UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:

INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:

Ingeniero de Sistemas

TEMA:

"DESARROLLO DE UNA APLICACIÓN WEB PARA LA GESTIÓN Y SEGUIMIENTO DE PROYECTOS DE ENSAYOS DE CAMPO DEL DEPARTAMENTO DE INVESTIGACIÓN DE UNA EMPRESA DE REGISTROS AGRÍCOLAS"

AUTOR:

JAIRO PATRICIO CASTELLANO BENALCÁZAR

TUTORA:

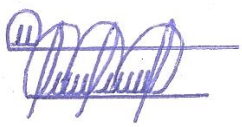
PAULINA ADRIANA MORILLO ALCÍVAR

Quito, marzo de 2021

CESIÓN DE DERECHOS DE AUTOR

Yo, JAIRO PATRICIO CASTELLANO BENALCÁZAR, con documento de identificación N° 1723028419, manifiesto mi voluntad y cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del trabajo de grado titulado: "DESARROLLO DE UNA APLICACIÓN WEB PARA LA GESTIÓN Y SEGUIMIENTO DE PROYECTOS DE ENSAYOS DE CAMPO DEL DEPARTAMENTO DE INVESTIGACIÓN DE UNA EMPRESA DE REGISTROS AGRÍCOLAS", mismo que ha sido desarrollado para optar por el título de: INGENIERO DE SISTEMAS, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia, suscribo este documento en el momento que hago la entrega del trabajo final en digital a la Biblioteca de la Universidad Politécnica Salesiana.



_____

JAIRO PATRICIO CASTELLANO BENALCÁZAR

CI: 1723028419

Quito, marzo de 2021

DECLARATORIA DE COAUTORÍA DE LA TUTORA

Yo, PAULINA ADRIANA MORILLO ALCÍVAR, declaro que bajo mi dirección y asesoría fue desarrollado el Proyecto Técnico, con el tema: "DESARROLLO DE UNA APLICACIÓN WEB PARA LA GESTIÓN Y SEGUIMIENTO DE PROYECTOS DE ENSAYOS DE CAMPO DEL DEPARTAMENTO DE INVESTIGACIÓN DE UNA EMPRESA DE REGISTROS AGRÍCOLAS", realizado por JAIRO PATRICIO CASTELLANO BENALCÁZAR, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana, para ser considerado como trabajo final de titulación.

Quito, marzo del 2021.

PAULINA ADRIANA MORILLO ALCÍVAR
CI: 1715646574

DEDICATORIA

Dedico este trabajo, a mi mamá, hermana y a Princesa.

Jairo Patricio Castellano Benalcázar

AGRADECIMIENTOS

Agradezco a mi alma máter, la Universidad Politécnica Salesiana donde he aprendido y comprendido que el ser profesional es más que sólo conocer la academia, va de la mano del ser y tener calidad humana.

A mi tutora, la Ing. Paulina Morillo, con quien llevo gran estima y fue quien me apoyó e instruyó constantemente a tener un proceso de mejora en mi formación académica.

A mi madre, quien tiene la paciencia de una civilización completa.

Jairo Patricio Castellano Benalcázar

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La gestión de la información ha cobrado gran importancia para las empresas en los últimos años, ya que se ha convertido en un activo, que puede mejorar los procesos y aumentar la rentabilidad del negocio. Sin embargo, la información generada en muchas ocasiones se torna difícil de administrar, debido a que los datos se dispersan en varios medios de almacenamiento como hojas de cálculo o medios no adecuados que dificultan su seguimiento. Por tanto, el presente trabajo tiene la finalidad de desarrollar una aplicación Web que permite administrar la información y datos generados en el departamento de Investigación Plaguicidas para una empresa de Registros Agrícolas, encargada de hacer el registro de productos y moléculas con una entidad de control. La empresa Agroambiente Cía. Ltda. sigue este giro de negocio, siendo el departamento de Investigación Plaguicidas donde se ejecutan los ensayos de campo para su posterior registro. De esta forma, se realizó un análisis y levantamiento de requerimientos con entrevistas, reuniones e información directa obtenida de los técnicos de oficina y campo de la empresa. Luego se construyó la aplicación, utilizando el Framework de desarrollo web Django, adaptando a la aplicación el proceso de elaboración documental y el proceso de ejecución de campo e informe final con la autoridad. Se construyó una base de datos para registrar el ingreso de responsables y actividades en cada una de las fases del proyecto. Finalmente, se logró construir un software Web adaptado al proceso de registro de proyectos plaguicidas que permite su registro y seguimiento desde su inicio hasta su cierre. La aplicación fue sometida a pruebas de caja negra, carga, rendimiento y estrés, mostrando su correcto funcionamiento y efectividad.

ABSTRACT

Information management has become very significant for companies in recent years, as it has become an asset, which can improve processes and increase the profitability of the business. However, the information generated on many occasions becomes hard to manage, because the data is scattered in several storage media such as spreadsheets or unsuitable media that make it difficult to monitor. The company Agroambiente Cía. Ltda. follows this line of business, being the Pesticides Research department where the field trials are carried out for their subsequent registration. In this way, an analysis and a survey of requirements were carried out with interviews, meetings, and direct information obtained from the company's office and field technicians. Therefore, this work aims to develop a Web application that allows managing the information and data generated in the Pesticide Research department for an Agricultural Registries company, in charge of registering products and molecules with a control entity. Then the application was built, using the Django web development framework, adapting the application to the document preparation process and the field execution process, and the final report with the authority. A database was built to record the entry of the responsible and the activities in each phase of the project. Finally, it was possible to construct a Web software adapted to the pesticide project registration process that allows its registration and monitoring from its beginning to its closure. The application was subjected to the black box, load, performance, and stress tests, showing its correct operation and effectiveness.

1. INTRODUCCIÓN

1.1. Antecedentes

A través de los años, las empresas han ido evolucionando al igual que las herramientas utilizadas para llevar a cabo su trabajo diario. Esto significa que, para que las pequeñas y medianas empresas logren crecer, mantener y progresar en un entorno global y cambiante, se deben plantear estrategias que permitan alcanzar su desarrollo empresarial en cuanto a su crecimiento económico, de cultura empresarial, de liderazgo y de gestión de áreas del conocimiento y de la innovación (Delfín y Acosta, 2016).

Por estas razones, las empresas consideran como una estrategia el uso de herramientas web para gestionar sus datos. Los beneficios para gestionar procesos basados en el uso de software web para pequeñas y medianas empresas, son una medida de eficiencia empresarial, aumento de productividad y funcionamiento efectivo para los procesos de la empresa (Ahmad y Sulaiman, 2011).

Tomando en cuenta que cada tipo de empresa genera sus propios tipos de datos e información se tiene que, las empresas dedicadas al campo agrícola generan datos llamados registros, los cuales se usan con el objetivo de controlar la gestión respecto a los cultivos; la información generada se utiliza para que los resultados técnicos y económicos ayuden a mejorar la eficiencia de estas empresas (Prieto y Escalante, 2019). Los datos convertidos luego en información son los elementos esenciales que alimentan el paso diario y la continuidad de una empresa de registros agrícolas (Roman y Chiang, 2017). En el caso de la Empresa Agroambiente Cía. Ltda., una empresa dedicada a Registros Agrícolas, tiene su departamento de investigación plaguicidas donde se genera información relacionada con proyectos de ensayos de campo. La empresa realiza investigaciones sobre cultivos y productos de diferentes empresas clientes, se designa técnicos responsables para trabajo de campo y se almacena esta información en documentos físicos, archivos y hojas de cálculo, así como los resultados de los

procesos y etapas del proyecto los cuales generan un flujo de información importante. A medida que la información relacionada con proyectos de ensayos de campo crece, dicha información se convierte en perecible, ya que no se puede manejar completamente, estudiar ni dar seguimiento a sus datos. Seguimiento que toda empresa busca tener, siendo que los datos representan aquella información útil que se usa como recurso estratégico para el desarrollo y crecimiento empresarial (Gallego, 2018).

Por ende, actualmente gran parte del sector empresarial como lo son las empresas de Registros Agrícolas comprenden la importancia de mantener toda su información de una forma administrada, organizada y que su recuperación sea lo más sencilla posible. Una empresa que no ha dedicado tiempo e interés suficiente para automatizar el manejo de sus datos, buscando medios adecuados para almacenarlos, tener acceso, actualizarlos, consultarlos y en general; mantener una administración efectiva de sus datos, corre el riesgo de perderlos (Gallego, 2018).

1.2. Justificación del Tema

Tener muchas formas de almacenamiento de datos, implica redundancia o gestión no adecuada de los mismos, lo cual impide una correcta recuperación de la información y que sus medidas de almacenamiento no sean óptimas (Sekar y Sethuraman, 2017). La desventaja de programas de escritorio y hojas de cálculo es que exigen recursos computacionales de los equipos en los que se encuentran instalados, a diferencia de una aplicación web con la que el equipo no sufre estos problemas sino por el contrario, consume recursos de un servidor externo reduciendo estos costos computacionales para sí mismo (Herrero y Carmona, 2011).

Para el caso de la empresa Agroambiente Cía. Ltda., la forma actual de almacenar y gestionar esta información de gran importancia dentro del departamento de investigación, para dar seguimiento a los proyectos de ensayos de campo plaguicidas, es usando métodos tradicionales como hojas de cálculo y programas de escritorio genéricos. Los cuales no se ajustan correctamente al giro del negocio y al área en particular, provocando un análisis de datos complejo que no involucra a todos los interesados debido a que la información no está al alcance e inmediatez, es ambigua, solo parcialmente organizada y presenta retrasos en su entrega.

Razón por la cual, se busca lograr un adecuado seguimiento de los proyectos de ensayos de campo en las distintas fases que se dispone para el esquema específico de trabajo que caracterizan a los proyectos plaguicidas. Y contemplando la seguridad de acceso al manejar el concepto de roles al sistema desarrollado. A su vez, esta futura implementación tendría repercusión en la forma en que se puedan analizar dichos datos, dado que permitirá identificar a mayor detalle las fases de cada proyecto y las actividades asociadas a cada una de ellas.

Por otro lado, ya que en la actualidad no se dispone de software que se ajuste a todas las especificaciones de los procesos llevados a cabo por el departamento de investigación de

una empresa de Registros Agrícolas, la aplicación web desarrollada buscará gestionar de manera efectiva los datos y dar seguimiento a los proyectos de ensayos de campo plaguicidas.

Por tanto, el presente trabajo tiene como finalidad el desarrollo de un sistema Web que permita tener un seguimiento para proyectos de Ensayos de Campo Plaguicidas de una empresa con su giro de negocio en Registros Agrícolas.

1.3. Objetivos

1.3.1. Objetivo General

Desarrollar una Aplicación Web para la gestión y seguimiento de proyectos de ensayos de campo del Departamento de Investigación de una empresa de Registros Agrícolas.

1.3.2. Objetivo Específico

Estudiar el estado actual de los procesos del departamento de Investigación de plaguicidas, para una empresa de Registros Agrícolas.

Analizar y diseñar un sistema web que cuente con las funcionalidades de acceso a usuarios, asignación de proyectos, creación de tareas y revisión del estado de los procesos.

Construir un sistema web que permita organizar y dar seguimiento a los proyectos de ensayos de campo del departamento de investigación de plaguicidas.

Realizar pruebas de caja negra, carga, rendimiento y estrés, para comprobar el funcionamiento del sistema web.

1.4. Marco Metodológico

Las distintas metodologías que apoyan el desarrollo de software como Scrum van a permitir que la realización del presente sistema web se mantenga en la línea de buenos tiempos para su culminación y dentro de parámetros eficientes (Choudhary, 2016).

Así, las metodologías seleccionadas serán uno de los marcos sobre los cuales se soporta la correcta ejecución del desarrollo de software, sus fases y consideraciones.

1.4.1. Scrum

Scrum dice ser un modelo para desarrollo ágil, el cual se caracteriza por adoptar el atributo de desarrollo incremental en lugar de una planificación y acción total. Scrum considera tener como base la calidad de los resultados en función del equipo con el cual trabaja e incluir la ejecución paralela de las diversas fases que tenga el proyecto a desarrollarse, en lugar de optar por fases secuenciales (Menzinsky et al., 2016).

Scrum está definido a su vez por una serie de prácticas y guías que brindan soporte a los principios de un desarrollo considerado como ágil. Los cuales son:

Gestión evolutiva del producto, que sustituye a la gestión tradicional o de predicción.

Calidad de resultado, con el conocimiento explícito de los participantes y clientes.

Estrategia de desarrollo incremental, la cual considera iteraciones (sprints) los que serán entregables funcionales.

Como explica Menzinsky (2016) Scrum lleva de manera muy práctica el crecimiento del proyecto tomando en cuenta 4 tácticas:

Revisión de Iteraciones, lo que significa que una vez terminado un sprint se evalúa la funcionalidad de este.

Desarrollo incremental, no existe un cierre de depuración del diseño o arquitectura, éste se puede adaptar conforme se va incrementando.

Autoorganización, según los factores que se presenten y sean impredecibles. Los equipos tienen suficiente organización para adoptar medidas necesarias.

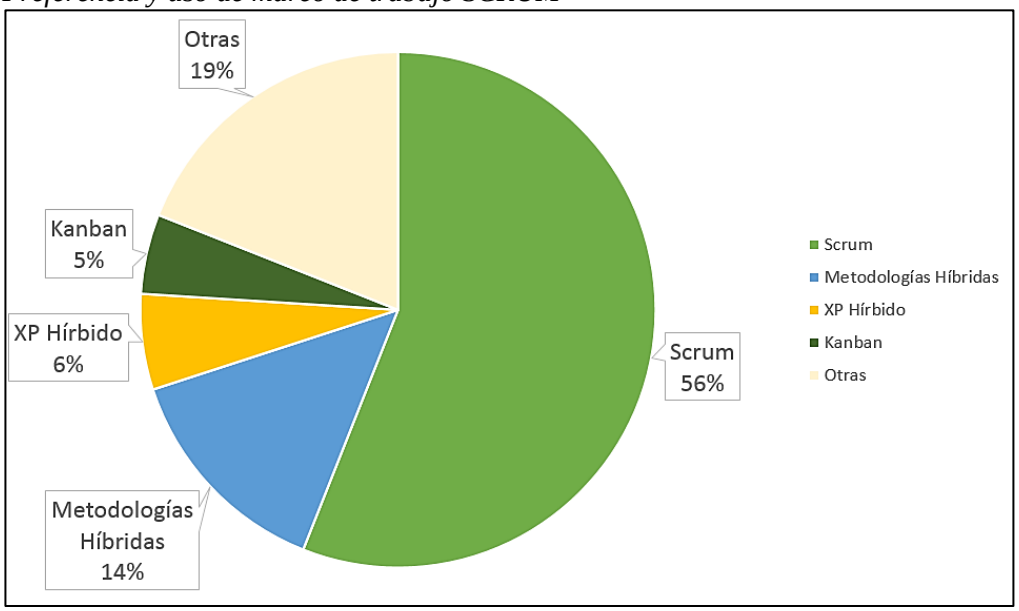
Colaboración, todo el equipo trabaja activamente y brinda las capacidades que dispone sin enmarcarse del todo en un rol definido.

1.4.2. ¿Por qué se eligió Scrum?

El marco de trabajo SCRUM es una de las formas ágiles mayormente utilizadas. (Ereiz y Music, 2019) exponen que SCRUM es con un 56%, líder por sobre metodologías Híbridas, XP y Kanban tal como se muestra en la Figura 1.

Figura 1

Preferencia y uso de marco de trabajo SCRUM



Nota. *La imagen muestra el porcentaje de uso de diferentes metodologías y SCRUM por parte de empresas encuestadas en el 2019 expuestas en la Conferencia Internacional de Ciencias de la computación e Informatización Educativa de la IEEE. Elaborado por: El autor.*

Scrum es utilizado para una gran variedad de disciplinas, siendo el marco de trabajo que adoptan las organizaciones principalmente en proyectos para desarrollo de software. Esto debido al valor añadido que suelen entregar a los clientes (Ereiz y Music, 2019).

Entre las características para adoptar a Scrum como marco de trabajo para el desarrollo del software se tuvo en consideración los siguientes beneficios:

Ofrece formas personalizadas de trabajo para proyectos de desarrollo de software.

Los ciclos incrementales o denominados sprints son sencillos de seguir.

La creación de requerimientos es flexible y puede cambiar en el camino, de modo que Scrum empuja al equipo de desarrollo e interesado del producto a estar comunicados recurrentemente de manera que estén de acuerdo con el producto entregado al final del proyecto.

Adicionalmente, Scrum funciona de forma organizada y se gestiona mediante los mencionados sprints. Cada sprint logra ayudar a que los procesos de creación de una aplicación se ejecuten en etapas y bajo un calendario estipulado. De modo que el dueño del producto comprueba regularmente si las expectativas de lo que espera sea el producto se estén cumpliendo y si no, modificarlo a tiempo (Wang, 2018).

1.5. Esquema del Sistema

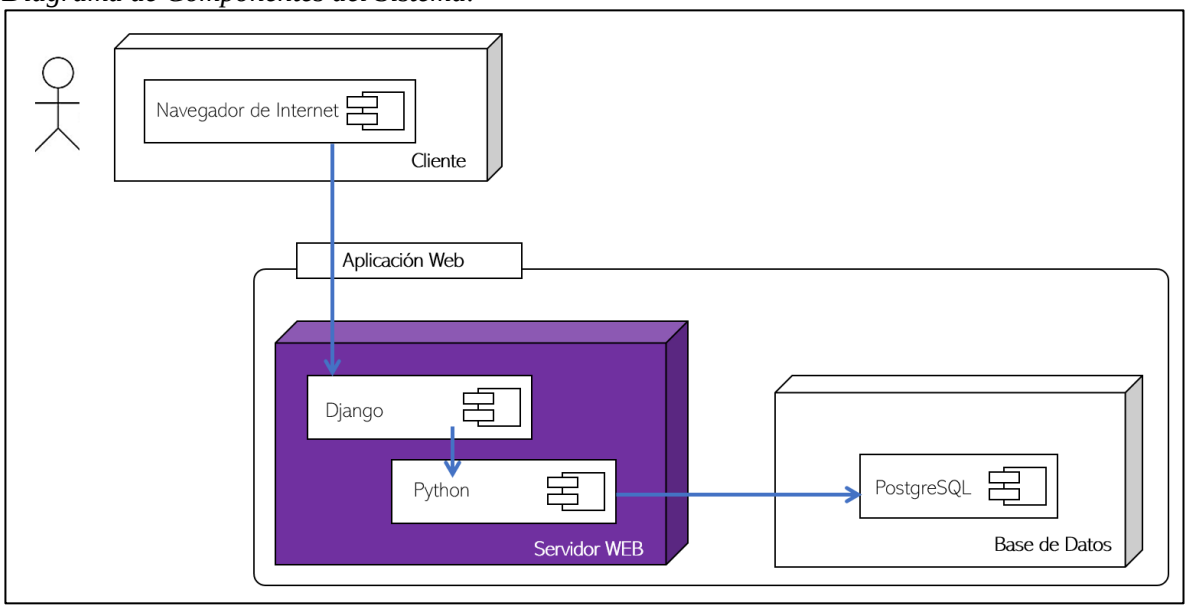
Lo que se planteó de inicio es definir cuál es el esquema de los componentes y cómo estos interactúan entre cada uno de ellos teniendo el panorama global del sistema. Con esto se logra identificar funcionalidades, la manera en que están organizados y los nodos que conforman el sistema global.

Como se puede ver en la Figura 2, los nodos representan en este caso los equipos físicos y se los muestra como cubos. Es decir, para este caso, se considera 3 nodos que van a procesarse en el equipo que se encuentren alojados. El nodo Cliente, procesará todo lo relacionado con el navegador de internet, por otro lado, el nodo del servidor web y de la base de datos, se procesarán del lado del servidor. Otro de los componentes diagramados son los denominados paquetes, que son los organizadores que agrupan los elementos de igual relación, en este caso el paquete denominado Aplicación Web, organiza tanto el servidor web como su base de datos.

Por último, en la misma Figura 2, se tienen las unidades lógicas denominadas componentes, del tipo componentes físicos como: el navegador de internet, PostgreSQL, Django y Python.

Figura 2

Diagrama de Componentes del Sistema.



***Nota**. El diagrama muestra 3 nodos, considerados como el hardware y representados por cubos, lo que significa que para desplegarlo se puede hacer uso de 3 equipos computacionales diferentes. Elaborado por: El autor.*

Con ayuda de este esquema se desarrolló el software Web considerando cuáles son los elementos del sistema y dónde se encontrarán cuando se comuniquen entre ellos.

1.6. Materiales y Métodos

1.6.1. Ubuntu

EL software Ubuntu es un sistema operativo de código abierto basado en el núcleo de Linux y distribuido bajo una licencia General Public License, disponible en múltiples idiomas. Uno de los beneficios de utilizar Ubuntu es que fue pensado para trabajar tanto como equipo de escritorio como para trabajar de servidor (Blanco, 2005).

Muchos optan por levantar sus servidores en este tipo de sistemas operativos ya que las distribuciones de Linux, y hablando de Ubuntu, tiene entre otros, estabilidad en su sistema, todas sus características configurables, y maneja una serie de librerías y paquetes que lo hacen aún más robusto (Ramírez, 2010).

En la Tabla 1 se muestra que Ubuntu es soportado por arquitecturas Intel, AMD y PowerPC.

Tabla 1

Arquitecturas soportadas por Ubuntu.

Arquitectura	Compatible	Sistema
Intel	X	x86, x64
AMD64	X	x64
PowerPC	X	Apple iBook, PowerBook, G4, G5

Nota. *Lista de arquitecturas que son soportadas por el sistema Operativo Ubuntu. La X indica que es compatible con el tipo de sistema de la columna final. Elaborado por: El autor, a través del Libro Manual Básico Ubuntu GNU-Linux página 8.*

1.6.2. Python

Norton et al. (2005) sostienen que el lenguaje de programación interpretado Python es un lenguaje rico en muchos atributos además de tener compatibilidad con varias plataformas y varios paradigmas. Una de las características de Python es su código sencillo y sintaxis basados en las de otro lenguaje llamado ABC. González (2018) describe 4 de estas características que más representan a Python y son preferidas por los desarrolladores, siendo:

La característica de tipado dinámico la cual no exige declarar variables asociadas a su tipo, sino que se adaptarán al valor que se le asigne al momento de ejecutar el programa.

El tipado fuerte que evita que una variable de un tipo cambie por otro tipo, no se adapta al contexto como en otros lenguajes, pero esto tiene el objetivo de prevenir errores.

El intérprete de Python es multiplataforma, compatible en Windows, Mac OS, y distribuciones de Linux.

Maneja el conocido paradigma de programación orientado a objetos, donde se usan objetos y clases.

1.6.3. Django

Django representa uno de los frameworks más utilizados en la actualidad, es muy competitivo cuando se trata de desarrollo ágil web y su uso ahorra un 10% de memoria al utilizarlo (Zhou et al., 2013).

Con Django se consigue producir y mantener aplicaciones de tipo Web que involucran la creación de un producto de alta calidad y que varios de los aspectos de desarrollo tradicionales mejoren significativamente (Holovaty y Kaplan-Moss, 2015).

1.6.4. PostgreSQL

PostgreSQL es un Sistema Gestor de Base de Datos (SGBD) relacionales desarrollado desde 1977. Se sabe que PostgreSQL es un proyecto de código abierto. Y es por esta misma característica que se tiene acceso a cifras reales de evaluación al compararlo con sus similares, como también se tiene acceso a estadísticas que presenta el SGBD relativo a su rendimiento, a diferencia de empresas de código cerrado que no hacen públicas sus cifras como es el caso de Oracle (Worsley y Drake, 2002).

Según el libro de Marqués (2011) se recomienda utilizar el motor de base de datos libres como PostgreSQL debido a que logra un balance adecuado cuando se considera variables de costo, características, funcionalidades y rendimiento. En la Tabla 2, se logra ver una

comparativa entre 4 de los principales SGBD donde se nota que PostgreSQL funciona en 6 de los 7 sistemas operativos más comunes que están listados.

Tabla 2

Compatibilidad de Gestores de Bases de Datos con sistemas operativos.

SISTEMA OPERATIVO	ORACLE	MYSQL	SQL SERVER	POSTGRESQL
AIX	X			
HP-UX	X			X
Linux	X	X	X	X
OS-X	X	X		X
Solaris	X	X		X
Windows	X	X	X	X
FreeBSD				X

Nota. *Tabla comparativa de compatibilidad de 4 de los principales Gestores de Bases de Datos con 7 de los sistemas operativos más comunes. Elaborado por: El autor.*

PostgreSQL también se sitúa en las 4 primeros SGBD relacionales más populares la según "db-engines.com".

1.6.5. JavaScript

(Dayley, 2014) JavaScript es considerado un lenguaje de programación como cualquier otro. Pero a diferencia de los lenguajes de programación conocidos, éste es construido, analizado gramaticalmente y ejecutado sobre un explorador de internet.

Esto implica, que al programar con este lenguaje se pueda desarrollar aplicaciones complejas que tendrán acceso con los distintos componentes del explorador de internet y servidor.

Uno de los beneficios de emplear este lenguaje, es que JavaScript es un código dinámico que maneja los elementos web sin necesidad de recargar la página.

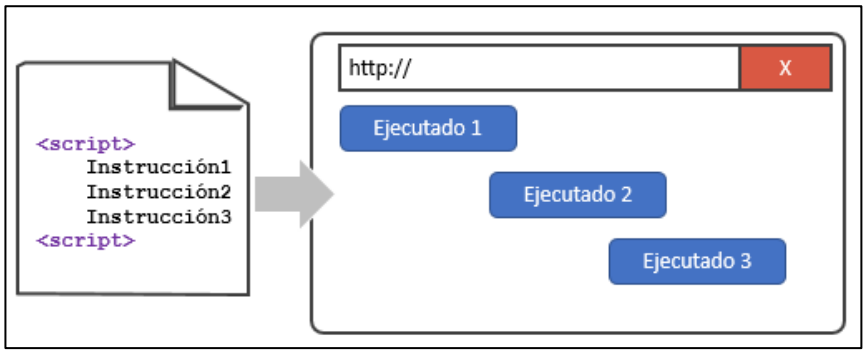
Como indica Gauchat (2017), la forma en que un código JavaScript funciona es que, una vez brindada la serie de instrucciones, el navegador hace que se ejecuten secuencialmente

indicando al sistema la acción que se pretende realizar y posteriormente mostrándola en pantalla. Este proceso se puede observar en la Figura 3.

JavaScript puede incorporarse a un documento usando técnicas distintas. En la técnica en línea, el código se inserta en cualquier elemento por medio de atributos. Insertar el código de forma externa se consigue utilizando archivos y haciendo un llamado a éstos.

Figura 3

Ejecución de JavaScript en un navegador.



Nota. *Ejecución de una serie instrucciones JavaScript mediante un navegador. Elaborado por: El autor.*

JavaScript comparte características similares a otros lenguajes de consulta y a la par su librería llamada JQuery puede equiparse a consultas del lenguaje SQL para bases de datos. Esta participación provee facilidades para diseñar y construir páginas web y aplicaciones que sean muy interactivas.

JavaScript es uno de los elementos más importante al momento de crear aplicaciones web por su comportamiento dinámico (Dayley, 2014).

1.6.6. Atom

Atom es una de las múltiples Interfaces para Entorno de Desarrollo o conocidos comúnmente como Editores de Código el cual fue desarrollado por GitHub. Siendo un editor de código libre, éste, se encuentra registrado bajo una licencia de código abierto, MIT y recibe mantenimiento por la comunidad de desarrolladores desde el 2015.

Atom es soportado y útil para el uso de múltiples lenguajes entre los que listan JavaScript y Python. Su forma de trabajo es sencilla y dispone de texto resaltado, identación y autocompletado lo cual facilita mucho la tarea de escribir código (github.com/atom, 2021).

1.6.7. Bootstrap

Como indica (Gallego, 2018), Bootstrap se considera como uno de los frameworks más conocidos y populares del medio. EL mismo que fue desarrollado por la empresa Twitter.

El objetivo de los frameworks de diseño y en particular Bootstrap es lograr brindar una experiencia de tipo responsiva, es decir que se adapta a diferentes dispositivos de diferentes dimensiones. Bootstrap dispone de complementos integrados de JavaScript, fuentes de letras y controladores para formularios.

Gallego (2018), Bootstrap es soportado por los navegadores Chrome, Firefox, Opera, Safari e Internet Explorer en su versión indicada en la Tabla 3.

Tabla 3

Navegadores soportados por Bootstrap.

NAVEGADOR	VERSIÓN SOPORTADA	REVISIÓN DE NAVEGADOR
Chrome	4 +	87.0.4280.88
Firefox	4 +	84.0.2
Opera	11.0 +	73.0.3856.284
Safari	14.0 +	14.0
Internet Explorer	8 +	Estable a partir de versión 7

Nota. *Tabla informativa sobre la versión de navegador que soporta Bootstrap. El símbolo (+) significa que Bootstrap es soportada por la versión de navegador indicada o versiones superiores. Elaborado por: El autor.*

1.6.8. Apache JMeter

Es una herramienta de código abierto para pruebas de software. Fue desarrollada por Apache Software Foundation en el lenguaje de programación java. Entre sus diversas

características están el poder crear ilimitado número de pruebas relacionadas con rendimiento, capacidad, tiempo, entre otras variables.

JMeter crea el número de peticiones indicadas y al ejecutarse pasa esas solicitudes al servidor, el cual responde y JMeter se encarga de guardar todas las respuestas nuevamente. Actualmente se encuentra en su versión 9 y es compatible con Ubuntu, Windows, Linux y Mac. Para poder ejecutar la aplicación también es necesario tener instalado el JDK de java (Rungta, 2019).

Tabla 4

Tabla comparativa para Software para Pruebas.

Software de Prueba	**¿Es Libre?**	**Plataformas Compatibles**			**Tipos de Pruebas**
		Windows	**Linux**	**Mac**	
Apache JMETER	Sí	X	X	X	Rendimiento Carga
MICRO FOCUS (HP) LOADRUNNER	No	X			Rendimiento Carga Calculador de carga Pruebas complejas
SMARTBEAR LOADUI	Versión libre limitada	X			Rendimiento Carga Pruebas complejas
IBM RATIONAL PERFORMANCE TESTER	No	X	X	X	Rendimiento Carga Protocolos SAP Pruebas complejas

Nota. *Tabla comparativa de Apache JMeter con 3 softwares de prueba. Elaborado por: El autor.*

En la Tabla 4, se puede observar que Apache JMeter dispone de los dos tipos de pruebas más comunes para pruebas de software y que es compatible con 3 Sistemas Operativos diferentes estando a la altura de programas licenciados como los de IBM.

CAPÍTULO I

2. MARCO TEÓRICO

2.1. Desarrollo de Software Profesional

El desarrollo de software se lo considera como una actividad de tipo profesional, donde el software que es desarrollado se lo realiza no de manera genérica sino como una respuesta puntual a un negocio en específico, la cual puede integrarse con otros tipos de software y dispositivos, como serían los sistemas de información. El software profesional está destinado para usarse no única ni exclusivamente por su desarrollador (lo cual carecería de sentido), sino por el contrario, para el uso de alguien más, se lleva un equipo de trabajo para su desarrollo en lugar de hacerlo de forma individual y una característica que posee es que puede cambiar y mejorarse al paso del tiempo (Sommerville, 2011).

2.2. Ingeniería de Software

La ingeniería de Software es la materia de la ingeniería que tiene por intención abordar los aspectos totales que intervienen para la realización de un software como lo plantea Sommerville (2011), considerando las primeras fases de su especificación hasta las fases finales como el mantenimiento, luego de su puesta en marcha. En la definición se enmarcan dos términos esenciales de disciplina y producción de software tal como lo indica Sommerville (2011):

Disciplina de la ingeniería, los profesionales ingenieros tienen la tarea de hacer que las cosas funcionen, con lo cual son capaces de aplicar conceptos, metodologías y también de usar herramientas apropiadas para el mejor desempeño. A su vez buscan solventar problemas, inclusive cuando no existen teorías o métodos que ayuden a un caso particular. Los profesionales ingenieros, a su vez, mantienen consciente el hecho de las posibles limitaciones financieras o políticas de cada empresa.

Aspectos completos de la producción de software, la ingeniería de software no únicamente se va a centrar por los atributos respecto a la parte técnica al momento de realizar el software, por el contrario, también toma interés en actividades de monitoreo del software, elaboración de adecuadas herramientas y la aplicación usando teorías y métodos que favorezcan y faciliten la fabricación del software en cuestión.

2.3. Software Libre

(Blanco, 2005) explica que el interés del software libre se dio en función de las notables limitaciones que existían en un software Propietario. Un software propietario es todo aquel programa, aplicativo, sistema operativo y todos sus afines los cuales tienen un contrato para un uso exclusivo o específico una vez hecha su compra. Es decir, los softwares propietarios requieren ser comprados o adquiridos por quien los creó de la manera en que acuerden las partes y que en general beneficia al creador y básicamente presta su uso a quien lo adquiere. Y en general cuando un software propietario presenta fallas, y a pesar, que una persona estuviera capacitada o tuviera el conocimiento suficiente para poder arreglarlo, no podía hacerlo debido a este mismo contrato que lo impedía.

Por el año de 1980 Richard Stallman, fundó el proyecto GNU para software libre debido a las constantes restricciones que se presentaban al utilizar software propietario mientras trabajaba en el Instituto de Tecnología de Massachusetts (MIT) que abandonó por esos mismos años.

Un software libre menciona estos mismos tipos de programas informáticos pero que a diferencia de los propietarios no existe un contrato de por medio ni intercambio monetario, sino por el contrario existe una licencia que permite copia, libre distribución, modificación y que es apoyado por una comunidad y por ende puede ser mejorado. Con esto se rompe los obstáculos de del intercambio, compartición y acceso al código (Fogel, 2007).

Como en todo, el software libre puede tener ventajas y desventajas. Las cuales se comparan en la Tabla 5.

Tabla 5

Ventajas y desventajas de usar software libre.

VENTAJAS	DESVENTAJAS
Adquisición de bajo precio o bajo ningún costo. Código fuente disponible para ser modificado y mejorado. Utilizar, reutilizar y copiar el software bajo una libre distribución o licencia de software libre. El software no es dependiente de ninguna empresa o corporación en particular sino mas bien de una comunidad que apoya sus problemas y mejoras. Brinda acceso a múltiples usuarios y empresas cuya adquisición con software propietario se dificultaría.	En la mayoría de los proyectos de software libre que son desarrollados por una comunidad no existen garantías. El costo de un software libre puede dirigirse a la capacitación que requiere para utilizar el software quizá en muchos casos no muy conocido. Las modificaciones sobre un código pueden hacerse sin previo aviso y sin requerir autorización aparente y versiones previas quedar sin soporte con el objetivo de migrar a una nueva versión.

Nota. *Comparativa entre ventajas y desventajas de usar software libre. Elaborado por: El autor, a través del Libro Manual Básico Ubuntu GNU-Linux páginas 13-15.*

2.4. Entornos Virtuales

En Docs.python.org. (2021) se explica que cuando se trabaja con Python y se crean determinadas aplicaciones, dicha aplicación por lo general está desarrollada y enriquecida en el ambiente de la versión de Python instalado. En otras palabras, las aplicaciones en ciertas ocasiones van a requerir que se tenga instalado una versión de Python que sea específica para que pueda funcionar normalmente. Esto a menudo, ya que es posible que las librerías y paquetes que hayan sido instalados para la versión en cuestión cambien, se renombren o dejen de funcionar en versiones posteriores de Python.

De modo que si se planea cambiar la misma aplicación a otra versión de Python pueda chocar con la compatibilidad y el trabajo de rehacerla sea arduo.

La solución brindada para estos casos es el denominado entorno virtual, el cual es un contenedor o en palabras sencillas, es un directorio en el cual se aloja una versión instalada de Python que será independiente hasta del mismo sistema operativo. De modo que maneja y trata de manera aislada todo lo que se instale dentro.

Los entornos virtuales no son obligatorios para iniciar un proyecto o aplicación en Python, sin embargo, son considerados y recomendables debido a que, al contener una versión específica de Python y un conjunto de paquetes o dependencias, no va a intervenir en lo absoluto con el Python originalmente instalado ni mucho menos con el sistema operativo u otros proyectos que se creen en el mismo computador.

Por lo cual, cuando se configura un ambiente virtual que sea diferente para los proyectos trabajados, los marcos de trabajo web también pueden ejecutarse en diversas versiones sin que esto genere conflictos (Raza et al., 2017).

2.5. Framework Web

Para contestar lo que significa un Framework Web se puede considerar lo que indica el libro de Adrian Holovaty y Jacob Kaplan-Moss (2015), sobre el desarrollo de aplicaciones web escritas en lenguaje Python. Si no se utiliza un framework se tendría el trabajo adicional de empezar su desarrollo desde prácticamente cero quizá utilizando un modelo estándar de Common Gateway Interface que se usa para el desarrollo de aplicaciones Web. De aquí entonces saltan una serie de interrogantes como:

Forma de Conexión a Base de Datos

Cierres y aperturas de conexión

Seguridad y Configuración de entorno para aplicaciones similares

Rediseño de la página.

Todos los anteriormente mencionados son los que un Framework Web trata solventar. De modo que el Framework Web brinda la infraestructura necesaria para lograr programar aplicaciones de manera que se pueda centrar en lo realmente importante como lo es el programar con código pulido y limpio sin necesidad de tener que crear o desarrollar esquemas que ya están creados y desarrollados.

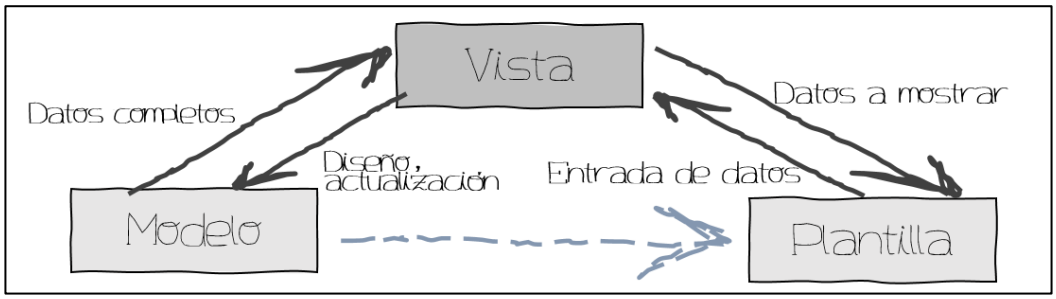
2.6. Modelo Vista Plantilla

Tanto el Modelo Vista Controlador (MVC por sus siglas en inglés), como el Modelo Vista Plantilla (MTV por sus siglas en inglés) son considerados como dos sistemas para la construcción de aplicaciones según indica (González, 2015).

Donde cada una de estas siglas son los elementos que manejan dichos aspectos. El Modelo, está relacionado con el manejo y gestión de la base de datos, Plantilla hace referencia al nivel de presentación todo lo que podría mostrarse en una página web y como final, Vista, la cual se encarga y podría decirse que es la capa intermedia entre las anteriores dos, ya que controla la lógica de negocios conectando por ejemplo qué modelo con cuál plantilla. Este esquema es el mostrado en la Figura 4 donde indica la interacción que tienen los componentes de este modelo cuando se comunican entre ellos y lo que entregan (Gour, 2019).

Figura 4

Patrón de diseño Modelo Vista Plantilla.



Nota. *Cada flecha de la imagen representa la relación directa y bidireccional que existe en este modelo, La vista usa se comunica con el modelo quien le pasa lo requerido si tiene permiso para mostrarlos en una plantilla que ha sido solicitada. Elaborado por: El autor.*

Holovaty y Kaplan-Moss (2015), indican que Django de hecho es un framework que usa el patrón de diseño MTV.

2.7. Bases de Datos Relacionales

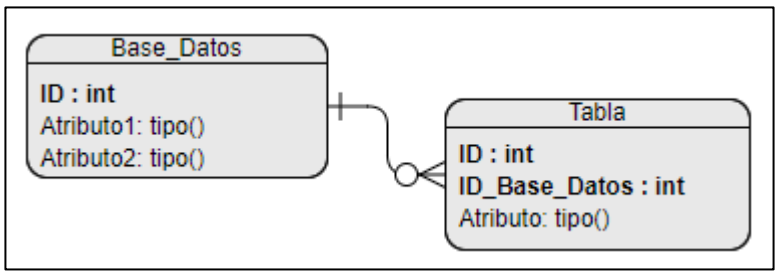
Una relación es cualquier vínculo que une de algún modo dos entidades. Las Bases de Datos Relacionales siguen un modelo relacional. En donde sus entidades se encuentran unidas entre sí. El modelo relacional es uno de los modelos de datos utilizado para aplicaciones comerciales y transacciones de datos. Tiene una alta posición de uso dada su simplicidad en comparación de modelos como los jerárquicos y de red.

Una base de datos relacional está constituida por un conjunto de tablas, donde cada una de estas tablas tienen un nombre único (La Figura 5 representa cómo está relacionada una tabla con la otra, por ejemplo, en este caso, la tabla denominada Tabla, tiene un atributo llamado ID_Base_Datos que hace referencia al atributo principal ID de la tabla denominada Base_Datos). En este tipo de bases cada fila supone una relación para el conjunto de valores, llamada entidad o registro y cada columna un representa un atributo de cada entidad. De modo que las tablas en palabras simples son un conjunto de entidades. La forma en que las tablas se

vinculan entre sí es mediante un atributo único de una primera tabla que se repetirá como atributo foráneo de una segunda tabla (Silberschatz, et al., 2011).

Figura 5

Relación entre tablas por un identificador.



Nota. *La figura muestra dos tablas relacionadas, donde el identificador ID de la primera tabla, pasa a una segunda tabla. Elaborado por: El autor.*

CAPÍTULO II

3. ANÁLISIS Y DISEÑO

Para diseñar la estructura que compone el sistema desarrollado usando el Framework Django, en la fase de análisis se tuvo reuniones con los encargados del departamento de Investigación Plaguicidas (dueños del producto) y con los Técnicos de campo y oficina. Adicionalmente se obtuvo información por medios escritos, de correo electrónico, archivos y con documentos que redactan los procedimientos que se manejan en el área, mediante el cual se consigue identificar el flujo de trabajo del departamento de Investigación Plaguicidas. Otra de las consideraciones para la construcción de la aplicación fueron el tamaño del departamento de investigación que no supera los 10 técnicos entre campo y oficina y que se ha adaptado a los recursos que se dispondría en la puesta en entorno real de ésta. Por esta razón también se ha considerado características físicas de gama media para el equipo servidor con Sistema Operativo Ubuntu de 64 Bits que soportó la aplicación web teniendo:

Memoria Física: 2.9 [GB]

Almacenamiento: 52,00 [TB]

Procesador: Core i3 de 4 núcleos de 2.9[GHz].

3.1. Procesos del departamento de Investigación Plaguicidas

Actualmente para que un trabajo, (refiriéndose a un Proyecto de investigación Plaguicidas que involucra realizar el registro de un producto con la Autoridad competente) llegue a su completitud, requiere de un conjunto de pasos consecutivos en los cuales intervienen los técnicos y los datos que generan en cada uno de estos pasos. Estos trabajos general alrededor 6 documentos por cada proyecto con un estimado de 38 a 90 hojas, teniendo alrededor de 15 proyectos por mes con un tiempo de realización de 3 a 12 semanas para completar un registro en el peor de los casos.

Con el análisis realizado se llegó a identificar las partes que componen el flujo completo de trabajo de lo que es, en resumen: Crear un proyecto de investigación Plaguicidas dentro de la empresa a pedido de una empresa cliente y concluir este mismo trabajo al entregar a la empresa cliente el registro realizado del producto con la Autoridad (La Autoridad es toda empresa ecuatoriana que se encarga de la regulación y control de productos Fito y Zoosanitarios). Para llevar a cabo este trabajo, existen dos procesos diferenciados. El primero descrito en la Figura 6, el cual tiene que ver con la Elaboración de documentación y Revisión. En este proceso se describen 14 pasos o etapas (los cuales pueden observarse a más detalle en la Figura 6) que tienen que ver con designar técnicos que elaboran documentos llamados Fichas Técnicas, Protocolos y Permisos de Importación o PIM y su respectiva revisión y aprobación, cada uno de los cuales tiene como requisito haber terminado el documento anterior. Estos documentos se caracterizan por:

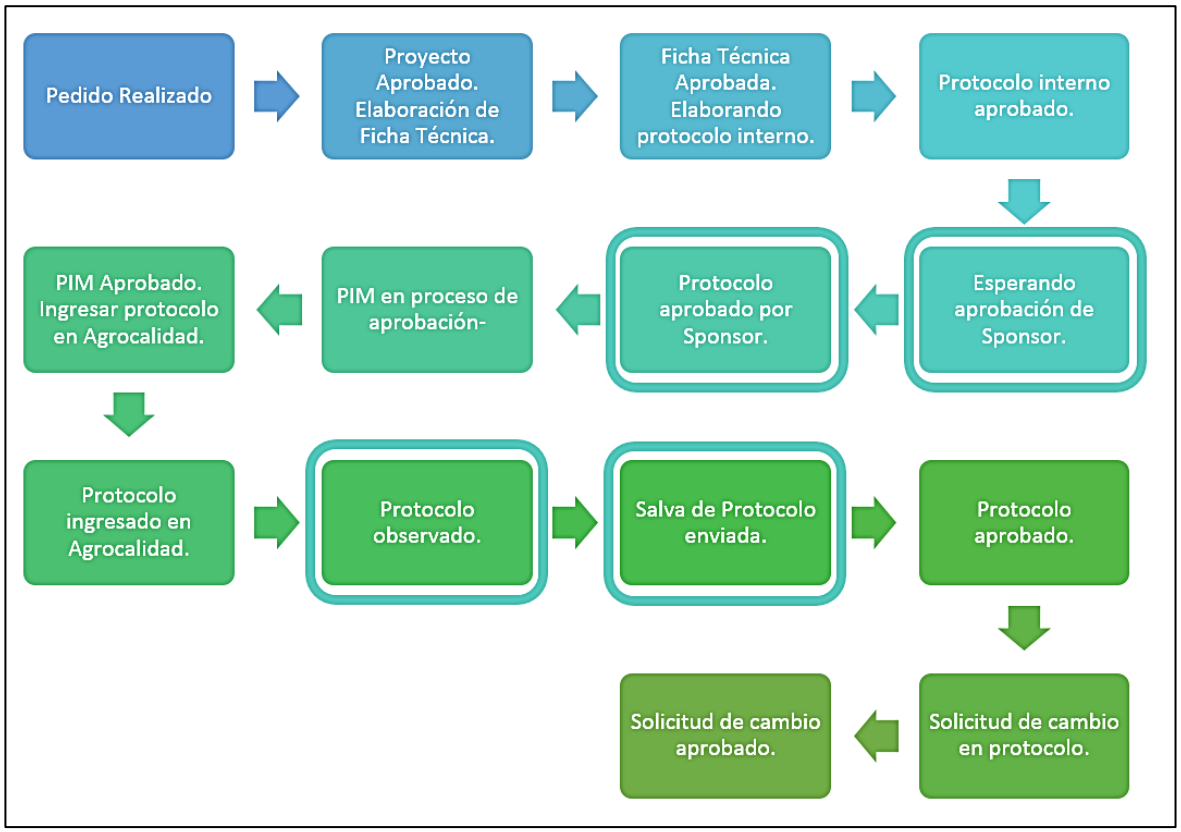
El documento Ficha Técnica requiere la aprobación de Pedido de Proyecto y toda la información asociada al producto a registrar y la empresa cliente.

El documento Protocolo requiere la Ficha Técnica para ser elaborado y aprobado de manera interna y posterior a esto poder ser ingresado a revisión con la Autoridad. En cuyo caso al no ser aprobado, requiere la elaboración de un documento adicional llamado Salva para poder ser ingresado nuevamente con la Autoridad. El mismo protocolo aprobado de manera interna en ocasiones puede ser solicitado por la empresa cliente o sponsor para su aprobación previo a ser ingresado.

El documento PIM requiere el Protocolo Interno elaborado para poder ingresar a revisión con la Autoridad.

Figura 6

Flujo de trabajo (Elaboración de documentación y Revisión).



Nota. *El diagrama describe el primer proceso para registro de un producto y sus 14 etapas. Las etapas con un doble recuadro pueden estar o no presentes en algunos casos. Elaborado por: El autor.*

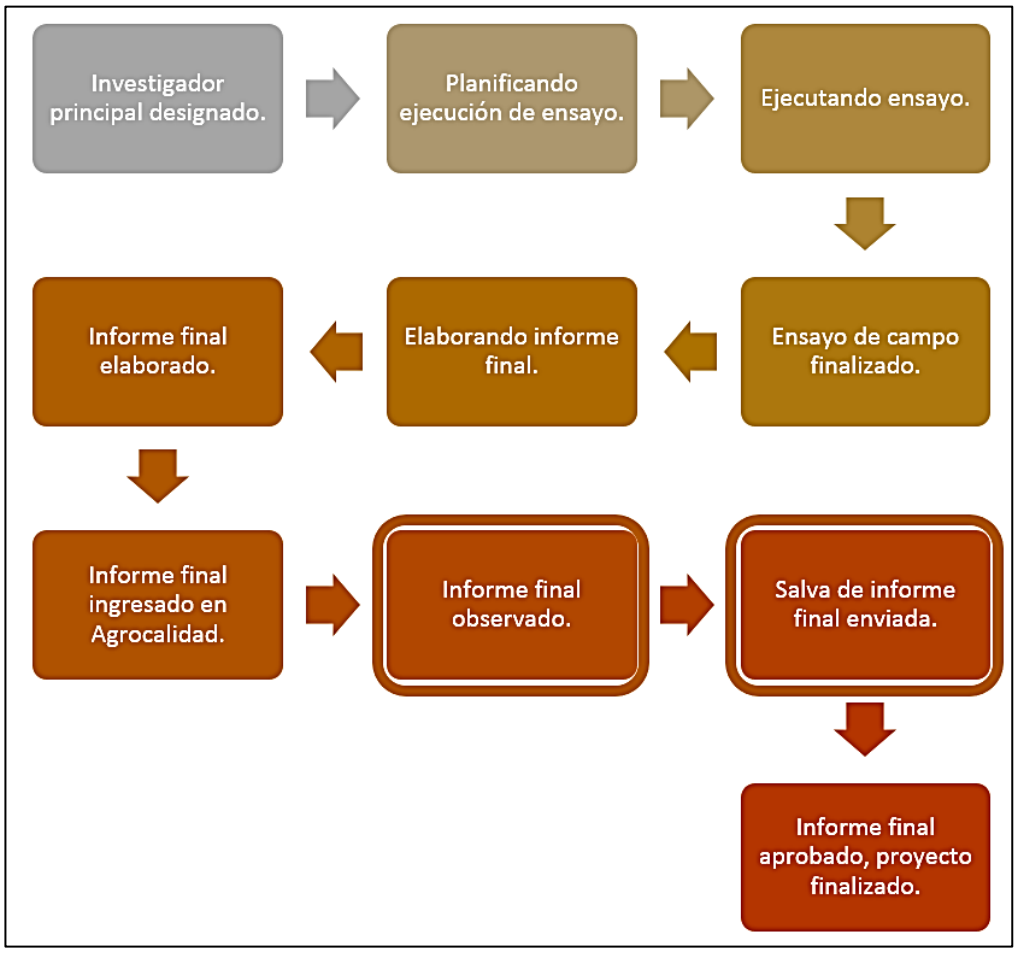
En la Figura 7 se encuentra el diagrama del segundo proceso. Este segundo Proceso de Ejecución de campo y Elaboración de informe final está compuesta de 10 etapas donde se designa a los técnicos responsables de ejecución de campo, se ejecuta el ensayo (se realiza el análisis y estudio del producto a registrar en campo y se obtienen resultados) y con el ensayo ejecutado se elabora el documento llamado Informe Final para su respectiva revisión y aprobación.

El documento Informe Final requiere de la aprobación hecha por la Autoridad del documento Protocolo y haber sido completada la ejecución de campo para poder ser elaborado. Posterior a esto es ingresado a revisión con la Autoridad. En cuyo caso al no ser aprobado,

requiere la elaboración de un documento adicional llamado Salva para poder ser ingresado nuevamente con la Autoridad.

Figura 7

Flujo de trabajo (Ejecución de campo y Elaboración de informe final).



Nota. *El diagrama describe el segundo proceso para registro de un producto y sus 10 etapas. Las etapas con un doble recuadro pueden estar o no presentes en algunos casos. Elaborado por: El autor.*

Los diagramas explican el comportamiento y la dirección que tendrá cualquier proyecto de investigación una vez iniciado. Dentro de este Flujo de trabajo también se identifica 3 elementos fundamentales que son:

El Evento (Explícito)

Datos Generados (Implícito)

Interventor (Implícito)

Cuando se habla del Evento, se refiere claro está a la descripción del estado en el cual se encuentra un proyecto determinado. El mismo que se identifica claramente con solo leerlo. Por ejemplo: "Esperando aprobación de Sponsor", lo cual significa que para que el Evento pueda cambiar, requerirá que un Cliente Sponsor apruebe un documento llamado Protocolo. Los Eventos cambian en función de su aprobación, asignación de responsables o entrega de documentos. Como segundo se tienen los datos generados, los cuales describen o bien a información como fechas y atributos, como también a documentación. Como final de este proceso se encuentran los Interventores que son los responsables de elaborar o designados a llevar el evento. Todos estos elementos con sus tareas, entregables y responsables se los puede ver en la Tabla 6.

Tabla 6

Procesos del Departamento de Investigación Plaguicidas.

Proceso	**Fase**	**Tareas**	**Salidas (Información o documentación generada)**	**Responsable**
Elaboración de documentación y Revisión	Previa	Petición de Proyecto (Registro de Producto). Datos de Empresa Cliente, Datos de Producto, Tipo de Trabajo, Fecha	Datos y Fechas.	Alta Gerencia
	Inicio de Proyecto Plaguicida	Aprobación para Inicio de Proyecto, Datos de Petición, Aprobación por, Fecha, Responsable de elaboración de Ficha Técnica.	Datos y Fechas.	Alta Gerencia
	Elaboración de Documentos	Ficha Técnica, Revisiones, Revisor, Fecha de Revisión, Fecha de Aprobación, Responsable de elaboración de Protocolo Interno. Protocolo Interno, Revisiones, Revisor, Fecha de Revisión, Fecha de Aprobación, Responsable de elaboración de Protocolo Externo.	Datos, Fechas y Documentos.	Técnico de Oficina y Técnico de Oficina Coordinador de Proyecto
	Entrega de Documentos	Entrega Protocolo Externo, Revisiones, Fecha de	Datos, Fechas y Oficios de Resolución.	Técnico de Oficina

		Revisión, Observaciones, Fecha de Aprobación.		
Fase de Ejecución de campo y Elaboración de informe final	Ejecución de Campo	Oficios de Resolución, Entidad supervisora, Responsable de Campo, Inicio y Fin de Trabajo, Documentos de campo	Datos, Fechas y Oficios de Resolución.	Investigador principal de Campo, Técnico de Campo
	Elaboración de Documentos	Informe Final, Revisiones, Revisor, Fecha de Revisión, Fecha de Aprobación, Responsable de entrega de documentos	Datos, Fechas y Documentos.	Investigador principal de Campo, Técnico de Campo
	Entrega de Documentos	Entrega Informe Final, Revisiones, Fecha de Revisión, Observaciones, Fecha de Aprobación.	Datos, Fechas y Oficios de Resolución.	Técnico de Campo

Nota. *En la tabla se hace una descripción más a detalle de los procesos del departamento de Investigación y sus fases. Elaborado por: El autor.*

3.2. Requerimientos

Para el desarrollo del sistema se han estudiado y analizado los requerimientos necesarios con el fin de obtener un sistema adecuado y ajustado a las necesidades planteadas. Desplegando así los dos tipos de requerimientos comunes, requerimientos funcionales abreviados como RF y no funcionales, abreviados como RNF.

3.2.1. Requerimientos funcionales

Todos los requerimientos siguientes son de tipo funcionales, estos requerimientos indican lo que el sistema debe hacer, se han englobado en 4 principales categorías. En la tabla 7 se muestra los requerimientos desglosados para cada categoría, notados RF seguido de su número de categoría y el número de requerimiento por categoría. Las categorías son las siguientes:

RF1: Gestión de Perfil

Se requiere que el usuario pueda acceder a su perfil y visualizar su información personal. Este tipo de requerimiento es para todos los usuarios.

RF2: Administración de Usuarios

Se requiere que el usuario administrador pueda listar usuarios. Este tipo de requerimiento es únicamente para el usuario con permisos elevados del sistema.

RF3: Administración de Datos

Se requiere que el usuario administrador pueda listar los proyectos. Este tipo de requerimiento es tanto para el usuario con permisos elevados del sistema como para el usuario común.

RF4: Gestión de Actividades

Se requiere que el usuario administrador pueda asignar proyectos. Este tipo de requerimiento es únicamente para el usuario con permisos elevados del sistema.

Tabla 7

Requerimientos Funcionales del sistema.

NRO.	FUNCIÓN	DESCRIPCIÓN	ENTRADAS Y SALIDAS	ACCIÓN	REQUERIMIENTOS
RF11	Visualiza el perfil	El usuario visualiza su perfil.	Ingreso al sistema. Mostrar los datos del usuario autenticado.	Llama a la plantilla de visualización de datos del usuario.	El usuario debe estar autenticado de manera exitosa.
RF12	Actualiza el perfil	El usuario visualiza su perfil y actualiza sus datos.	Ingreso de datos para cambiar del perfil. Mostrar los datos del usuario autenticado actualizados.	Llama a la plantilla de actualización de datos del usuario y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario a actualizar.
RF21	Lista usuario	El usuario visualiza el perfil de los usuarios creados en el sistema.	Ingreso al sistema. Mostrar los datos de todos los usuarios existentes.	Llama a la plantilla de plantilla de visualización de datos de usuarios.	El usuario debe estar autenticado con permisos elevados de manera exitosa.
RF22	Crea usuario	El usuario crea nuevo usuario.	Ingreso de datos para nuevo usuario.	Mostrar los datos del usuario creado en la lista de usuarios. Llama a la plantilla de creación de datos del usuario y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF23	Elimina usuario	El usuario lista los usuarios y elimina uno de ellos.	Código identificador único de usuario a eliminar. Listar los usuarios sin disponer del usuario eliminado.	Llama a la plantilla de eliminación de datos del usuario y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF24	Cambiar contraseña de usuario	El usuario visualiza el perfil de un usuario y actualiza su contraseña.	Ingreso de datos para actualizar contraseña. Listar los usuarios.	Llama a la plantilla de actualización de datos del usuario y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF31	Lista proyecto	El usuario visualiza los proyectos creados en el sistema.	Ingreso al sistema.	Mostrar los datos los proyectos existentes. Llama a la plantilla de plantilla de visualización de datos de proyectos.	El usuario debe estar autenticado manera exitosa.
RF32	Crea proyecto	El usuario crea nuevo proyecto.	Ingreso de datos para nuevo proyecto.	Mostrar los datos del proyecto creado en la lista de proyectos.	El usuario debe estar autenticado de manera exitosa y corresponder al

				Llama a la plantilla de creación de datos del proyecto y como siguiente a la plantilla de visualización de datos.	usuario con permisos elevados.
RF33	Elimina proyecto	El usuario lista los proyectos y elimina uno de ellos.	Código identificador único del proyecto a eliminar. Listar los proyectos sin disponer del proyecto eliminado.	Llama a la plantilla de eliminación de datos del proyecto y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF34	Actualiza proyecto	El usuario lista los proyectos existentes y actualiza los datos de un proyecto.	Ingreso de datos para actualizar el proyecto. Listar los datos del proyecto actualizados.	Llama a la plantilla de actualización de datos del proyecto y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF35	Lista empresa cliente	El usuario visualiza las empresas clientes creadas en el sistema.	Ingreso al sistema. Mostrar los datos las empresas clientes existentes.	Llama a la plantilla de plantilla de visualización de datos de las empresas clientes.	El usuario debe estar autenticado manera exitosa.
RF36	Crea empresa cliente	El usuario crea una nueva empresa cliente.	Ingreso de datos para nueva empresa cliente. Mostrar los datos de la nueva empresa cliente creada en la lista de empresas clientes.	Llama a la plantilla de creación de datos de empresa cliente y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF37	Elimina empresa cliente	El usuario lista las empresas clientes y elimina una de ellas.	Código identificador único de la empresa cliente a eliminar. Listar las empresas clientes sin disponer de la empresa cliente eliminada.	Llama a la plantilla de eliminación de datos de la empresa cliente y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF38	Actualiza empresa cliente	El usuario lista las empresas clientes existentes y actualiza los datos de una empresa cliente.	Ingreso de datos para actualizar la empresa cliente.	Listar los datos de la empresa cliente actualizados. Llama a la plantilla de actualización de datos de la empresa cliente y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.

RF41	Asigna proyecto	El usuario asigna un proyecto a otro usuario.	Ingreso de datos de asignación del proyecto. Mostrar los datos de asignación de proyecto en la lista de proyectos asignados.	Llama a la plantilla de creación de datos de asignación de proyectos y como siguiente a la plantilla de visualización de datos.	El usuario debe estar autenticado de manera exitosa y corresponder al usuario con permisos elevados.
RF42	Lista asignación de proyecto	El usuario visualiza las asignaciones de proyecto creadas en el sistema.	Ingreso al sistema. Mostrar los datos asignación de proyecto existentes.	Llama a la plantilla de plantilla de visualización de datos de los proyectos asignados.	El usuario debe estar autenticado manera exitosa.
RF43	Realiza el proyecto asignado	El usuario visualiza las asignaciones de proyecto creadas en el sistema y realiza el proyecto asignado.	Ingreso al sistema. Mostrar los datos asignación de proyecto existentes realizados.	Llama a la plantilla de plantilla de visualización de datos de los proyectos asignados completados.	El usuario debe estar autenticado manera exitosa.

Nota. *Tabla con requerimientos funcionales del sistema, se han definido 4 categorías con nomenclatura tipo RF21, en este caso indica categoría 2, primer requisito. Elaborado por: El autor.*

3.2.2. Requerimientos No Funcionales

Los requerimientos No Funcionales son la parte implícita del sistema. Se han considerado las características básicas para que un sistema web funcione correctamente y brinde experiencia de usuario satisfactoria. Dentro de los requerimientos No Funcionales se tiene:

RNF1: Rendimiento

El sistema soporta una carga mínima y máximo aceptable para las acciones comunes que conllevan el uso habitual del sistema.

RNF2: Usabilidad

El sistema puede utilizarse y manejarse sin mayor esfuerzo. Mantiene una apariencia responsiva y su utilización es cómoda.

RNF3: Simplicidad

No se requiere ser un técnico experto para manejar el sistema y la consecución de objetivos están en un alcance máximo de 3 subniveles. Con una capacitación corta se puede manejar y usar toda la capacidad del sistema con la menor complejidad posible.

RNF4: Navegabilidad

Puede intercambiar entre una vista y otra sin problema. E interactúa entre sus distintos componentes del sistema web desplazándose por todas sus páginas.

RFN5: Seguridad

El sistema guarda y recupera los datos sin perderlos. Se maneja un acceso de seguridad basado en clave contraseña y niveles de acceso.

RNF6: Mantenibilidad

El sistema se ha diseñado lo mayor organizado y lógico posible de modo que la inserción de nuevos módulos o mejora de uno implica un menor esfuerzo.

3.3. Diseño

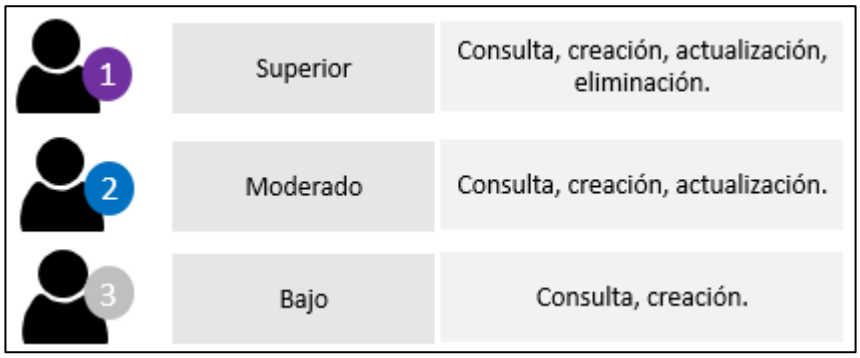
A continuación, se detallan los atributos y elementos que se consideraron para la construcción y desarrollo del sistema.

3.3.1. Roles de Usuarios del Sistema

Para los roles del sistema se ha manejado un nivel de jerarquía y permisos. Considerando acciones como las de edición para los niveles más elevados y acciones de consulta y creación por lo general para los niveles menores. Este concepto se lo muestra en la Figura 8 para el nivel de jerarquía en el sistema.

Figura 8

Nivel de jerarquía en el sistema.



Nota. *Niveles de tipos de permisos y acciones que puede realizar el usuario de nivel (1), (2) y (3) según su jerarquía. Elaborado por: El autor.*

3.3.2. Rol Administrador

El usuario con rol Jefe de Proyecto dispondrá de permisos altamente elevados. Los cuales le facultan a:

Modificación de Código.

Visualización de perfil.

Actualización de datos de perfil.

Acceso a tablero de control.

Creación/Eliminación, actualización de Módulos.

Creación, actualización y eliminación de datos empresariales, de clientes y productos.

Creación, actualización y eliminación de usuarios.

Gestión de contraseñas de todos los usuarios.

Creación, eliminación, actualización de roles y permisos.

Creación, actualización y eliminación de Proyectos.

Visualización de Fases de Proyectos.

3.3.3. Rol Jefe de Proyecto

El usuario con rol Jefe de Proyecto dispondrá de permisos moderadamente elevados. Los cuales le facultan a:

Visualización de perfil.

Actualización de datos de perfil.

Modificación de su contraseña.

Creación y edición de datos empresariales, de clientes y productos.

Creación, actualización de roles y permisos.

Creación, actualización y asignación de Proyectos.

Creación, actualización y revisión de tareas para Técnicos.

Acceso a tablero principal.

Creación de reportes.

Visualización de Fases de Proyectos.

Visualización de tareas realizadas y tareas pendientes (propias y de usuarios técnicos).

3.3.4. Rol Técnico

El usuario con rol Técnico dispondrá de permisos limitados. Los cuales le facultan a:

Visualización de perfil.

Modificación de su contraseña.

Actualización de datos de perfil.

Actualización de sus tareas de Técnico.

Acceso parcial a tablero principal (según sus tareas).

Creación de reportes.

Visualización de Fases de Proyectos en los que interviene.

Visualización de tareas realizadas y tareas pendientes (propias).

3.4. Diagramas de Casos de uso

Los diagramas de caso permiten comprender la interacción de los actores sobre el sistema, cada uno de estos diagramas representan las partes constitutivas del sistema con su comportamiento.

3.4.1. Casos de Uso Administrador

Figura 9, Subgráfica (a), El actor que corresponde al administrador del sistema puede ingresar y visualizar su perfil una vez ingresados los datos de autenticación.

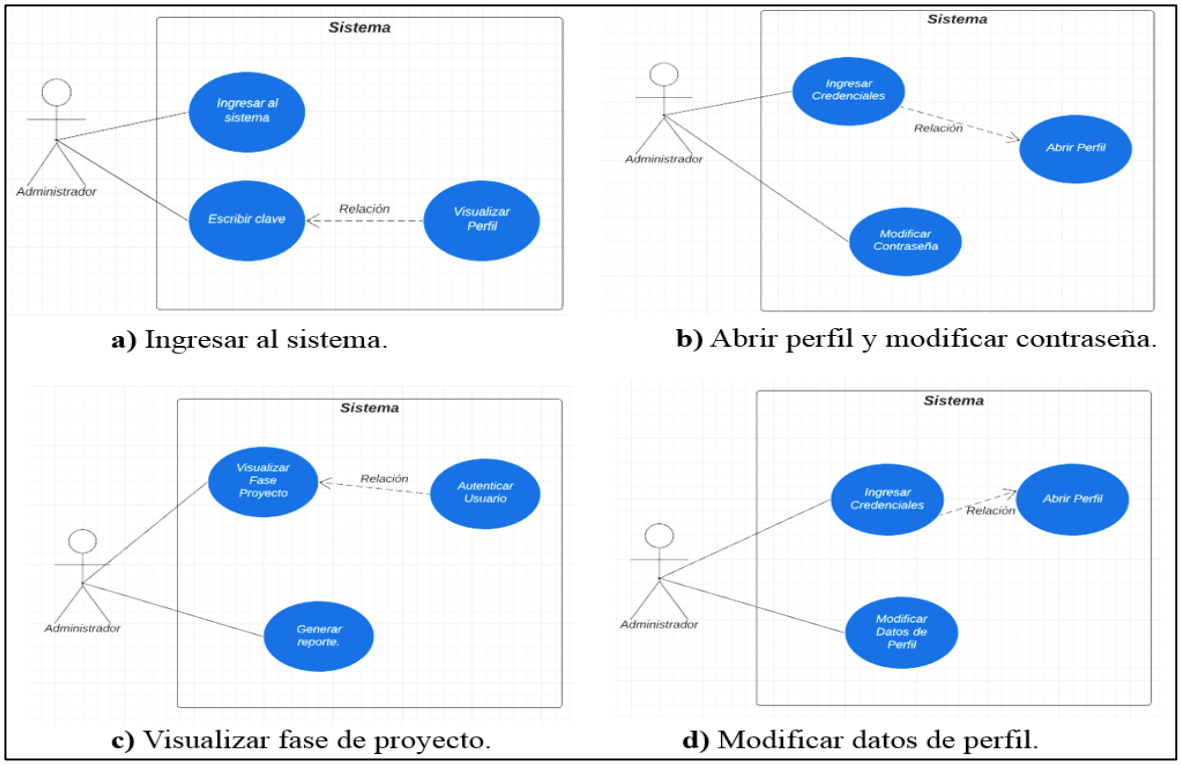
Figura 9, Subgráfica (b), El actor que corresponde al administrador del sistema puede ingresar y modificar su contraseña una vez ingresado y abierto su perfil después de haberse autenticado.

Figura 9, Subgráfica (c), El actor que corresponde al administrador del sistema puede visualizar la fase del proyecto y generar un reporte.

Figura 9, Subgráfica (d), El actor que corresponde al administrador del sistema una vez ingresados los datos de autenticación puede ingresar, visualizar su perfil y actualizar sus datos.

Figura 9

Interacciones con el sistema como usuario administrador.



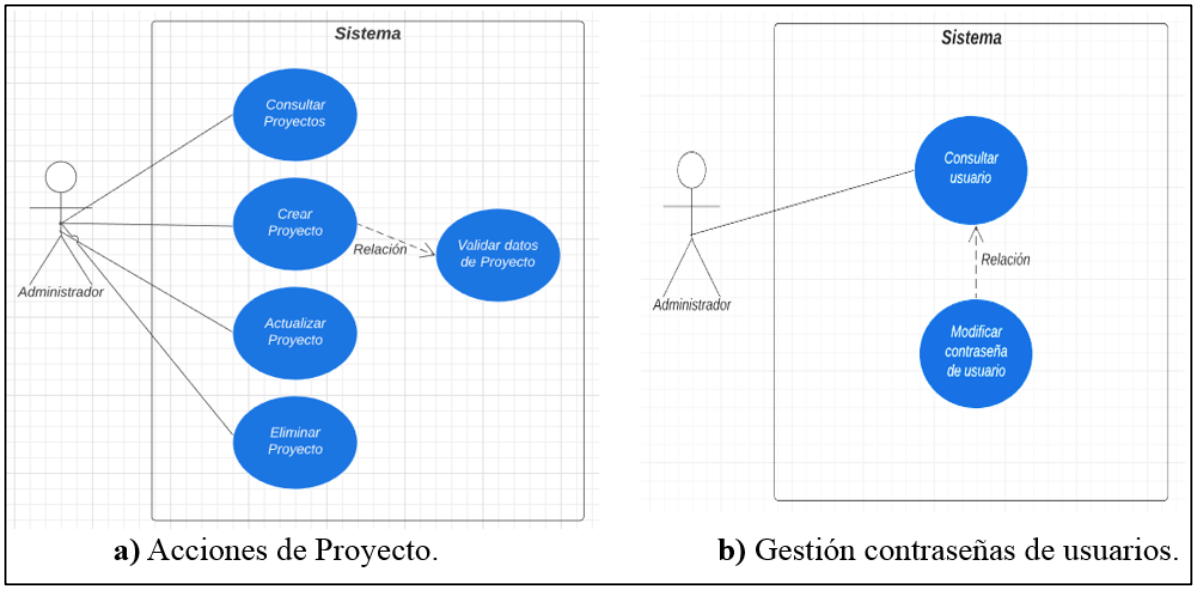
Nota. *Funciones de actualización, creación y consulta por un usuario administrador. Cada diagrama corresponde a un caso de uso, Elaborado por: El autor.*

Figura 10, Subgráfica (a), El actor que corresponde al administrador del sistema puede realizar las acciones de crear, actualizar y eliminar proyectos.

Figura 10, Subgráfica (b), El actor que corresponde al administrador del sistema una vez ingresados los datos de autenticación puede ingresar, consultar perfiles de usuario y modificar la contraseña de uno de ellos.

Figura 10

Interacciones con el sistema como usuario administrador.



Nota. *Funciones de actualización, creación, eliminación y consulta por un usuario administrador. Cada diagrama corresponde a un caso de uso. Elaborado por: El autor.*

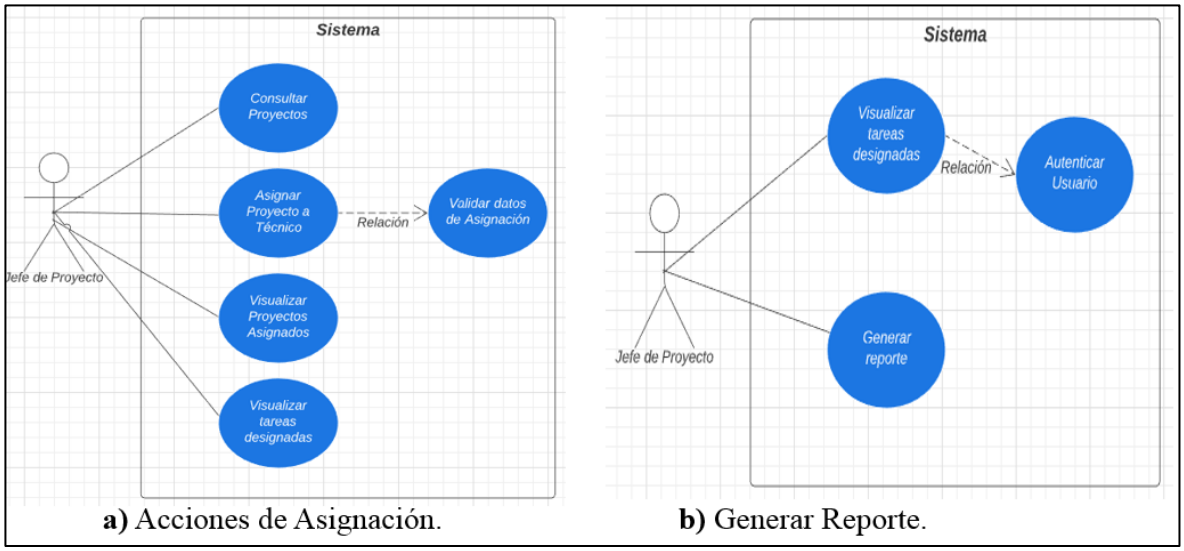
3.4.2. Caso de Uso Jefe de Proyecto

Figura 11, Subgráfica (a), El actor que corresponde al administrador del sistema puede ingresar y asignar proyectos a usuarios técnicos una vez ingresados los datos de autenticación.

Figura 11, Subgráfica (b), El actor que corresponde al administrador del sistema puede ingresar y generar un reporte después de haberse autenticado.

Figura 11

Interacciones con el sistema como usuario jefe de proyecto.



Nota. *Funciones de actualización, creación y de asignación de tareas. Cada diagrama corresponde a un caso de uso. Elaborado por: El autor.*

3.4.3. Casos de Uso Técnicos

Figura 12, Subgráfica (a), El actor que corresponde al usuario común o técnico puede modificar su contraseña una vez ingresadas sus credenciales y visualizado su perfil.

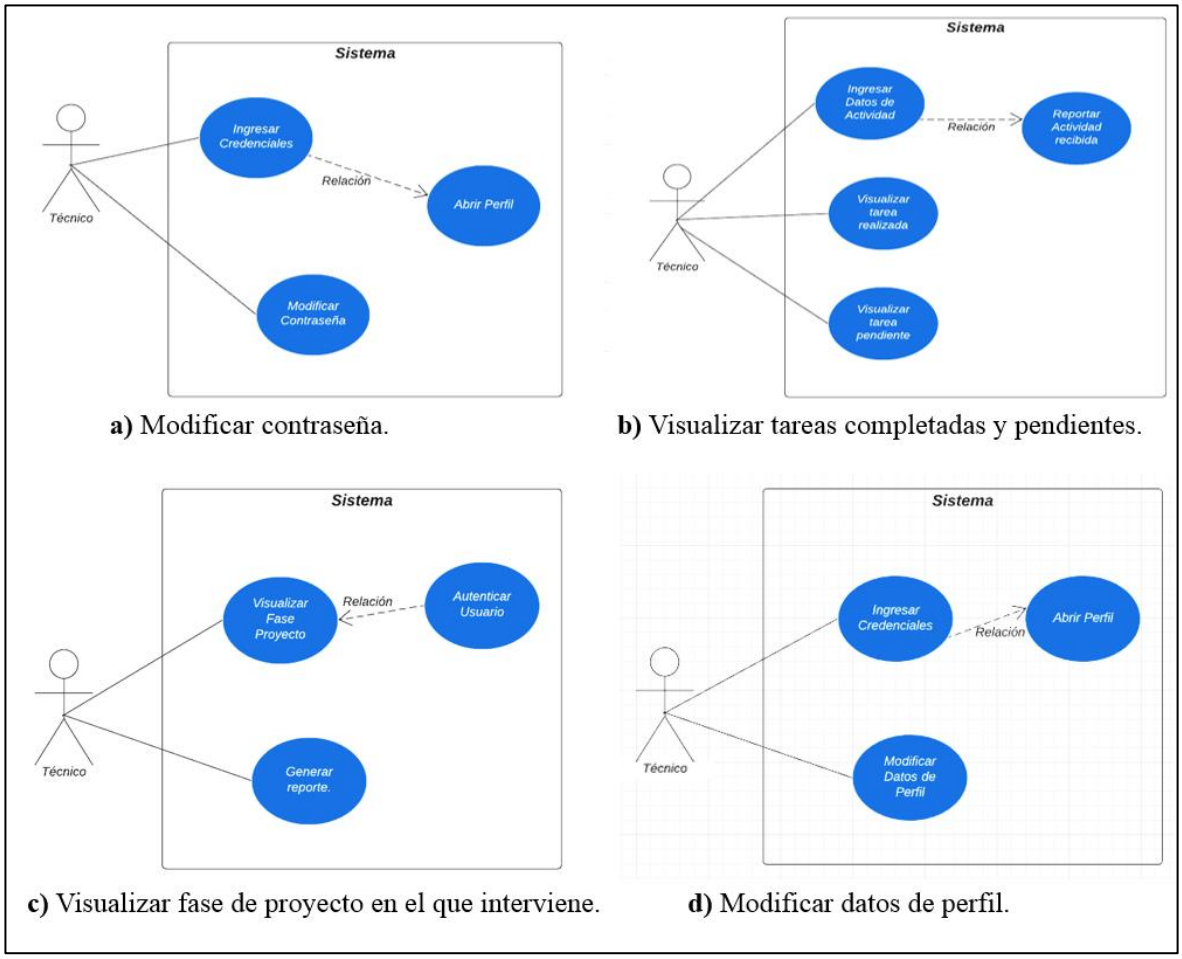
Figura 12, Subgráfica (b), El actor que corresponde al usuario común o técnico puede realizar las tareas asignadas una vez ingresadas sus credenciales y visualizar sus tareas pendientes.

Figura 12, Subgráfica (c), El actor que corresponde al usuario común o técnico puede visualizar la fase del proyecto en el que interviene.

Figura 12, Subgráfica (d), El actor que corresponde al usuario común o técnico puede modificar datos de su perfil siempre y cuando se encuentre debidamente autenticado.

Figura 12

Interacciones con el sistema como usuario técnico.



Nota. *Funciones de actualización, creación y consulta por un usuario técnico. Cada diagrama corresponde a un caso de uso. Elaborado por: El autor.*

3.5. *Construcción de Base de Datos*

Una base de datos es el contenedor por excelencia para los distintos datos que componen el sistema. Debido a la naturaleza relacional de los datos y que la mayor parte de la información generada guarda los mismos atributos (diferenciado únicamente por sus valores) se ha optado por una base de datos relacional. La elección para la base de datos de PostgreSQL se hizo en función de su capacidad de crecimiento, por su adaptabilidad, optimización y velocidad de consulta (Katzburg, Golander y Weiss, 2018).

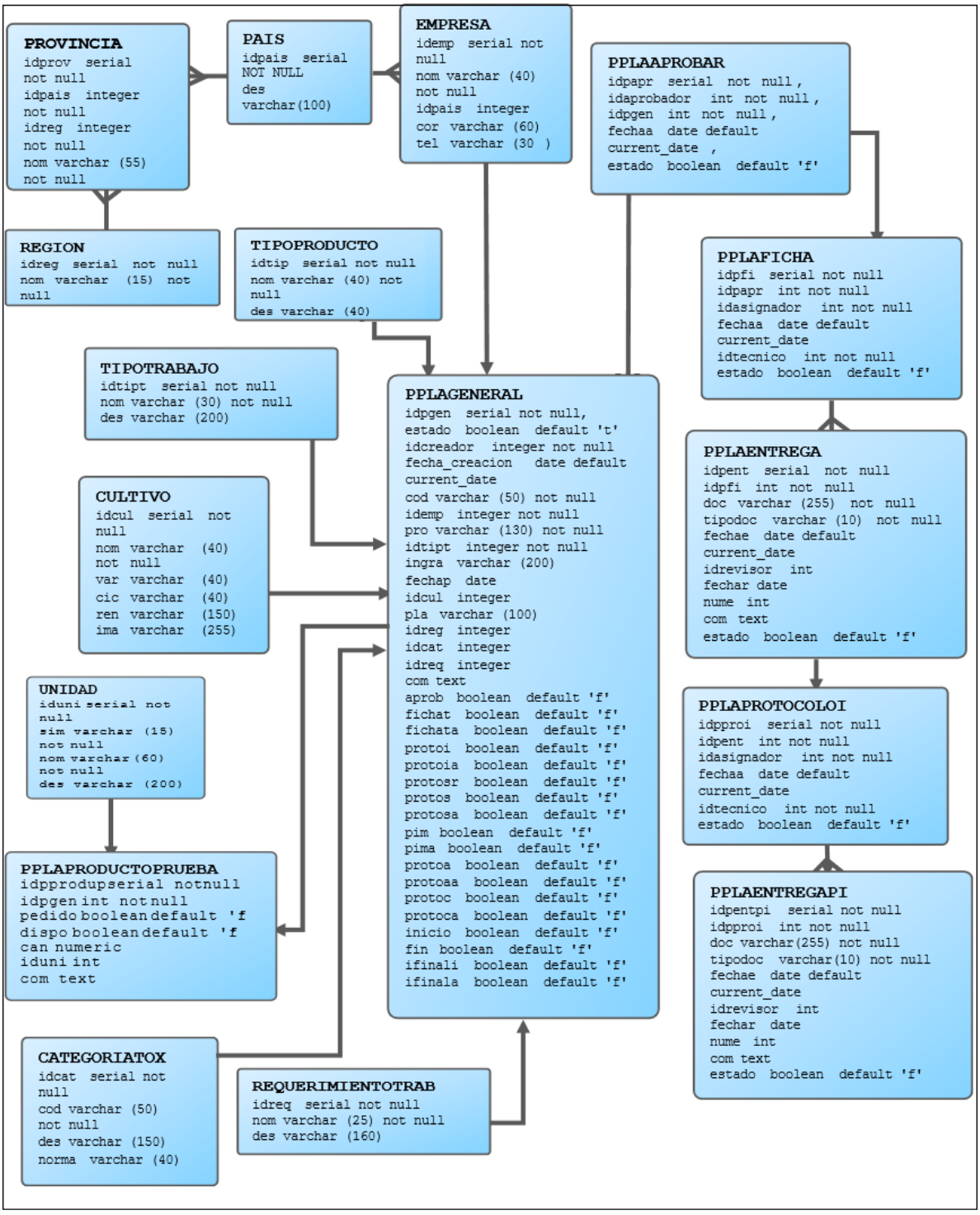
Para la construcción de las tablas y atributos de la base de datos, se tomó como guía dos partes. Como primera parte, el Diagrama de Flujo de Trabajo, el cual se compone de dos procesos. En la Figura 6 se muestra el diagrama para el primer proceso y en la Figura 7 para el segundo proceso. Como segunda parte se utilizó la tabla de los Procesos del Departamento de Investigación Plaguicidas mostradas en la Tabla 6.

Así, estas dos partes se utilizaron para hacer una exploración con la que se obtuvo el panorama de interacción global a mayor detalle y con esta información se armaron las relaciones, la lógica para el funcionamiento del sistema y se identificó los atributos necesarios que cada tabla iba a requerir para que el sistema sea organizado, haciéndolo fácil de mantener y de modo que pueda añadirse nuevas tablas a futuro de manera escalable y sin afectar de forma drástica la lógica ya planteada.

El diagrama construido para la base de datos se puede observar en la Figura 13, que muestra una parte de la Base de Datos relacional para el Proceso de Elaboración hasta la entrega de un segundo documento. La Base de Datos completa se la puede ver en el Anexo 1, de la parte Anexos.

Figura 13

Base de Datos relacional del sistema.



Nota. *Base de Datos relacional del sistema desarrollado, cuenta con tablas relacionadas como también con tablas independientes que se ha denominado moleculares. En la figura se observa la tabla PPLAGENERAL que es la tabla base para la creación y registro de un proyecto. Elaborado por: El autor.*

Cabe mencionar que, al momento de construir la base de datos, se manejó las tablas considerando su naturaleza en 2 tipos.

Tablas moleculares, siendo las más básicas y no tienen claves foráneas.

Tablas relacionadas, teniendo una o más de una relación con otras tablas.

Esto se hace sumamente importante al momento de la creación de relaciones debido a que las tablas moleculares deben ser creadas primero, y el resto en orden de su relación con las anteriores.

3.6. Backend

El Backend es la parte tras bastidores o del desarrollo en sí que tiene el sistema. Desde este punto se manejaron diferentes partes de la lógica que va por detrás y para que los componentes se interrelacionen en situaciones tales como conexiones a la base de datos, peticiones y solicitudes. Todo esto se lo configura con cambios puntuales sobre los archivos creados después de la instalación de Django y creación del primer proyecto y aplicación, y el resto de funcionamiento por detrás es gestionado por el Framework. En la Figura 14 se puede observar la parte del backend considerando como referencia el recuadro central (que representa el Framework Django) y hacia la izquierda.

Django, es el framework para Backend que se utilizó, su instalación como del entorno virtual puede encontrarse en el Anexo 2. Django fue levantado sobre el Sistema Operativo Ubuntu en su versión 18.0.4 el cual funcionó como servidor. Para hacer correr a Django desde la terminal de Ubuntu los primeros pasos fueron la creación y activación del entorno virtual, entorno donde se instaló la versión de Django 3.1 y con esto el conjunto de librerías que se necesitó.

3.6.1. Conexión a la Base de Datos

Al crear un proyecto y una aplicación en Django, automáticamente se crea un directorio con archivos entre los cuales se tiene el archivo de configuración. Dentro de este cambiamos el motor de Base de Datos que por defecto se encuentra en SQLITE, por POSTGRESQL.

Los parámetros que se deben colocar para el bloque de código mostrados en la Figura 14 corresponden a parte del motor que se ha utilizado PostgreSQL, el nombre de la base de datos, el usuario dueño de la base de datos creada en PostgreSQL, la contraseña creada para la conexión con la Base de Datos, la dirección del host en el cual se encuentra alojada la base de datos para este caso es la dirección local donde se encuentra instalada y creada la Base de Datos y como final el número de puerto que utiliza PostgreSQL que es 5432.

Figura 14

Fragmento del archivo settings.py.

```
DATABASES = {
'default': {
    'ENGINE': 'django.db.backends.postgresql',
    'NAME': 'nombreBaseDatos',
    'USER': 'jairo',
    'PASSWORD': 'Contraseña',
    'HOST': 'localhost',
    'PORT': '5432',
}
```

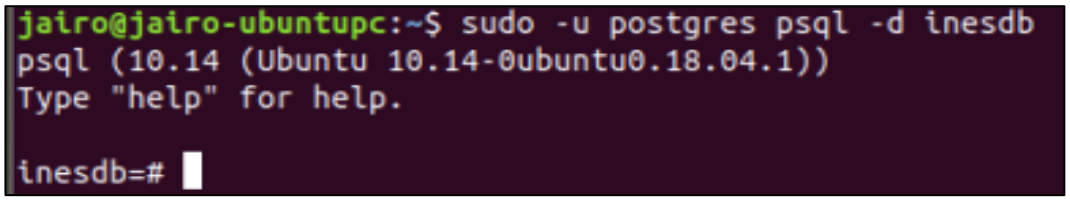
Nota. *Configuración del Archivo seetings.py para utilizar la Base de Datos PostgreSQL. Elaborado por: El autor.*

Para comprobar la conexión exitosa con la base de datos INESDB que se creó en PostgreSQL desde el terminal con permisos de superusuario se logra conectar con la llamada al comando psql -d y el nombre de nuestra base de datos.

Al conectarse a la base de datos simplemente aparecerá el nombre de la base de datos el cual se encuentra seguido de un signo igual y una almohadilla como se ve en la Figura 15.

Figura 15

Conexión a Base de Datos del Sistema.



Nota. *Ejecución por consola de comando psql del Gestor de base de Datos PostgreSQL, mediante el cual hace conexión con la Base de Datos inesdb. Elaborado por: El autor.*

3.6.2. Librerías necesarias para Django

Django dispone de un conjunto de liberías propias que son alojadas en la carpeta django.conf, dentro del cual contiene una serie de ficheros donde cada cual le brinda funcionalidades distintas al framework. Para administrar usando la libería admin, autenticar con la librería auth, o proteger las peticiones de falsificaciones con la libería crsf.

A parte de las librerías propias de Django, para el caso del desarrollo se han instalado otro conjunto de paquetes que han sido necesarios para que funcione el sistema. Algunos de estos paquetes son funcionales dentro del Administrador de Django el cual es una interfaz propia del framework que permite administrar los diferentes modelos o tablas que se hayan registrado en el archivo Admin.py. Para el Administrador de Django se han instalado paquetes como Django-ckeditor y Django-import-export, la primera para hacer ediciones sobre el contenido de los campos y el segundo para restaurar y respaldar datos de la base de datos. Por otro lado, para utilizar el gestor de base de datos PostgreSQL se instaló la librería Psycopg2 que comunica la base de datos con el framework.

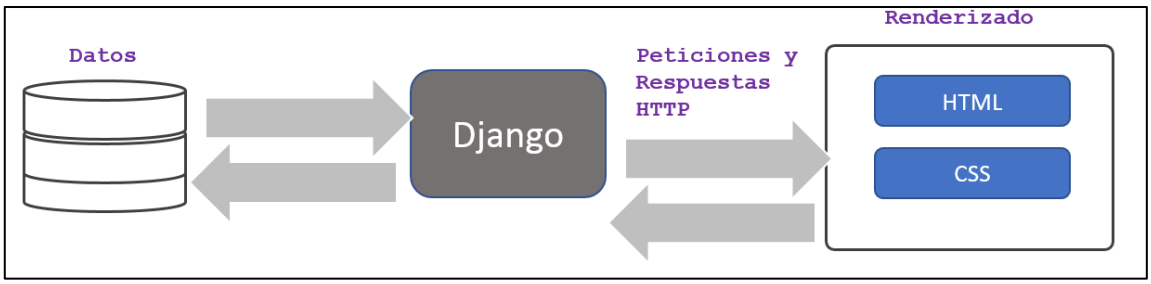
Librerías como Pytz para gestionar la zona horaria y Sqlparse para hacer validación de sentencias de tipo SQL son del tipo adicional que también tienen su funcionalidad e importancia dentro del sistema. Todas estas librerías pueden encontrarse descritas en el Anexo 2.

3.7. Frontend

El frontend es la otra parte de la moneda de la aplicación web. Esta parte del sitio web es la que tiene comunicación directa con el usuario. Cada vez que pulsa, abre o cierra algo y es completamente visible para el usuario final, significa que se encuentra trabajando con el frontend. Las interfaces, plantillas y páginas que son con las que el usuario interactúa, son consideradas parte del frontend. En la Figura 16 se puede dar a entender el concepto de forma gráfica tomando como referencia el recuadro central (que representa el Framework Django) y hacia la derecha todo forma parte del frontend.

Figura 16

Backend y Frontend.



Nota. *Relación existente y comunicación entre los componentes utilizando el framework Django. Tomando como referencia el recuadro central hacia la derecha se considera el Frontend y del recuadro hacia su izquierda el Backend. Elaborado por: El autor.*

3.7.1. Interfaces

De acuerdo con los roles de usuario, cada usuario tiene una pantalla principal donde se activan determinadas funciones según el tipo de usuario que ingrese. Dichas funciones se pueden acceder a través de un menú lateral con botones que despliegan las diferentes vistas de acciones o tareas a realizar.

Si se considera las funciones a las cuales puede acceder un usuario según su rol, éstas pueden ser del tipo genéricas y funciones específicas.

Las funciones genéricas a todos los usuarios son:

Login o Ingreso al Sistema, Visualización y actualización de perfil y Modificación de contraseña.

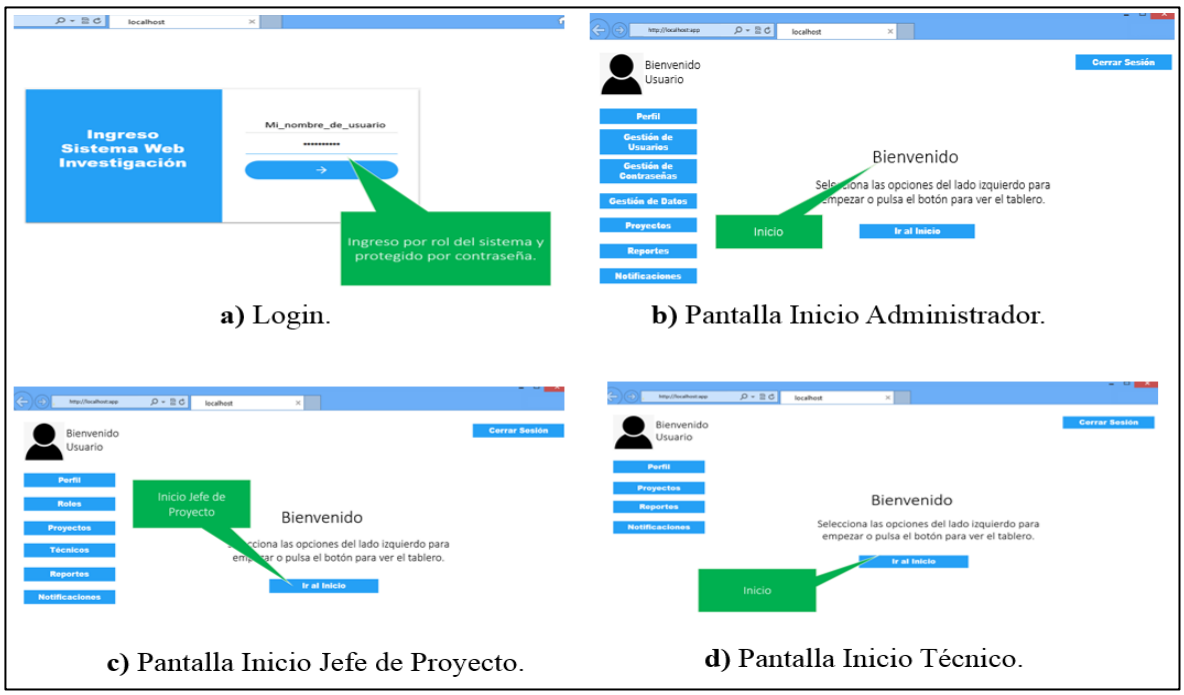
Las funciones que son exclusivas para usuarios administradores son:

Eliminación y Asignación.

Como se puede ver en la Figura 17, en la Subgráfica (a) se muestra la interfaz de Login que es común a todos los usuarios, mientras que las interfaces de Inicio serán particulares para el tipo de usuario. En la misma Figura 17, Subgráfica (b), se observa la pantalla de inicio para Administrador, en la Subgráfica (c) para el Jefe de Proyecto y finalmente la Subgráfica (d) para el usuario Técnico.

Figura 17

Pantalla de Login y de Inicio.



Nota. *La imagen muestra una de las vistas comunes a todos los usuarios, se requiere que se encuentre autenticado. Las subgráficas b, c, d, corresponden a la pantalla inicial de cada tipo de usuario para Administrador, Jefe de Proyecto y Técnico respectivamente. Elaborado por: El autor.*

3.7.2. Interfaces Administrador

Creación de Usuarios

Esta interfaz mostrada en la Figura 18, Subgráfica (a) permite la creación de un usuario nuevo. Para lo cual, la interfaz solicita llenar sus datos los que incluyen una contraseña y guardado los cambios se listarán junto con los demás usuarios ya creados.

Creación de Empresas Clientes

Esta interfaz permite la creación de nuevas empresas clientes. Para lo cual, la interfaz solicita llenar los datos los que incluyen información relacionada a la empresa como datos de contacto, país de procedencia y al guardar los cambios se listarán junto con las demás empresas clientes ya creadas.

3.7.3. Interfaces Jefe de Proyecto

Creación de Proyecto

En esta interfaz que se puede ver en la Figura 18, Subgráfica (b) se podrá crear un proyecto nuevo. Para lo cual, la interfaz solicita llenar sus datos los que incluyen una información de la empresa, producto, cultivo. Una vez guardado los cambios se listarán junto con los demás proyectos ya creados.

Aprobación de Proyectos

Esta interfaz permite que el usuario cambie el estado de un proyecto de solicitado a aprobado. Una vez guardado los cambios se listarán junto con los demás proyectos que ya han sido aprobados.

Asignación de Proyectos

Los proyectos podrán ser asignados una vez que hayan sido aprobados mediante la interfaz de Aprobación de Proyectos. En esta interfaz se muestran los proyectos ya aprobados para seleccionar a un Técnico a designar.

Una vez guardado los cambios se listarán junto con los demás proyectos ya asignados.

Revisión de Entregas

En esta interfaz el usuario podrá visualizar las tareas entregadas por el usuario Técnico y poder aprobarlas. Una vez guardado los cambios se listarán junto con los demás tareas ya aprobadas o tareas por corregir.

3.7.4. Interfaces Técnico/Técnico de Proyecto

Visualizar Proyectos

Mediante esta interfaz el usuario Técnico podrá ver la información general de los proyectos y el detalle de los proyectos a su cargo.

Entrega de Tareas

En esta interfaz mostrada en la Figura 18, Subgráfica (c) el usuario logra ver los proyectos asignados con su tarea por realizar. El usuario Técnico podrá ver y realizar las tareas respectivas para los proyectos asignados a su cargo. Al cargar los datos de entrega y una vez guardado los cambios se listarán junto con los demás proyectos ya entregados por el usuario.

Consulta de Entregas

En esta interfaz el usuario logra ver las tareas entregas y el estado en el que se encuentran. Si una tarea no se ha aprobado deberá entregarla nuevamente por la interfaz de Entrega de Tareas.

Figura 18

Interfaces de usuarios.

a) Interfaz administrador. **b)** Interfaz Jefe de Proyecto.

c) Interfaz Técnico.

Nota. *Se muestra la variación que puede existir entre los tipos de usuarios y las acciones que pueden realizar. La Subgráfica (a) de administrador muestra la creación de un usuario. La Subgráfica (b) de Jefe de Proyecto muestra la creación de un proyecto. Mientras la Subgráfica (c) de Técnico muestra la entrega de una tarea. Elaborado por: El autor.*

CAPÍTULO III

4. CONSTRUCCIÓN Y PRUEBAS

En este capítulo se van a realizar pruebas de la Aplicación Web desarrollada en el Framework Django, en los títulos siguientes se tiene la descripción del tipo de pruebas empleadas para la aplicación y parámetros considerados para ejecutarlas.

4.1. Tipos de Pruebas realizadas

Una de las primeras pruebas fueron las pruebas de tipo funcionales o pruebas de caja negra, donde, se las ha ido implementado consecutivamente mientras se desarrollaba el software de modo que añadida una nueva funcionalidad se procedía a ver su comportamiento al usar entradas y corregirlo en caso de error o salidas no esperadas.

Las pruebas adicionales consideradas para el software fueron 4.

- Pruebas de caja negra
- Pruebas de carga
- Pruebas de rendimiento
- Pruebas de estrés

A continuación, se describen en qué consisten cada una de estas pruebas.

Pruebas de Caja negra, este tipo de pruebas permiten conocer la funcionalidad de un software donde para un conjunto de entradas se observa el comportamiento para sus salidas (Mera, 2016). Un ejemplo sencillo de esto es pensar en probar la función “sumar en una calculadora”. Para el ejemplo como entradas ingresa un primer número, luego el operador suma, después un segundo número y finalmente pulsa el operador igual para ver su resultado. La salida debería ser el resultado de sumar estos dos números y sin duda se esperaría que estén correctamente sumados. Con lo cual se determina si cumple o no dicha prueba.

Pruebas de Carga, con este tipo de pruebas el objetivo es comprobar que el sistema realiza una carga sin que se vea comprometido su rendimiento o en otras palabras que pueda

atender a un número de solicitudes determinado sin que haya problemas como que se sobrecargue el servicio o no pueda atender a todas las solicitudes.

Pruebas de Rendimiento, se encargan en medir, sobre todo, los tiempos en que tarda en responder un sistema cuando se encuentra con una carga particular y como consecuencia notar qué tan fiable es en las operaciones y ejecuciones que realiza.

Pruebas de Estrés, una de las pruebas más conocidas y de interés por parte de los administradores de servidores, dado que permite conocer cuánto soporta un sistema cuando está en presencia de grandes volúmenes de demandadas de servicio concurrentes.

Las pruebas de carga, rendimiento y estrés se las realizó utilizando la herramienta de software libre Apache JMeter.

4.2. Parámetros para Pruebas

Las pruebas se han armado en escenarios similares a los esperados en producción. Se consideró 5 de las páginas que manejan transacciones en la aplicación web. Y se hizo la planeación del número de hilos correspondientes a peticiones simultáneas al servidor mostradas en la Tabla 8 que se ajustan a los requerimientos actuales de la empresa.

Tabla 8
Número de hilos usados para las pruebas.

	NÚMERO DE HILOS					
PRUEBA	20	50	100	150	500	1000
CARGA	X	X	X			
RENDIMIENTO	X	X	X	X		
ESTRÉS	X	X	X	X	X	X

Nota. *Los números de hilos corresponden al número de peticiones simultáneas que hará la herramienta para comprobar las variables principales de rendimiento, carga y estrés. Elaborado por: El autor.*

Una vez planificado lo anterior, dentro de la herramienta Apache JMeter se creó el elemento Grupo de Hilos el cual contiene las peticiones de tipo creación, asignación, listado y login. Las cuales corresponden al grupo de solicitudes comunes dentro del sistema, por

ejemplo, la operación de creación es la misma tanto para crear una nueva empresa, como para crear un nuevo proyecto. Lo único que cambian son el número de atributos entre una y otra, pero el método de operación es la mismo es decir un método POST.

Al utilizar la herramienta se inicia con un proyecto de tipo Test Plan o Plan de pruebas. En el cual se configura tanto el nombre del Plan de Pruebas como las variables de usuario, estos datos se los puede ver en la Tabla 9. Las variables de interés serán la dirección IP (donde la herramienta realizará las peticiones) y el número de puerto correspondiente al puerto utilizado por el servidor de Django.

Tabla 9

Valores del proyecto para plan de pruebas usados.

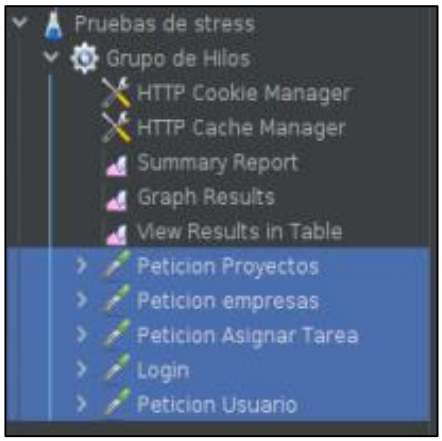
Nombre	**Valor**
Dirección	Localhost
Puerto	8080

Nota. *La dirección corresponde al valor IP del host donde se encuentra alojado el servidor, al encontrarse realizando las pruebas con Apache JMeter instalado en el servidor se coloca localhost, de otro modo será el número de la dirección IP. Elaborado por: El autor.*

En la Figura 19 se observa que, para el caso del Grupo de Hilos que es una subelemento creado debajo del proyecto Plan de Pruebas, se creó 3 elementos para análisis de resultados que son: Gráfica de Resultados, Árbol de resultados y Reporte resumen.

Figura 19

Peticiones creadas y reportes de resultados.



Nota. *Creación de 5 Peticiones de tipo HTTP para la ejecución de pruebas en la herramienta JMeter. Elaborado por: El autor.*

Con esto preparado ya se dispuso de la información y escenarios para empezar las pruebas.

4.3. Resultados de Pruebas

4.3.1. Pruebas de Caja Negra

Para estas pruebas se ha considerado comprobar el funcionamiento de la aplicación haciendo pruebas de caja negra, se consideró el resultado de la prueba y el criterio de aceptación está determinado si cumple o no y si es el resultado esperado. Las pruebas se han notado como P de Prueba, G para General, es decir todos los usuarios, J para Jefe de Proyecto y T para Técnico. Los resultados de esta prueba se muestran en la Tabla 10.

Tabla 10

Resultados de pruebas de caja negra.

Prueba	Intención	Acción realizada	¿Lo logró?		Esperado	
			Sí	No	Sí	No
PG1	Ingresar al sistema.	Ingresó usuario correcto y contraseña incorrecta.		X	X	
PG2	Ingresar al sistema.	Ingresó usuario incorrecto y contraseña correcta.		X	X	
PG3	Ingresar al sistema.	Ingresó usuario y contraseña incorrectos.		X	X	
PG4	Ingresar al sistema.	Ingresó usuario y contraseña correctos.	X		X	
PG5	Ingresar al sistema después de cerrar sesión.	Recarga la página, pulsa la opción de perfil.		X	X	
PJ1	Consultar proyectos pulsando la opción de consulta de proyectos.	Pulsar la opción consulta de proyectos.	X		X	
PJ2	Crear proyecto sin escribir un campo obligatorio.	El Jefe de Proyecto dejó en blanco un campo obligatorio de la creación de proyectos y pulsó guardar.		X	X	
PJ3	Crear proyecto escribiendo campos obligatorios y dejando en blanco campos opcionales.	El Jefe de Proyecto dejó en blanco campos opcionales de la creación de proyectos y pulsó guardar.	X		X	
PJ4	Asignar una ficha técnica sin proyectos aprobados.	El Jefe de Proyecto abrió la opción de asignar fichas y no encontró nada.		X	X	
PJ5	Designar la entrega de un protocolo sin elegir un técnico.	El jefe de proyecto ingresa a la opción asignar, no seleccionó el técnico y pulsó guardar.		X	X	
PJ6	Designar la entrega de un protocolo eligiendo a un técnico.	El jefe de proyecto ingresa a la opción asignar, seleccionó el técnico y pulsó guardar.	X		X	
PJ7	Revisar último informe final asignado a un técnico.	Como jefe de proyecto ingresa a la opción informe final asignados.	X		X	
PT1	Entregar una Ficha Técnica sin adjuntar el documento de Ficha Técnica.	El técnico ingresa a la opción entregar ficha técnica, no adjuntó el documento de Ficha Técnica y pulsó guardar.		X	X	
PT2	Entregar una Ficha Técnica adjuntando el documento de Ficha Técnica.	El técnico ingresa a la opción entregar ficha técnica, adjuntó el documento de Ficha Técnica y pulsó guardar.	X		X	
PT3	Guardar la ejecución de campo ingresando textos en un campo numérico.	El Técnico ingresa en el campo duración de ensayo un texto y pulsó guardar.		X	X	
PT4	Guardar la ejecución de campo ingresando un número en un campo numérico.	El Técnico ingresa en el campo duración de ensayo un número y pulsó guardar.	X		X	
		Totales	7	9	16	0

***Nota**. Resultados obtenidos al realizar 16 pruebas de caja negra a la aplicación web, el criterio de aceptación es el resultado esperado, teniendo 0 en resultados no esperados se puede llegar a la conclusión de que se tuvo 100% resultados esperados. Elaborado por: El autor.*

Con estas pruebas se consiguen 7 respuestas de logro exitoso y 9 respuestas de logro infructuoso con un total de 16 entre ambas. Lo cual no significa que haya fallado el sistema cuando no logró el objetivo de la prueba, sino que la correcta funcionalidad queda determinada por el criterio de aceptación si fue o no esperado, obteniendo 16 respuestas esperadas, de las 16 pruebas representando el 100% de resultados esperados.

4.3.2. Pruebas de carga

El objetivo de este tipo de pruebas es comprobar que el software realiza la tarea común de cargar o atender a un número de solicitudes determinada.

Debido a que no tiene por intención exceder las capacidades del software se ha tomado 20, 50 y 100 hilos de peticiones simultáneas, éste último con todos los valores que arroja la herramienta JMeter está dispuesto en la Tabla 11.

Tabla 11

Reportes de resultados con porcentaje de error.

Peticiones	**Cantidad de Muestras**	**Tiempo medio [ms]**	**Tiempo mínimo [ms]**	**Tiempo máximo [ms]**	**Porcentaje de Error**	**Rendimiento [s]**
Petición Proyectos	100	4071	50	15149	0%	6.3
Petición Empresas	100	3440	23	12497	0%	5.9
Petición Asignar Tarea	100	967	67	2296	0%	6.0
Login	100	456	10	1229	0%	6.1
Petición Usuario	100	932	55	1858	0%	6.2

Nota. *Resultados obtenidos mediante la herramienta JMeter para 100 muestras (hilos) la herramienta lo ejecutó en 17 segundos. Elaborado por: El autor.*

El objeto creado de Reporte Resumen en la herramienta JMeter para las peticiones de 100 hilos con un tiempo de ejecución de 17 segundos. En la tabla 11 permite ver que la carga tiene un 0% de error al recibir solicitudes simultáneas. Lo que para condiciones normales no representa problemas de carga. Sucede lo mismo con 20 y 50 hilos.

4.3.3. Pruebas de Rendimiento

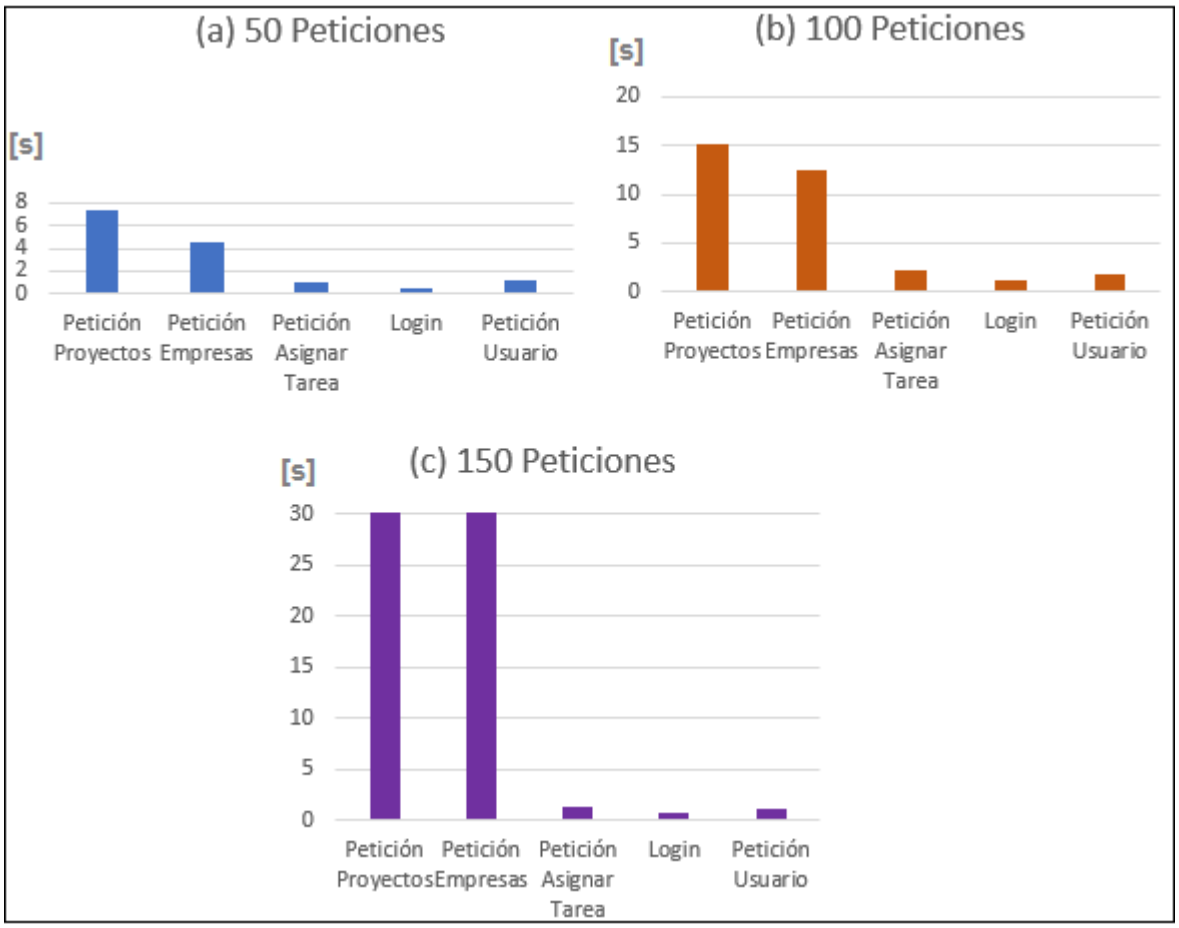
En esta prueba se buscó la caracterización de variables que indiquen que la aplicación web es fiable en operación, tiempo y ejecución. Para esta prueba se usó hasta 150 hilos de peticiones como se muestra en la Tabla 12.

Tabla 12
Resultados de tiempo máximo de hasta 150 hilos.

Peticiones/[s]	**(a) 50 Peticiones**	**(b) 100 Peticiones**	**(c) 150 Peticiones**	**Porcentaje de Error**
Petición Proyectos	7.315	15.149	30.119	0%
Petición Empresas	4.53	12.497	30.089	0%
Petición Asignar Tarea	1.039	2.296	1.387	0%
Login	0.536	1.229	0.692	0%
Petición Usuario	1.141	1.858	1.084	0%

***Nota**. Resultados obtenidos mediante la herramienta JMeter para 50, 100 y 150 muestras (hilos), la herramienta lo ejecutó en 8, 17 y 30 segundos respectivamente sin ningún error. Elaborado por: El autor.*

Figura 20
Resultados para 50, 100 y 150 hilos.



Nota. *Comparación de datos de tiempo máximo de respuesta de la aplicación para 50, 100 y 150 peticiones. El tiempo de respuesta se mantiene dentro de 30 segundos para peticiones simultáneas. Las peticiones para gestión de proyectos y empresas requieren más tiempo para ser atendidas. Datos obtenidos de la herramienta JMeter. Elaborado por: El autor.*

Las Figura 20 muestra 3 Subgráficas, la tupla Petición vs. Tiempo máximo que tarda su respuesta, Subgráfica (a) para 50 usuarios concurrentes, Subgráfica (b) para 100 usuarios concurrentes y Subgráfica (c) para 150 usuarios concurrentes. Para este caso no se busca hacer una prueba de estrés, pero como se puede observar mientras el número de hilos (peticiones) es mayor (Subgráfica (c)) el tiempo de respuesta es mayor, no obstante, por lo obtenido en la herramienta JMeter se considera el rendimiento estable ya que el porcentaje de error es cero

(mostrado en la Tabla 12), es decir se atienden las cargas y la reducción en rendimiento no es abrupta de modo que el servidor puede continuar operando normalmente.

4.3.4. Prueba de Estrés

Esta prueba permite conocer las capacidades máximas o los límites a los cuales se puede enfrentar la aplicación. En el primer escenario el servidor se encontraba con 3 aplicaciones corriendo, en el segundo 2 aplicaciones corriendo y en el último escenario el servidor se encontraba ejecutándose con 4 tareas a la par y un video de internet.

Tabla 13
Resultados para prueba de estrés.

Carga de 1000 Peticiones	**Escenario 1**	**Escenario 2**	**Escenario 3**	**Promedio**
Número de peticiones máximas alcanzadas con porcentaje de error 0.	485	535	295	438.33
Tiempo en atender todas las peticiones [m]	2.38	2.24	1.51	2.04

Nota. *Pruebas de estrés realizadas mediante 3 pasadas usando JMeter con 1000 muestras (peticiones concurrentes) en 3 escenarios diferentes. El tiempo se encuentra en minutos. Elaborado por: El autor.*

Con esta prueba cuyos resultados se recopila en la Tabla 13 se pudo determinar que las peticiones simultáneas en el rango de 1 a 294 peticiones no tendrán incidencia en el funcionamiento normal del sistema, el servidor los continuará atendiendo. Así también, el número máximo de peticiones simultáneas que se esperaría soporte el servidor y la aplicación es de 438 en un tiempo de estimado de 2 minutos. Es decir, las peticiones después de 438 empezarían a tener problemas al no lograr ser atendidas simultáneamente.

CONCLUSIONES

Se ha desarrollado una Aplicación Web que permite la gestión y seguimiento de proyectos de ensayos de campo del Departamento de Investigación que integra la lógica de negocio de una empresa de Registros Agrícolas. La aplicación se ha adaptado a los recursos de la empresa y es escalable a nuevos procesos.

Se realizó el análisis del estado actual del departamento de Investigación plaguicidas, lo que permitió comprender cómo interactúan las fases de cada proceso para completar un proyecto plaguicida con sus interventores y qué información y tipos de documentos generan cada uno de ellos. Este conocimiento previo sobre el giro de negocio de la empresa fue uno de los elementos clave que ayudó a optimizar tiempo en la descripción y levantamiento de requerimientos.

El uso de metodologías ágiles complementó el trabajo y desarrollo de la presente aplicación. El manejo correcto de estas herramientas también contribuyó a que se generen mayores detalles en la marcha, hacer correcciones y adicionar características implícitas para el correcto funcionamiento de la aplicación. El desarrollo del software en conjunto con los interesados permitió aplicar mejoras al producto final, las cuales brindaron características deseables y esperadas al usuario final, sin tener conflictos posteriores.

El uso del framework Django facilitó significativamente el desarrollo de este proyecto, ya que dispone de un modelo MTV equiparable al modelo tradicional MVC, pero mucho más simple, que además es limpio, organizado y en donde las situaciones relacionadas con conexiones son casi transparentes. En este framework, se sustituye el lenguaje SQL por sentencias ORM que son igual de legibles y casi inmediatas de

comprender, a través de un conjunto simple de sentencias, se pueden llegar a diseñar consultas tan complejas como se desee, aprovechando toda la funcionalidad que Django ofrece. Del mismo modo, la sintaxis de Django con Python permite compartir y aplicar características que son compatibles con lenguajes más elementales de un software web como son HTML, JavaScript entre otros. Otra ventaja de la arquitectura que maneja es la facilidad para reutilización de código.

A través de la herramienta Apache JMeter, se logró configurar diferentes pruebas que permitieron evidenciar el correcto funcionamiento de la aplicación web. La conservación y vida del software se puede llevar a cabo con un mantenimiento adecuado. Las pruebas de aceptaciones de la aplicación web quedan más claras al corroborar que responde ante las peticiones y la interacción del usuario con el sistema.

RECOMENDACIONES

Es importante tomar un tiempo apropiado para entender la lógica del negocio de la empresa que requiere una solución informática, ya que, bien es cierto que la mayoría de las aplicaciones de software y similares requerirán adecuaciones luego de la entrega, el plantear desde el inicio del proyecto un esquema adecuado, diagramas de flujo, análisis de requerimientos y las necesidades reales de la empresa, simplifica el trabajo y su ejecución.

Para el desarrollo se podría usar editores de código o IDEs adaptados al lenguaje. Estos IDEs pueden ayudar a que el trabajo sea más simple, organizado y limpio. Además, incluyen opciones como las de autocompletado, que permite que el tipeo de código sea más preciso y rápido.

Es recomendable que al integrar herramientas para medir el funcionamiento de la aplicación software se considere tomar una referencia inicial, para verificar su correcto funcionamiento al final del desarrollo del proyecto.

Se recomienda utilizar alguna herramienta para encapsular la aplicación, es decir utilizar entornos virtuales, debido a que previenen temas de incompatibilidad y actualizaciones. Por otro lado, el uso de herramientas por consola como Freeze de Python, permite guardar de manera sencilla la configuración del entorno virtual y todas las librerías y dependencias requeridas en el desarrollo de la aplicación. De igual forma, facilita la recuperación de la configuración del sistema, al migrar o cambiar de versión, o incluso el sistema operativo.

Se recomienda también someter a la aplicación a pruebas de accesibilidad y usabilidad web, ya que, aunque se ha verificado el funcionamiento de la herramienta en diferentes equipos, no se realizaron pruebas especificadas con este enfoque.

GLOSARIO

HTML: Lenguaje de Marcado de Hipertexto

IDE: Entorno de Desarrollo Integrado.

MVC: Modelo Vista Controlador.

MVT: Modelo Vista Plantilla.

ORM: Mapeo Relacional de Objetos.

SGBD: Sistema Gestor de Base de Datos.

SQL: Lenguaje de Consulta Estructurado.

REFERENCIAS

Artículos Académicos

Mera, J. (2016). Análisis del proceso de pruebas de calidad de software. Ingeniería Solidaria, 12(20), 163-176. https://doi.org/10.16925/in.v12i20.1482

Prieto, R., & Escalante, J. C. (2019, enero). REGISTRO DE PRODUCCIÓN AGRÍCOLA COMO HERRAMIENTA PARA LA TOMA DE DECISIONES. https://doi.org/10.13140/RG.2.2.17149.61926

Roman, A., & Chiang, K. (2017). Data Development: Making It Organized. Monetizing Your Data, 164-180. https://doi.org/10.1002/9781119356271.ch10

Bibliografía

Blanco, S. (2005). Manual básico Ubuntu GNU/Linux Versión (BETA) Breezy (1.a ed., Vol. 1). licencia libre Creative Commons denominada Reconocimiento-NoComercial-CompartirIgual 2.1 España.

Dayley, B. (2014). JQuery and JavaScript phrasebook (pp. 1,2). Upper Saddle River, NJ: Addison-Wesley.

Delfín Pozos, F., & Acosta Márquez, M. (2021). Analysis and relevance in business development. Retrieved 8 February 2020

Fogel, K. (2007). Producing Open-Source Software: How to Run a Successful Free Software Project (1.a ed., Vol. 1). O'Reilly.

Gauchat, J. (2017). El gran libro de HTML5, CSS3 y Javascript (3rd ed.). Barcelona: Marcombo.

González, H. (2015). MVC 4 con .Net desde cero: Guía práctica para implementar MVC 4 con C# y Visual Studio 2012/2013 (1.a ed., Vol. 1). Createspace Independent Publishing Platform.

Holovaty, A., Kaplan-Moss, J., & García, S. (2015). La guía definitiva de Django: Desarrolla aplicaciones Web de forma rápida y sencilla. (3rd ed.). Apress.

Marqués, M. (2011). Bases de datos (1st ed., pp. 11-12). Castelló de la Plana: Universitat Jaume I, Servei de Comunicació i Publicacions.

Marzal, A., García, P., & Gracia, I. (2014). Introducción a la programación con Python 3 (2nd ed.). Castellón de la Plana: Universitat Jaume I. Servei de Comunicació i Publicacions.

Menzinsky, A., López, G., & Palacio, J. (2016). Scrum Manager SafeCreative. 15-16.

Norton, P. C., Samuel, A., Aitel, D., Foster-Johnson, E., Richardson, L., Diamond, J., Parker, A., & Roberts, M. (2005). Beginning Python. Wiley.

Ramírez, E. (2010). Manual Básico Ubuntu GNU-Linux Introducción al software libre: Uso básico de sistema operativo y programas de ofimática (1.a ed., Vol. 1). Licencia Atribución-NoComercial 3.0 Costa Rica (CC BY-NC 3.0).

Rungta, K. (2019). Learn Jmeter in 1 Day: Definitive Guide to Learn Jmeter for Beginners (2nd ed.).

Silberschatz, A., Korth, H., & Sudarshan, S. (2011). Database systems concepts (6th ed., pp. 45-48). Estados Unidos: McGraw-Hill Companies, Inc.

Sommerville, I., Velázquez, S. F., & Olguín, V. C. (2011). Ingeniería de software. Pearson (México).

Wang, Z. (2018). Teamworking Strategies of Scrum Team. Proceedings Of The 2018 2Nd International Conference On Computer Science And Artificial Intelligence - CSAI '18. doi: 10.1145/3297156.3297179

Worsley, J. C., & Drake, J. D. (2002). Practical Postgresql (Pap/Cdr ed.). Oreilly & Associates Inc.

Conferencias

Ahmad, W., Sulaiman, S., & Johari, F. (2010). Usability Management System (USEMATE): A web-based automated system for managing usability testing systematically. 2010 International Conference On User Science And Engineering (I-User). doi: 10.1109/iuser.2010.5716734

Choudhary, B., & Rakesh, S. K. (2016). An approach using agile method for software development. 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH), 155-156. https://doi.org/10.1109/iciccs.2016.7542304

Ereiz, Z., & Music, D. (2019). Scrum Without a Scrum Master. 2019 IEEE International Conference On Computer Science And Educational Informatization (CSEI). doi: 10.1109/csei47661.2019.8938877

Herrero, J. L., Lucio, F., & Carmona, P. (2011). Web services and web components. 2011 7th International Conference on Next Generation Web Services Practices, 2-3. https://doi.org/10.1109/nwesp.2011.6088171

Katzburg, N., Golander, A., & Weiss, S. (2018). NVDIMM-N Persistent Memory and its Impact on Two Relational Databases. 2018 IEEE International Conference On The Science Of Electrical Engineering In Israel (ICSEE). doi: 10.1109/icsee.2018.8646020

Sekar, K. R., & Sethuraman, J. (2017). Optimal component selection for rich internet applications in web engineering. 2017 International Conference on Networks &

Advances in Computational Technologies (NetACT), 1-2. https://doi.org/10.1109/netact.2017.8076808

Zhou, J., Chen, L., Ding, H., Tu, J., & Xu, B. (2013). A Method of Optimizing Django Based on Greedy Strategy. 2013 10Th Web Information System And Application Conference. doi: 10.1109/wisa.2013.41

Sitios Web

DB-Engines Ranking. (2021). Retrieved 21 January 2021, from https://db-engines.com/en/ranking

Entornos Virtuales y Paquetes — documentación de Python - 3.9.1. Docs.python.org. (2021). Retrieved 8 January 2021, from https://docs.python.org/es/3/tutorial/venv.html.

Gallego, A. (2018, 8 abril). Bootstrap 4. PDFManuales. https://www.pdf-manual.es/programacion-web/css/177-bootstrap-4.html

González, R. (2018). Tutorial de Python. Python para todos. Mundogeek. http://mundogeek.net/tutorial-python/

Gour, R. (2019, 16 abril). Working Structure of Django MTV Architecture - Towards Data Science. Medium. https://towardsdatascience.com/working-structure-of-django-mtv-architecture-a741c8c64082

Raza, A., Jain, A., Bendoraitis, A., & Tyapkov, A. (2017). Aprendizaje Django. Retrieved 21 January 2021, from https://riptutorial.com/Download/django-es.pdf

Releases · atom/atom. GitHub. (2021). Retrieved 8 January 2021, from https://github.com/atom/atom/releases?after=v1.0.0.

ANEXOS

Anexo 1. Base de Datos del Sistema

Anexo 2. Instalación del Framework Django y Librerías utilizadas.