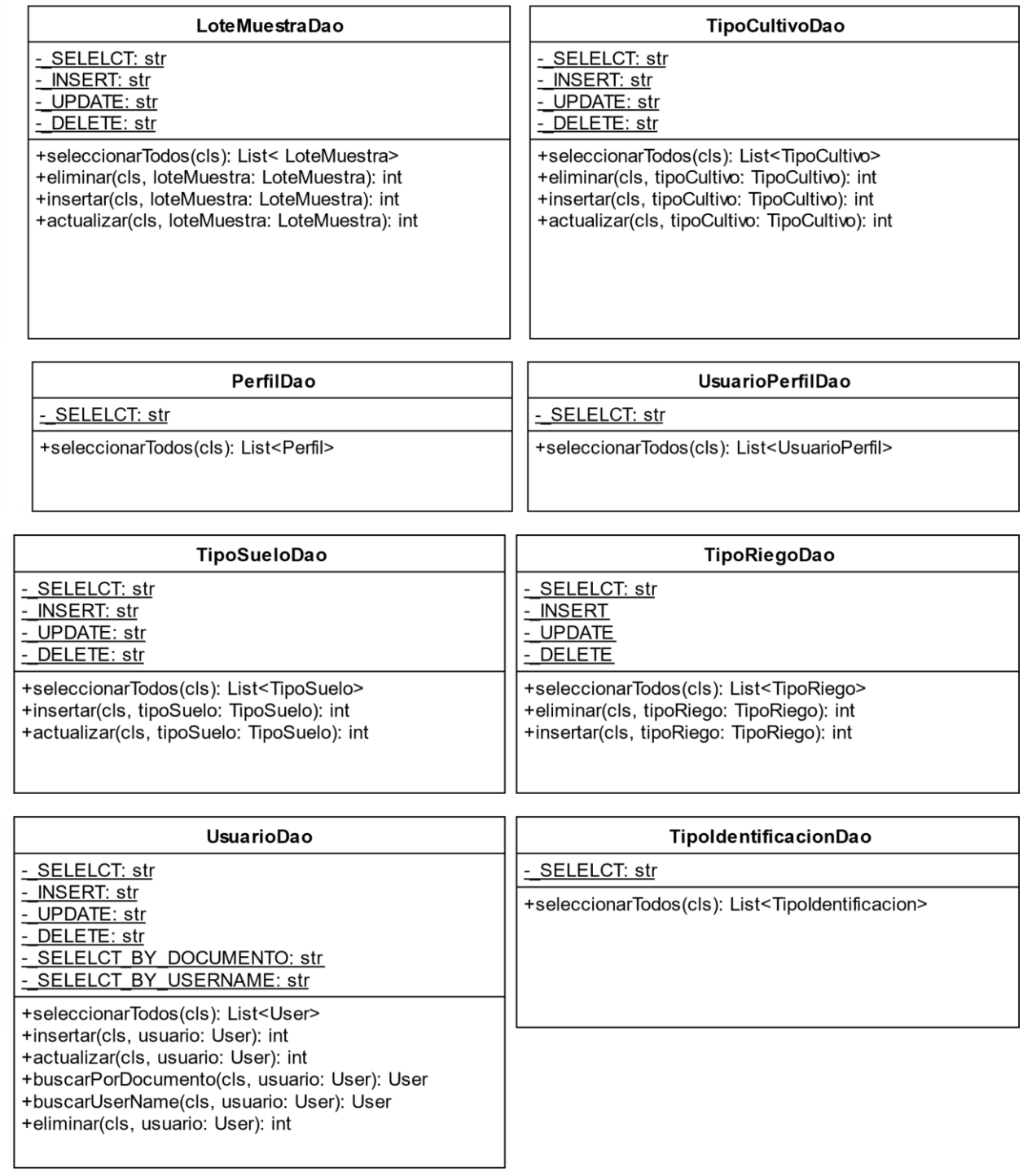

TipoCultivo
-_id: int
-_nombre: str
-_variedad: str
-_referencia: str
-_total: date
-_inicial: date
-_desarrollo: date
-_maduracion: date
-_final: date
+id(self): int
+id(self, id): None
+nombre(self): str
+nombre(self, nombre): None
+variedad(self): str
+variedad(self, variedad): None
+referencia(self): str
+referencia(self, referencia): None
+total()
+total(self, total): None
+inicial(self)
+inicial(self, inicial): None
+desarrollo(self)
+desarrollo(self, desarrollo): None
+maduracion(self)
+maduracion(self, maduracion): None
+final(self)
+final(self, final): None
+__str__(self): str
TipoDispositivo
-_id: int
-_categoria: str
-_caracteristica: str
+id(self): int
+id(self, id): None
+categoria(self): str
+categoria(self, categoria): None
+caracteristica(self): str
+caracteristica(self, caracteristica): None
+__str__(self): str
TipoIdentificacion
-_id: int
-_nombre: str
-_descripcion: str
+id(self): int
+id(self, id): None
+nombre(self): str
+nombre(self, nombre): None
+descripcion(self): str
+descripcion(self, descripcion): None
+__str__(self): str
TipoRiego
+_id: int
+_nombre: str
+_efectividad: float
+_descripcion: str
+id(self): int
+id(self.id): None
+nombre(self): str
+nombre(self, nombre): None
+efectividad(self): float
+efectividad(self, efectividad): None
+descripcion(self): str
+descripcion(self, descripcion): str
+__str__(self): str

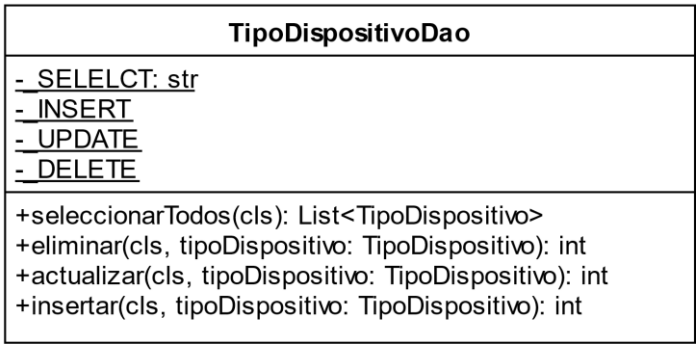

User
-_id: int
-_nombre: str
-_apellido: str
-_documento: int
-_celular: int
-_direccion: str
-_user: str
-_password: str
-_correo: str
-_tipoldentificacion : int
+id(self): int
+id(self, id)
+nombre(self): str
+nombre(self, nombre)
+apellido(self): str
+apellido(self, apellido)
+documento(self): int
+documento(self, docuemento)
+celular(self): int
+celular(self, celular)
+direccion(self): str
+direccion(self, direccion)
+user(self): str
+user(self, user)
+password(self): str
+password(self, password)
+correo(self): str
+correo(self, correo)
+tipoldentificacion(self): int
+tipoldentificacion(self, tipoldentificacion)
UsuarioPerfil
-_id: int
-_usuario: int
-_perfil: int
+id(self): int
+id(self, id)
+usuario(self): int
+usuario(self, usuario)
+perfil(self): int
+perfil(self, perfil)
TipoSuelo
-_id: int
-_nombre: str
-_velocidad: float
+_pmp: float
+_cp: float
+id(self): int
+id(self, id)
+nombre(self): str
+nombre(self, nombre)
+velocidad(self): float
+velocidad(self, velocidad)
+pmp(self): float
+pmp(self, pmp)
+cp(self): float
+cp(self, cp)

