

MynimaList



MynimaList

Tabla de contenidos

Tabla de contenidos	2
Integrantes del grupo	3
Introducción	4
Roles	5
Gestión de riesgos	6
Tipo de riesgo: Personal	6
Tipo de Riesgo: Requisitos	6
Tipo de Riesgo: Estimación	6
Tipo de Riesgo: Organización	6
Tipo de Riesgo: Tecnológico	6
Planificación	7
Modelo de proceso software elegido: Scrum	7
Tableros Trello	8
Requisitos	10
Funcionales	10
No funcionales	14
Requisitos Opcionales	16
Esquema en MagicDraw	16
Casos de Uso	17
Herramientas software	23
Herramienta de comunicación	23
Herramienta de trabajo colaborativo	23
Herramienta de elaboración de documentos	23
Diagrama de clases	24

Integrantes del grupo

- Jacobo Elichá Garrucho (jacoboeg@uma.es)
- Julia Pérez Barreales (juliaperez@uma.es)
- Álvaro Sánchez Hernández (alvaro.s.h@uma.es)
- Jesús Escudero Moreno (xexu65@uma.es)
- Isidro Javier García Fernández (isidrojova@uma.es)
- Juan Manuel García Delgado (juanma01@uma.es)
- José Antonio Luque Salguero (joseantonioluquesalguero1212@uma.es)
- David Ramírez Palacios (ramirezpalaciosd@uma.es)

Enlace repositorio GitHub: <https://github.com/Jacobo-EG/g10-MynimaList>

Introducción

MynimaList consiste en una aplicación web minimalista e intuitiva para realizar anotaciones colaborativas como listas de tareas, ideas de proyectos o recordatorios.

Buscamos crear un espacio en el que se puedan gestionar distintas listas, tanto privadas, como compartidas entre varios usuarios que tengan necesidades comunes.

La idea inicial es que un usuario pueda crear una lista de tareas o con anotaciones en la que posteriormente se puedan añadir más editores si se trata de una lista compartida.

Nos decidimos por esta idea ya que, en nuestro grupo de estudio echamos en falta una forma sencilla de poder escribir en tiempo real las tareas que necesitamos realizar y poder compartirlas con el resto de nuestros compañeros.

De esta manera, se facilita la organización y permite mantener informados a los integrantes del grupo en todo momento.

Roles

- Álvaro Sánchez Hernández: tester, implementador
- David Ramírez Palacios: Scrum Master, implementador
- Isidro Javier García Fernández: diseñador, Product Owner
- Jacobo Elicha Garrucho: diseñador, Scrum Master
- Jesús Escudero Moreno: Product Owner, tester
- José Antonio Luque Salguero: implementador, tester.
- Juan Manuel García Delgado: tester, implementador
- Julia Pérez Barreales: diseñadora, implementadora

Gestión de riesgos

- Tipo de riesgo: **Personal**
 - Descripción del riesgo: Personal clave está enfermo o no está disponible en momentos críticos.
 - Probabilidad: moderada.
 - Efectos del riesgo: Serio.
 - Estrategia para mitigarlo: Reorganizar el equipo de desarrollo para que haya más superposición de trabajo y, por lo tanto, el personal conozca el trabajo de otros compañeros.

- Tipo de Riesgo: **Requisitos**
 - Descripción del riesgo: Se proponen unos cambios de requisitos que necesitan un importante rediseño.
 - Probabilidad: Baja.
 - Efecto del riesgo: Serio.
 - Estrategia para mitigarlo: Obtener la información de trazabilidad para evaluar el impacto de los cambios en los requisitos, Maximizar la ocultación de la información en el diseño.

- Tipo de Riesgo: **Estimación**
 - Descripción del riesgo: Se subestima el tiempo necesario para el desarrollo del software.
 - Probabilidad: Alto.
 - Efecto del riesgo: Serio.
 - Estrategia para mitigarlo: Realizar una estimación por exceso.

- Tipo de Riesgo: **Organización**
 - Descripción del riesgo: Una reestructuración de la organización provoca un cambio de los gestores responsables del proyecto.
 - Probabilidad: Bajo.
 - Efecto del riesgo: Tolerable.
 - Estrategia para mitigarlo: Preparar un documento informativo que muestre la forma en la cuál el proyecto realiza una importante contribución a los objetivos.

- Tipo de Riesgo: **Tecnológico**
 - Descripción del riesgo: La base de datos usada en el sistema no es capaz de procesar tantas transiciones por segundo como se esperaba.
 - Probabilidad: Moderado.
 - Efecto del riesgo: Serio.
 - Estrategia para mitigarlo: Investigar la posibilidad de comprar una base de datos de alto rendimiento.

Planificación

Modelo de proceso software elegido: Scrum

Hemos elegido este modelo de trabajo, ya que fomenta la comunicación entre los miembros del grupo, conocimiento de las potencialidades, optimización del tiempo y adaptabilidad a cambios.

Mediante este modelo de proceso software podemos llevar a cabo un consenso de opiniones e ideas acerca de cómo desarrollar las distintas partes del proyecto o cómo resolver los problemas que se puedan presentar.

Pensamos que es buena idea utilizar este método, pues en un principio nuestro proyecto parte de una cierta incertidumbre.

Vamos a intentar realizar ciclos cortos de Sprint para intentar reducir los posibles riesgos que puedan suceder a lo largo del desarrollo de nuestra aplicación web.

