**Entregable Tarea 1**

**INF331 - Pruebas de Software**

Icono

Descripción generada automáticamente

Integrantes: Carlos Vega Muñoz | Diego Veas Bastías

Profesor Oscar Reyes.

Segundo Semestre 2024, Universidad Federico Santa María.

Repositorio: <https://github.com/MyTask-PruebasDeSoftware331/.Task1>

**1) Requerimiento**

"Un cliente nos ha solicitado una aplicación para gestionar sus tareas ":

Esta aplicación funcionará a través de la línea de comandos, en donde se nos indicará cuando una entrada u operación es inválida, a través de mensajes de error claros.

* Gestión de Tareas:

Desarrollar un sistema que permita a los usuarios crear, consultar, actualizar y eliminar tareas a través de la línea de comandos. Cada tarea debe tener un título, una descripción, una fecha de vencimiento, y una etiqueta que facilite la organización.

Los títulos y descripciones de las tareas tendrán una longitud máxima, 100 caracteres en título y 500 en descripción.

Las etiquetas para seleccionar serán fijas, las cuales son: Prioridad alta, media o baja; académica, trabajo, personal. Se podrán utilizar una combinación de 2 etiquetas (prioridad y tipo)

Las fechas de vencimiento tendrán el formato AAAA-MM-DD XX:YY:ZZ (formato TimeStamp)

Las tareas iniciarán con un estado de “Pendiente” y después podrán ser modificadas a otros estados como “Atrasada”, “En progreso”, “Completada” o “Archivada” (El profesor mencionó un estado “abierta” en el foro, pero como no nos quedó claro su uso, haremos el supuesto de que “abierta” y “pendiente” son el mismo estado. El estado “atrasada” se cambiará de nombre a “vencido” en la entrega)

* Filtrado y Búsqueda:

Incluir funcionalidades para que los usuarios puedan filtrar y buscar tareas las tareas que han sido creadas, en función de varios criterios: como la fecha de vencimiento, etiquetas, título o estado de la tarea.

Al buscar por fechas, se podrá ingresar un rango de fechas para buscar.

El resultado de las búsquedas será toda la información asociada a la tarea que cumpla con alguno de los parámetros de búsqueda.

* Gestión de Estados de Tareas:

Implementar un sistema de gestión de estados para las tareas que permita a los usuarios marcar una tarea como "en progreso" o "completada". Las tareas que no se completan antes de su fecha de vencimiento, quedarán con un estado de “vencido”

Las tareas completadas pueden ser archivadas y consultadas posteriormente (quedarían con el estado “archivada”), asimismo, estas pueden ser eliminadas en cualquier momento. (no es necesario archivarlas una vez que se completan)

Los estados de las tareas tendrán un flujo normal de "pendiente" --> "en progreso" --> "completado" --> "archivado"

* Autenticación:

Proteger el acceso al sistema con un mecanismo de autenticación mediante un nombre de usuario y contraseña. Cada usuario tendrá su propia lista de tareas. Los nombres de usuarios no se podrán repetir.

Se usará un sistema de codificación de claves a través de Hashing, con una base de datos local.

El sistema estará diseñado para uso personal o equipos pequeños, no para manejar grandes volúmenes de tareas o usuarios.

**2) Tarea**

1. ¿Cómo especificarías mejor el requerimiento? (Validación: ¿Estamos resolviendo el problema correcto?)

Por la descripción del requerimiento, sabemos que el problema principal que queremos resolver es gestionar tareas a través de una aplicación. Para especificar mejor el requerimiento, hicimos aclaraciones con respecto a lo que se espera que hiciera la aplicación. Hicimos preguntas por el foro relacionadas a los requisitos del problema a tratar (gestión de tareas, filtrado y búsqueda, gestión de estados de tareas y autenticación), pues el enunciado las plantea desde una perspectiva muy general. Nuestras preguntas y supuestos se enfocaron en asegurarse que el programa hiciera lo que se esperaba, teniendo en cuenta las posibles “restricciones” que pueden existir al crear una aplicación.

1. ¿Cómo asegurarías que el programa cumpla el requerimiento? (Verificación: ¿Estamos resolviendo correctamente el problema?)