Paquete Dao:

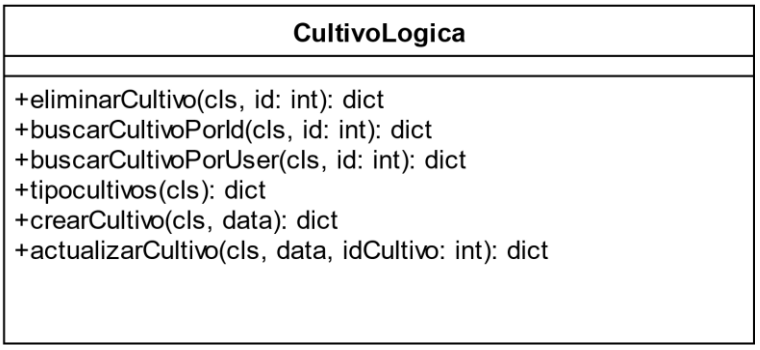

CultivoDao
-_SELELCT: str
-_SELECT_BY_USER: str
-_SELECT_BY_ID: str
-_INSERT: str
-_UPDATE: str
+seleccionarTodos(cls): List<Cultivo>
+buscarPorUsuario(cls, cultivo: Cultivo): List<Cultivo>
+buscarPorId(cls, cultivo: Cultivo): Cultivo
+eliminar(cls, cultivo: Cultivo): int
+insertar(cls, cultivo: Cultivo): int
+actualizar(cls, cultivo: Cultivo): int
AdminRiegoDao
-_SELELCT: str
-_SELELCT_BY_SECTO: str
-_INSERT: str
-_UPDATE: str
-_DELETE: str
+seleccionarTodos()
+buscarSector(cls, adminRiego: AdminRiego): AdminRiego
+eliminar(cls, adminRiego: AdminRiego): int
+insertar(cls, adminRiego: AdminRiego): int
+actualizar(cls, adminRiego: AdminRiego): int


FincaDao
- _SELELCT: str
- _SELELCT_BY_USUARIO: str
- _SELELCT_BY_ID: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<Finca>
+buscarFincaPorUsuario(cls, finca: Finca): List<Finca>
+buscarFincaPorId(cls, finca: Finca): Finca
+eliminar(cls, finca: Finca): int
+insertar(cls, finca: Finca): int
+actualizar(cls, finca: Finca): int
LoteDao
- _SELELCT: str
- _SELELCT_BY_ID: str
- _SELELCT_BY_FINCA: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<Lote>
+buscarPorId(cls, lote: Lote): Lote
+buscarPorFinca(clc, lote: Lote): List<Lote>
+eliminar(cls, lote: Lote): int
+insertar(cls, lote: Lote): int
+actualizar(cls, lote: Lote): int
RiegoDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<Riego>
+eliminar(cls, riego: Riego): int
+insertar(cls, riego: Riego): int
+actualizar(cls, riego: Riego): int
SectoresDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<Sector>
+eliminar(cls, sector: Sector): int
+insertar(cls, sector: Sector): int
+actualizar(cls, sector: Sector): int
DispositivoDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<Dispositivo>
+eliminar(cls, dispositivo: Dispositivo): int
+insertar(cls, dispositivo: Dispositivo): int
+actualizar(cls, dispositivo: Dispositivo): int
DispositivoMedidaDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<DispositivoMedida>
+eliminar(cls, dispositivoMedida: DispositivoMedida): int
+insertar(cls, dispositivoMedida: DispositivoMedida): int
+actualizar(cls, dispositivoMedida: DispositivoMedida): int

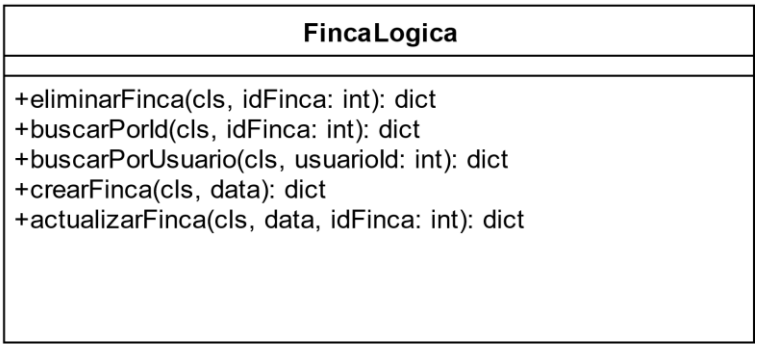

LoteMuestraDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List< LoteMuestra>
+eliminar(cls, loteMuestra: LoteMuestra): int
+insertar(cls, loteMuestra: LoteMuestra): int
+actualizar(cls, loteMuestra: LoteMuestra): int
TipoCultivoDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<TipoCultivo>
+eliminar(cls, tipoCultivo: TipoCultivo): int
+insertar(cls, tipoCultivo: TipoCultivo): int
+actualizar(cls, tipoCultivo: TipoCultivo): int
PerfilDao
- _SELELCT: str
+seleccionarTodos(cls): List<Perfil>
UsuarioPerfilDao
- _SELELCT: str
+seleccionarTodos(cls): List<UsuarioPerfil>
TipoSueloDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
+seleccionarTodos(cls): List<TipoSuelo>
+insertar(cls, tipoSuelo: TipoSuelo): int
+actualizar(cls, tipoSuelo: TipoSuelo): int
TipoRiegoDao
- _SELELCT: str
- _INSERT
- _UPDATE
- _DELETE
+seleccionarTodos(cls): List<TipoRiego>
+eliminar(cls, tipoRiego: TipoRiego): int
+insertar(cls, tipoRiego: TipoRiego): int
UsuarioDao
- _SELELCT: str
- _INSERT: str
- _UPDATE: str
- _DELETE: str
- _SELELCT_BY_DOCUMENTO: str
- _SELELCT_BY_USERNAME: str
+seleccionarTodos(cls): List<User>
+insertar(cls, usuario: User): int
+actualizar(cls, usuario: User): int
+buscarPorDocumento(cls, usuario: User): User
+buscarUserName(cls, usuario: User): User
+eliminar(cls, usuario: User): int
TipoIdentificacionDao
- _SELELCT: str
+seleccionarTodos(cls): List<TipoIdentificacion>



