**Distribución Inicial de Roles y Responsabilidades**

Desde el inicio del proyecto se estableció una clara división de tareas entre los integrantes, lo que permitió enfocar los esfuerzos en áreas específicas y garantizar la calidad en cada una de ellas. En concreto, se asignaron los siguientes roles:

* **Control:** Encargado de gestionar la lógica y los procesos centrales del sistema. Esta función fue responsable de coordinar cómo interactúan los distintos componentes y asegurar que la aplicación respondiera de manera adecuada a las acciones del usuario y a la ejecución de procesos internos.
* **Modelo:** Responsable de la creación y definición de las clases, así como de sus atributos. Esta área se enfocó en estructurar los datos y representar la información de manera coherente y lógica, lo que es fundamental para la integridad y el correcto funcionamiento del sistema.
* **Vista:** Dedicada al desarrollo de la interfaz de usuario, garantizando que la experiencia visual resultara coherente y funcional. La labor del equipo de vista fue clave para asegurar que la presentación del producto fuera atractiva y amigable, facilitando la interacción del usuario.

**Uso de Ramas y Control de Versiones**

Con la finalidad de mantener el código organizado y estable, se adoptó la siguiente estrategia respecto al control de versiones:

1. **Creación de Ramas Específicas:**  
   En lugar de trabajar directamente en la rama principal (main), cada miembro creó una rama específica para desarrollar sus respectivas funcionalidades o realizar modificaciones en su área. Este método, ejecutado a través del comando:

* git switch -c nombre-de-la-rama

permitió mantener un entorno aislado para cada tarea, facilitando la identificación y seguimiento de los cambios realizados en cada componente del sistema.

1. **Commits Frecuentes:**  
   Durante el desarrollo, se realizaron commits de manera constante. Esta práctica tuvo dos objetivos claros:

* **Documentar el avance:** Cada commit sirvió como un registro detallado del progreso, permitiendo conocer la evolución del código.
* **Describir las modificaciones:** Se incluyeron descripciones precisas de los cambios efectuados, lo que facilitó el seguimiento del desarrollo y la detección de posibles errores o discrepancias.

1. **Integración Controlada y Revisión de Código:**  
   Antes de incorporar cualquier cambio a la rama principal, se llevaron a cabo pruebas y, en numerosos casos, se realizaron revisiones de código (code reviews) entre compañeros. Este proceso colaborativo aseguró que:

* Cada nueva función se implementara correctamente y estuviera libre de errores.
* Se mantuvieran los estándares del proyecto en todo momento.

1. **Fusión a la Rama Principal:**  
   La rama principal (main) se mantuvo en un estado estable durante todo el desarrollo y nunca fue modificada de forma directa. Una vez comprobada la compatibilidad y correcto funcionamiento de los cambios en una rama específica, se procedió a fusionarla con main utilizando los siguientes comandos:

* git checkout main
* git merge nombre-de-la-rama

Este flujo de trabajo permitió integrar gradualmente las nuevas funcionalidades y correcciones, manteniendo la integridad del código base, cuando el proyecto ya se había acabado subimos la versión de la main al github;