Para el desarrollo de MynimaList mediante el modelo Scrum deberemos implementar los **documentos** propios de dicha metodología de trabajo, como lo son:

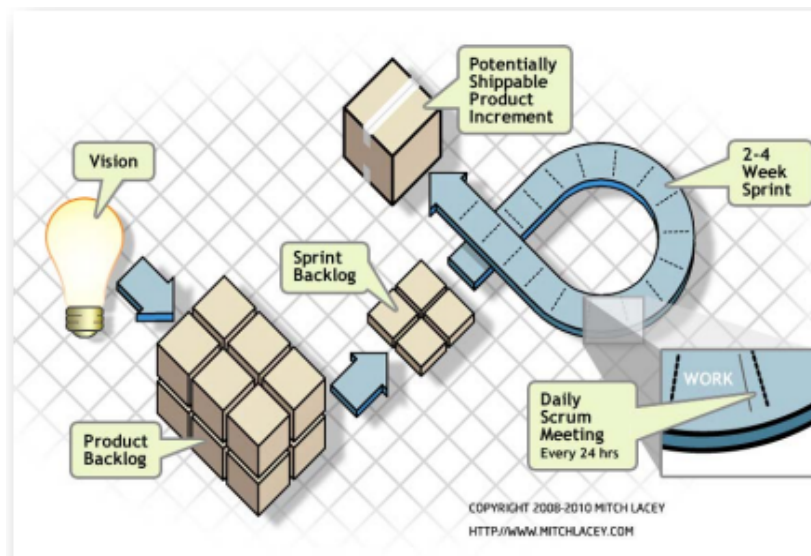
- Product Backlog → reunir los requisitos del proyecto y describir las funcionalidades
- Sprint Backlog → requisitos que se desarrollaran en el siguiente sprint
- Burndown Chart → gráfica que mide la cantidad de requisitos en el Backlog

También hemos seguido la división de cada uno de los miembros del equipo en distintos **roles** que se utilizan en el método Scrum. Como hemos mencionado anteriormente, nos hemos asignado roles como:

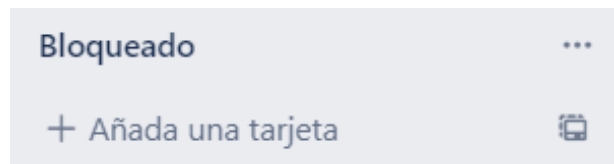
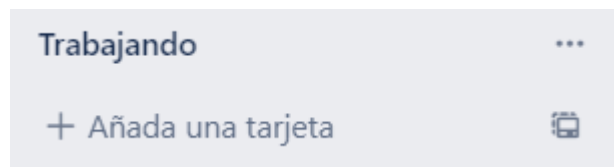
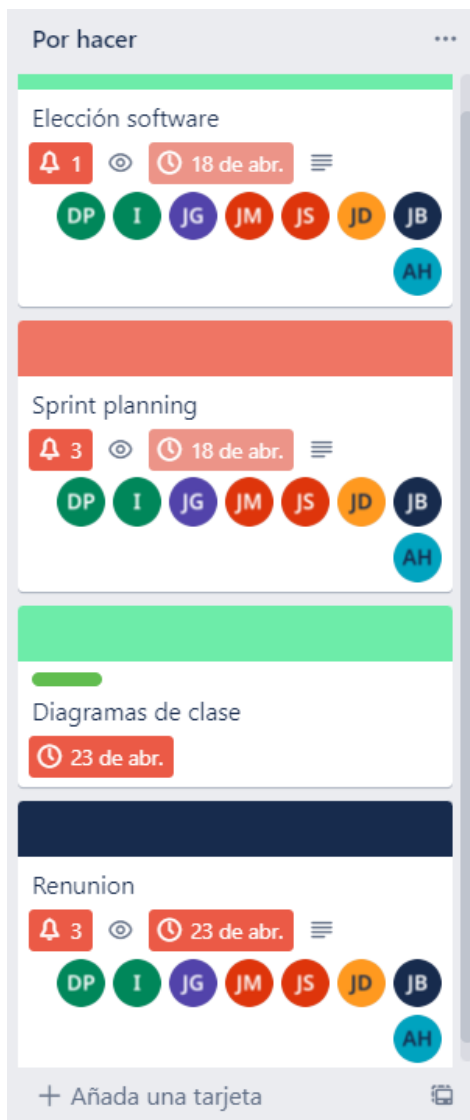
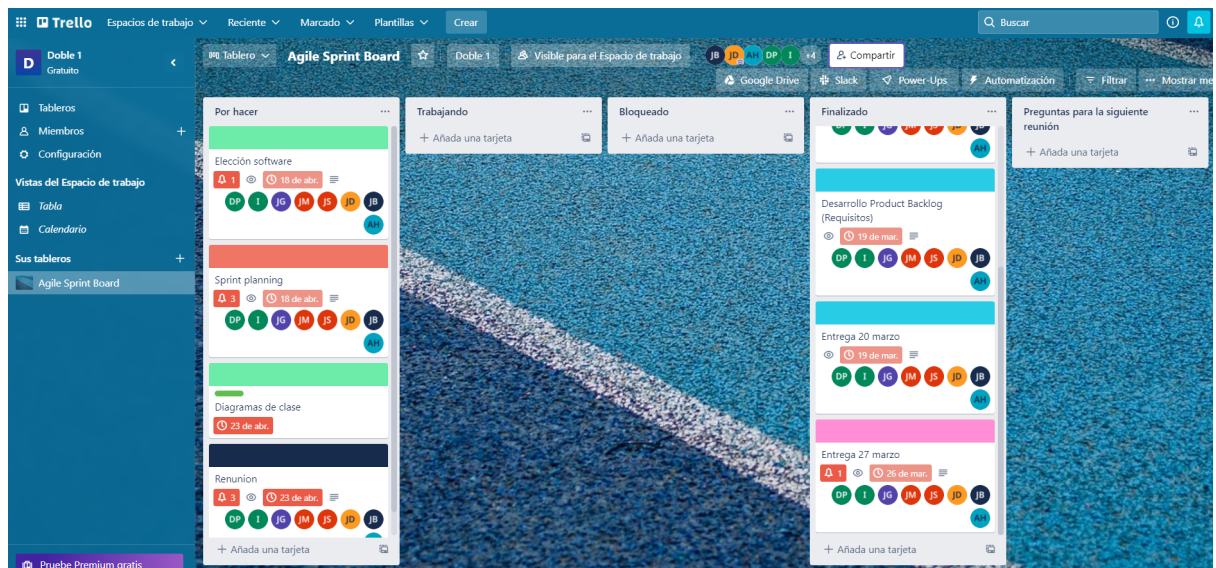
- Producto Owner
- Scrum Master
- Equipo de desarrollo: diseñador, implementador, tester