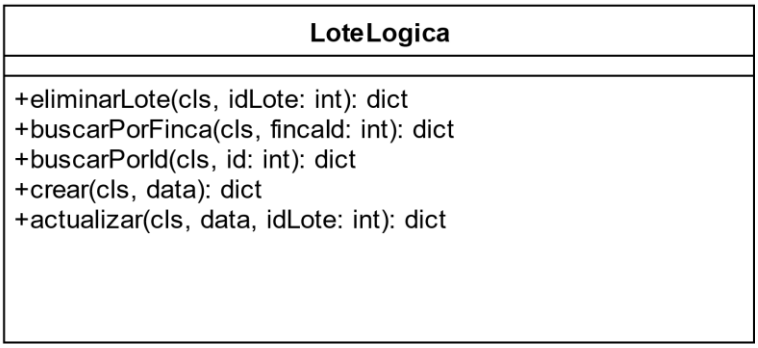
Paquete cultivo:



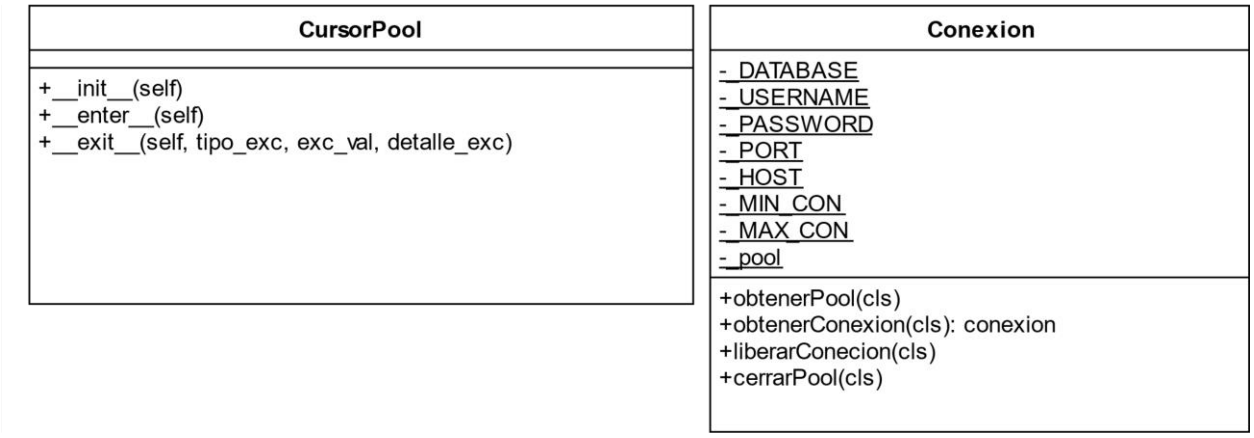
Paquete finca:



Paquete lote:



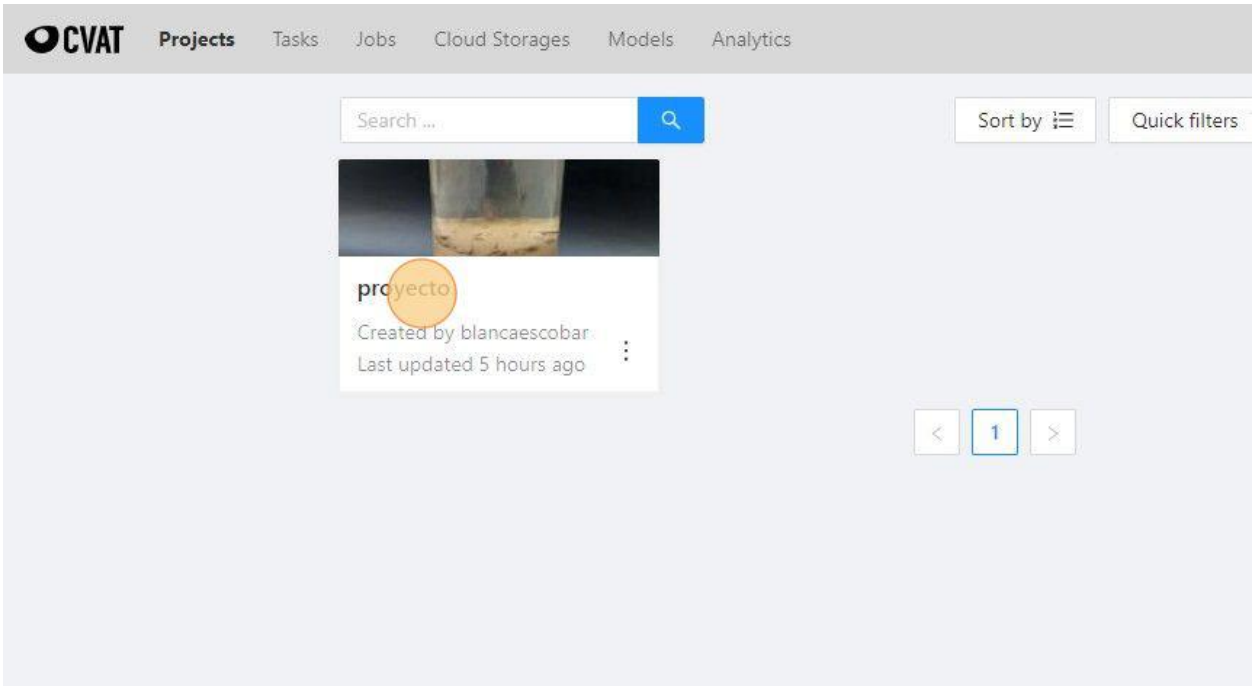
Paqute utilities:



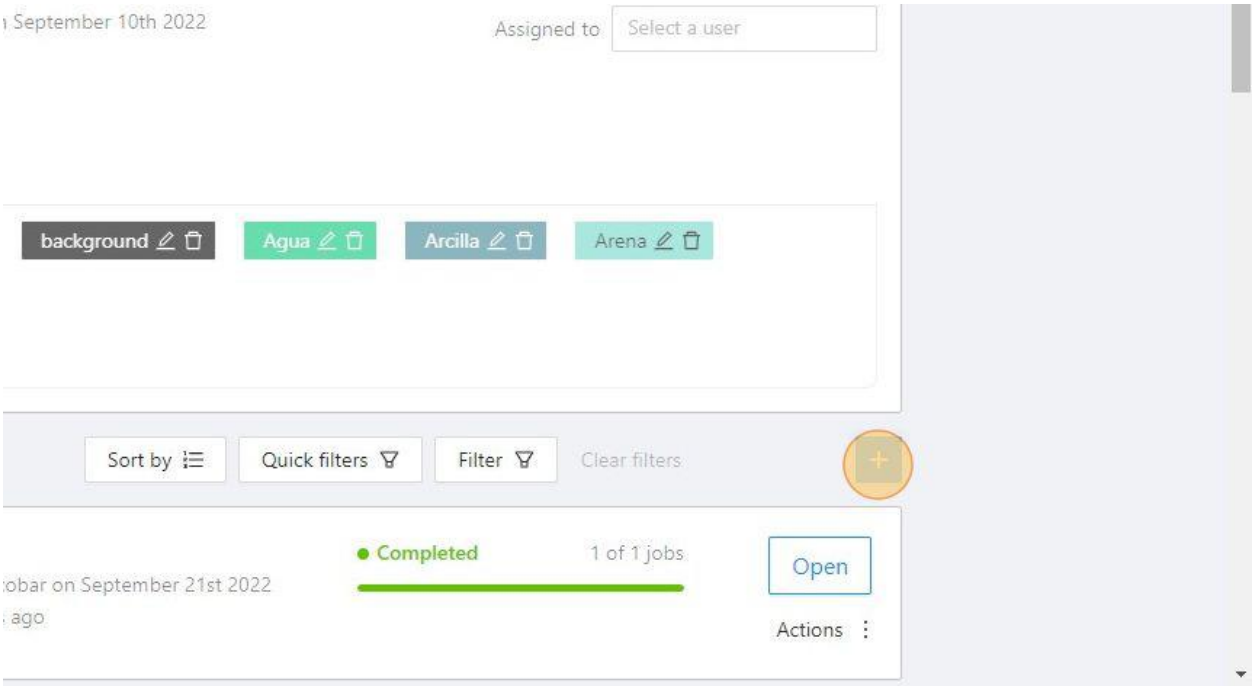
15.3 Inteligencia Artificial

15.3.1 Proceso de anotación de las fotografías en CVAT

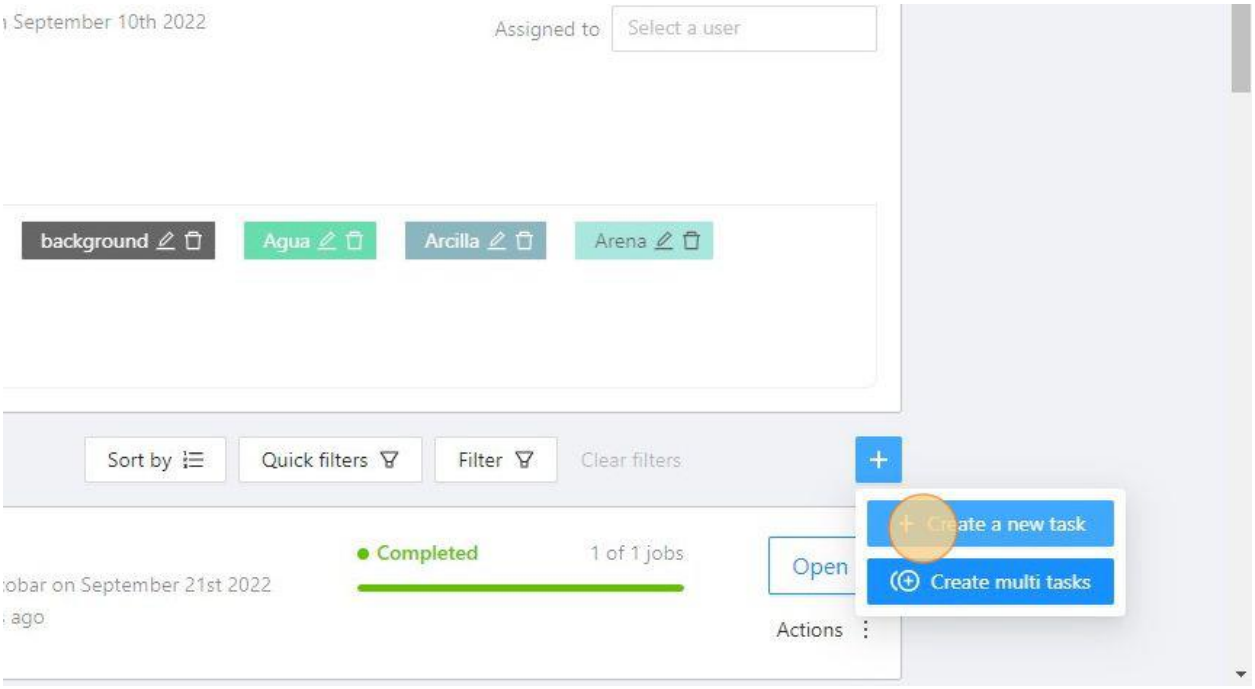
1. Click en el proyecto creado en CVAT en este caso "proyecto".



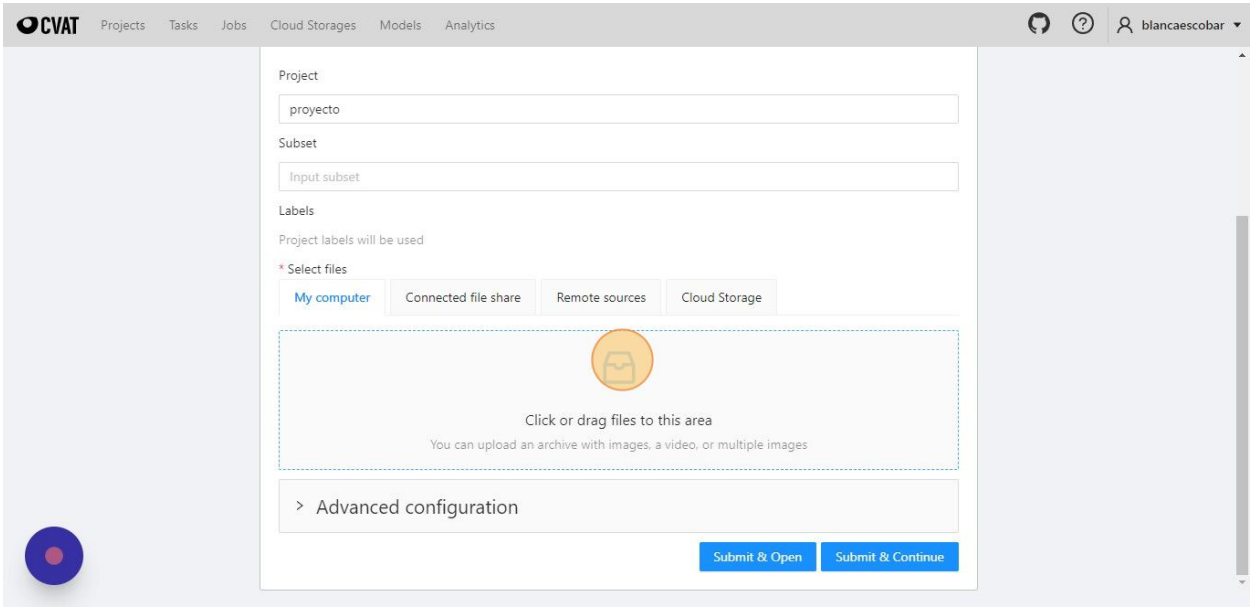
2. Click en este icono.