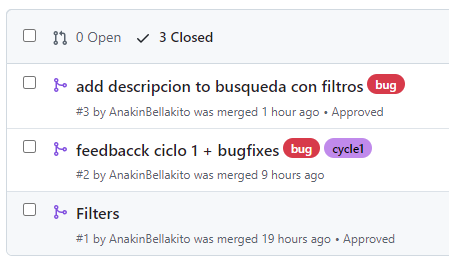
Para asegurarnos que el programa cumpla el requerimiento, separamos en distintas historias de usuario cada uno de los requisitos del trabajo, con tal de dejar una separación explícita entre estas. Esto nos permite saber que cada parte a trabajar se está haciendo como se tiene planeado que funcionen. Si alguna parte de los requisitos no funciona correctamente, esta podrá ser arreglada sin afectar a las otras partes.

1. Organización, explicar cómo se organizó el proyecto y el flujo de trabajo de éste.

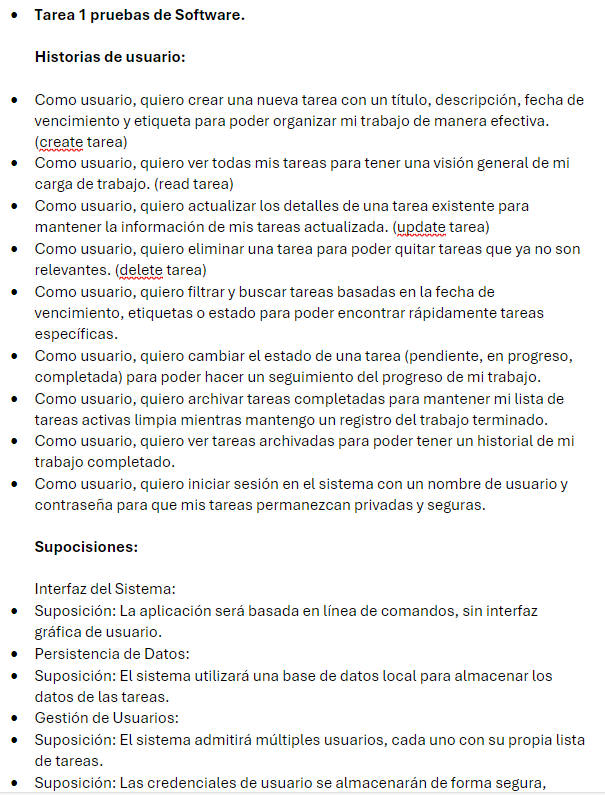
Nuestro flujo de trabajo inició con una conversación entre los integrantes del grupo para entender qué es lo que se esperaba hacer y cómo hacerlo, asimismo definimos qué aspectos del requerimiento necesitaban mayor explicación, con tal de poder preguntar por el foro de aula o hacer los supuestos correspondientes. A continuación, nos dividimos la carga de trabajo, uno de nosotros se enfocó en el desarrollo del programa mientras que el otro trabajó en los aspectos formales de la entrega y las pruebas a realizar en el código. En otras palabras Carlos es el desarrollador y Diego hizo la labor de QA.

1. Incluir evidencia de flujo de trabajo y configuraciones realizadas (Imágenes de pantalla).

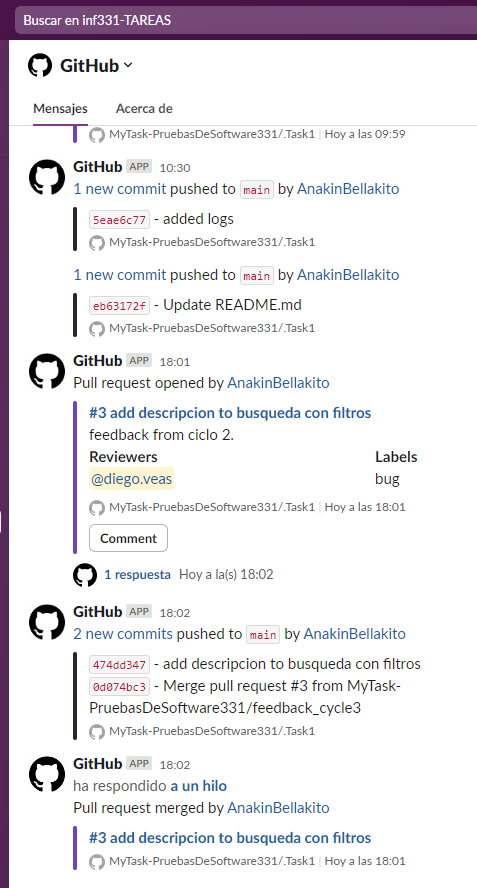
* Uso de Pull Requests para confirmar los cambios a efectuar.