También llevaremos a cabo **reuniones** correspondientes al modelo de proceso software Scrum, con el objetivo de establecer la situación actual del proyecto, problemas que se nos presenten y posibles soluciones a los mismos. Desarrollaremos documentos como:

- Daily Scrum / Stand-up meeting → reunión sobre el estado del proyecto
- Sprint Planning → establecer qué trabajo se hará
- Sprint Review Meeting → Inspeccionar el trabajo realizado
- Sprint Retrospective → Análisis del propio sprint



Tableros Trello



Finalizado

Desarrollar logotipo

JGJB

Desarrollar documento. Portada, Secciones 1, 2 y 3.

DP I JG JM JS JD JB

AH

Entrega 13 de marzo

DP I JG JM JS JD JB

AH

Desarrollo Product Backlog (Requisitos)

🔔 19 de mar.

DP I JG JM JS JD JB

AH

Finalizado

DP I JG JM JS JD JB

AH

Desarrollo Product Backlog (Requisitos)

🔔 19 de mar.

DP I JG JM JS JD JB

AH

Entrega 20 marzo

🔔 19 de mar.

DP I JG JM JS JD JB

AH

Entrega 27 marzo

🔔 1

🔔 26 de mar.

DP I JG JM JS JD JB

AH

Requisitos

Funcionales

RF1: Recuadros inicio de sesión

Como Product Owner

quiero tener una página de inicio con dos recuadros en los que los usuarios inserten su nombre de usuario/correo electrónico y contraseña

Para iniciar sesión al entrar en la aplicación.

Pruebas de Aceptación

- Ver cómo en el recuadro de contraseña está no aparece con letras, es decir, aparece con puntos negros.

RF2: Botón inicio de sesión (Deriva de RF1)

Como Product Owner

quiero tener un botón de inicio que cuando el usuario haya previamente introducido su correo electrónico y contraseña, acceda a la web.

Para iniciar sesión al entrar en la aplicación.

Pruebas de Aceptación

- Al pulsar el botón si los datos introducidos son correctos iniciar sesión, si no lo son que informe del error.

RF3: Botón para ir a la página de registro

Como Product Owner

quiero tener un botón de registro que nos dirija a la página de registro (donde el nuevo usuario podrá registrarse).

Para iniciar sesión al entrar en la aplicación.

Pruebas de Aceptación

- Probar que el botón efectivamente nos redirige a la página de registro.

RF4: Registro (Deriva de RF3)

Como Product Owner

quiero tener una página de registro con tres recuadros en los que los usuarios inserten su nombre de usuario, su correo electrónico y su contraseña, en dicho orden.

Para registrarse en la aplicación.

Pruebas de Aceptación

- Comprobar que efectivamente dichos recuadros aparecen y que además en el de contraseña esta aparece con puntos negros.

RF5: Botón de registro (Contenido en RF4)

Como Product Owner

quiero tener un botón de registro que cuando el usuario haya previamente introducido su usuario, su correo electrónico y su contraseña se registre en la web (Todo esto desde la página de registro).

Para registrarse en la aplicación.

Pruebas de Aceptación

- Registrar un usuario nuevo.
- Registrar un usuario cuyo nombre de usuario ya haya sido registrado y se recibe un mensaje de error.

- Registrar un usuario cuyo correo ya haya sido utilizado para otro usuario y se recibe un mensaje de error.
- No se permiten registros de usuarios que dejen sin valor algún dato.

RF6: Página principal de las listas (Deriva de RF1)

Como Product Owner

quiero tener una página principal donde poder ver todos mis listas, crearlas...

Para seleccionar o crear una lista.

Pruebas de Aceptación

- Observar que aparece el botón de crear listas.
- Observar que aparece un menú con las listas creadas.

RF7: Menú de listas (Contenido en RF6)

Como Product Owner

quiero un menú donde se encuentren las listas creadas.

Para organizar las diferentes listas que tengo y acceder a ellas.

Pruebas de Aceptación

- Seleccionar una lista y acceder a ella.
- Eliminar una lista y observar que desaparece del menú.
- Crear listas y observar que aparecen en el listado de las listas.

