

# Sprint Backlog

## 1. Poder crear tareas compuestas de otras subtareas:

En esta tarea vamos a añadir la posibilidad de que una tarea esté compuesta de otras tareas, a las que llamaremos subtareas. Pese a que en este documento haremos referencia a ellas como *subtareas*, a nivel de programación y de base de datos, seguirán siendo tareas. Vamos a explicar a continuación cómo haremos para implementar esta nueva característica.

### 1.1 Al modelo Tarea le **añadiremos dos atributos**:

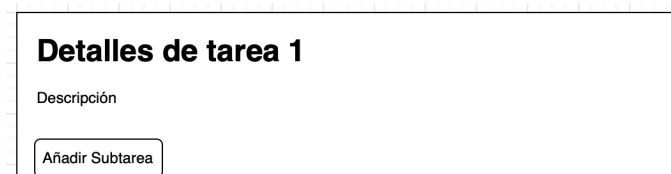
1. Un array de tareas: En este array se guardan las tareas hijas (subtareas), para saber que tareas tiene asociada la tarea padre y poder listarlas.
2. Un atributo Integer: Para identificar el id de la tarea padre. En el caso de que la tarea no tenga padre, es decir, que no sea una subtask, este campo será null, en el caso de ser una subtask este campo tendrá el id de la tarea padre.

### 1.2 **Añadir una subtask a una tarea**:

Es necesario que ambas tareas esten creadas previamente, la que llamaremos subtask simplemente tiene que estar creada como una tarea.

Una vez tenemos creadas tanto la tarea padre como la tarea que será hija, vamos a la vista de detalles de la tarea padre. En esta vista veremos un botón que será Añadir subtask. Si pulsamos dicho botón veremos un desplegable con todas las tareas existentes, cada tarea de este desplegable tendrá un checkbox, en el que si queremos que se convierta en una subtask, seleccionaremos ese checkbox. Lo veremos mejor con un boceto.

En primer lugar vemos el botón para añadir una subtask.



The mockup shows a rectangular box representing a task detail view. At the top left, it is titled 'Detalles de tarea 1'. Below the title is a text input field labeled 'Descripción'. At the bottom left of the box is a button labeled 'Añadir Subtask'.

En segundo lugar vemos que al pulsar el botón tenemos la lista de tareas con sus respectivos checkbox para asignar subtareas.

### Detalles de tarea 1

Descripción

Añadir Subtarea

☒ Tarea 1

☐ Tarea 2

☐ Tarea 3

☐ Tarea 4

### 1.3 Ver listado de subtareas:

A parte de añadir una subtarea, en una tarea podremos **ver en un listado las tareas hijas** que tiene cada tarea, este listado será igual que el que se ve cuando se listan las tareas principales. A continuación vemos un ejemplo de ello:

### Detalles de tarea 1

Descripción

Añadir Subtarea

ID	Descripción	Acción
1	Descripción descripción descripción	<div><div>Editar</div><div>Eliminar</div></div>

### Condiciones de satisfacción

Para esta tarea, consideraremos que cumple con las condiciones de satisfacción si al ver los detalles de una tarea se pueda añadir subtasks y que una vez añadidas, en esta misma vista, se puedan ver las subtasks que tiene asignadas.

## 2. Mejorar interfaz para mostrar equipos a los que pertenece un usuario

En esta tarea se añadirá una opción de filtro para la lista de equipos, de manera que un usuario pueda ver todos los equipos existentes o solamente aquellos a los que pertenezca.

LISTADO EQUIPOS

☐ Mis Equipos  
☐ Todos los equipos

Id	Nombre	Acciones
----	--------	----------

### Condiciones de satisfacción

Por defecto, cuando se acceda al listado, aparecerán todos los equipos existentes. En caso de marcar la casilla “Mis equipos”, solo aparecerán aquellos a los que pertenezca el usuario registrado. En caso de marcar la casilla “Todos los equipos” aparecerán todos los equipos existentes.

### 3. Añadir un estado a las tareas, por ejemplo: todo, in progress, done

En esta tarea se añadirá un estado a las tareas, de manera que podamos saber en cual se encuentra. Cada tarea podrá tener 3 estados: pendiente, en progreso y acabada. Por defecto, cada vez que se cree una tarea nueva se le asignará el estado ‘pendiente’. Una vez creada, el usuario podrá modificar la tarea, poniendo el estado que considere.

El estado de las tareas aparecerá en la lista de las mismas, así como si abrimos sus detalles.

LISTADO TAREAS

id	Tarea	Prioridad	Estado	Acciones
1	Tarea 1	Alta	Pendiente	<input type="checkbox"/> <input type="checkbox"/>

EDITAR TAREA

Titulo:

Estado:

Pendiente	✓
En progreso	
Acabada	

## Condiciones de satisfacción

Las tareas que se creen tendrán como estado 'pendiente'. Una vez creada, se podrá cambiar su estado entre 'en progreso' y 'acabada'.

## 4. Asignar prioridad a las tareas

En esta funcionalidad se implementará la posibilidad de asignar un nivel de prioridad a cada tarea. Este nivel de prioridad será visible tanto en el momento de crear una tarea como en el listado de tareas.