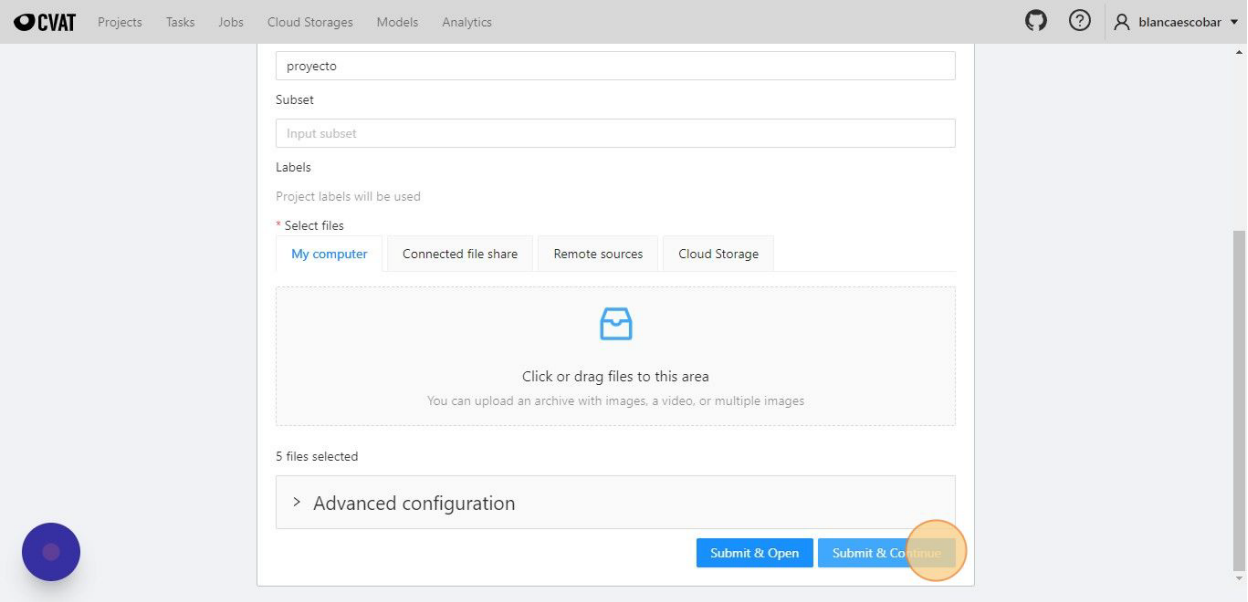
3. Click en “Create a new task”.



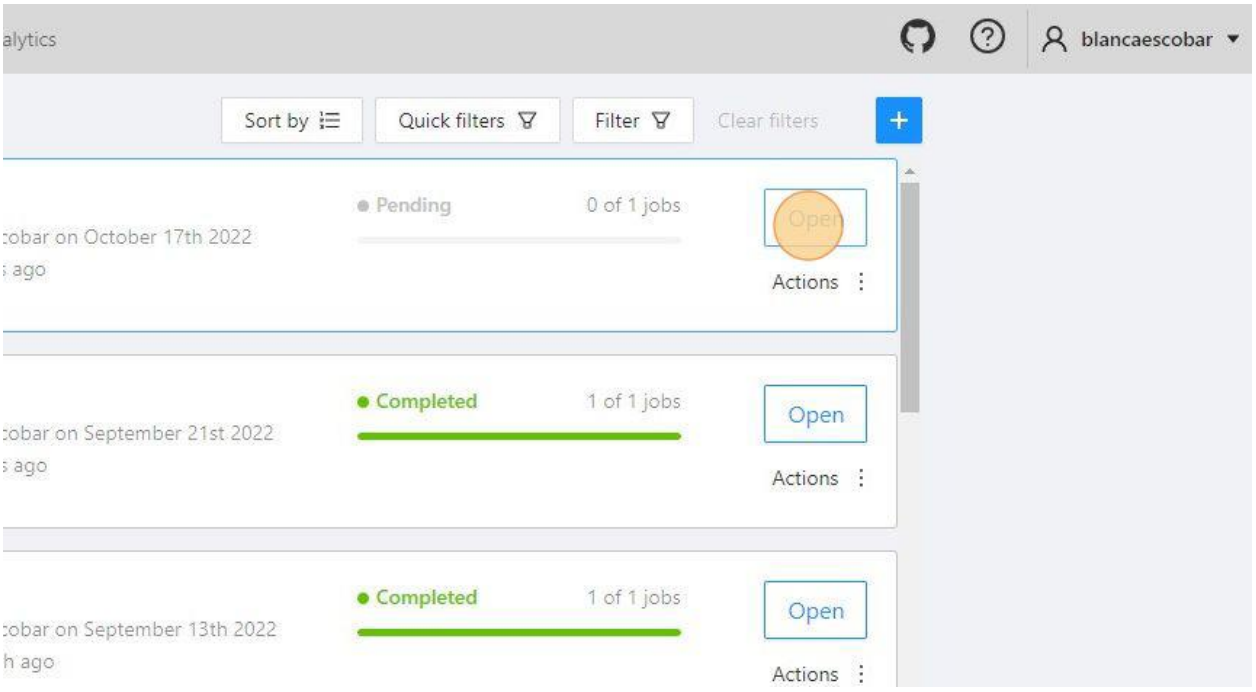
4. Click en este espacio para subir las fotografías de las muestras.



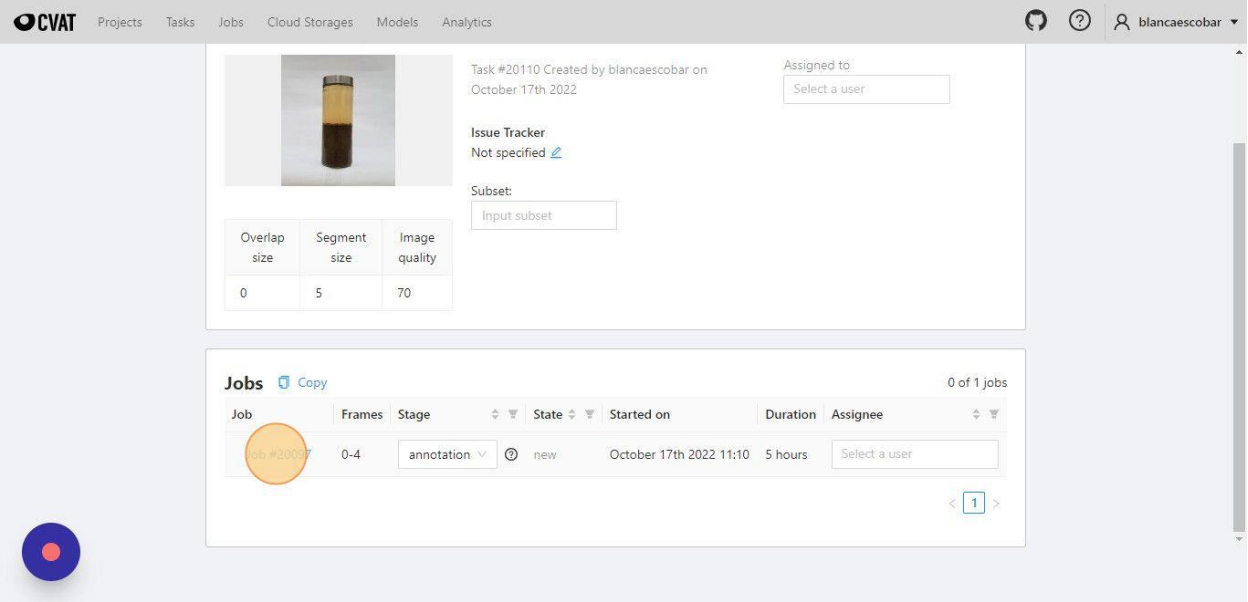
5. Luego de subir las imágenes Click en "Submit & Continue".