RF8: Botón para crear listas (Contenido en RF7)

Como Product Owner

quiero tener una opción en el menú donde poder crear nuevas listas.

Para poder crear listas.

Pruebas de Aceptación

- Pulsar el botón y crear una lista.

RF9: Guardado manual de la lista (Contenido en RF6)

Como Product Owner

quiero poder guardar las listas creadas y sus modificaciones pulsando un botón.

Para no perder las listas creadas.

Pruebas de Aceptación

- Si recargamos la página del navegador donde trabajamos con la lista, ésta aparecerá actualizada con los cambios realizados.

RF10: Icono/texto que indica si la lista se encuentra Guardada o No Guardada (Contenido en RF6)

Como Product Owner

quiero poder indicar al usuario el estado de la lista.

Para que el usuario sepa el estado de su lista y no perderla.

Pruebas de Aceptación

- Símbolo de lista guardada implica que la lista se mantendrá actualizada.
- Símbolo de lista no guardada implica que los cambios realizados se perderán al cerrar la ventana de la lista.

RF11: Botón cierre de sesión (Contenido en RF7)

Como Product Owner

quiero que haya un botón en el menú de cerrar sesión.

Para poder cambiar de cuenta o que nadie acceda a mi cuenta.

Pruebas de Aceptación

- Si se desea cambiar de cuenta, será una cuenta distinta de la que se intenta cerrar sesión
- Se redirigirá al usuario a la página principal de inicio de sesión de MynimaList

RF12: Botón tareas con fecha de entrega (Contenido en RF6)

Como Product Owner

quiero que haya un botón con el que se pueda seleccionar en un calendario una fecha de entrega de las tareas.

Para ayudar a la organización de los usuarios.

Pruebas de Aceptación

- Para estudiantes, la fecha de entrega deberá seleccionarse en un rango correspondiente al curso académico

RF13: Botones fuentes/tamaño de letra (Contenido en RF7)

Como Product Owner

quiero que haya un botón que permite cambiar la fuente y otro el tamaño de la letra de las listas.

Para poder personalizar las listas.

Pruebas de Aceptación

- Si se cambia la fuente, todo lo ya anotado y lo que se siga anotando se encontrará en esa fuente.
- Si se cambia el tamaño de la letra, todo lo ya anotado y lo que se siga anotando aparecerá en ese tamaño.

RF14: Modo oscuro (Contenido en RF7)

Como Product Owner

quiero que haya un botón para poder poner la página en un modo oscuro.

Para personalizar la aplicación.

Pruebas de Aceptación

- Se encuentra la página en modo claro, si se pulsa el botón cambia a modo oscuro.
- Se encuentra la página en modo oscuro, si se pulsa el botón cambia a modo claro.

RF15: Botón Compartir (Contenido en RF6)

Como Product Owner

quiero que haya un botón para poder compartir la lista con otros usuarios.

Para compartir las listas.

Pruebas de Aceptación

- Al pulsar el botón, aparece un pop up con un enlace que permite acceder a la lista.

RF16: Botón Borrar lista (Contenido en RF6)

Como Product Owner

quiero que haya un botón para poder borrar la lista creada (en el caso de ser administrador de la lista). En cambio, si un usuario no-administrador desea eliminar la lista le aparecerá un pop up indicando que no puede realizar esa acción.

Para borrar las listas.

Pruebas de Aceptación

- Un usuario administrador pulsa el botón eliminar lista, aparece una solicitud de confirmación de la eliminación, y se elimina la lista.
- Un usuario no administrador pulsa eliminar lista, aparece un pop up que indica que no se puede eliminar.

RF17: Pop up miembros (Deriva de RF15)

Como Product Owner

quiero que al pulsar el botón compartir aparezca un pop up con los usuarios que estén inscritos en la lista, junto con el link único de cada lista.

Para poder ver el número de miembros e invitar a otros usuarios.

Pruebas de Aceptación

- Un usuario pulsa el botón pop up miembros, y aparece un pop up con los usuarios que se encuentran inscritos en la lista.

RF18: Botón confirmar borrado (Deriva de RF16)

Como Product Owner

quiero al pulsar el botón de eliminar lista, aparezca un pop up preguntando si realmente se desea eliminar la lista a modo de confirmación.

Para evitar que los usuarios borren sus listas involuntariamente.

Pruebas de Aceptación

- Un usuario administrador pulsa el botón eliminar lista, aparece una solicitud de confirmación de la eliminación para una doble verificación.
- Un usuario no administrador pulsa eliminar lista, aparece un pop up que indica que no se puede eliminar.

RF19: Botón salir de lista (Contenido en RF6)

Como Product Owner

quiero que haya un botón para poder salir de la lista.

Para que un usuario pueda salirse de una lista siempre que lo desee.

Pruebas de Aceptación

- Un usuario pulsa el botón salir de lista, aparece una solicitud de confirmación de salida de la lista, y se sale de la lista.

RF20: Botón confirmar salir de lista (Deriva de RF19)

Como Product Owner

quiero al pulsar el botón de salir lista, aparezca un pop up preguntando si realmente se desea salir de la lista a modo de confirmación.

Para evitar que los usuarios salgan de las listas involuntariamente.

Pruebas de Aceptación

- Un usuario pulsa el botón salir de lista, aparece una solicitud de confirmación de salida de la lista para una doble verificación.

No funcionales

RNF1: Autoguardados frecuentes (Satisface RF9)

Como Product Owner

quiero que el autoguardado se realice en intervalos de tiempo lo más breve posibles.

