UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y MANTENIMIENTO DE VEHÍCULOS EN UNA EMPRESA PÚBLICA BASADOS EN LA METODOLOGÍA SNAIL

NAGUA PULLAGUARI DARWIN GUSTAVO
INGENIERO DE SISTEMAS

MACHALA
2024

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y MANTENIMIENTO DE VEHÍCULOS EN UNA EMPRESA PÚBLICA BASADOS EN LA METODOLOGÍA SNAIL

NAGUA PULLAGUARI DARWIN GUSTAVO
INGENIERO DE SISTEMAS

MACHALA
2024

UTMACH

FACULTAD DE INGENIERÍA CIVIL

CARRERA DE INGENIERÍA DE SISTEMAS

EXAMEN COMPLEXIVO

DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y MANTENIMIENTO DE VEHÍCULOS EN UNA EMPRESA PÚBLICA BASADOS EN LA METODOLOGÍA SNAIL

NAGUA PULLAGUARI DARWIN GUSTAVO
INGENIERO DE SISTEMAS

ZEA ORDOÑÉZ MARIUXI PAOLA

MACHALA, 30 DE ENERO DE 2024

MACHALA
30 de enero de 2024

Trabajo titulación

por Darwin Nava

Fecha de entrega: 18-ene-2024 10:12p.m. (UTC-0500)

Identificador de la entrega: 2273646331

Nombre del archivo: DESARROLLO_DE_SOFTWARE_PARA_LA_GESTI_N_DE_VEH_CULOS20240118.pdf (837.97K)

Total de palabras: 4848

Total de caracteres: 30685

CLÁUSULA DE CESIÓN DE DERECHO DE PUBLICACIÓN EN EL REPOSITORIO DIGITAL INSTITUCIONAL

El que suscribe, NAGUA PULLAGUARI DARWIN GUSTAVO, en calidad de autor del siguiente trabajo escrito titulado DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y MANTENIMIENTO DE VEHÍCULOS EN UNA EMPRESA PÚBLICA BASADOS EN LA METODOLOGÍA SNAIL, otorga a la Universidad Técnica de Machala, de forma gratuita y no exclusiva, los derechos de reproducción, distribución y comunicación pública de la obra, que constituye un trabajo de autoría propia, sobre la cual tiene potestad para otorgar los derechos contenidos en esta licencia.

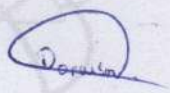
El autor declara que el contenido que se publicará es de carácter académico y se enmarca en las dispociones definidas por la Universidad Técnica de Machala.

Se autoriza a transformar la obra, únicamente cuando sea necesario, y a realizar las adaptaciones pertinentes para permitir su preservación, distribución y publicación en el Repositorio Digital Institucional de la Universidad Técnica de Machala.

El autor como garante de la autoría de la obra y en relación a la misma, declara que la universidad se encuentra libre de todo tipo de responsabilidad sobre el contenido de la obra y que asume la responsabilidad frente a cualquier reclamo o demanda por parte de terceros de manera exclusiva.

Aceptando esta licencia, se cede a la Universidad Técnica de Machala el derecho exclusivo de archivar, reproducir, convertir, comunicar y/o distribuir la obra mundialmente en formato electrónico y digital a través de su Repositorio Digital Institucional, siempre y cuando no se lo haga para obtener beneficio económico.

Machala, 30 de enero de 2024

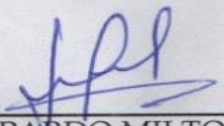


NAGUA PULLAGUARI DARWIN GUSTAVO
0704420579

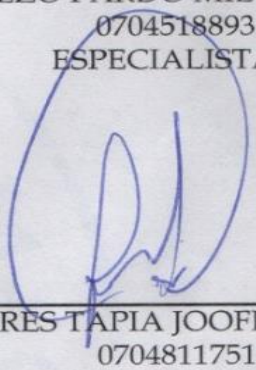
Nota de aceptación:

Quienes suscriben, en nuestra condición de evaluadores del trabajo de titulación denominado DESARROLLO DE SOFTWARE PARA LA GESTIÓN Y MANTENIMIENTO DE VEHÍCULOS EN UNA EMPRESA PÚBLICA BASADOS EN LA METODOLOGÍA SNAIL, hacemos constar que luego de haber revisado el manuscrito del precitado trabajo, consideramos que reúne las condiciones académicas para continuar con la fase de evaluación correspondiente.

ZEA ORDOÑEZ MARIUXI PAOLA
0702801598
TUTOR - ESPECIALISTA 1



VALAREZO PARDO MILTON RAFAEL
0704518893
ESPECIALISTA 2



HONORES TAPIA JOOFRE ANTONIO
0704811751
ESPECIALISTA 3

Fecha de impresión: Wednesday 31 de January de 2024 - 21:42

Trabajo titulación

INFORME DE ORIGINALIDAD

6%	5%	0%	2%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	repositorio.utmachala.edu.ec Fuente de Internet	3%
2	Submitted to Universidad Argentina John F. Kennedy Trabajo del estudiante	2%
3	oa.upm.es Fuente de Internet	1%
4	www.scielo.org.co Fuente de Internet	1%

Excluir citas	Activo	Excluir coincidencias	< 1%
Excluir bibliografía	Activo		

Dedicatoria

A mi amada familia,

A lo largo de este significativo trayecto, mi corazón se llena de gratitud al dedicar este logro a quienes han sido mis pilares inquebrantables. A mi madre y a mi padre, cuyo amor incondicional y sabias enseñanzas han iluminado cada paso de mi camino académico. A mi esposa, compañera leal, por su paciencia inquebrantable y constante aliento.

En particular, este trabajo está dedicado a mis hijos y hermanas, testigos de mi dedicación y esfuerzo. Aspiro a que este esfuerzo sirva como inspiración, guiándolos hacia la realización de sus propios sueños y aspiraciones

Nagua Pullaguari Darwin Gustavo

Agradecimiento

A mis padres,

No tengo palabras suficientes para expresar lo agradecido que estoy por su apoyo incondicional durante este proceso de titulación. Sus palabras de aliento fueron mi fuerza, y su presencia constante, mi ancla en los momentos más desafiantes. Simplemente, no sé cómo agradecerles lo suficiente.

A mi esposa,

Gracias por estar a mi lado, incluso en los días más largos y noches más oscuras. Tu apoyo constante ha sido mi roca, y sin importar los desafíos, siempre estuviste allí para sostenerme.

Este logro no solo es mío; es de quienes siempre creyeron en mí.

Nagua Pullaguari Darwin Gustavo

Resumen

Desarrollo de software para la gestión y mantenimiento de vehículos en una empresa pública basados en la metodología SNAIL.

Nagua Pullaguari Darwin Gustavo, 0704420579

Este informe detalla el desarrollo de un software web para la gestión y mantenimiento de vehículos en una empresa pública, bajo la metodología Snail. Considerando la magnitud del software y las necesidades de la empresa, se han seleccionado las siguientes herramientas: Visual Studio Code como entorno de desarrollo integrado (IDE), Python como lenguaje de programación y el marco de desarrollo web Flask. Para la base de datos, se ha optado por PostgreSQL, con la mirada puesta en futuras expansiones. El software se enfoca en la gestión de usuarios y vehículos, incluyendo funciones comunes como el ingreso y control de ambos. Sin embargo, también incorpora algoritmos innovadores que aportan una relevancia significativa, como el reconocimiento facial para el inicio de sesión y la clasificación de imágenes de vehículos. La estructura del proyecto se basa en una arquitectura de tres capas, siguiendo el patrón de diseño MVC, con el objetivo de lograr una programación ordenada y coherente que se refleje en un software web útil y fluido.

Palabras Clave: Python, Flask, PostgreSQL, SNAIL, MVC, Reconocimiento facial.

.

Abstract

Development of software for the management and maintenance of vehicles in a public company based on the SNAIL methodology.

Nagua Pullaguari Darwin Gustavo, 0704420579

This report details the development of web software for the management and maintenance of vehicles in a public company, under the Snail methodology. Considering the magnitude of the software and the needs of the company, the following tools have been selected: Visual Studio Code as an integrated development environment (IDE), Python as a programming language and the Flask web development framework. For the database, PostgreSQL has been chosen, with an eye on future expansions. The software focuses on the management of users and vehicles, including common functions such as entry and control of both. However, it also incorporates innovative algorithms that provide significant relevance, such as facial recognition for login and vehicle image classification. The project structure is based on a three-layer architecture, following the MVC design pattern, with the aim of achieving orderly and coherent programming that is reflected in useful and fluid web software.

Keywords: Python, Flask, PostgreSQL, SNAIL, MVC, Facial recognition.

Contenido

Contenido de Tablas

Contenido de Gráficos

Contenido de Imágenes

Introducción

En la actual era digital, las empresas buscan constantemente optimizar la eficiencia operativa mediante la incorporación de tecnologías avanzadas. Un ejemplo paradigmático se halla en el sector de transporte y logística, donde la introducción de sistemas de seguimiento y gestión de flotas ha generado transformaciones sustanciales en las operaciones de empresas líderes. El análisis de las operaciones de una destacada compañía de logística en Brasil revela claramente cómo estas herramientas tecnológicas han impactado positivamente en su desempeño, ilustrando de manera inequívoca cómo la adopción de soluciones tecnológicas, particularmente en la gestión de flotas, puede aportar mejoras significativas en la eficiencia operativa y la gestión integral de la cadena de suministro.[1]

En paralelo, el sector ferroviario también enfrenta desafíos similares en la gestión de flotas. La implementación de mantenimiento predictivo y basado en condiciones podría reducir significativamente los costos, aunque se destaca el desafío de la transición hacia una gestión holística de la flota. Este ejemplo propone una solución que optimiza dinámicamente el mantenimiento de activos considerando el tiempo de vida útil restante, requisitos operativos y limitaciones de capacidad de mantenimiento, promoviendo una gestión más eficiente y una reducción de paradas de activos.[2]

Este proyecto se enfoca en mejorar la gestión y mantenimiento de vehículos en una empresa pública. Se utiliza la metodología SNAIL, que destaca por su simplicidad, comunicación y planificación del código. El sistema, desarrollado con tecnologías como Python, Flask y PostgreSQL, incorpora patrones de diseño como el Modelo-Vista-Controlador (MVC) y principios SOLID. Además, se integran medidas de seguridad como tokens CSRF y reconocimiento facial mediante bibliotecas como Haar Cascade, OpenCV y Face_Recognition.

La metodología SNAIL se desglosa desde la identificación de requisitos hasta la clausura del proyecto, adaptándose a las necesidades del desarrollo. Los resultados obtenidos confirman el cumplimiento de requisitos para la gestión de usuarios y vehículos en la empresa, respaldados por pruebas de sistema, integración y específicas.

La efectividad del algoritmo de clasificación de imágenes se evalúa mediante pruebas específicas, contribuyendo a la evaluación global del sistema. En conjunto, estas pruebas responden a la pregunta de investigación sobre si la aplicación del software basado en la metodología SNAIL facilita la gestión de información, tiempos y actividades en los vehículos.

El trabajo detalla el marco teórico y metodológico, así como los resultados que respaldan la eficacia del enfoque implementado.

1. Marco teórico

1.1. Desarrollo Web con Python y Flask:

1.1.1. Visual Studio

Visual Studio Code es definido como una plataforma de código abierto, un editor de código de multiplataforma que pertenece a Microsoft y proporciona todos los componentes necesarios de un IDE como: IntelliSense, depuración, control de versiones, creación de plantillas y API de extensiones que brindan muchas facilidades a los desarrolladores.[3]

1.1.2. Python

Python es un lenguaje de programación de propósito general, que permite una fácil extensión de las habilidades aprendidas a diferentes áreas de aplicación, como análisis de imágenes, procesamiento de lenguaje natural, minería de datos o aprendizaje automático en conferencias posteriores que se basan en nuestro curso introductorio.[4]

Python funciona sin costo es Windows, Unix, Linux, y otros sistemas operativos con una sintaxis más simple y elegante que la de otros lenguajes de programación.[5]

1.1.3. Flask

Flask es una plataforma micro web basada en Python que permite a los usuarios agregar funcionalidades de aplicaciones como si estuvieran integradas en el propio marco. [6] Crear una aplicación web con Python y Flask mejora la robustez del sistema, garantizando su solidez y confiabilidad.[7]

1.1.4. Patrones de Diseño

1.1.4.1. Patrones de Diseño MVC (Modelo Vista Controlador)

El patrón de diseño MVC, proporciona diversas formas de visualizar e interactuar con los datos, por lo que para el diseño del sistema web SIMEFF se toma como referencia principal este patrón de diseño, sin embargo, es posible aplicar las características de los patrones cliente-servidor y el de capas.[8]

La implementación de la arquitectura Modelo Vista Controlador (MVC) en Flask facilita el desarrollo de aplicativos webs al organizar la lógica en capas, proporcionando mayor velocidad de carga de información sin aumentar el tamaño total de los archivos.[9] En resumen, MVC en Flask

ofrece una estructura eficiente para optimizar el rendimiento y la gestión de proyectos web.

1.1.4.2. Principios de desarrollo SOLID

SOLID es un acrónimo formado por la primera letra de cada principio. Son los siguientes:

Tabla 1: Principios SOLID

	Principio	Descripción
S	Single Responsibility Principle	Principio de Responsabilidad Única.
O	Open/Closed Principle	Establece que el código debe ser abierto a la ampliación, pero cerrado a la modificación.
L	Liskov Substitution Principle	Una subclase debe poder ser sustituida por su superclase sin que el programa deje de funcionar.
I	| Interface Segregation Principle	Si una interfaz es demasiado general y hace que no todas las clases que la usan empleen todas las funciones, debe ser dividida.
D	| Dependency Inversion Principle	Establece que las implementaciones deben depender de abstracciones, no de concreciones ni detalles.

Fuente: Elaboración propia en base al Artículo “Metodologías ágiles de desarrollo aplicadas a la enseñanza de la programación”. [10]

1.2. Base de Datos y Persistencia:

1.2.1. PostgreSQL

La elección de PostgreSQL como estructura de base de datos back-end se fundamenta en su arquitectura cliente-servidor, filosofía de código abierto, alta confiabilidad, robustez y soporte para la accesibilidad multiusuario.[11]

1.2.1.1. PgAdmin

Es una aplicación gráfica de diseño y manejo de base de datos PostgreSQL. PgAdmin se diseña para responder a las necesidades de la mayoría de los usuarios, desde escribir simples consultas SQL hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y hace simple la administración.[12]

1.2.2. Alembic

Alembic es una herramienta orientada a la creación e implementación de migraciones a bases de datos, específicamente realizada para SQLAlchemy, otra librería popular dirigida al mapeo de relaciones de objetos.[13]

1.2.3. Migraciones con Flask-Migrate

Extensión de flask que maneja la gestión de la base de datos de SQLAlchemy, como por ejemplo añadir tablas, campos a la base de datos, etc., usando Alembric.[14]

Tabla 2: *Comandos utilizados con Flask-Migrate*

Comandos	Descripción
flask db init:	Inicializa una carpeta en el servidor principal destinada a scripts que actualizan la estructura de la base de datos, sin afectar los datos existentes.
flask db migrate -m	Genera un script de migración utilizado para actualizar la estructura de las tablas en la BD. En la primera ejecución, también crea la base de datos.
flask db upgrad:	Ejecuta el script de migración para actualizar la base de datos, dejándola creada después de este paso.

Fuente: Elaboración propia en base
al Trabajo de grado de Jesús Gálvez Guerrero. [14]

1.3. Seguridad y Autenticación:

1.3.1. Token CSRF

El ataque CSRF abusa del hecho de que los navegadores envían automáticamente información de autenticación, como cookies, a cualquier solicitud al sitio web autenticado, independientemente del origen de la solicitud. Por lo general, un atacante engaña a un usuario para que navegue a un sitio controlado por el atacante, por ejemplo, www.attacker.com, que luego envía una solicitud al sitio víctima www.vulnerable.com, mientras el usuario inicia sesión en www.vulnerable.com. en el mismo navegador. [15]

Un token CSRF debe cumplir con los siguientes requisitos: debe ser único, secreto e impredecible, es decir, debe de generarse de manera completamente aleatoria mediante un método seguro. Se deben generar

siempre en el lado del servidor, pudiéndose crear una vez por sesión de usuario o por cada petición mandada. Este segundo caso es más seguro ya que el intervalo que posee un atacante para explotarlo es mínimo.[16]

1.4. Reconocimiento Facial

El reconocimiento facial es una herramienta muy importante en el medio que permite identificar a través de ciertas características a un individuo, aunque a veces resulta beneficioso el reconocimiento facial, no se debe olvidar que el mal uso del reconocimiento facial afecta de gran manera al desarrollo del individuo.[17]

1.4.1. Haar Cascade

Clasificador encargado de detectar la presencia de un rostro en un fotograma de video.[18]

1.4.2. OpenCV

OpenCV (Visión por computadora de código abierto) es una biblioteca de funciones de programación dirigidas principalmente a la visión por computadora en tiempo real, desarrollada originalmente por Intel, luego fue respaldada por Willow Garage e Itseez. La biblioteca es multiplataforma y de uso gratuito bajo la licencia BSD de código abierto.[19]

1.4.3. Face_Recognition

Face_Recognition es una librería de Python que emplea una red neuronal convolucional tipo ResNet de 27 capas y el descriptor HOG para el reconocimiento y manipulación de rostros en imágenes. Ofrece funciones variadas, desde detectar rostros en imágenes o vídeos hasta identificar personas específicas y medir similitudes entre caras.[20]

2. Marco Metodológico

2.1. Metodología de Desarrollo

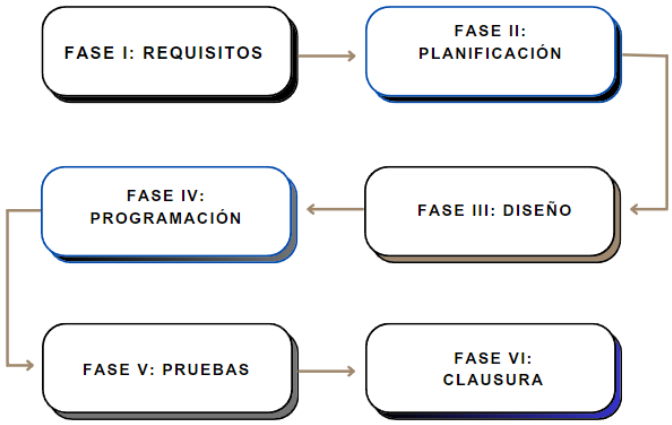
2.1.1. Metodología SNAIL

SNAIL (Software Nativo de Arquitectura Iterativa Lógica), es una metodología híbrida de desarrollo de aplicaciones web, que se basa en la simplicidad, comunicación y planificación del código desarrollado, su nombre nace de la forma que tiene el modelo de sus fases, ya que, al ser un modelo en espiral, toma una forma similar a la de un caracol.[21]

2.1.2. Fases de SNAI

Para la correcta elaboración de este proyecto, se tomó en cuenta cada una de las fases que nos brinda la metodología SNAIL.

Gráfico 1: Fases de la metodología SNAIL

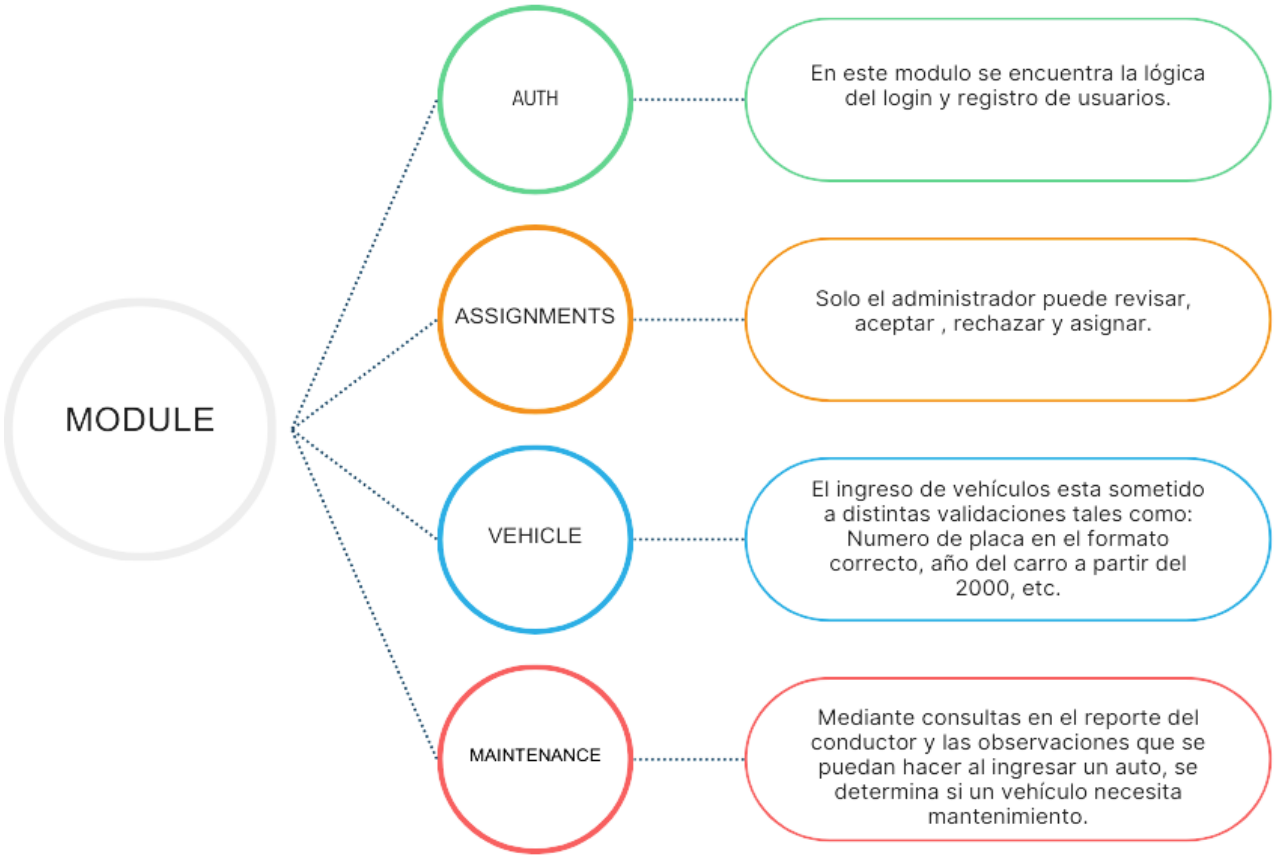


Fuente: Elaboración propia en base al Libro “Una metodología híbrida para el desarrollo de aplicaciones web”.[21]

2.1.2.1. Fase 1: Requisitos

Durante la fase de requisitos, se identificaron los módulos que componen el sistema web. Además, se especificaron requisitos particulares, como la implementación del reconocimiento facial para el ingreso de usuarios.

Gráfico 2: Módulos del software



Fuente: Elaboración propia

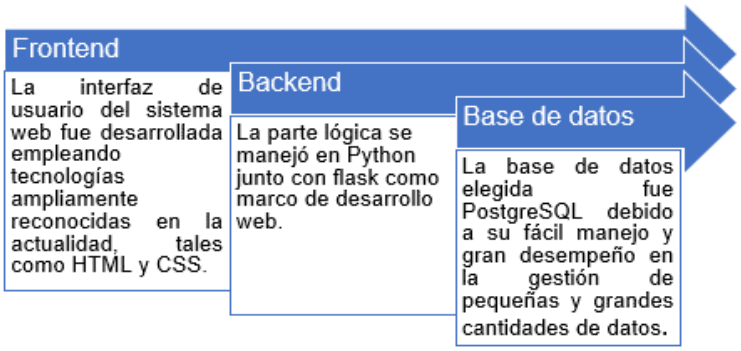
2.1.2.2. Fase 2: Planificación

La duración estimada del proyecto es de alrededor de dos meses, con el objetivo de desarrollar un software que satisfaga los requisitos de una empresa pública en la gestión y mantenimiento de vehículos. Durante la implementación de este software web, se anticipa la necesidad de gastos que podrían incluir la adquisición de un nombre de dominio (DNS) y otras herramientas de asistencia virtual.

2.1.2.3. Fase 3: Diseño

Se definió la arquitectura del proyecto, estructurándolo en tres capas fundamentales: Frontend, Backend y Base de Datos:

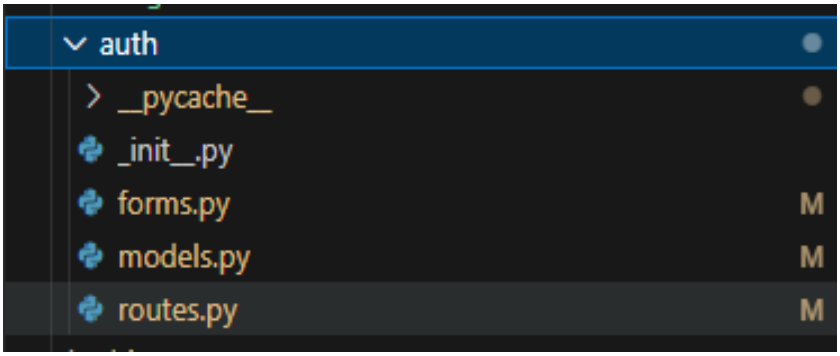
Gráfico 3: Arquitectura del Sistema: Componentes y Funciones



Fuente: Elaboración Propia

Además, se implementa el patrón de diseño Modelo-Vista-Controlador (MVC), que proporciona una estructura organizada y modular para facilitar el desarrollo y la mantenibilidad del software. Esta arquitectura en capas, combinada con el patrón de diseño MVC, contribuirá a la eficaz implementación y gestión del proyecto a lo largo de su ciclo de vida.

Imagen 1: Implementación del patrón de diseño MVC



Fuente: Elaboración Propia

2.1.2.4. Fase 4: Programación

Durante la Fase de Programación, se llevaron a cabo las instalaciones necesarias de bibliotecas fundamentales para el desarrollo de la aplicación. Entre estas bibliotecas se incluyen Flask, Flask-Migrate, Flask-Login, y request, que desempeñan roles cruciales en la creación de la aplicación web y la gestión de la autenticación de usuarios.

2.1.2.5. Fase 5: Pruebas

Durante esta etapa, se implementaron dos tipos de pruebas:
Las Pruebas de Integración se enfocaron en verificar el funcionamiento adecuado de cada ruta de la aplicación, incluyendo las validaciones del sistema.

En cuanto a las Pruebas del Sistema, se solicitó a usuarios externos que interactuaran con el sistema con el propósito de descubrir posibles errores no identificados en las etapas previas de pruebas.

2.1.2.6. Fase 6: Clausura

En este punto, se realiza una revisión exhaustiva para asegurar que todos los objetivos del proyecto se hayan cumplido según lo planeado. Cada fase, desde la definición de requisitos hasta la implementación y las pruebas, se evalúa para garantizar la coherencia y el logro de los resultados deseados

3. Resultados

Después de concluir el desarrollo del sistema web y realizar un análisis exhaustivo de cada fase según la metodología SNAIL, confirmamos que la aplicación cumple con los requisitos establecidos para la gestión de usuarios y vehículos en la empresa. Además, proporciona información detallada sobre posibles mantenimientos necesarios para cada vehículo.

Tabla *3:* *Pruebas de sistema a los diferentes módulos*

	Validaciones	Interfaz
Modulo Autentificación	☑	☒
Modulo vehículos	☑	☑
Módulos Asignaciones	☑	☒
Módulos Mantenimiento	☒	☒

Fuente: Elaboración Propia

La Tabla 3 fue realizada en base a los comentarios positivos o negativos de personas externas al proyecto. Estas pruebas ayudan a prestar un especial enfoque en la interfaz del sistema.

Tabla 4: Pruebas de Integración para el registro de vehículos

#	**Funcionalidad**	**Requisitos / Componentes involucrados**	**Prueba Exitosa**
1	Autentificación	Módulo de Usuarios + Base de Datos	Correcto
2	Registro de Vehículos	Módulo de Vehículos + Base de Datos	Correcto
3	Asignación de Tareas	Módulo de Asignaciones + Módulo de Usuarios	Correcto
4	Mantenimiento	Módulo de Mantenimiento + Base de Datos	Correcto

Fuente: Elaboración Propia

La tabla 4 es el resultado de las pruebas de integración, en estas pruebas se corroboro el correcto funcionamiento de los módulos integrados con la base de datos.

Tabla 5: Imágenes para la tabla de pruebas.

#	Imagen	Nombre
1		Imagen 1
2		Imagen 2
3		Imagen 3
4		Imagen 4
5		Imagen 5
6		Imagen 6

Fuente: Elaboración Propia

Imagen 2: Tabla de prueba en Excel - Prueba #1

TÍTULO DE LA PRUEBA	PRIORIDAD	ID DE CASO DE PRUEBA	NÚMERO DE PRUEBA	FECHA DE LA PRUEBA
Algoritmo Iscar()	Alto	1	1	01/18/2024
DESCRIPCIÓN DE LA PRUEBA		**PRUEBA DISEÑADA POR**	**PRUEBA EJECUTADA POR**	**FECHA DE EJECUCIÓN**
Ingreso de imágenes con el algoritmo Iscar()		Darwin Nagua Pullaguari	Darwin Nagua Pullaguari	01/18/2024

DESCRIPCIÓN DE LA PRUEBA	DEPENDENCIAS DE PRUEBA	CONDICIONES DE PRUEBA	CONTROL DE PRUEBAS
Ingreso de imágenes con el algoritmo Iscar()	Modulo Vehiculo - Ingreso de vehiculos	Imágenes de prueba con y sin automóviles	Validaciones para que el campo imagen no este vacio.

ID DE PASO	DESCRIPCIÓN DEL PASO	FECHA DE LA PRUEBA	RESULTADOS ESPERADOS	RESULTADOS REALES	APROBAR / REPROBAR	NOTAS ADICIONALES
1	Ingreso de la imagen 1 que se encuentra en la tabla 5.	01/18/2024	La imagen es un carro por lo que debe reconocerla e ingresarla.	No la detecto como carro	x	La imagen es de un carro de dibujo, talvez por ese motivo no lo reconoce.
1	Ingreso de la imagen 2 que se encuentra en la tabla 5.	01/18/2024	La imagen es un carro por lo que debe reconocerla e ingresarla.	Ingreso Correcto	ok	
1	Ingreso de la imagen 4 que se encuentra en la tabla 5.	01/18/2024	La imagen no es un carro, se espera que la rechaze.	La detecto como carro	ok	El algoritmo se enfoca en datalles especificos de un carro, en este caso la placa.
1	Ingreso de la imagen 5 que se encuentra en la tabla 5.	01/18/2024	La imagen no es un carro, se espera que la rechaze.	Rechazo correcto	x	

Fuente: Elaboración Propia

Imagen 3: Tabla de prueba en Excel – Prueba #2

TÍTULO DE LA PRUEBA	PRIORIDAD	ID DE CASO DE PRUEBA	NÚMERO DE PRUEBA	FECHA DE LA PRUEBA
Algoritmo Iscar()	Alto	1	2	01/19/2024
DESCRIPCIÓN DE LA PRUEBA		**PRUEBA DISEÑADA POR**	**PRUEBA EJECUTADA POR**	**FECHA DE EJECUCIÓN**
Ingreso de imágenes con el algoritmo Iscar()		Darwin Nagua Pullaguari	Darwin Nagua Pullaguari	01/19/2024

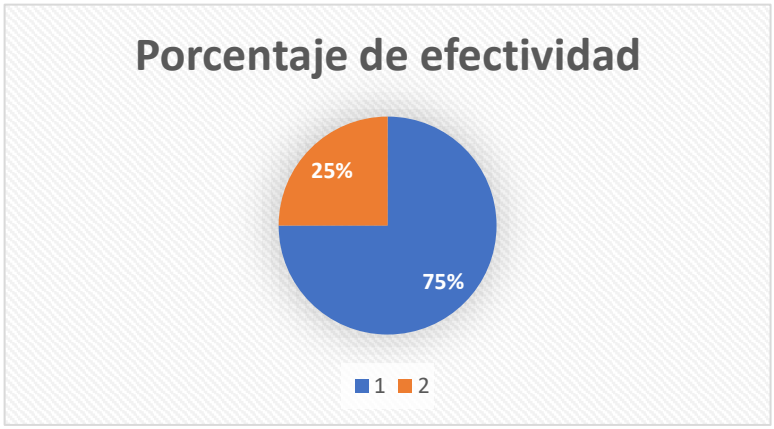
DESCRIPCIÓN DE LA PRUEBA	DEPENDENCIAS DE PRUEBA	CONDICIONES DE PRUEBA	CONTROL DE PRUEBAS
Ingreso de imágenes con el algoritmo Iscar()	Modulo Vehiculo - Ingreso de vehiculos	Imágenes de prueba con y sin automóviles	Validaciones para que el campo imagen no este vacio.

ID DE PASO	DESCRIPCIÓN DEL PASO	FECHA DE LA PRUEBA	RESULTADOS ESPERADOS	RESULTADOS REALES	APROBAR / REPROBAR	NOTAS ADICIONALES
1	Ingreso de la imagen 2 que se encuentra en la tabla 5.	01/19/2024	La imagen es un carro por lo que debe reconocerla e ingresarla.	No la detecto como carro	ok	
2	Ingreso de la imagen 3 que se encuentra en la tabla 5.	01/19/2024	La imagen es un carro por lo que debe reconocerla e ingresarla.	Ingreso Correcto	ok	
3	Ingreso de la imagen 5 que se encuentra en la tabla 5.	01/19/2024	La imagen no es un carro, se espera que la rechaze.	Rechazo correcto	ok	
4	Ingreso de la imagen 6 que se encuentra en la tabla 5.	01/19/2024	La imagen no es un carro, se espera que la rechaze.	Ingreso correcto	ok	

Fuente: Elaboración Propia

Las pruebas unitarias realizadas para el algoritmo IsCar (), el cual se encarga de clasificar si una imagen es carro o no, arrojaron que, de un total de 8 intentos, en 6 ocasiones el algoritmo realizo la correcta clasificación de imagen, y tan solo en 2 intentos fracaso.

Gráfico 4: Porcentaje de efectividad del algoritmo ISCAR ()



Fuente: Elaboración Propia

Finalizando el informe de las pruebas unitarias, tenemos que el algoritmo IsCar() tiene un 75% de efectividad al momento de clasificar un a imagen, permitiendo solo el ingreso de imágenes de carros al sistema.

En resumen, la combinación de pruebas de sistema, pruebas de integración y pruebas específicas proporciona una evaluación completa del sistema web desarrollado. Estas pruebas son fundamentales para asegurar que la aplicación cumple con los requisitos establecidos y puede desempeñar eficientemente sus funciones en la gestión de usuarios, vehículos y asignaciones dentro de la empresa.

4. Conclusión

La aplicación del enfoque SNAIL en el desarrollo del sistema web, ha demostrado ser efectiva al proporcionar una estructura lógica y una planificación iterativa. El diseño basado en fases, desde la conceptualización de requisitos hasta las pruebas exhaustivas, ha permitido una gestión clara del proyecto.

Las pruebas realizadas, tanto de integración como de sistema, junto con las pruebas específicas, han proporcionado una evaluación completa del sistema web. La tabla de pruebas muestra resultados positivos en la mayoría de los casos, validando la funcionalidad del sistema en áreas críticas como autenticación, registro de vehículos y asignación de tareas

Bibliografía

[1] M. Vivaldini, S. R. I. Pires, y F. B. de Souza, «Improving Logistics Services Through the Technology Used in Fleet Management», *JISTEM: Journal of Information Systems and Technology Management*, vol. 9, n.º 3, pp. 541-562, 2012.

[2] A. Crespo del Castillo, J. A. Marcos, y A. K. Parlikad, «Dynamic fleet maintenance management model applied to rolling stock», *Reliability Engineering & System Safety*, vol. 240, p. 109607, dic. 2023, doi: 10.1016/j.ress.2023.109607.

[3] J. E. M. R. Campos, C. S. C. Rodríguez, L. D. A. Luján, y A. C. M. de los Santos, «Sistema de reconocimiento facial para el control de accesos mediante Inteligencia Artificial», *Innovación y Software*, vol. 4, n.º 1, pp. 24-36, 2023.

[4] J. Lasser, D. Manik, A. Silbersdorff, B. Säfken, y T. Kneib, «Introductory data science across disciplines, using Python, case studies, and industry consulting projects», *Teaching Statistics*, vol. 43, n.º S1, pp. S190-S200, 2021, doi: 10.1111/test.12243.

[5] C. L. Vidal-Silva *et al.*, «Experiencia académica en desarrollo rápido de sistemas de información web con Python y Django», *Formación universitaria*, vol. 14, n.º 5, pp. 85-94, oct. 2021, doi: 10.4067/S0718-50062021000500085.

[6] N. Ahmed *et al.*, «Machine learning based diabetes prediction and development of smart web application», *International Journal of Cognitive Computing in Engineering*, vol. 2, pp. 229-241, jun. 2021, doi: 10.1016/j.ijcce.2021.12.001.

[7] I. Anshori *et al.*, «Web-based surface plasmon resonance signal processing system for fast analyte analysis», *SoftwareX*, vol. 18, p. 101057, jun. 2022, doi: 10.1016/j.softx.2022.101057.

[8] K. D. L. C. Silva, J. F. G. Portillo, A. D. D. J. Islao, y F. J. G. Mata, «Diseño del sistema web SIMEFF para la gestión de presupuestos sanitarios de acuicultura», *Ciencia Latina Revista Científica Multidisciplinar*, vol. 7, n.º 1, pp. 7463-7479, mar. 2023, doi: 10.37811/cl_rcm.v7i1.4979.

[9] A. Piedrahíta-Carvajal, P. A. Rodríguez-Marín, D. F. Terraza-Arciniegas, M. Amaya-Gómez, L. Duque-Muñoz, y J. D. Martínez-Vargas, «Aplicación web para el análisis de emociones y atención de estudiantes», *TecnoLógicas*, vol. 24, n.º 51, pp. 1-15, 2021.

[10] M. Benito y J. Felipe, «Metodologías ágiles de desarrollo aplicadas a la enseñanza de la programación», jun. 2021, Accedido: 15 de enero de 2024. [En línea]. Disponible en: http://rua.ua.es/dspace/handle/10045/116132

[11] J. R. Klaasse, L. C. Alewijnse, y J. van der Weerd, «TraceBase; A database structure for forensic trace analysis», *Science & Justice*, vol. 61, n.º 4, pp. 410-418, jul. 2021, doi: 10.1016/j.scijus.2021.03.001.

[12] P. Hernández Rodríguez, E. Blanco Camejo, O. Y. Rojas Grass, y M. Á. Sánchez Palmero, «Sistema para la Dirección de Compras de la UCI», *Serie Científica de la Universidad de las Ciencias Informáticas*, vol. 12, n.º 9, pp. 73-82, 2019.

[13] Á. Martínez Quiroga, «APP4AUCTIONS», jul. 2023, Accedido: 8 de enero de 2024. [En línea]. Disponible en: https://eciencia.urjc.es/handle/10115/23128

[14] J. Gálvez Guerrero, «Diseño y programación de la interfaz web y la gestión de la conexión con el exterior, de los laboratorios remotos para el aprendizaje del microcontrolador LCP4088 y la placa Arduino Mega.», bachelor thesis, 2022. Accedido: 15 de enero de 2024. [En línea]. Disponible en: https://rodin.uca.es/handle/10498/27317

[15] K. Peguero y X. Cheng, «CSRF protection in JavaScript frameworks and the security of JavaScript applications», *High-Confidence Computing*, vol. 1, n.º 2, p. 100035, dic. 2021, doi: 10.1016/j.hcc.2021.100035.

[16] P. F. Gayol, «Estudio de los Principales Tipos de Ataques por Inyección de Código a Aplicaciones Web y Sistema para Determinar si un Código Fuente es Vulnerable a SQL Injection».

[17] J. A. Cadena Moreano, R. H. Montaluisa Pulloquinga, G. A. Flores Lagla, J. C. Chancusig Chisag, y O. A. Guaypatín Pico, «Reconocimiento facial con base en imágenes», *Boletín Redipe*, vol. 6, n.º 5, pp. 143-151, 2017.
[18] F. Oliveira-Teixeira, T. P. Donadon-Homem, y A. Pereira-Junior, «Aplicación de inteligencia artificial para monitorear el uso de mascarillas de protección», *Revista Científica General José María Córdova*, vol. 19, n.º 33, Art. n.º 33, ene. 2021, doi: 10.21830/19006586.725.
[19] A. V. Nuñez y L. N. Nuñez, «Reconocimiento facial para el encendido automático de vehículos basado en Raspberry Pi», *REVISTA ODIGOS*, vol. 1, n.º 2, Art. n.º 2, jun. 2020, doi: 10.35290/ro.v1n2.2020.326.
[20] D. A. Caro Barranco y A. C. López Roncancio, «Sistema Inteligente para el Registro de Asistencia Basado en Procesamiento Digital de Imágenes y Redes Neuronales Convolucionales», dic. 2018, Accedido: 8 de enero de 2024. [En línea]. Disponible en: https://manglar.uninorte.edu.co/handle/10584/8485
[21] J. R. Molina Ríos, M. P. Zea Ordóñez, F. F. Redrován Castillo, N. M. Loja Mora, M. R. Valarezo Pardo, y J. A. Honores Tapia, *SNAIL, Una metodología híbrida para el desarrollo de aplicaciones web*, 1.ª ed. Editorial Científica 3Ciencias, 2018. doi: 10.17993/IngyTec.2018.38.

Anexos

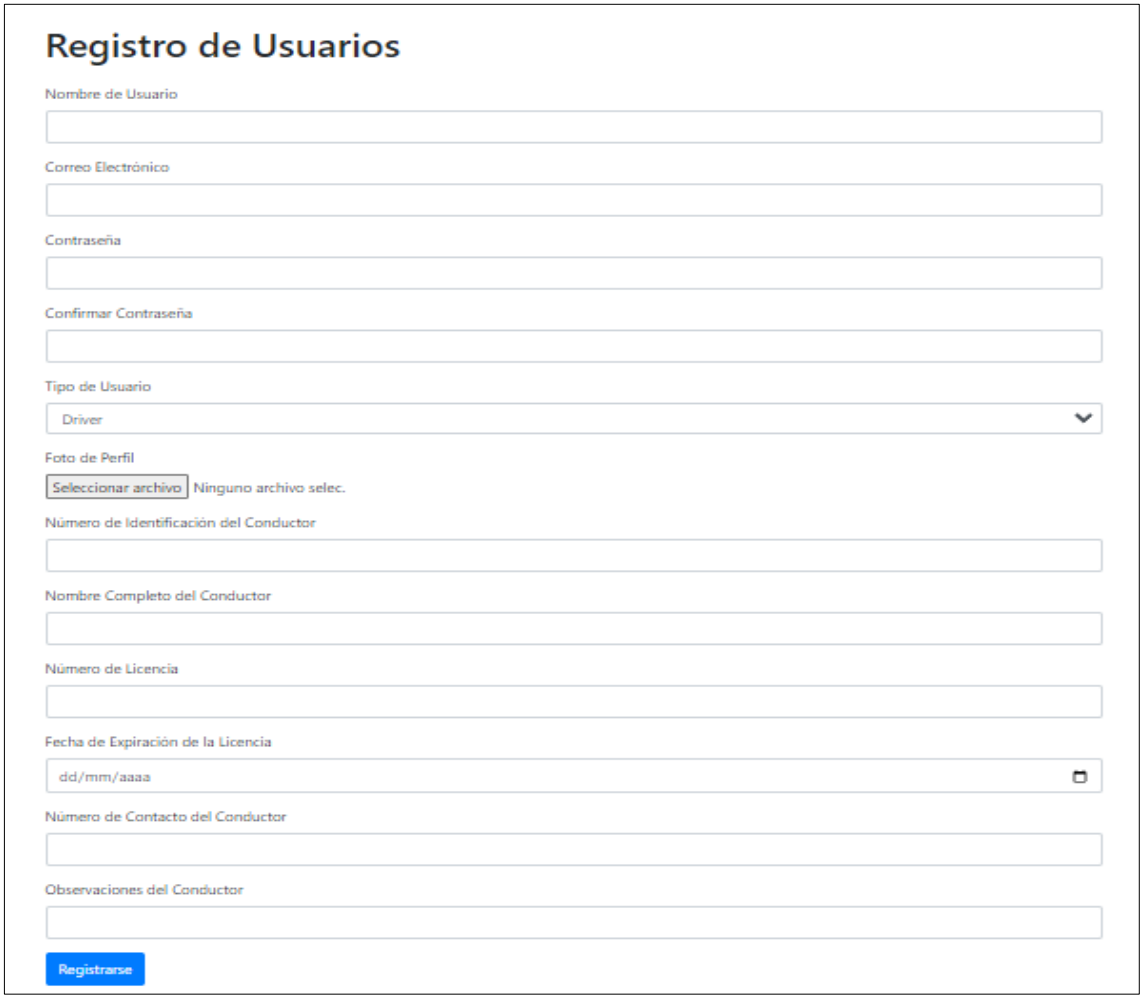
Imagen 4: Login

Fuente: Elaboración Propia

Imagen 5: Página Principal – Administrador

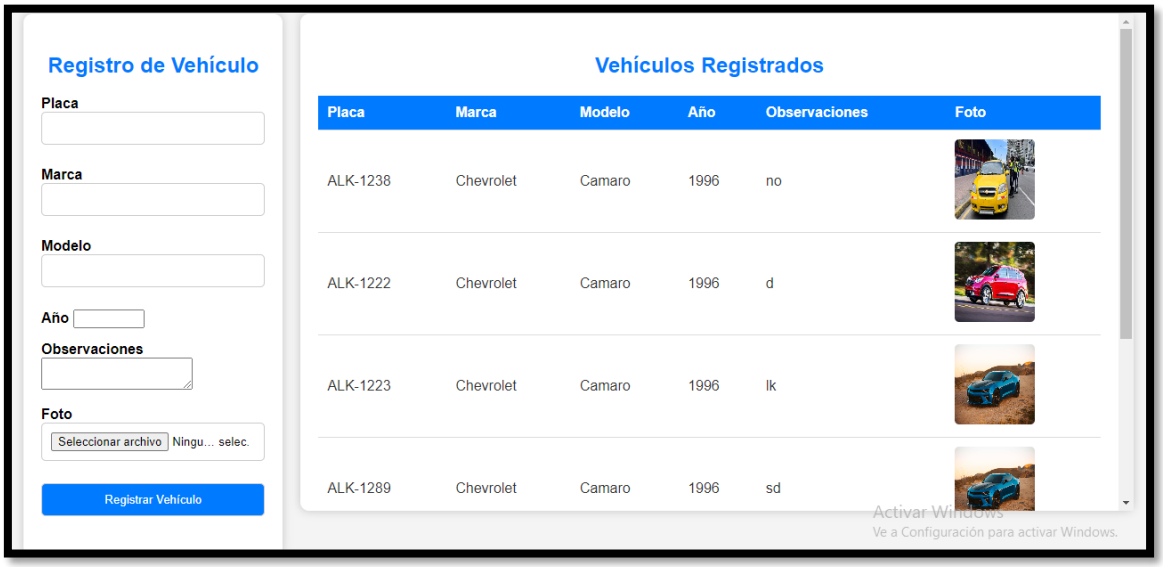
Fuente: Elaboración Propia

Imagen 6: Registro de Usuarios



Fuente: Elaboración Propia

Imagen 7: Registro de Vehículos



Fuente: Elaboración Propia