### 1. Modificación del modelo Tarea

Para soportar esta funcionalidad, se añadirá un atributo al modelo **Tarea**:

- **Atributo de prioridad (String):** Este atributo almacenará el nivel de prioridad de la tarea. Sus valores posibles serán:
  - Alta
  - Media
  - Baja

La prioridad de una tarea será asignada en el momento de su creación. Este atributo será obligatorio y no podrá ser modificado posteriormente.

### 2. Asignar prioridad al crear una tarea

Al crear una nueva tarea, el usuario deberá seleccionar un nivel de prioridad utilizando un desplegable que contendrá las opciones "Alta", "Media" y "Baja". La selección de prioridad será obligatoria para completar el proceso de creación de la tarea.

En el mockup a continuación, podemos ver cómo se integra este desplegable en el formulario de creación de tareas:

**Nueva tarea para el usuario Ejemplo**

Titulo de la tarea

Descripción

Prioridad  

Alta  
Media  
Baja

### 3. Visualizar la prioridad en el listado de tareas

Una vez creada una tarea, su nivel de prioridad será visible en el listado general de tareas. En este listado, cada fila correspondiente a una tarea mostrará la prioridad junto a otros datos relevantes (como el ID, la descripción, etc.).

En el siguiente mockup, se observa un ejemplo del listado de tareas con su prioridad.

Listado de tareas de usuario				
Id	Tarea	Descripción	Prioridad	Acciones
1	Tarea 1	Añadir estado a las tareas.	Alta.	Editar <b>Borrar</b> Ver detalles

### 4. Restricciones de la funcionalidad

- **Prioridad inmutable:** Una vez asignada la prioridad al crear la tarea, no podrá ser modificada posteriormente.
- **Visualización consistente:** La prioridad será visible en todas las vistas donde se listan las tareas.

### Condiciones de satisfacción

Esta funcionalidad será considerada completa si cumple con los siguientes criterios:

1. Al crear una tarea, el usuario puede seleccionar un nivel de prioridad mediante un desplegable.
2. La prioridad asignada a cada tarea es visible en el listado general de tareas.
3. La prioridad de una tarea no puede ser modificada una vez creada.

### 5. Editar Perfil

Esta funcionalidad permite a los usuarios visualizar, editar algunos de sus datos personales, y cambiar su contraseña desde una sección dedicada dentro de la aplicación. Además, se han introducido mejoras importantes en la seguridad del manejo de contraseñas en el backend.

## 1. Visualizar datos

En la vista de cuenta los usuarios podrán visualizar la información de su perfil.

**Datos de la cuenta**

**Nombre:** Ejemplo Nombre  
**Email:** ejemplo@ua.com  
**Fecha de Nacimiento:** 01-16-2002

Editar

## 2. Datos que se pueden editar

En el formulario de edición de datos, los usuarios podrán modificar los siguientes campos:

- **Nombre:** Se puede actualizar el nombre completo del usuario.
- **Email:** Se permite cambiar la dirección de correo electrónico.
- **Fecha de nacimiento:** Se puede ajustar la fecha de nacimiento del usuario.

En el mockup siguiente se puede observar el diseño del formulario de edición de datos:

**Datos de la cuenta**

Editar datos de la cuenta

Nombre  
Ejemplo Nombre

Email  
Ejemplo Nombre

Fecha de nacimiento  
01/16/2002

Guardar Cambios

## 3. Cambiar contraseña

En la misma vista, se incluye un apartado para permitir al usuario cambiar su contraseña. Para garantizar que este proceso sea seguro, el formulario solicita:

- a. **Contraseña actual:** Verificación para asegurarse de que el usuario tiene acceso a su cuenta.
- b. **Nueva contraseña:** Contraseña que el usuario desea establecer.
- c. **Confirmación de la nueva contraseña:** Para evitar errores de escritura en la nueva contraseña.

El diagrama muestra una interfaz de usuario para cambiar la contraseña. En la parte superior, hay un botón rectangular etiquetado "Guardar Cambios". Debajo de este, hay un contenedor principal con el título "Cambiar contraseña". Dentro de este contenedor, hay tres campos de entrada de texto, cada uno precedido por un texto descriptivo: "Contraseña actual:", "Nueva contraseña:" y "Confirmar nueva contraseña:". En la parte inferior del contenedor, hay dos botones: "Cambiar Contraseña" y "Cancelar".

#### 4. Implementación de cifrado de contraseñas

Durante el diseño de esta funcionalidad, se identificó una vulnerabilidad en el manejo de contraseñas, ya que estas no estaban cifradas ni en el proceso de registro ni en el de inicio de sesión. Para solucionarlo:

- Se implementará **cifrado seguro** de contraseñas en el backend.
- Se actualizará el registro para almacenar las contraseñas de forma cifrada.
- El proceso de inicio de sesión será modificado para validar las contraseñas cifradas.

#### 5. Flujo de interacción

- a. El usuario accede a la sección "Datos de la cuenta".
- b. Visualiza sus datos actuales en un formato de solo lectura.
- c. Al hacer clic en el botón **Editar**, se habilita un formulario para modificar los datos permitidos.
- d. Si desea cambiar la contraseña, utiliza el formulario específico para ello.
- e. Tras guardar los cambios, se actualizan los datos en la base de datos y el usuario recibe una notificación de éxito.