Para evitar la pérdida de datos al cerrar las listas sin guardarlas manualmente.

Pruebas de Aceptación

- Si recargamos la página del navegador donde trabajamos con la lista, ésta aparecerá actualizada con los cambios realizados.
- Deberá aparecer si se encuentra el usuario conectado a la red para poder realizar el guardado y que el usuario sea consciente de que no es posible realizar el guardado en caso contrario.

RNF2: Encriptado de contraseñas

Como Product Owner

quiero que las contraseñas se encuentren encriptadas.

Para la seguridad y privacidad de nuestros usuarios.

Pruebas de Aceptación

- Comprobar que al almacenar la contraseña en la base de datos esta se encuentra encriptada.

RNF3: Almacenamiento en nuestra BBDD

Como Implementador

quiero que toda la información se almacene en nuestra BBDD.

Para guardar la información.

Pruebas de Aceptación

- Cuando la página deba almacenar información comprobar que se almacena en la base de datos.

RNF4: Página intuitiva y sencilla

Como Product Owner

quiero que la página sea simple, intuitiva y minimalista.

Para que sea atractiva y fácil de usar para los usuarios.

Pruebas de Aceptación

- Comprobar que un público general puede hacer uso de la aplicación sin que sea necesario buscar información fuera de esta.

RNF5: Licencia de software

Como Product Owner

Quiero una licencia de software.

Para poder desarrollar la aplicación con una serie de términos y condiciones de uso.

RNF6: Máximo de creación de listas/caracteres diario (Contenido en RF6)

Como Product Owner

quiero limitar la creación de listas por usuario al día.

Para impedir la sobrecarga de información en nuestra Base de Datos.

Pruebas de Aceptación

- Cuando se supere el número de lista deberá aparecer un mensaje que bloquee la creación de más

- Cuando se supere el número de caracteres deberá aparecer un mensaje que bloquee el poder insertar más caracteres.

RNF7: Control de nombres de usuario (Deriva de RF5)

Como Administrador

quiero no permitir caracteres extraños en el usuario ni que se repitan nombres de usuario

Para evitar errores a la hora de iniciar sesión.

Pruebas de Aceptación

- Error al intentar registrar un usuario con correo o usuario o contraseña que contenga caracteres no alfanuméricos.

RNF8: Guardar datos de usuario (Deriva de RNF3)

Como Administrador

quiero guardar el nombre de usuario, correo electrónico y contraseña de cada usuario.

Para que sea posible iniciar sesión.

Pruebas de Aceptación

- Observar que la creación de un usuario se almacena en la base de datos correctamente.

RNF9: Responsive design (Todos los botones, menus y recuadros estarían contenidos aquí no??)

Como Product Owner

quiero que la página web se adapte correctamente a los distintos tipos de formato de pantalla manteniendo la estética, claridad y funcionalidad.

Para que la aplicación se pueda usar en todos los dispositivos.

Pruebas de Aceptación

- Comprobar que la aplicación se ve correctamente al entrar desde un ordenador
- Comprobar que la aplicación se ve correctamente al entrar desde un teléfono móvil
- Comprobar que la aplicación se ve correctamente al entrar desde otro dispositivo portátil (por ejemplo una tablet)

Requisitos Opcionales

Requisitos funcionales

RF21: Ocultar menú

Como Product Owner

quiero poder ocultar o no ocultar el menú

Para personalizar la aplicación.

Pruebas de Aceptación

- Si el menú está oculto(no aparece desglosado en la pantalla), cuando se pulse sobre el botón de menú, dejará de estar oculto.
- Si el menú no está oculto, al pulsar en el botón de menú se ocultará.

Requisitos no funcionales

RNF10: Corrección automática de Errores

Como Product Owner

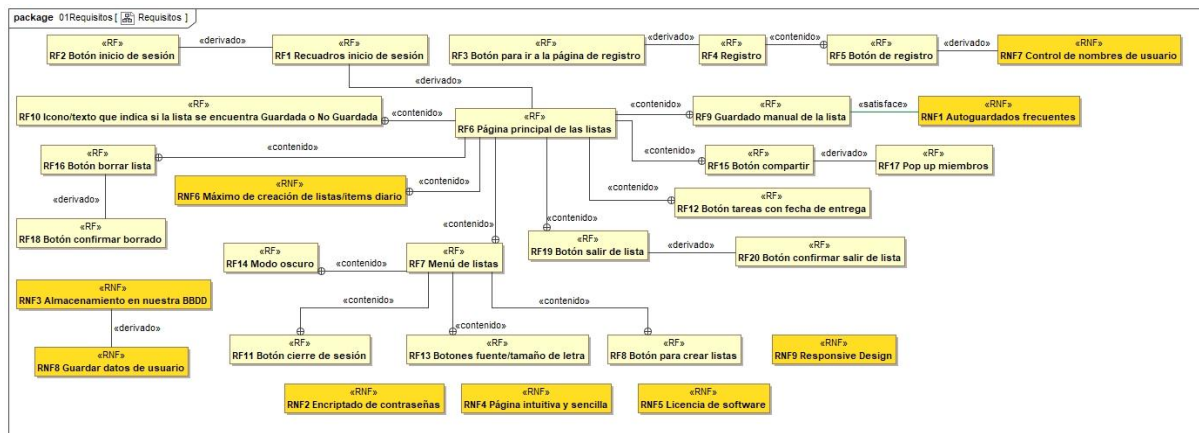
quiero tener una herramienta que corrija los errores ortográficos

Para fomentar el uso correcto del lenguaje.