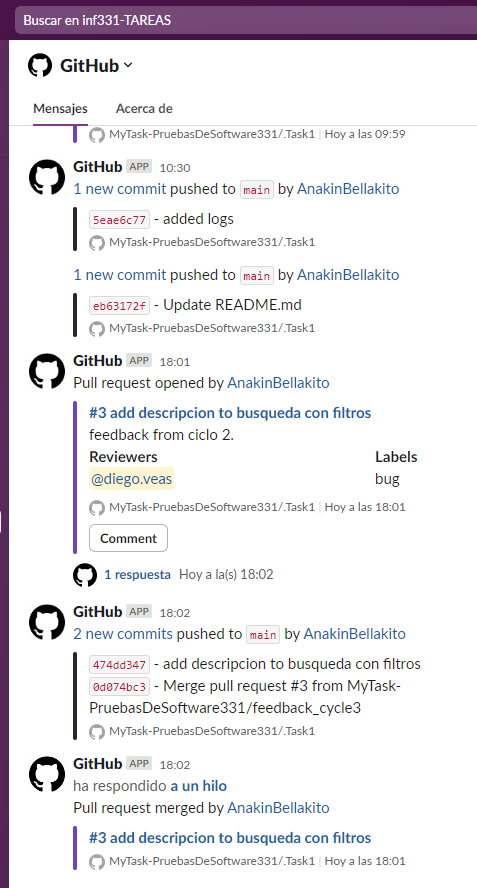


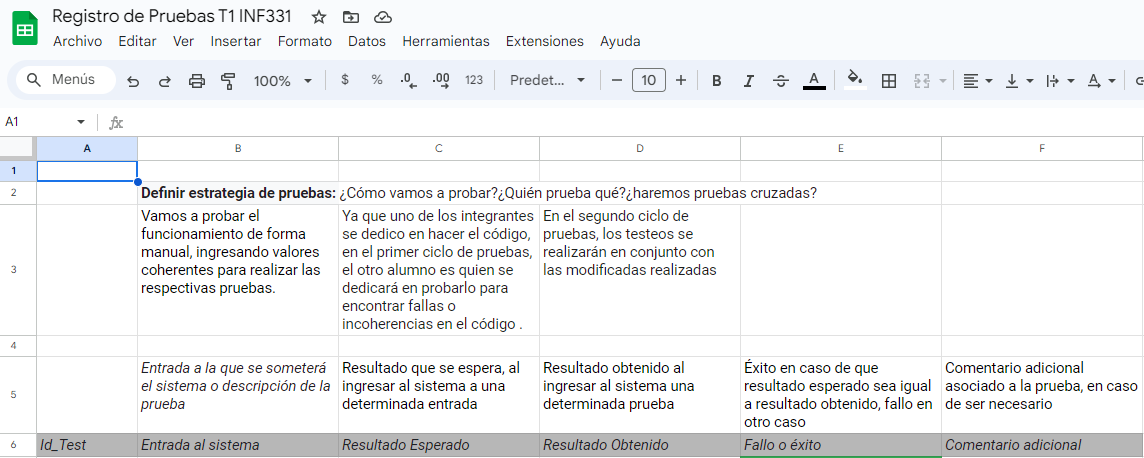
* Documento inicial con historias de usuario y suposiciones a plantear/preguntar.



* Uso de un espacio de trabajo en Slack con integración a Github para mantener un seguimiento de los cambios.





* Registro de los ciclos de pruebas realizados.

1. Problemas encontrados y cómo se solucionaron.

Algunos de los problemas que encontramos, fueron principalmente relacionados en aspectos necesarios para cumplir con lo definido en la tarea, más que complicaciones de código.

Al inicio hubo que descargar y aprender a usar Sqlite para poder trabajar con una base de datos personal, lo cual implicó un pequeño nivel de dificultad al implementarlo en Windows. Esto se solucionó buscando tutoriales de instalación.

Durante el proceso de desarrollo no hubo mayores problemas, pero al momento de realizar los ciclos de pruebas pudimos encontrar ciertos detalles que no habían sido correctamente filtrados inicialmente (como por ejemplo que la información no se mostraba como se tenía planificado, o que los usuarios podían ser creados con los mismos datos). Esto se soluciona una vez que estos detalles fueron detectados, a través de ciertos cambios en los atributos de las funciones o agregando restricciones correspondientes.