#### Condiciones de satisfacción

Esta funcionalidad será considerada completa si cumple con los siguientes criterios:

1. Los usuarios pueden visualizar y modificar su nombre, email, y fecha de nacimiento.

2. Los usuarios pueden cambiar su contraseña mediante el formulario correspondiente.
3. Las contraseñas están cifradas de manera segura en el sistema.
4. Los cambios realizados son reflejados correctamente en el sistema y son persistentes.

## 6. Calendario de tareas

En esta tarea se añadirá un calendario mensual, con opciones para elegir tanto el mes como el año. En la fecha seleccionada se mostrará el calendario con las tareas de cada día. Se podrá seleccionar una tarea para ver su contenido. Además incluirá una lista con las tareas de dicho mes.

CALENDARIO

MES AÑO BUSCAR

L	M	X	J	V	S	D

LISTA DE TAREAS

### Condiciones de satisfacción

El usuario debe poder ver el calendario en las fechas que quiera, así como las tareas del mes elegido y acceder a los detalles de las mismas.

## 7. Añadir y Editar Foto de Perfil

Con esta funcionalidad, los usuarios podrán personalizar su cuenta añadiendo una foto de perfil durante el registro o editándose posteriormente en la sección de edición de datos.

### 1. Foto de perfil en el registro

Durante el proceso de registro, el usuario tendrá la opción de subir una foto de perfil. Esto se implementa añadiendo un campo de carga de imagen (input tipo file) al formulario de registro. Las características del manejo de la foto son:

- **Opcionalidad:** El campo de foto de perfil no es obligatorio. Si el usuario no sube una foto, se asignará una imagen predeterminada.



- **Validación:** El sistema validará que el archivo sea una imagen en formato permitido (como JPEG o PNG) y que no supere un tamaño máximo (por ejemplo, 2 MB).

## 2. Visualización de la foto de perfil

Una vez registrado, el usuario podrá ver su foto de perfil en la vista "cuenta". La foto aparecerá junto con los demás datos personales (nombre, email, etc.), tal como se muestra en el siguiente mockup:


**Datos de la cuenta**

**Nombre:** Adrian García

**Email:** adrian@gmail.com

**Fecha de nacimiento:** 23/06/2002

**Foto**



Editar

## 3. Edición de la foto de perfil

En la sección de edición de datos, el usuario tendrá la posibilidad de actualizar su foto de perfil. Esto se realiza añadiendo un campo de carga de imagen al formulario de edición. Las características de esta funcionalidad son:

- **Sustitución:** Al guardar los cambios, la nueva foto sustituirá a la anterior en el sistema.
- **Validación:** Se aplican las mismas reglas de validación que en el registro (formato y tamaño).

## 4. Cambios en el backend

Para soportar esta funcionalidad, se realizaron los siguientes cambios en el backend:

1. **Almacenamiento de imágenes.**
2. **Actualización de la imagen:** Durante la edición de los datos de la cuenta, si el usuario sube una nueva foto, la anterior será reemplazada en el sistema.

## 5. Flujo de interacción

1. **Registro:**
  - El usuario sube su foto de perfil (opcional).
  - Si no sube ninguna foto, se asigna una predeterminada.
2. **Visualización de cuenta:**
  - El usuario ve su foto junto con los datos personales.

### 3. Edición de datos:

- El usuario sube una nueva foto de perfil para sustituir la actual.
- La nueva foto es previsualizada antes de confirmar los cambios.
- Al guardar, la foto es actualizada en la base de datos y reflejada en la interfaz.

## Condiciones de satisfacción

Esta funcionalidad será considerada completa si:

1. Los usuarios pueden subir una foto de perfil al registrarse.
2. Los usuarios pueden ver su foto de perfil en la sección "Datos de la cuenta".
3. Los usuarios pueden actualizar su foto de perfil en la sección de edición de datos.
4. Las fotos de perfil cumplen con las reglas de validación de formato y tamaño.
5. Las imágenes predeterminadas se asignan correctamente a los usuarios que no suben una foto.

## 8. Filtrar y ordenar tareas

En esta tarea se añadirá una opción de filtro para la lista de tareas, de manera que un usuario pueda ver las tareas que considere. Se podrá buscar una tarea por su nombre, así como filtrar por sus distintos campos: prioridad, estado y fecha.

id	Tarea	Prioridad	Estado	Acciones
1	Tarea 1	Alta	Pendiente	<input type="checkbox"/> <input type="checkbox"/>

## Condiciones de satisfacción

El usuario debe poder filtrar las tareas por los campos que considere.

## 9. Comentarios en Tareas

En esta tarea vamos a añadir la posibilidad de que una tarea pueda tener una composición de comentarios. Vamos a explicar a continuación cómo haremos para implementar esta nueva característica.

### 1.1 Crear nuevo modelo **Comentario**:

1. Crear el nuevo modelo con los siguientes atributos:
  - a. Usuario usuario
  - b. Tarea tarea
  - c. String comentario
  - d. LocalDateTime fecha
2. Crear el DTO, repositorio y capa de servicios correspondiente.
3. Añadir un atributo **List<Comentario> comentarios** al modelo Tarea.

### 1.2 Añadir un comentario a una tarea:

Una vez tenemos creadas la tarea padre, vamos a la vista de detalles de la tarea. En esta vista veremos un botón que será Añadir comentario junto a un textbox. Si rellenamos ese textbox con alguna cadena y pulsamos el botón de enviar se añadiría el comentario dentro de la tarea, lo veremos mejor con un boceto.

En primer lugar vemos el botón para añadir una subtarea.

### Detalles de la tarea 1

Descripción

Enviar

### 1.3 Ver listado de comentarios:

A parte de los detalles de una tarea, podremos **ver un listado de comentarios** (si los hay) que tiene cada tarea. A continuación vemos un ejemplo de ello:

## Detalles de la tarea 1

Descripción

Enviar

**Juan** 15/12/24 11:34

Este es mi primer comentario en la web!

### 1.4 Editar o borrar el comentario:

Una vez en el **listado de comentarios**, deben aparecer dos botones.

- Borrar: Simplemente saldrá una alerta preguntando al usuario si está seguro de querer borrar la tarea.
- Editar: Una vez pulsado el botón, el texto del comentario se volverá un textbox con el texto del comentario precargado. El usuario podrá hacer los cambios en el texto que quiera y cuando acabe darle a Enviar. Esto actualiza el comentario

## Detalles de la tarea 1

Descripción

Enviar

**Juan** 15/12/24 11:34

Este es mi primer comentario en la web!



Enviar

## **Condiciones de satisfacción**

El usuario debe de poder comentar en todas las tareas en las que esta asignado y poder ver todos los comentarios que haya en una tarea.

Además debe de poder borrar y editar el comentario.

## 8. Asignar deadline a las tareas

En esta funcionalidad se implementará la posibilidad de asignar una fecha de vencimiento a cada tarea. Esta fecha límite será visible en la vista de detalles de tareas incluyendo los días restantes.

### 1. Modificación del modelo Tarea

Para soportar esta funcionalidad, se añadirá un atributo al modelo **Tarea**:

- **Deadline (LocalDateTime):** Este atributo almacenará la fecha de vencimiento de la tarea.

La fecha de vencimiento de la tarea podrá ser asignada en el momento de su creación. Este atributo será opcional y podrá ser modificado posteriormente.

### 2. Asignar prioridad al crear una tarea

Al crear una nueva tarea, el usuario deberá seleccionar una fecha y hora usando el componente de HTML **"datetime-local"**


En el mockup a continuación, podemos ver cómo se integra este componente en el formulario de creación de tareas:

### Nueva tarea para el usuario Ejemplo

Título

Descripción

Fecha de vencimiento



Crear Tarea

Cancelar

### 3. Visualizar la prioridad en el listado de tareas

Una vez creada una tarea, su deadline será visible en la vista general de detalles de tareas.

En el siguiente mockup, se observa un ejemplo del listado de tareas con su prioridad.

## Detalles de la tarea 1

Descripción

**Fecha de vencimiento:**

30/12/2024 11:30 ¡Quedan 3 días, 9 horas y 22 minutos!

### Condiciones de satisfacción

Esta funcionalidad será considerada completa si:

1. Se puede seleccionar una deadline cuando se crea la tarea con el componente mencionado.
2. Se puede ver el deadline y el tiempo restante en detalles de tareas.
3. Se puede editar la fecha de vencimiento en el formulario de editar tarea.

# Puesta en Producción

Para la puesta en producción utilizamos el mismo contenedor de producción que el de la práctica anterior.

Primero realizamos el script de migración a partir del schema de la versión 1.3.0 y la 1.4.0. Este script contiene las instrucciones para actualizar nuestra base de datos en producción.

Primero realizamos una copia de seguridad de los datos y actualizamos con el fichero de migración. Para ello, dentro del bash del contenedor ejecutamos el comando “psql -U mads mads < /mi-host/schema-1.3.0-1.4.0.sql”.

Si ahora comprobamos nuestra base de datos veremos que se han actualizado los cambios y podremos usar la aplicación sin ningún tipo de problema.



# Informe sobre las sesiones de Pair Programming

## Sesión 1

En la primera sesión vamos a realizar la tarea de “Poder crear tareas compuestas de otras subtareas”. Hemos decidido realizar la técnica Driver-Navigator. Esta tarea estaba asignada a Álvaro, por lo que es uno de los participantes, el otro miembro involucrado es Victoria. En este caso Álvaro ha sido Driver y Victoria ha ocupado el papel de Navigator.

En los primeros 20 minutos hemos añadido los dos atributos, modificado el constructor y creado sus getters y sus setters. Además hemos creado unos test para comprobar que los cambios añadidos funcionarán correctamente.

Después de la primera pausa hemos conseguido que pasen los test (ya que dieron un error) y hemos investigado y aprendido qué flujo utilizar en la creación de ramas y sus respectivas acciones, en este caso hemos estado 40 minutos.

Una vez ya teníamos las funciones básicas, los test pasaban y teníamos claro el flujo que seguir en git, hemos implementado la vista para ver las subtareas asignadas a una tarea. Hemos hecho un condicional para que en el caso de que no tenga sub tareas asignadas, se vea un mensaje comunicando que no tiene tareas asignadas. En este caso le hemos dedicado 30min.

## Sesión 2

En esta sesión hemos trabajado en varias funcionalidades relacionadas con la gestión de tareas y mejoras visuales. La técnica utilizada sigue siendo Driver-Navigator.

En los primeros 30 minutos hemos creado la vista y el controlador necesarios para añadir nuevas tareas al sistema. Esto incluyó el diseño de un formulario sencillo para ingresar los datos de la tarea y la implementación de la lógica en el controlador para procesar las solicitudes de creación.

Durante los siguientes 40 minutos, hemos implementado las funcionalidades para los botones de eliminar y editar tareas. Esto implicó vincular los botones en la vista con los métodos correspondientes en el controlador y comprobar que las operaciones se realizaban correctamente tanto en la base de datos como en la interfaz.

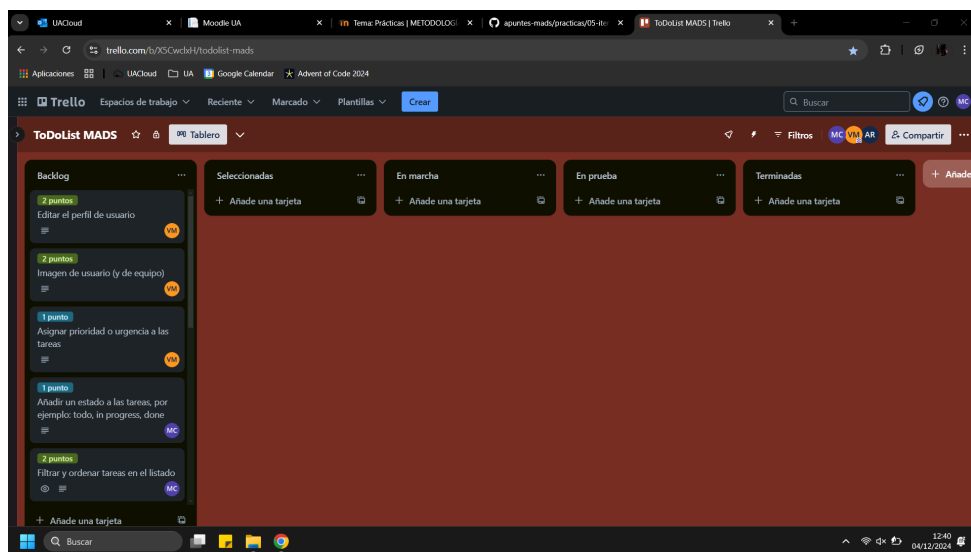
Después de una pausa, dedicamos 30 minutos a mejorar el proceso general de gestión de tareas, centrándonos en detalles que simplificaran la interacción del usuario. También realizamos ajustes en la interfaz, mejorando la disposición y la apariencia de los elementos en las vistas.

Finalmente, durante los últimos 30 minutos, resolvimos todos los conflictos que surgieron al intentar integrar los cambios realizados en esta sesión con los de la rama develop. Esto incluyó una revisión detallada de los archivos afectados y la ejecución de pruebas para garantizar que todo funcionara correctamente tras la fusión.

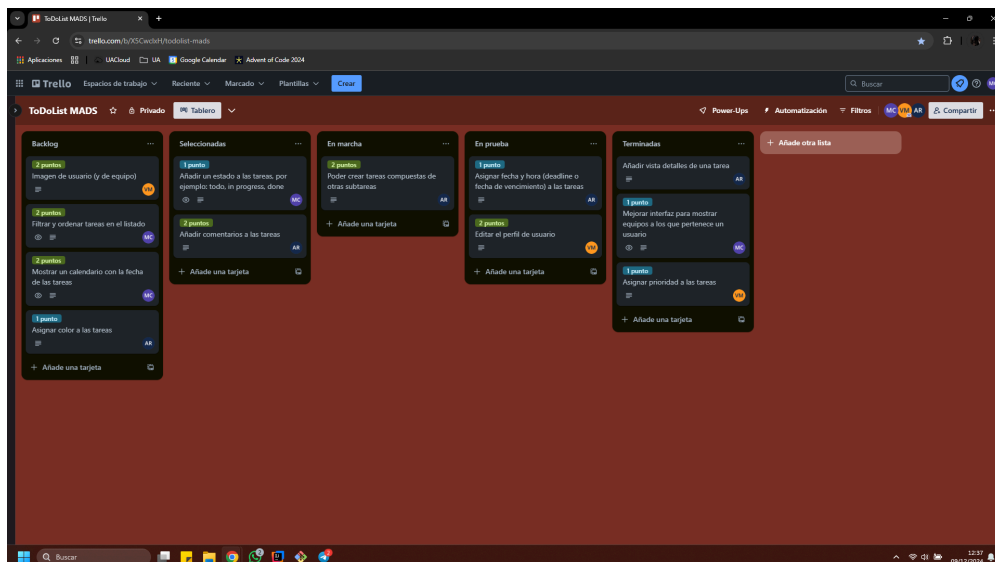
# Informe sobre la evolución del desarrollo

Las capturas del tablero de Trello muestran la evolución del trabajo utilizando la metodología Kanban. Cada columna representa un estado en el flujo de trabajo, como "Backlog", "Seleccionadas", "En Marcha", "En Prueba", "Terminadas". Al comparar las capturas en diferentes momentos, podemos observar cómo las tareas avanzaron a través del tablero.

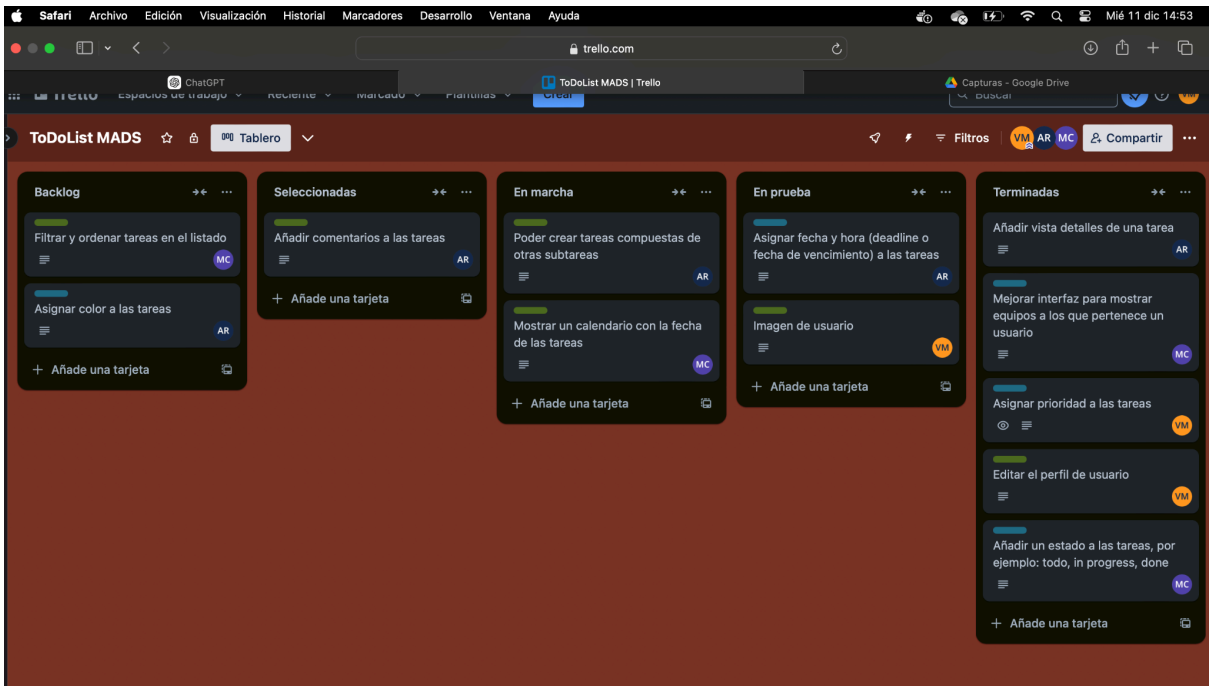
## Captura Tablero 04/12/24:



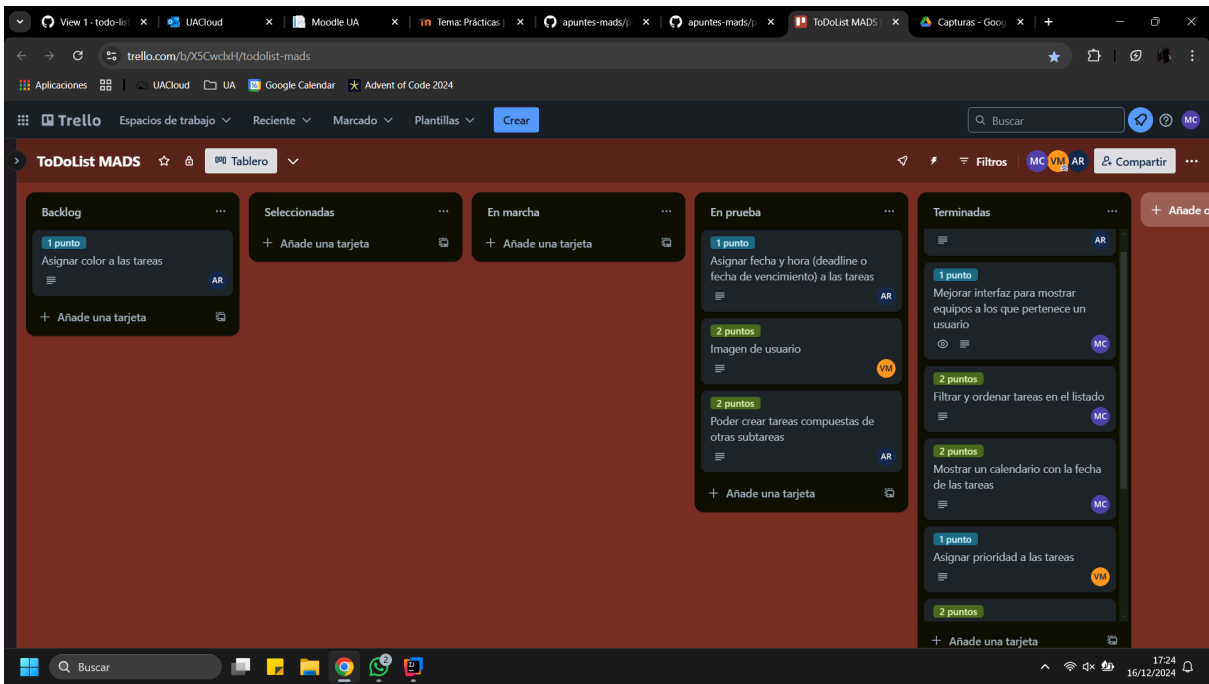
## Captura Tablero 09/12/24:



# Captura Tablero 11/12/24:

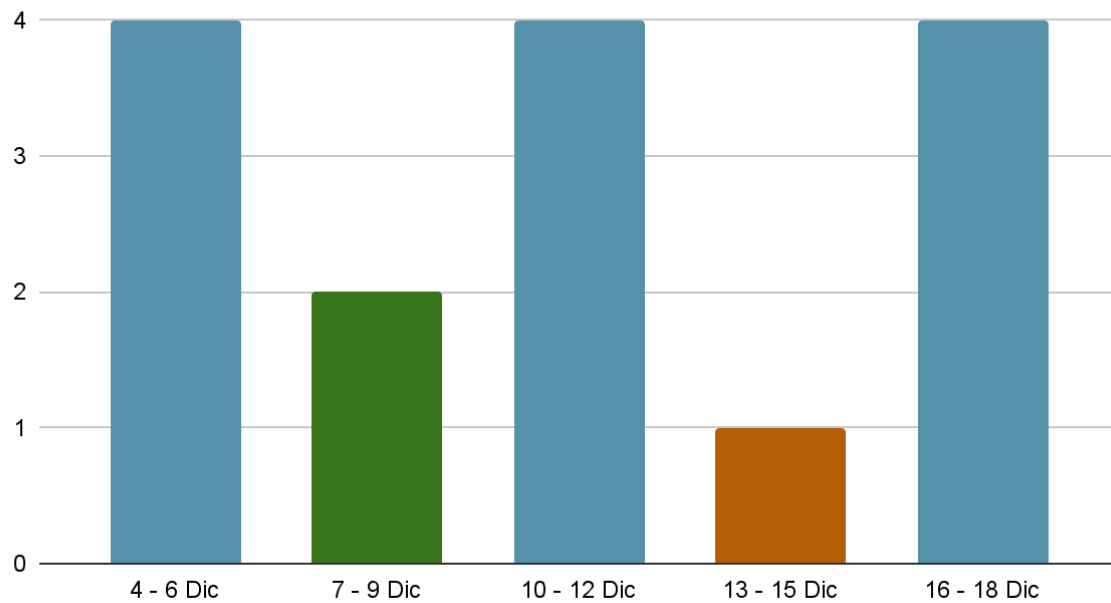


# Captura Tablero 16/12/24:



## Gráfica

Pull Requests



- Esta gráfica representa la cantidad de pull requests (PRs) creados en un repositorio de GitHub, divididos en cinco rangos de dos días cada uno. El objetivo es visualizar la actividad de desarrollo durante el período analizado, identificando picos o patrones de contribuciones.

# Funciones implementadas

## Mejorar interfaz para mostrar equipos a los que pertenece un usuario

### **Descripción para el usuario:**

Se podrá filtrar la lista de equipos de forma que se puedan ver todos o solamente aquellos de los que se forme parte.

### **Descripción técnica:**

Se añade un parámetro a la ruta que informa de la forma en la que queremos visualizar la lista. Dependiendo de la opción elegida esta cambiará entre todos los equipos o solo a los que el usuario logueado forme parte.

## Añadir un estado a las tareas, por ejemplo: todo, in progress, done

### **Descripción para el usuario:**

Al crear una tarea se le asigna por defecto un estado "Pendiente" el cual puede ser editado para poder realizar un seguimiento de la misma.

### **Descripción técnica:**

- Se añadió un atributo estado al modelo Tarea para almacenar el nivel de prioridad. Este atributo es de tipo String y solo acepta tres valores: "Pendiente, En progreso, Finalizada".
- Al crear la tarea se le asigna por defecto el estado "Pendiente".
- En el listado general de tareas, se modificó la vista para mostrar el estado junto con otros datos de cada tarea.
- El estado de una tarea puede ser modificado una vez creada para poder hacer un seguimiento de la misma.

## Asignar prioridad a las tareas

### **Descripción para el usuario:**

Permite asignar un nivel de prioridad (Alta, Media o Baja) a las tareas en el momento de su creación. Esta prioridad será visible en el listado de tareas y ayudará a organizar las tareas según su importancia.