Pruebas de Aceptación

- Si hay una palabra mal escrita, esta se sombreadrá, y al colocar el ratón sobre ella, aparecerá un cuadro con la corrección propuesta
- Si una palabra no está mal escrita, no se sombreadrá

Esquema en MagicDraw



Casos de Uso

CU1. Iniciar sesión

- Contexto de uso:
CU1. Un usuario desea acceder a su cuenta.
- Precondiciones y activación:
Estar registrado y conectado a internet.
- Garantías de éxito / Postcondición:
Consigue acceder.
- Escenario principal:
 1. Escribir el usuario o correo electrónico.
 2. Escribir en la parte inferior la contraseña.
 3. Seleccionar 'Entrar' o pulsar el retorno de carro.
 4. El sistema comprueba la validez del usuario y la contraseña.
- Escenarios alternativos:
 - 1.b Usuario incorrecto o usuario no registrado.
 - 2.b El usuario es correcto pero la contraseña es incorrecta.

CU2. Crear lista

- Contexto de uso:
CU2. Cuando un usuario lo desee podrá crear una lista nueva.
- Precondiciones y activación:
El usuario está conectado a la web y se encuentra registrado.
- Garantías de éxito / Postcondición:
Se ha creado una lista nueva con el usuario como administrador.
- Escenario principal:
 1. Iniciar sesión en MynimaList
 2. El usuario pulsa el desplegable 'Administrar listas'.
 3. El sistema comprueba que el usuario no ha excedido el máximo diario.
 4. El sistema muestra el desplegable.
 5. En el desplegable selecciona 'Crear nueva lista'.
 6. La web muestra un formulario para crear la lista.
 7. Posteriormente selecciona las características de la lista rellenando el formulario estableciendo una configuración inicial para la lista.
 8. Guardamos la lista.
 9. La lista se guarda en la web y se muestra un mensaje de éxito.
- Escenarios alternativos:
 - 1.b Credenciales de inicio de sesión erróneas
 - 3.b El usuario ha excedido el límite diario de listas diarias y se le impide crear una nueva mostrando un mensaje de error.

CU3. Visualizar lista

- Contexto de uso:
CU3. El usuario desea visualizar el contenido de la lista
- Precondiciones y activación:
Estar registrado y conectado a internet.
- Garantías de éxito / Postcondición:
Debe tener un enlace de acceso a una lista ya creada.
- Escenario principal:
Lista creada por el mismo usuario:
 1. Inicia sesión en MynimaList.
 2. Selecciona la lista.
 3. El sistema accede a la lista seleccionada.
 4. Visualiza el contenido de la lista mostrado en pantalla por el sistema.Lista creada por otro usuario:
 1. Obtener enlace de la lista creada por otro usuario.
 2. Acceder al enlace mediante un navegador web.
 3. Visualiza el contenido de la lista mostrado en pantalla por el sistema.
- Escenarios alternativos:
 - 2.2.b Enlace de la lista incorrecto porque la lista ya ha sido borrada u otro motivo
 - 1.1.b Credenciales de inicio de sesión erróneas

CU4. Modificar lista

- Identificador único:
CU4. Modificar lista
- Contexto de uso:
Se desea modificar el contenido de una lista ya creada
- Precondiciones y activación:
Se debe de iniciar sesión para poder modificar una lista. La lista ya debe existir.
- Garantías de éxito / Postcondición:
La lista debe haber sido creada por el mismo usuario o por otro. Se debe tener acceso a dicha lista.
- Escenario principal:
Lista creada por el mismo usuario:
 1. Inicia sesión en MynimaList.
 2. Selecciona la lista creada.
 3. Pulsa en la línea en la que desea escribir.
 4. Modifica el contenido de la lista.
 5. Selecciona guardar los cambios realizados.
 6. El sistema guarda en la base de datos los cambios realizados.Lista creada por otro usuario:
 1. Obtener enlace de la lista creada por otro usuario.
 2. Acceder al enlace mediante un navegador web.
 3. Visualizar la lista creada.
 4. Modificar el contenido de la lista.
 5. Selecciona guardar los cambios realizados.
 6. El sistema guarda en la base de datos los cambios realizados.
- Escenarios alternativos:
 - 1.3.b No es posible modificar la lista mientras otro usuario la está modificando, el sistema muestra por pantalla mensaje de error.

- 2.3.b No es posible modificar la lista mientras otro usuario la está modificando, el sistema muestra por pantalla mensaje de error.
- 2.2.b Enlace de la lista incorrecto.
- 1.1.b Credenciales de inicio de sesión erróneas.

CU5: Invitar lista

- Identificador único:
CU5. Invitar Lista
- Contexto de uso:
Se desea compartir una lista con otro usuario.
- Precondiciones y activación:
Se debe iniciar sesión para poder invitar. La lista ya debe existir.
Solo puede compartir la lista un administrador.
- Garantías de éxito / Postcondición:
El usuario se ha unido satisfactoriamente a la lista.
- Escenario principal:
 1. Iniciar sesión en MynimaList
 2. Pulsar el botón de crear enlace.
 3. El sistema genera un enlace de invitación
 4. Copiar el enlace de invitación
 5. Mandar el enlace al invitado
- Escenarios alternativos:
 - 1.b Credenciales de inicio de sesión erróneas
 - 2.b Enlace erróneo o antiguo.
 - 2.b Usuario bloqueado previamente.

