**UNIVERSIDAD CATÓLICA DE EL SALVADOR**

**FACULTAD DE INGENIERÍA Y ARQUITECTUR**

**Imagen que contiene Diagrama

Descripción generada automáticamente**

**ASIGNATURA:**

**TÉCNICAS DE PRODUCCIÓN DE SISTEMAS “A”**

**TEMA**

**PLATAFORMA DE GESTIÓN DE PROYECTOS PARA EQUIPOS COLABORATIVOS**

**DOCENTE**

**ING. KARLA MICHELLE LÓPEZ DE QUINTANA**

|  |  |
| --- | --- |
| **PRESENTAN:** | **CARNET** |
| 1.- Francisco José Valle Salazar | 2022-VS-601 |
| 2.- Alejandra Isabel Hernández Rodríguez | 2022-HR-651 |
| 3.- Joel Enrique Paniagua García | 2022-PG-651 |
| 4.- Oscar Edgardo Cárcamo Ibarra | 2022-CI-650 |
| 5.- Marlon Steven Interiano Villeda | 2022-IV-650 |
| 6.- Diego Ernesto Portillo Alfaro | 2023-PA-651 |
| 7.- Miguel Ángel Arévalo Ramos | 2022-AR-652 |

**SANTA ANA, OCTUBRE DE 2025**

Índice

[1. Introducción 3](#_Toc210245721)

[2. Descripción del Branching y Ramas 3](#_Toc210245722)

[3. Uso de Herramientas de Versionamiento 5](#_Toc210245723)

[3.1. Creación del Repositorio y Gestión de Ramas 5](#_Toc210245724)

[4. Historial de Cambios (Commits) 5](#_Toc210245725)

[5. Archivo CHANGELOG.md 9](#_Toc210245726)

[6. Conclusiones 10](#_Toc210245727)

[7. Enlace a GitHub 10](#_Toc210245728)

# 1. Introducción

En este informe se presenta el primer avance del proyecto, enfocado en el uso de herramientas de control de versiones. En esta etapa se trabajó con Git y GitHub para gestionar el repositorio, organizar las ramas del proyecto y mantener un registro claro de los cambios realizados durante el desarrollo.

**Objetivo General**

Implementar un sistema de control de versiones en el proyecto de software, utilizando Git y una plataforma de repositorios en línea, con el fin de gestionar de manera organizada el desarrollo, integrar nuevas funcionalidades y documentar el historial de cambios.

**Objetivos Específicos**

1. Configurar una estructura de ramas (main, development y feature) que permita gestionar las distintas fases del proyecto y facilite la colaboración en equipo.
2. Documentar los cambios realizados mediante commits descriptivos y un archivo CHANGELOG.md, garantizando la trazabilidad y el registro claro de las modificaciones.

# 2. Descripción del Branching y Ramas

En el proyecto se utilizan tres ramas principales para organizar el desarrollo:

**main**

* Rama principal del proyecto.
* Contiene el código estable, probado y listo para producción.
* Solo recibe cambios desde development cuando se valida que el sistema es funcional y estable.

**development**

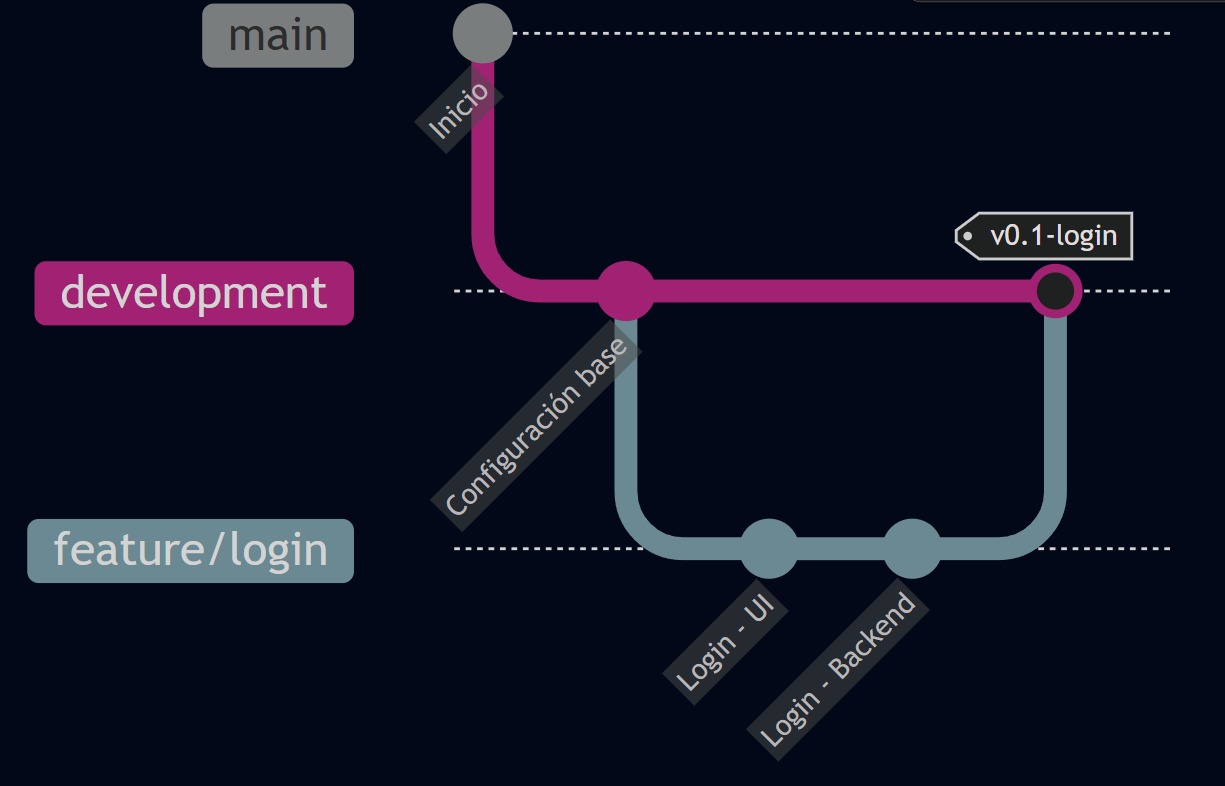
* Rama destinada al desarrollo activo.
* Aquí se integran todas las funcionalidades creadas en ramas feature/\*.
* Sirve como punto intermedio antes de pasar a producción.

**feature/nombre**

* Ramas creadas desde development para implementar funcionalidades específicas.
* Ejemplo: feature/login, feature/dashboard.
* Al terminar el desarrollo, se integran nuevamente en development.

Actualmente no se tiene ninguna rama feature asignada, pero la estructura ya está lista para cuando se creen nuevas funcionalidades.

**Flujo de trabajo de las ramas:**



# 3. Uso de Herramientas de Versionamiento

En este proyecto se utilizó Git como sistema de control de versiones distribuido, junto con la plataforma GitHub para la gestión en línea del repositorio. Git permite llevar un control detallado del historial de cambios, mientras que GitHub facilita la colaboración en equipo, la creación de ramas y la integración de nuevas funcionalidades de manera organizada.

## 3.1. Creación del Repositorio y Gestión de Ramas

El repositorio fue creado directamente en la plataforma GitHub y posteriormente clonado en el entorno local de desarrollo. La estructura de ramas definida fue la siguiente:

* main: rama principal que contiene la versión estable del proyecto.
* development: rama donde se integran los cambios y nuevas características antes de pasarlos a la versión estable.
* feature/[nombre-de-la-funcionalidad]: esta rama aun no esta creada debido a que no hay funciones nuevas agregadas al proyecto, de momento.

El flujo de trabajo establecido consiste en desarrollar nuevas funcionalidades en ramas feature, integrarlas primero en development y, una vez probadas y estables, realizar la integración final en la rama main.

Nota: Durante la gestión del repositorio se detectó que, al subir y descargar el proyecto completo desde GitHub, este en ocasiones se corrompe. Para solventar esta situación, se decidió manejar diferentes versiones del proyecto comprimido a fin de garantizar siempre una copia funcional.

# 4. Historial de Cambios (Commits)

**- Subida inicial del proyecto de Visual Studio**

* **Rama:** main
* **Cambios implementados:** Se realizó la primera carga del proyecto generado en Visual Studio, incluyendo la estructura base y los archivos iniciales necesarios para ejecutar la aplicación.
* **Contribución:** Este commit marca el inicio formal del proyecto en el repositorio, permitiendo que el equipo tenga un punto de partida común.

**- Migraciones hechas**

* **Rama:** main
* **Cambios implementados:** Se añadieron o actualizaron las migraciones del proyecto, lo cual indica modificaciones en la base de datos para reflejar nuevas entidades o cambios en las existentes.
* **Contribución:** Garantiza que la base de datos esté alineada con el modelo del proyecto, lo que facilita la correcta persistencia y manejo de la información.

**- Merge pull request #1 from development**

* **Rama:** main (merge desde development)
* **Cambios implementados:** Se integraron los avances desarrollados en la rama de development hacia la rama principal.
* **Contribución:** Centraliza las mejoras y nuevas funcionalidades en la rama principal, asegurando que el proyecto se mantenga actualizado con el trabajo realizado de forma colaborativa.

**- Subiendo cambios en el proyecto**

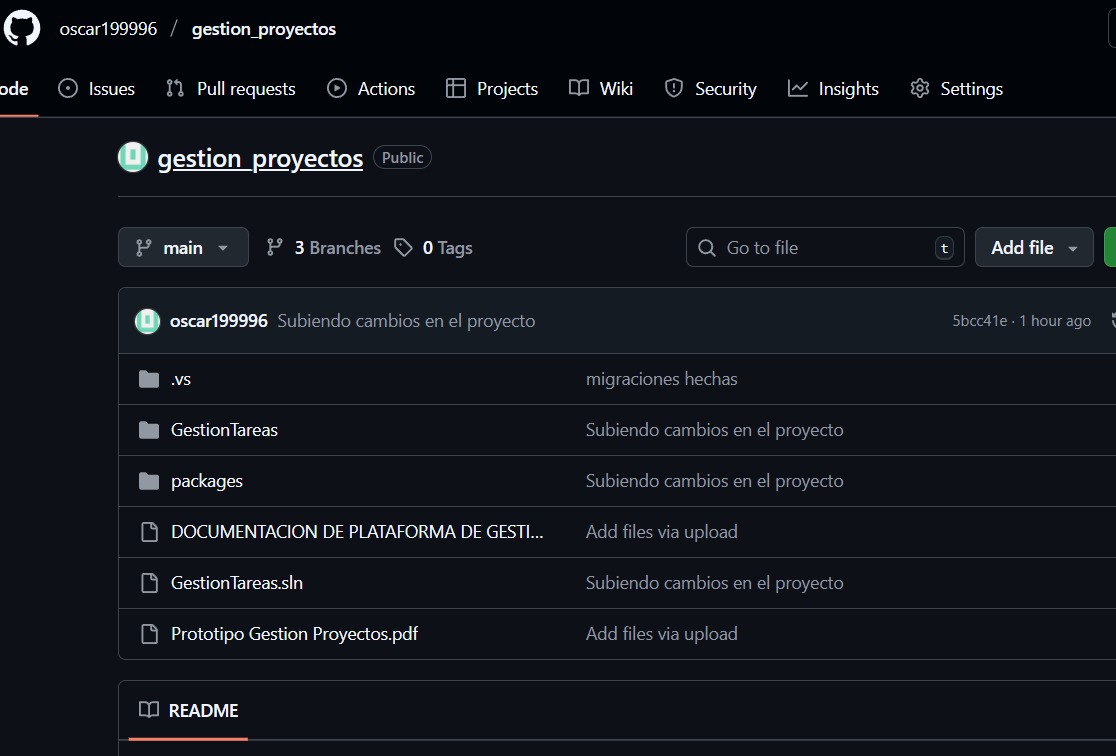
* **Rama:** main
* **Cambios implementados:** Se realizaron ajustes en diferentes archivos del proyecto, posiblemente corrigiendo errores y mejorando la organización del código.
* **Contribución:** Este commit es importante porque muestra la evolución continua del sistema, aplicando correcciones necesarias para mantener la estabilidad del proyecto.

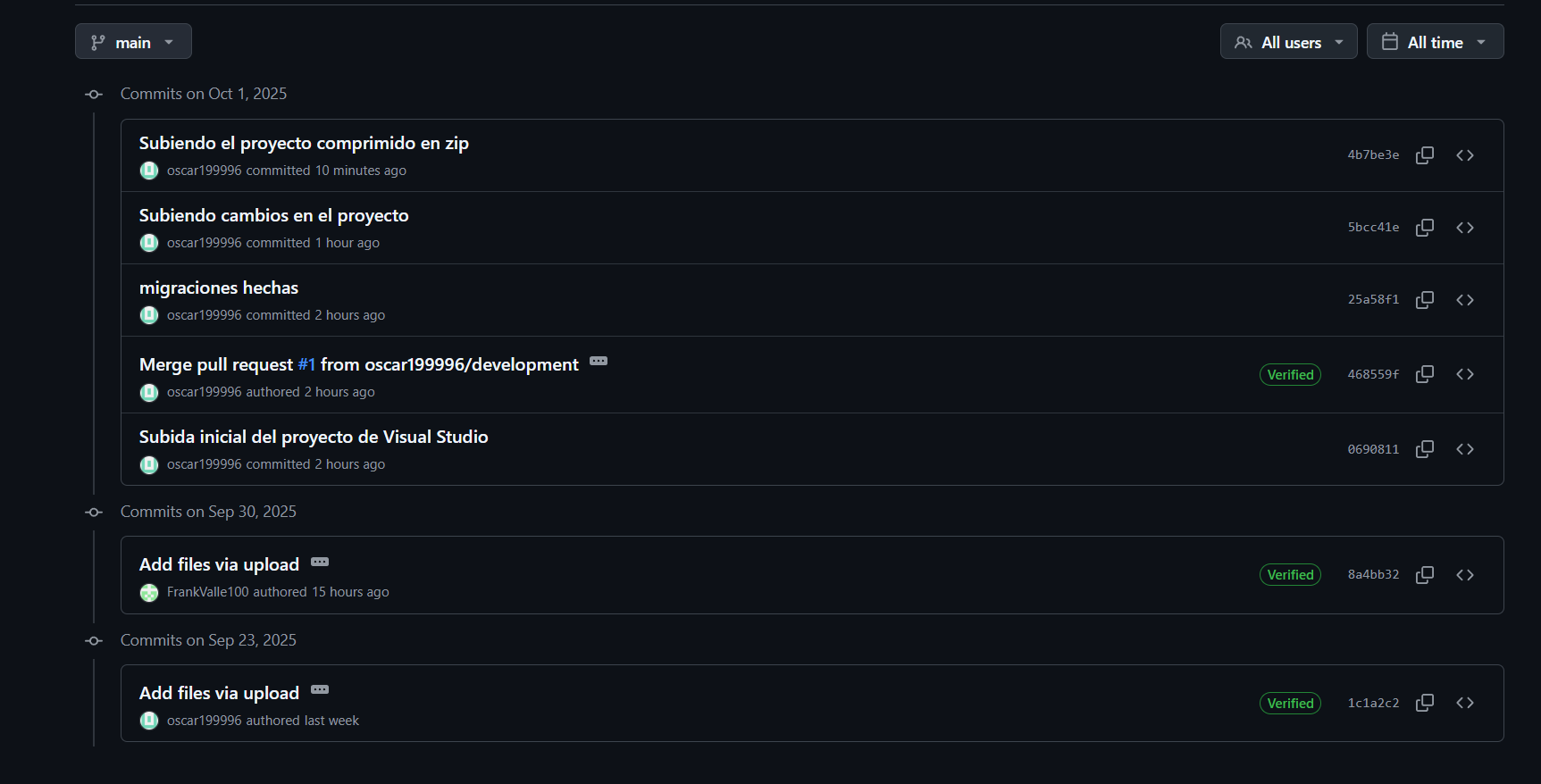
**- Subiendo el proyecto comprimido en zip**

* **Rama:** main
* **Cambios implementados:** Se añadió al repositorio una versión comprimida del proyecto en formato ZIP.
* **Contribución:** Este cambio permite contar con un respaldo del proyecto completo, facilitando su descarga y distribución de manera rápida.

**- Add files via upload (documentación y prototipo)**

* **Rama:** main
* **Cambios implementados:** Se cargaron archivos adicionales como la documentación de la plataforma y un prototipo en PDF.
* **Contribución:** Aporta valor al proyecto al incluir material de soporte y diseño, lo que ayuda tanto a desarrolladores como a futuros usuarios a comprender la finalidad y estructura del sistema.





# 5. Archivo CHANGELOG.md

Con el fin de documentar de manera estructurada el historial de cambios del proyecto, se creó el archivo CHANGELOG.md en el repositorio. Este archivo registra las versiones del proyecto, las fechas de los cambios y una breve descripción de cada modificación realizada, clasificándolas en categorías de acuerdo con su naturaleza.

# Changelog

Todas las modificaciones importantes del proyecto se documentan en este archivo.

## [1.0.0] - 2025-09-15

### Added

- Se implementó la estructura base del proyecto.

## [1.0.1] - 2025-09-16

### Added

- Se configuró la base de datos inicial.

## [1.1.0] - 2025-09-29

### Added

- Se desarrolló el módulo de autenticación de usuarios.

### Fixed

- Se corrigieron errores de visualización en la interfaz de usuario.

### Changed

- Se reorganizó la estructura de carpetas del proyecto para mejorar la mantenibilidad.

# 6. Conclusiones

El uso de herramientas de control de versiones, como Git, ha sido fundamental para el desarrollo colaborativo del proyecto. Estas han permitido que el equipo trabaje de manera simultánea y organizada, manteniendo un flujo de trabajo claro mediante el uso de ramas como main, development y feature.

Además, el control de versiones ha garantizado orden y trazabilidad en cada cambio realizado, facilitando la recuperación del código en caso de errores y asegurando que siempre exista una versión estable del sistema.

# 7. Enlace a GitHub

<https://github.com/oscar199996/gestion_proyectos.git>