### **Descripción técnica:**

- Se añadió un atributo prioridad al modelo Tarea para almacenar el nivel de prioridad. Este atributo es de tipo String y solo acepta tres valores: "Alta", "Media" y "Baja".
- El formulario de creación de tareas incluye un desplegable obligatorio para seleccionar la prioridad.
- En el listado general de tareas, se modificó la vista para mostrar la prioridad junto con otros datos de cada tarea.
- La prioridad asignada a una tarea no puede ser modificada una vez creada.

## Editar Perfil

### Descripción para el usuario:

Permite a los usuarios visualizar y modificar ciertos datos personales, como su nombre, correo electrónico y fecha de nacimiento. Además, incluye una opción para cambiar su contraseña con medidas de seguridad mejoradas, asegurando una experiencia más confiable y segura.

### Descripción técnica:

- Visualización de datos: Se implementó una vista de solo lectura en la que los usuarios pueden consultar su información personal.
- Formulario de edición: Se desarrolló un formulario que permite modificar campos específicos del perfil, como el nombre, correo electrónico y fecha de nacimiento.
- Cambio de contraseña:
  - El formulario incluye validaciones para la contraseña actual, nueva contraseña y confirmación de la nueva contraseña.
  - El backend valida que la contraseña actual proporcionada coincida con la almacenada en el sistema.
- Cifrado de contraseñas:
  - Se integró un mecanismo de cifrado para proteger las contraseñas almacenadas.
  - Se actualizaron los procesos de registro e inicio de sesión para manejar las contraseñas cifradas.
- Persistencia de datos: Los cambios realizados en el perfil se actualizan en la base de datos y son reflejados inmediatamente en las vistas.
- Notificaciones: Al guardar los cambios, el sistema envía una confirmación visual al usuario sobre el éxito de la operación.

## Calendario de tareas

### Descripción para el usuario:

Permite a los usuarios tener una vista de calendario en las que aparecerán marcados los días que tengan tareas de un mes seleccionado. También se verá una lista con dichas tareas, con las mismas opciones que en la vista "Listado de tareas".

**Descripción técnica:**

- Se añade una vista con un calendario. Cada vez que se carga, recoge las tareas del usuario logueado para el mes que haya seleccionado.
- En el calendario se muestran remarcadas las fechas con tareas mientras que en la lista aparecen dichas tareas con sus atributos y opciones.

## Añadir y Editar foto de perfil

**Descripción para el usuario:**

Permite a los usuarios personalizar su cuenta añadiendo una foto de perfil durante el registro o editándola posteriormente desde la sección de edición de datos. Si no se sube ninguna foto, se asignará automáticamente una imagen predeterminada.

**Descripción técnica:**

- Registro con foto de perfil:
  - Se añadió un campo opcional de carga de imagen en el formulario de registro.
  - El sistema valida que el archivo sea una imagen en formatos permitidos (JPEG o PNG).
- Visualización de la foto de perfil:
  - La foto de perfil del usuario se muestra junto con su información personal en la vista de cuenta.
- Edición de la foto de perfil:
  - En el formulario de edición de datos, se añadió un campo de carga de imagen que permite al usuario reemplazar su foto de perfil actual.
  - Se aplican las mismas reglas de validación que en el registro (formato y tamaño).

## Filtrar y ordenar tareas

**Descripción para el usuario:**

Se podrá filtrar la lista de tareas según el nombre de la misma, su estado y/o su prioridad.

**Descripción técnica:**

- Se añaden como parámetros a la ruta el nombre que debe tener la tarea, su estado y su prioridad.
- En caso de estar vacío algún campo no se tendrá en cuenta para la búsqueda.
- Se listarán todas las tareas que cumplan las condiciones de los campos rellenos.

# Retrospectiva

Qué ha ido bien:

**Colaboración efectiva:** La implementación de las tareas en sesiones de pair programming utilizando la técnica Driver-Navigator fue un éxito. Esto permitió un flujo de trabajo organizado, resolución rápida de problemas y aprendizaje mutuo entre los participantes.

**Cumplimiento de objetivos:** Todas las funcionalidades clave planificadas para el sprint se completaron con las condiciones de satisfacción definidas, incluyendo la creación de tareas compuestas de subtareas, la asignación de prioridades y la edición de perfil.

**Calidad técnica:** Se realizaron pruebas exhaustivas para garantizar el correcto funcionamiento de las nuevas funcionalidades, destacando el cifrado de contraseñas y las validaciones de datos en el backend.

**Gestión del tiempo:** Las tareas se distribuyeron adecuadamente a lo largo del sprint, con ajustes continuos basados en las necesidades de cada funcionalidad.

**Mejora continua:** Se dedicaron esfuerzos específicos a resolver conflictos de ramas en Git y a realizar integraciones con éxito, asegurando un código limpio y funcional.

Que podría mejorar:

**Planificación inicial ajustada:** El retraso en la finalización de la práctica anterior afectó la disponibilidad de tiempo para este sprint, lo que nos llevó a trabajar con un margen más estrecho. Esto pudo haber limitado nuestra capacidad para realizar ajustes más detallados o abordar tareas con mayor profundidad.

**Aplicación de TDD:** Aunque se llevaron a cabo pruebas en las funcionalidades implementadas, la falta de un enfoque basado en TDD desde el principio podría haber reducido el esfuerzo requerido para corregir errores y mejorado la calidad del código desde las primeras iteraciones.