CU6: Establecer admin

- Identificador único:
CU6. Establecer admin.
- Contexto de uso:
Se desea establecer a un usuario como administrador de una lista.
- Precondiciones y activación:
Se debe iniciar sesión para poder establecer un administrador y el usuario al que queremos establecer como admin debe pertenecer a lista.
La lista ya debe existir con anterioridad.
Sólo puede establecer admin otro admin.
- Garantías de éxito / Postcondición:
El nuevo usuario es ahora un admin.
- Escenario principal:
 1. Iniciar sesión en Mynimalist
 2. Visualizar la lista.
 3. Pulsar el botón de establecer admin
 4. El sistema despliega una ventana
 5. Escribir el nombre de usuario que queremos establecer como admin
 6. El sistema comprueba la validez del usuario y que pertenezca a la lista
 7. El sistema establece como admin al usuario
- Escenarios alternativos:
 - 1.b Credenciales de inicio de sesión erróneas
 - 6.b Usuario no pertenece a la lista

CU7: Editar Aspecto y Características Lista

- Identificador único:
CU7. Editar Aspecto Lista
- Contexto de uso:
Cambiar estilo de letra, tamaño... de una lista.
- Precondiciones y activación:
Se debe iniciar sesión como admin de la lista y existir la lista.
- Garantías de éxito / Postcondición:
Se ha cambiado el aspecto de lista satisfactoriamente.
- Escenario principal:
 1. Iniciar sesión en Mynimalist.
 2. El admin pulsa el botón de ajustes de la lista.
 3. Modificar el estilo de letra o el tamaño.
 4. Guardar los cambios.
 5. El sistema guarda los cambios
- Escenarios alternativos:
 - 1.b Credenciales de inicio de sesión erróneas
 - 3.b Error al modificar.
 - 5.b Error al guardar los cambios.

CU8: Eliminar Lista

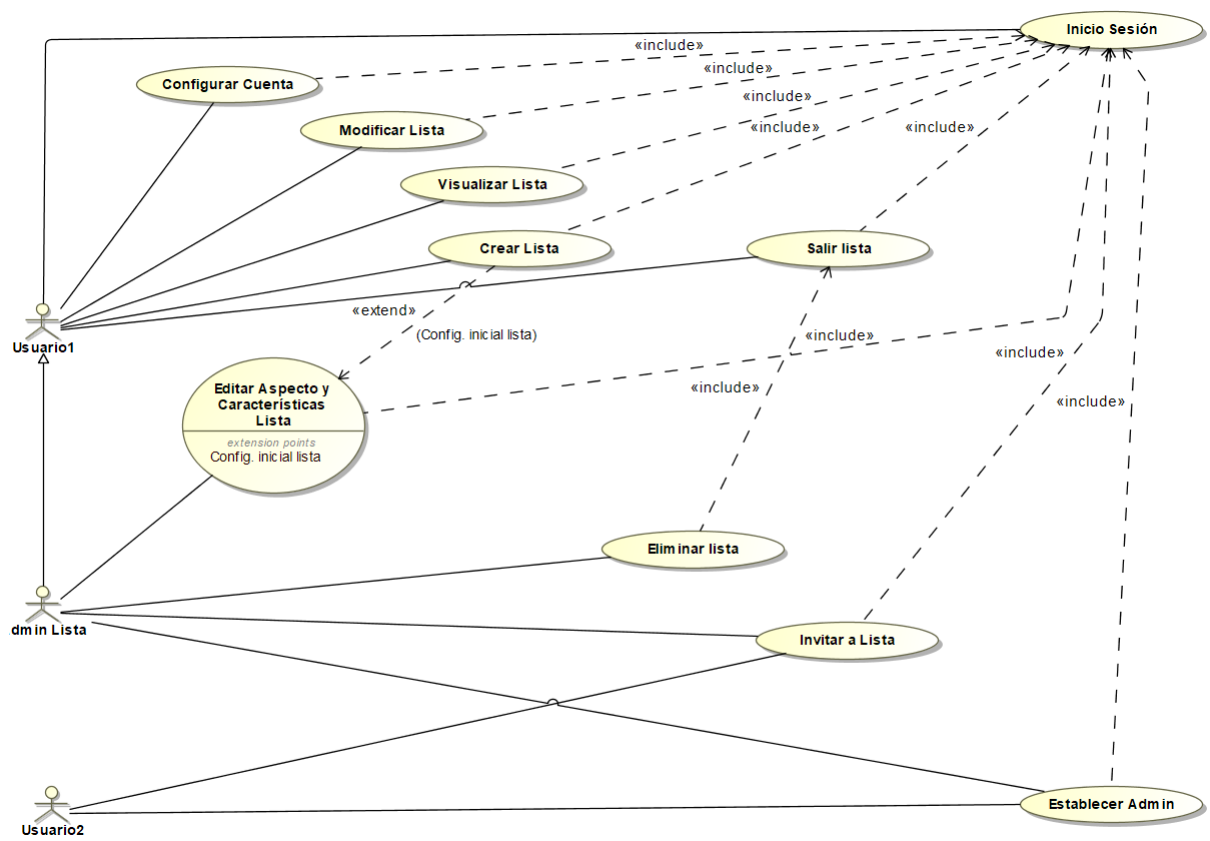
- Identificador único:
CU8. Eliminar Lista
- Contexto de uso:
Se desea eliminar una lista.
- Precondiciones y activación:
Se debe iniciar sesión para eliminar la lista. La lista ya debe existir. Debe ser administrador de la lista.
- Garantías de éxito / Postcondición:
Se elimina la lista deseada tanto para el administrador como para todos los usuarios que estuvieran incluidos como invitados.
- Escenario principal:
 1. Iniciar sesión en MynimaList.
 2. Seleccionar una lista de la cual sea administrador.
 3. Seleccionar el botón de eliminar lista.
 4. El sistema muestra un desplegable.
 5. Seleccionar el botón confirmar borrado.
 6. El sistema borra la lista.
- Escenarios alternativos:
 - 1.b Credenciales de inicio de sesión erróneas.
 - 3.b No es administrador de la lista, luego no aparece dicho botón de eliminar la lista.