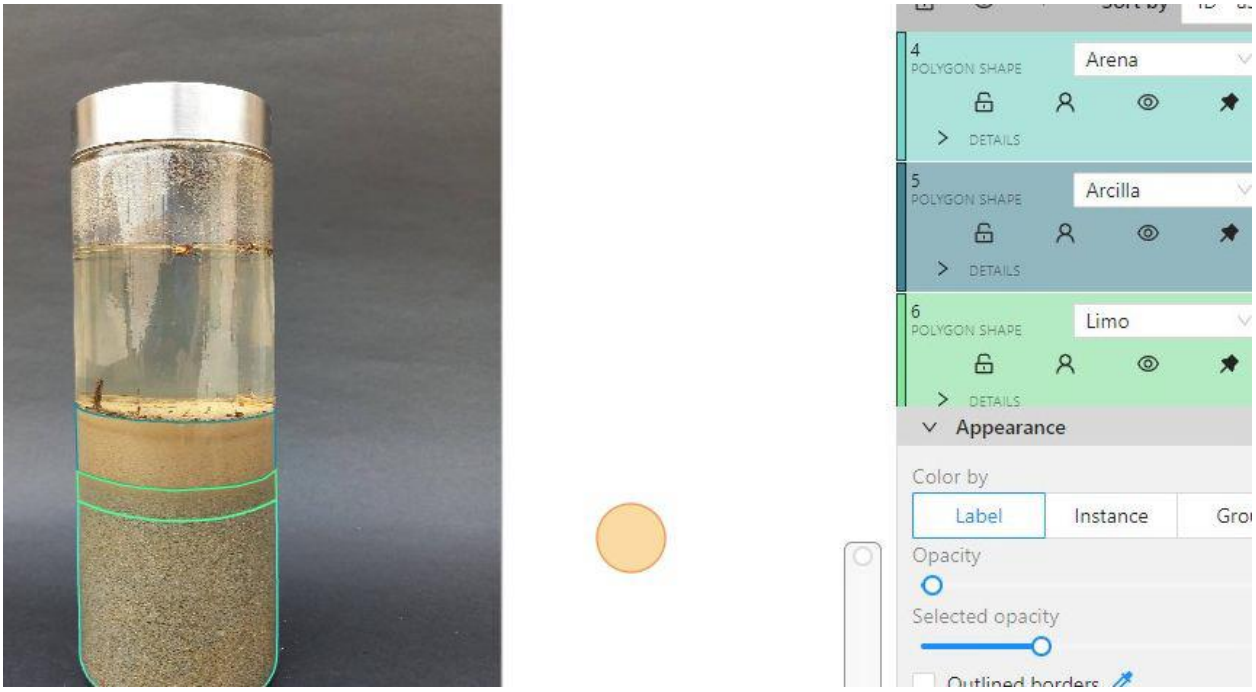
6. Después ir a Task, buscar la nueva tarea creada y dar click en "Open".



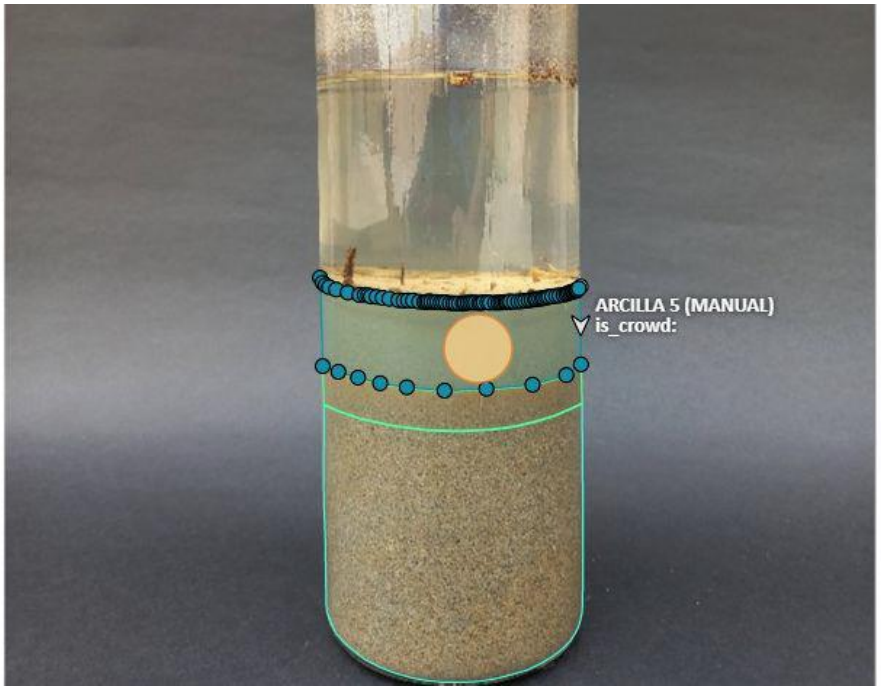
7. Click en este link "Job #20097".



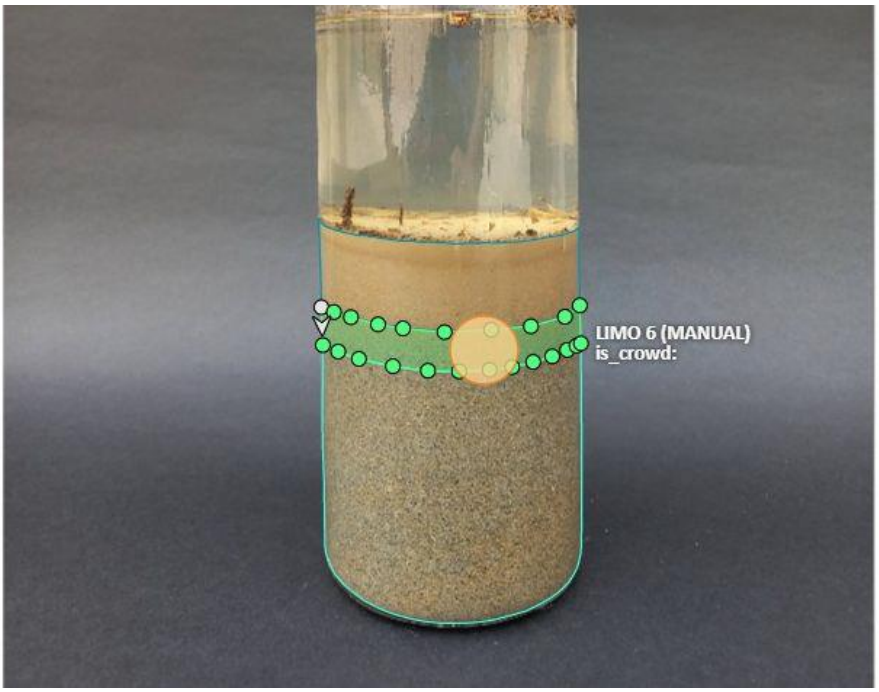
8. Aquí se ve las etiquetas de Arena, Arcilla y Limo de la muestra.



