Creación y mantenimiento del repositorio y manejo de issues:

Repositorio Github:

Para el correcto desarrollo del proyecto y manejo del versionado con técnicas de DevOps, se creó un repositorio en Github. En este repositorio, en primera instancia se agregaron en la rama Main:

- Carpeta Backend: Con el código fuente del Backend de la aplicación en C# y .Net, también incluye pruebas unitarias.
- Carpeta Frontend: Con el código fuente del Frontend de la aplicación en el framework Angular.
- Carpeta Base de Datos: Con los archivos .bak y .sql con la definición de la BD y sus datos de prueba
- Carpeta documentación: Con toda la documentación generada para el obligatorio.

Versionado:

Para el versionado el equipo debatió entre la utilización de gitflow y la metodología trunk based. Si bien los miembros del equipo ya habíamos utilizado gitflow y no teníamos experiencia con la metodología trunk based, nos decantamos por esta ultima ya que entendimos que se complementa mejor con KANBAN, el contexto utilizado para el proceso de ingeniería.

Trunk based se enfoca en mantener una única rama principal lo que simplifica la gestión del flujo de trabajo y hace que sea más fácil visualizar y controlar el trabajo en curso. El equipo entendió que esto es fundamental para el tipo de proyecto a realizar.

Sin embargo, si bien nos basaremos en trunk based para el versionado, lo haremos con algunos cambios, si bien respetamos no tener muchas ramas simultaneas abiertas y también el realizar commits rapidos a la rama principal, decidimos tener la rama develop abierta además de main para tener más seguridad y protección contra errores en el versionado.

El equipo trabajará en la rama develop pero hará merges constantes a main de manera similar a lo estipulado en la metodología trunk based.

Issues:

El equipo hará uso de la herramienta provista por github para el manejo de issues, para esto se definen tags para categorizar los issues y un modelo para especificarlos.

Tags:

Para tipos de Issues:

Bug: El issue representa un error en el código

AC: El issue representa una porción de código que funciona, pero no está bien escrita.

Missing Requirements: Requerimientos no implementados.

Para clasificar issues de tipo bug dependiendo de que tan grave es para el sistma:

Severity:minor: Prácticamente no afecta el funcionamiento del sistema.

Severity:mild: Afecta el funcionamiento del sistema levemente.

Severity:mayor: Afecta mucho al funcionamiento del sistema.

Severity:critical Imposibilita el funcionamiento correcto del sistema .

Para clasificar issues de tipo bug dependiendo la prioridad para solucionarlos:

Priority:low: El bug no tiene prioridad sobre ningún otro.

Priority:médium: El bug será atendido cuando sea conveniente.

Priority:high: El bug debe ser atendido lo antes posible.

Prioriy:inmediate: El bug debe ser atendido inmediatamente.

Para clasificar issues de tipo AC dependiendo de su gravedad:

Seriousness:low: Problemas con el código poco graves.

Seriousness:high Problemas con el código graves.

Para invalidar un Issue:

Invalid: El issue no es correcto y no se debe tener en cuenta