CU9: Configurar Cuenta

- Identificador único:
CU9. Configurar Cuenta
- Contexto de uso:
Como usuario, se desea modificar algún aspecto de la cuenta
- Precondiciones y activación:
Tener una cuenta creada e iniciar sesión.
- Garantías de éxito / Postcondición:
Guardar los cambios realizados.
- Escenario principal:
 1. El usuario inicia sesión.
 2. Pulsa el botón de ajustes.
 3. El sistema abre un menú donde el usuario puede visualizar y modificar tanto su usuario como su correo electrónico y contraseña.
 4. El usuario realiza los cambios que desee.
 5. El sistema valida la unicidad del usuario y del correo electrónico, y valida los cambios realizados guardándolos en la base de datos.
- Escenarios alternativos:
 - 1.b Credenciales de inicio de sesión erróneas.
 - 5.b El usuario y/o el correo electrónico modificados ya existen en la base de datos. Luego el sistema vuelve a mostrar las credenciales y muestra un mensaje de error.

CU10: Salir Lista

- Identificador único:
CU10. Salir Lista
- Contexto de uso:
Como usuario de una lista, se desea salir de una lista.
- Precondiciones y activación:
Se debe iniciar sesión y tener acceso a una lista ya existente.
- Garantías de éxito / Postcondición:
Dejar de tener acceso a una lista a la que el usuario ha sido invitado.
- Escenario principal:
 1. Iniciar sesión en MynimaList
 2. Seleccionar una lista.
 3. Seleccionar el botón de salir lista.
 4. El sistema muestra una ventana.
 5. Seleccionar el botón confirmar.
 6. El sistema elimina el usuario de la lista.
- Escenarios alternativos:
 - 1.b Credenciales de inicio de sesión erróneas.



Herramientas software

Herramientas software usadas: grupo de Whatsapp, Github, Trello, Google Docs, Figma, Discord.

- **Herramienta de comunicación**

Las aplicaciones elegidas para la comunicación entre los integrantes del grupo son Whatsapp, donde hemos creado un grupo para coordinarnos, y Discord, donde hemos creado un servidor para poder realizar reuniones a través de videollamadas.

- **Herramienta de trabajo colaborativo**

Las herramientas en línea seleccionadas para la realización del proyecto son Github, donde se ha creado un repositorio al que se irán subiendo los documentos relacionados con el proyecto; Trello, en el que hemos hecho un espacio de trabajo donde registramos las tareas *por hacer*, *en proceso* y *hechas* mediante tarjetas; y Figma, que utilizaremos para probar distintos diseños hasta encontrar el más adecuado para nuestro proyecto.

- **Herramienta de elaboración de documentos**

Para la realización de documentos se ha elegido Google Docs, ya que nos permite realizar cambios al mismo tiempo en un mismo documento.

Diagrama de clases

Las clases que vamos a implementar son las siguientes:

Usuario
<i>Attributes</i> <ul style="list-style-type: none">- nombreUsuario : String- contraseña : String- email : String- listasAdmin : String []
<i>Operations</i> <ul style="list-style-type: none">+ getNombreUsuario() : String+ getContraseña(): String+ getEmail() : String+ setNombreUsuario(nombreUsuario: String) : Void+ setContraseña(contraseña: String) : Void+ setEmail(email: String) : Void+ makeAdmin(listasAdmin: String []) : Void+ isAdmin(codigoLista: String) : Boolean

Usuario representa un usuario en nuestra aplicacion, que tendra capacidad de crear una lista. El usuario si no existe, no puede crear dicha lista. Un usuario puede crear muchas listas.

Lista
<i>Attributes</i> <ul style="list-style-type: none">- nombre : String- codigo: String- usuarios : Usuarios[]
<i>Operations</i> <ul style="list-style-type: none">+ getNombre() : String+ setNombre(nombre: String) : Void+ getCodigo() : String+ deleteLista() : Void+ exitLista(usuarios: Usuario[]) : Void+ getUsuarios() : Usuario[]+ addUsuarios(usuarios: Usuario[]) : Void

La clase Lista representa las diferentes listas que pueden crear los usuarios. Dichas listas se componen de tareas. Una lista que no existe no puede tener tareas. Una lista puede tener ninguna o muchas tareas.

Tarea
<i>Attributes</i> <ul style="list-style-type: none">- texto : String- codigo: String
<i>Operations</i> <ul style="list-style-type: none">+ getTexto() : String


```
+ setTexto(texto: String) : Void
+ getCodigo() : String
+ deleteTarea() : Void
```

La clase tarea representa una cadena de caracteres que compone a cada tarea.

Estas clases las hemos representado en un diagrama de clases en MagicDraw, con las relaciones explicadas anteriormente:

