

**FACULTAD DE INGENIERIA DE SISTEMAS**

**CAMPUS LIMA SUR**

**PROYECTO DE HERRAMIENTAS DE DESARROLLO**

**"Desarrollo Colaborativo con Control de Versiones"**

**Integrantes:**

DAVILA GARCIA, JAIME JHAIR

MALPARTIDA CASANOVA, ANNIE ARIADNA

FALCON TRILLO, VERONIKA CRISTINA

**Docente:**

Burga Montañez, Roberto Eduardo

**Sección:**

44696

**LIMA -PERÚ 2024**

**Sistema De Control de Versiones Para Proyectos Web**

**CAPITULO I: PLANEACION**

**Definición De Alcance**

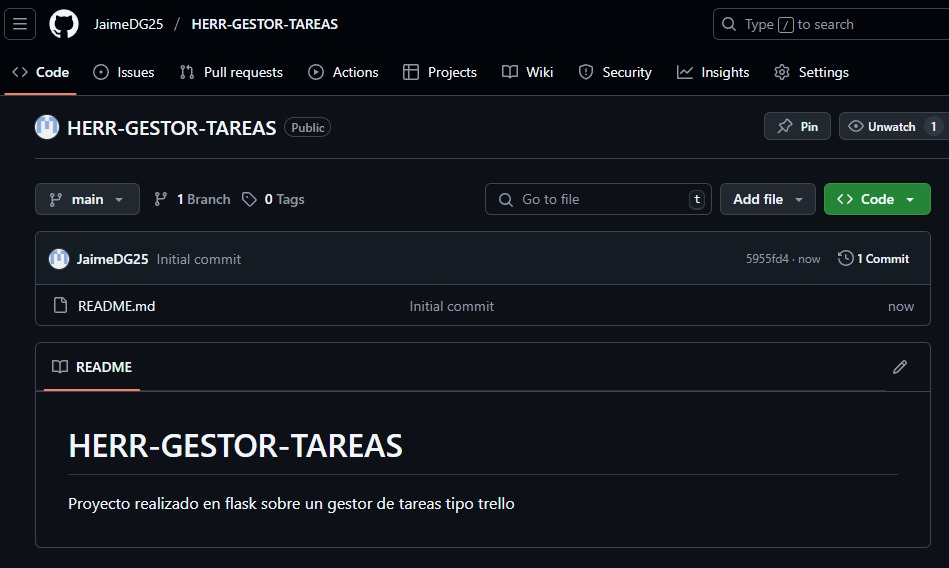
Este proyecto consiste en la implementación del uso de un sistema de control de versiones Git dentro de un equipo de desarrollo de software que permitirá llevar un control de los cambios ejecutados dentro de un IDE y facilitar la colaboración de múltiples integrantes de equipo y mantener un historial seguro de las versiones del proyecto.

**Repartición De Roles**

|  |  |  |
| --- | --- | --- |
| Nombre del integrante | Actividad asignada | Rama asignada |
| Annie Ariadna Malpartida Casanova | Interfaz de usuario y estilos CSS | feature/ui |
| Jaime Jhair Davila Garcia | Lógica de negocio en backend | feature/backend |
| Jaime Jhair Davila Garcia | Conexión con base de datos | feature/database |
| Veronika Cristina Falcon Trillo | Pruebas unitarias y validación | feature/testing |
| Todos | Documentación y revisión cruzada | docs, review/\* |

**Creación Del Repositorio GitHub**

Utilizaremos un repositorio remoto para guardar las versiones de nuestro proyecto de una forma mas segura y llevar un mejor control de las versiones.



**Figura 1**

**CAPITULO II: DESARROLLO INICIAL**

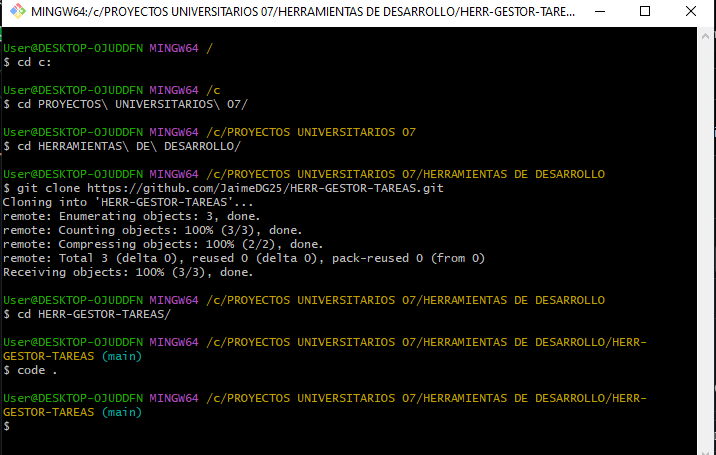
**Configuración De Git**

Antes de iniciar nuestro proyecto debemos realizar las configuraciones globales de los usuarios en un repositorio local.

* Configuración del nombre: git config –global user.name “Algún nombre”
* Configuración del email: git config –global user.email “Algún correo”
* Configuración del editor de texto: git config --global core.editor "code --wait"
* Mostrar configuraciones: git config –list

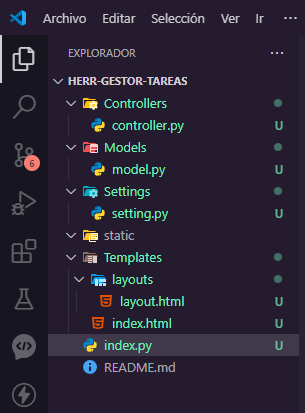
**Implementación De La Rama Principal (Master O Main)**

Para comenzar a trabajar con el proyecto alojado en GitHub, es necesario clonar el repositorio remoto hacia nuestro equipo local. Este proceso se realiza mediante el comando: git clone <URL-del-repositorio>. Este comando crea una copia exacta del repositorio remoto en nuestra máquina local, incluyendo todo el historial de cambios, ramas, archivos y configuraciones.

****

**Figura 2**

**Estructura del Proyecto en Visual Studio Code**



**Figura 3**

Nuestro proyecto maneja la arquitectura MVC tanto para el manejo e interacción con los datos mediante el lenguaje Python con su framework Flask.

* **Controllers:** En esta carpeta Archivo donde defines funciones que procesan las solicitudes del frontend, por ejemplo, listar tareas, crear nuevas, eliminar, etc.
* **Models:** Archivo donde defines las clases que representan las tablas (como Usuario, Tarea, Estado, etc.). Aquí también se importa y configura el objeto db de SQLAlchemy.
* **Settings:** Aquí normalmente defines funciones como get\_connection() o get\_sqlalchemy\_uri() que se usan para configurar la base de datos en Flask.
* **Static:** Carpeta destinada a los recursos estáticos como imágenes, CSS, JavaScript, íconos, etc.
* **Templates:** Contiene las plantillas HTML que se renderizan desde Flask usando render\_template.
* **Index.py:** Este es el archivo principal del proyecto, el punto de entrada de la aplicación Flask.

**CAPITULO III: DESARROLLO COLABORATIVO**

**CAPITULO IV: INTEGRACION**

**CAPITULO II: OPTIMIZACION**

**CAPITULO II: PODA DE RAMAS**