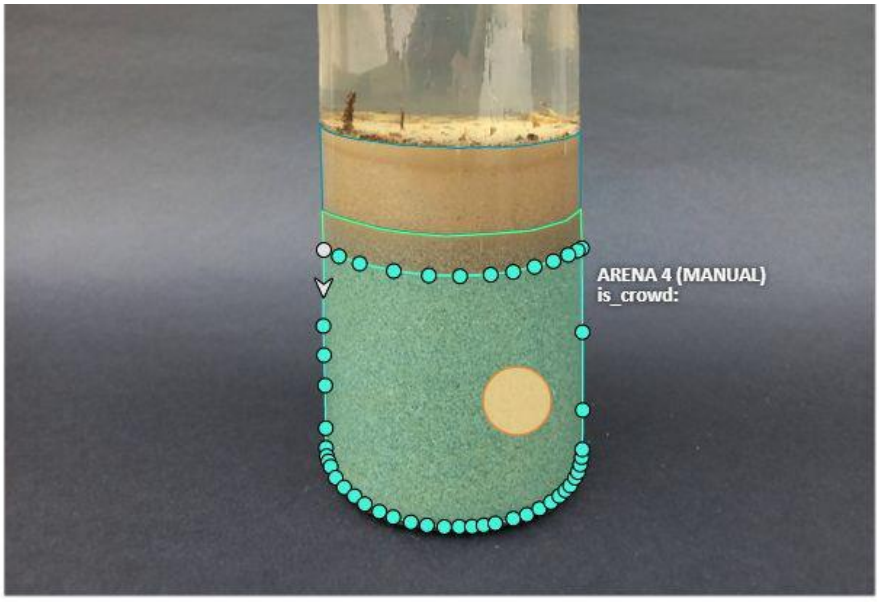
9. Etiqueta de Arcilla en la muestra.



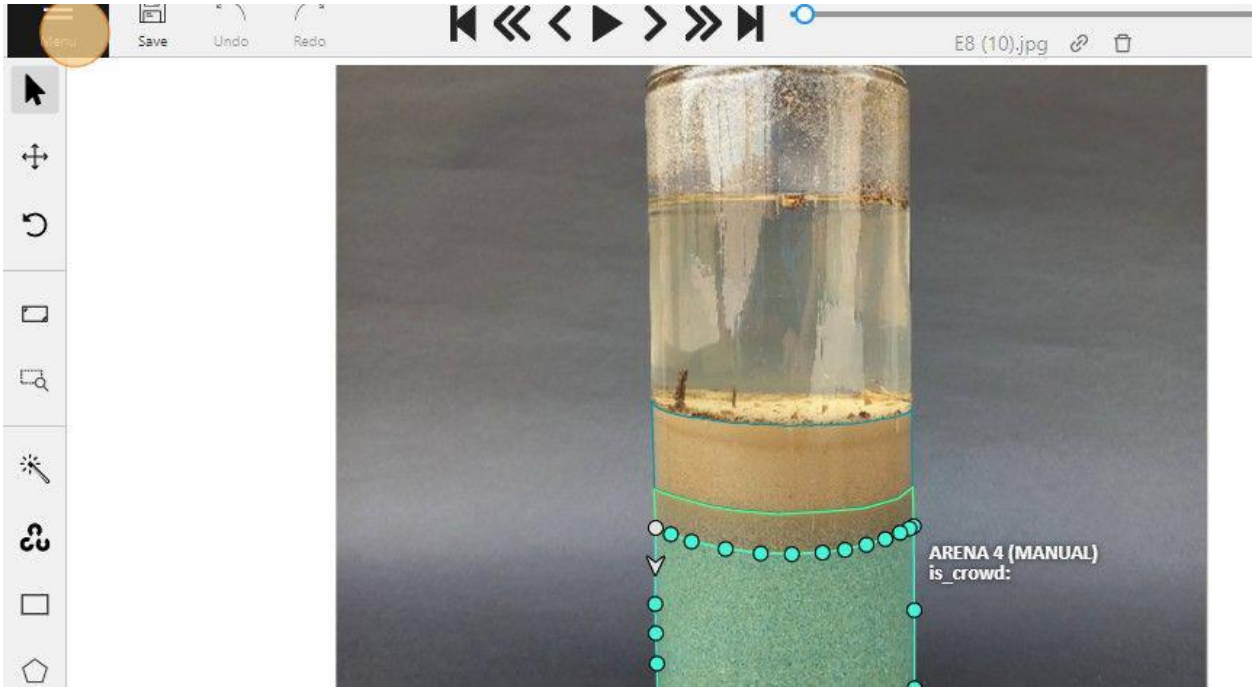
10. Etiqueta de Limo en la muestra.



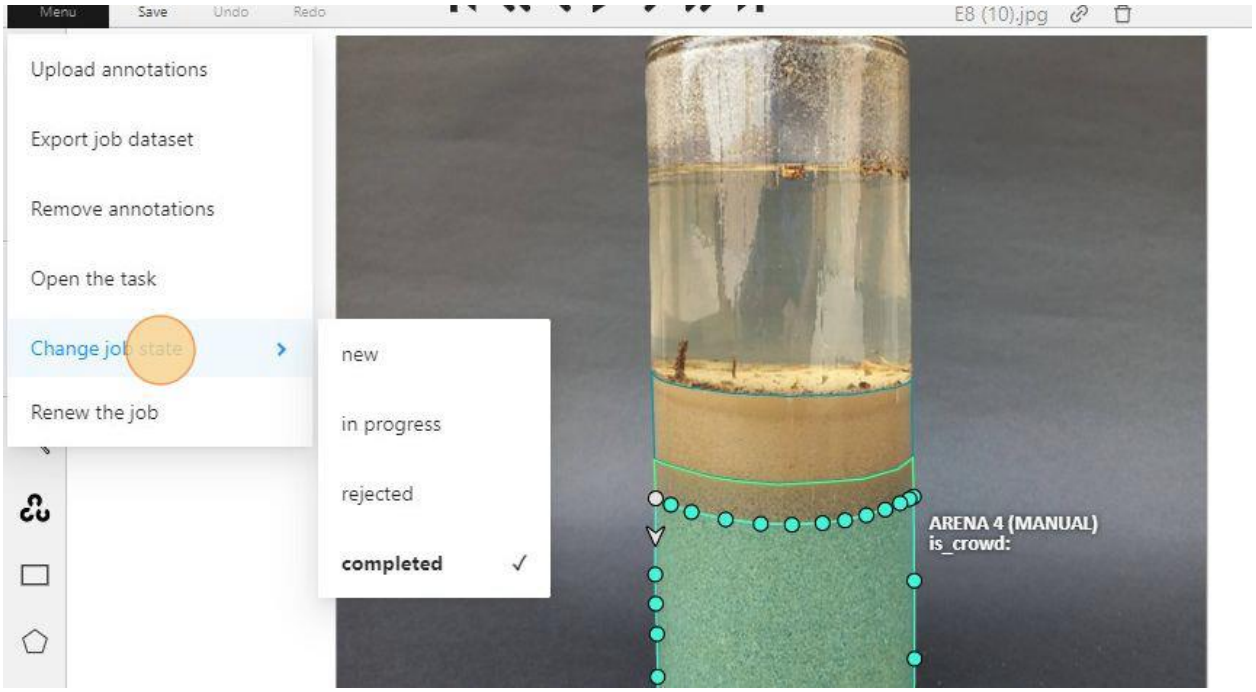
11. Etiqueta de Arena en la muestra.



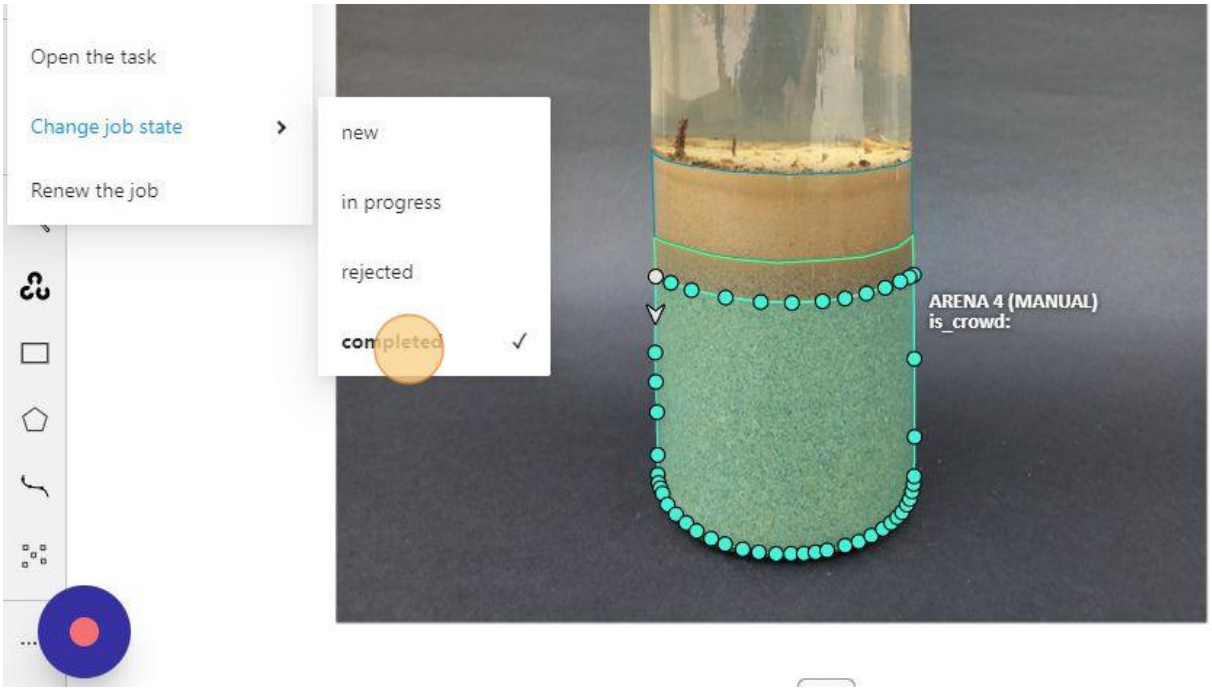
12. Click en “Menú”.



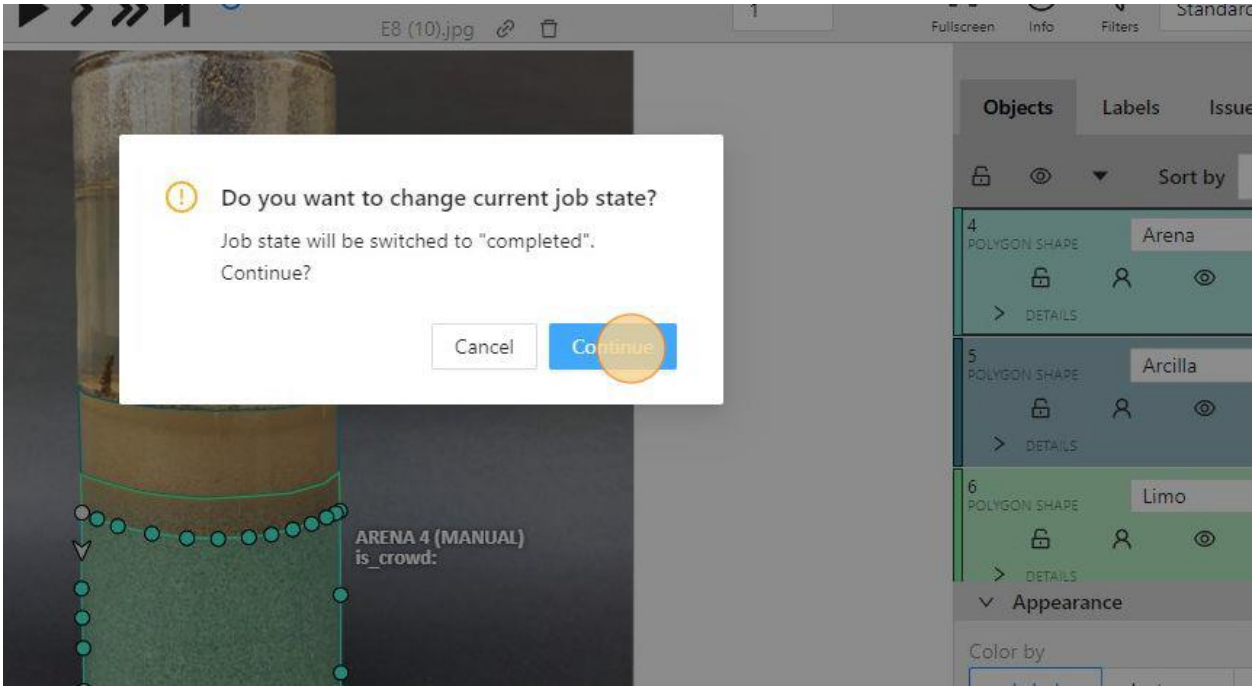
13. Click en “Change job state”.



14. Click en “completed”.



15. Y por último Click en “Continue”.


E8 (10).jpg
Fullscreen
Info
Filters
Objects
Labels
Issue
Sort by
4
POLYGON SHAPE
Arena
DETAILS
5
POLYGON SHAPE
Arcilla
DETAILS
6
POLYGON SHAPE
Limo
DETAILS
Appearance
Color by
Do you want to change current job state?
Job state will be switched to "completed".
Continue?
Cancel
Continue
ARENA 4 (MANUAL)
is_crowd: