



Análisis y Diseño de Sistemas de Información

Ingeniería Informática de Gestión y Sistemas de Información

Biblioteca

Autores:

Xabier Gabiña Barañano
Janire Veganzones García
Ainhize Martinez Duran
Javier Criado Garcia
Kepa Reches Urrutia
Jon Valdes Granado
Mohamed El Basri Dehbi

Índice general

1. Changelog	3
1.1. Versión 1 (2023-10-08)	3
1.2. Versión 2 (2023-10-22)	3
1.3. Versión 3 (2023-11-05)	3
1.4. Versión 4 (2023-11-15)	3
1.5. Versión 5 (2024-01-08)	3
2. Introducción	4
3. Casos de Uso	5
3.1. General	5
3.2. Extendidos	6
3.2.1. Gestión de reservas	6
3.2.2. Reseñas	9
3.2.3. Red de amigos	12
3.2.4. Administrador	15
3.2.5. Foros	23
3.2.6. Recomendaciones del sistema	28
3.2.7. Recomendaciones de amigos	29
4. Modelo de Dominio	30
4.1. Diagrama de modelo de dominio	30
4.2. Desarrollo del modelo de dominio	31
4.2.1. Entidades	31
4.2.2. Relaciones	32
5. Plan de Pruebas	39
5.1. Gestión de reservas	39
5.2. Reseñas	40
5.3. Red de amigos	41
5.4. Administrador	42
5.5. Foros	44
5.6. Recomendaciones del sistema	45
5.7. Recomendaciones de amigos	46
6. Diagrama relacional	48
7. Diagrama de clases	49

8. Diagramas de comunicación	50
8.1. Gestión de reservas	50
8.2. Reseñas	51
8.3. Red de amigos	54
8.4. Administrador	55
8.5. Foros	60
8.6. Recomendaciones del sistema	61
8.7. Recomendaciones de amigos	63
9. Diagrama de secuencias	65
9.1. Gestión de reservas	65
9.2. Reseñas	66
9.3. Red de amigos	67
9.4. Administrador	68
9.5. Foros	73
9.6. Recomendaciones del sistema	74
9.7. Recomendaciones de amigos	75
10. Problemas de implementación	76
10.1. Gestión de reservas	76
10.2. Reseñas	76
10.3. Red de amigos	76
10.4. Administrador	76
10.5. Foros	76
10.6. Recomendaciones del sistema	76
10.7. Recomendaciones de amigos	76
11. Conclusiones	78
12. Repositorio	79

Changelog

1.1. Versión 1 (2023-10-08)

- Añadido caso de uso general.
- Añadidos los casos de uso extendidos.

1.2. Versión 2 (2023-10-22)

- Añadido modelo de dominio.
- Modificado el caso de uso general.
- Modificados los casos de uso extendidos.

1.3. Versión 3 (2023-11-05)

- Añadido plan de Pruebas.
- Modificado el modelo de dominio.

1.4. Versión 4 (2023-11-15)

- Añadido diagrama relacional.
- Añadido diagrama de clases.
- Añadido diagrama de comunicación.

1.5. Versión 5 (2024-01-08)

- Añadido diagrama de secuencia.
- Añadido conclusiones.
- Correcciones varias.

Introducción

En un mundo cada vez más digitalizado, las bibliotecas modernas están buscando formas innovadoras de adaptarse a las necesidades cambiantes de sus usuarios. La biblioteca que nos ocupa se encuentra en este proceso de transformación, reconociendo la importancia de expandir sus servicios en línea y brindar a sus usuarios una experiencia más enriquecedora y conectada. Para llevar a cabo esta ambiciosa misión, han solicitado la colaboración de nuestro equipo de desarrollo.

En su estado actual, la biblioteca cuenta con un servicio web que permite a los usuarios consultar su extenso catálogo de libros. Además, ofrece un apartado de usuarios que pueden acceder a una sección privada. Sin embargo, esta plataforma carece de funcionalidades adicionales que pueden mejorar significativamente la experiencia de los usuarios y promover una mayor interacción entre ellos.

En este proyecto, nuestro objetivo es expandir y mejorar el servicio web de la biblioteca, dotándolo de una serie de funcionalidades clave. Estas nuevas características incluyen la gestión de reservas, la capacidad de dejar reseñas y comentarios en los libros, la creación de una red de amigos entre los usuarios, la introducción de un rol de administrador con capacidades de gestión, la implementación de foros para discusión y la incorporación de sistemas de recomendación, tanto basados en el historial de lecturas como en la afinidad de intereses entre usuarios.

A lo largo de este documento, exploraremos en detalle cada una de estas funcionalidades, presentando soluciones técnicas y estratégicas para su implementación. Este proyecto representa una emocionante oportunidad para ayudar a la biblioteca a evolucionar y brindar un servicio en línea más completo y enriquecedor a su comunidad de usuarios ávidos de conocimiento y lectura.

Casos de Uso

3.1. General

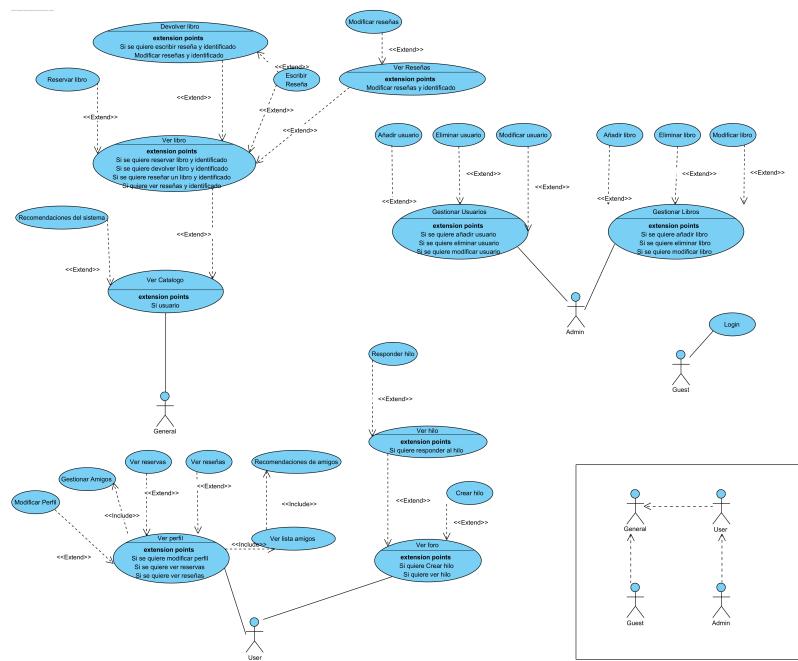


Figura 3.1: Diagrama de casos de uso general

3.2. Extendidos

3.2.1. Gestión de reservas

Responsable: Mohamed El Basri
<p>The diagram illustrates an extension mechanism. A general actor (User) interacts with a use case 'Ver catálogo'. This use case has an extension point labeled 'Si usuario'. Another use case, 'Ver libro', extends this point. 'Ver libro' also has its own extension point labeled 'Si se quiere reservar libro e identificado'. A third use case, 'Reservar libro', extends this second point. Dashed arrows indicate the extension relationships between the use cases.</p>
Nombre: Ver libros recomendados
Descripción: Permite a los usuarios realizar reservas de libros.
Actores: Usuario
Precondiciones: Estar identificado en el sistema.
Requisitos no funcionales: Ninguno
Flujo de Eventos: <ol style="list-style-type: none">El usuario selecciona el libro deseado en el catálogo y pulsa "Reservar". Se abre la interfaz de reserva (Figura 3.2.1.2).Si el usuario pulsa aceptar, si el tiempo de reserva es menor a 2 meses aparece una interfaz indicando que la reserva se ha realizado con éxito (Figura 3.2.1.3).<ol style="list-style-type: none">Si el usuario pulsa aceptar, si el tiempo de reserva es mayor a 2 meses aparece una interfaz indicando error (Figura 3.2.1.4).Si el usuario pulsa cancelar, se cierra la interfaz.
Poscondiciones: Ninguna
Interfaz Gráfica:



Figura 3.2.1.2: Interfaz de reserva



Figura 3.2.1.3: Interfaz de reserva con éxito

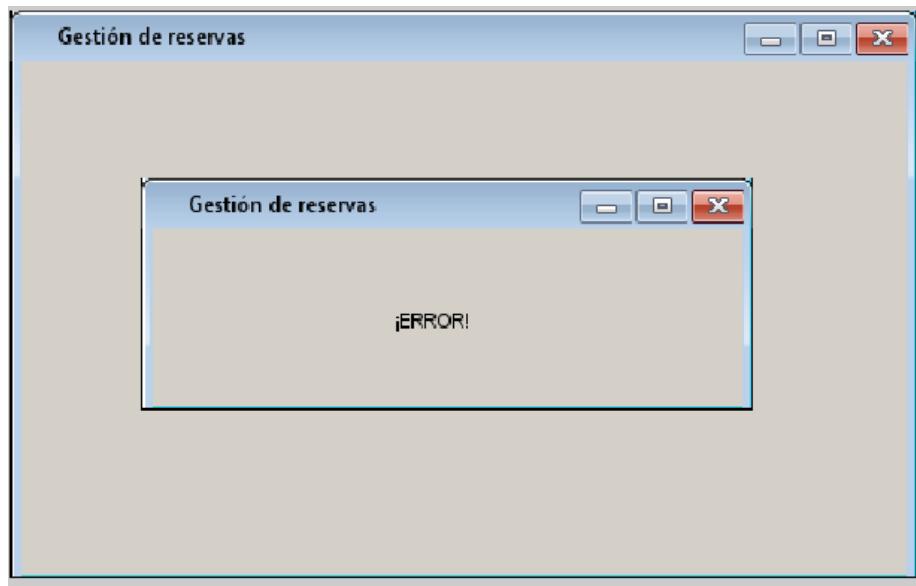


Figura 3.2.1.4: Interfaz de reserva con error

3.2.2. Reseñas

Responsable: Javier Criado
<pre> graph LR Actor[General] --> VerCatalogo([Ver Catalogo extension points Si usuario]) VerCatalogo --<<Extend>>--> EscribirReseña([Escribir Reseña]) VerCatalogo --<<Extend>>--> DevolverLibro([Devolver libro extension points Si se quiere escribir una reseña y identificado Modificar reseñas y identificado]) VerCatalogo --<<Extend>>--> ModificarResenas([Modificar reseñas extension points Si se quiere devolver libro y identificado Si se quiere reservar libro y identificado Si se quiere reseñar un libro y identificado Si quiere ver reseñas y identificado]) </pre>
Nombre: Reseñas Descripción: Permite al cliente puntuar y comentar en un libro una vez lo haya devuelto, también permite modificar estos datos posteriormente. Actores: Usuario Precondiciones: Estar identificado en el sistema Requisitos no funcionales: Ninguno
Flujo de Eventos: <ol style="list-style-type: none"> 1. El usuario entra en el catalogo para ver los libros disponibles. a) El usuario selecciona uno de los libros. [Figura 3.2.2.2] <ol style="list-style-type: none"> 1) Al seleccionar 'Devolver libro', se devolvera el libro y se podra añadir una nueva reseña. 2) Al seleccionar 'Escribir reseña', se abrirá una pestaña donde se podrá escribir una reseña sobre el libro. [Figura 3.2.2.3] 3) Al seleccionar el botón 'Ver reseñas', el usuario podrá ver las reseñas sobre el libro. [Figura 3.2.2.4] <ol style="list-style-type: none"> a' Al hacer click sobre una de las reseñas esta se mostrara en una ventana aparte. [Figura 3.2.2.5]
Poscondiciones: Ninguna Interfaz Gráfica:

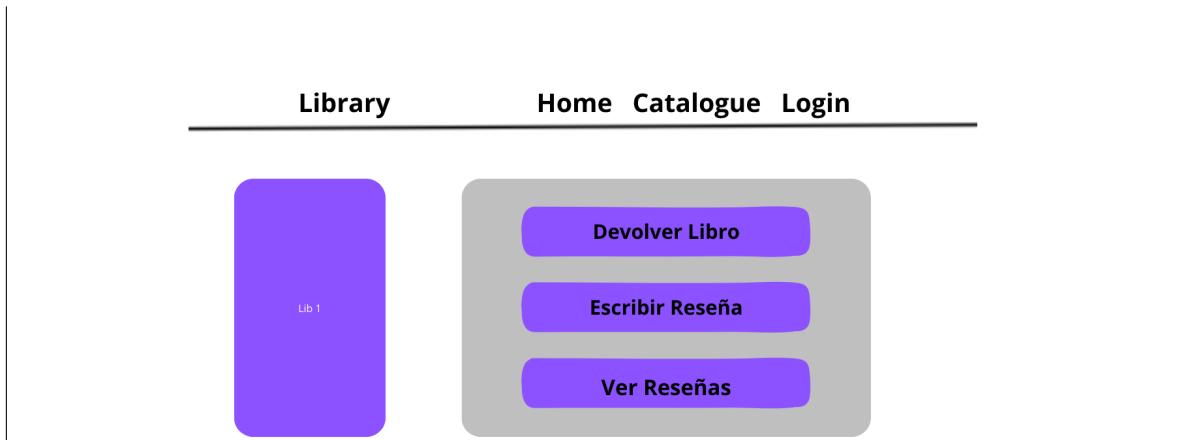


Figura 3.2.2.2: Interfaz de ver libro.

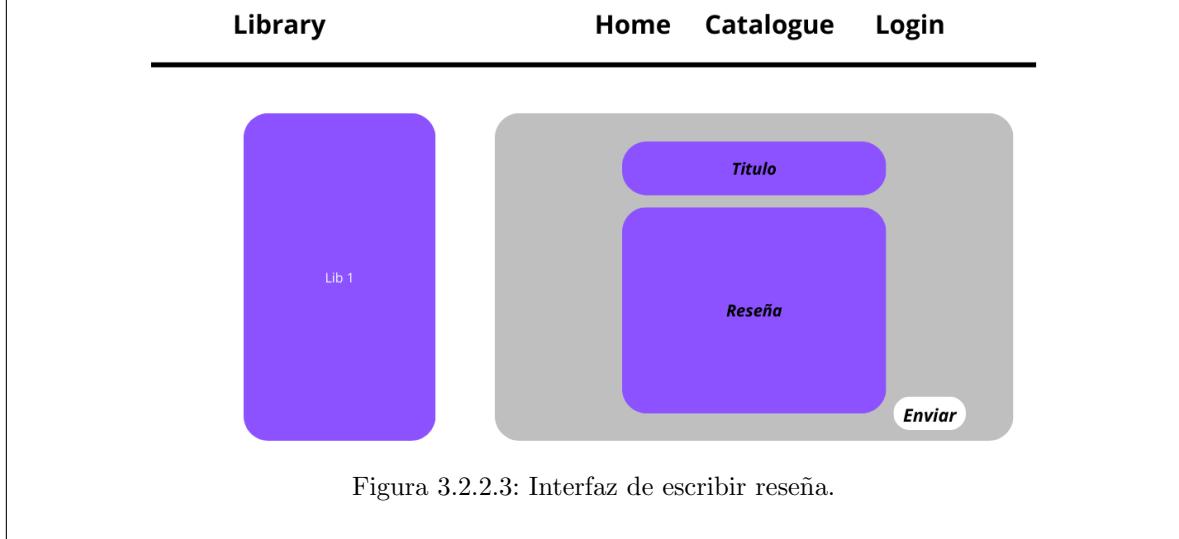


Figura 3.2.2.3: Interfaz de escribir reseña.

Library

Home Catalogue Login

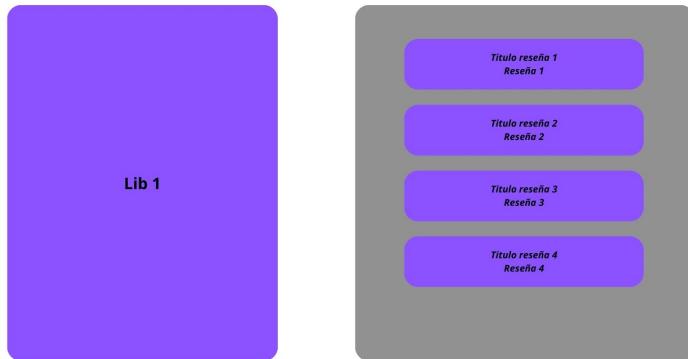


Figura 3.2.2.4: Interfaz para ver las reseñas.

Library

Home Catalogue Login

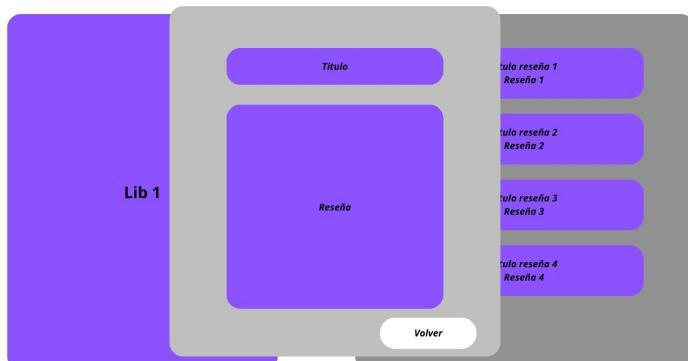


Figura 3.2.2.5: Interfaz para ver una reseña.

3.2.3. Red de amigos

Responsable: Jon Valdes
<pre> classDiagram actor User usecase VerPerfil { <<extension points>> Si se quiere modificar perfil Si se quiere ver reservas Si se quiere ver reseñas } usecase ModificarPerfil usecase GestionarAmigos usecase VerReservas usecase VerReseñas usecase VerListaAmigos User --> VerPerfil : VerPerfil --> <<Extend>> ModificarPerfil VerPerfil --> <<Extend>> GestionarAmigos VerPerfil --> <<Extend>> VerReservas VerPerfil --> <<Extend>> VerReseñas VerPerfil --> <<Include>> VerListaAmigos </pre>
Figura 3.2.3.1: Diagrama de casos de uso de la subfuncionalidad de red de amigos
Nombre: Red de amigos.
Descripción: Permite al usuario gestionar su perfil, de modo que puede tanto consultar el historial de reservas y de reseñas, como añadir o eliminar amigos. También puede editar sus datos personales.
Actores: Usuario.
Precondiciones: Estar identificado en el sistema.
Requisitos no funcionales: Ninguno
Flujo de Eventos: <ol style="list-style-type: none"> 1. El usuario accede a su perfil. [Figura 3.2.3.2] <ol style="list-style-type: none"> a) El usuario puede pulsar 'Editar perfil' para editar sus datos. [Figura 3.2.3.5] b) El usuario puede pulsar 'Consultar reservas' para ver su historial de reservas. c) El usuario puede pulsar 'Consultar reseñas' para ver su historial de reseñas. d) El usuario puede ver su lista de amigos. e) El usuario puede pulsar 'Añadir amigo' para añadir un amigo. [Figura 3.2.3.3] f) El usuario puede pulsar 'Solicitudes de amistad' para ver las solicitudes de amistad pendientes. g) El usuario puede pulsar 'Eliminar amigo' para eliminar un amigo. [Figura 3.2.3.4]
Poscondiciones: Ninguna
Interfaz Gráfica:

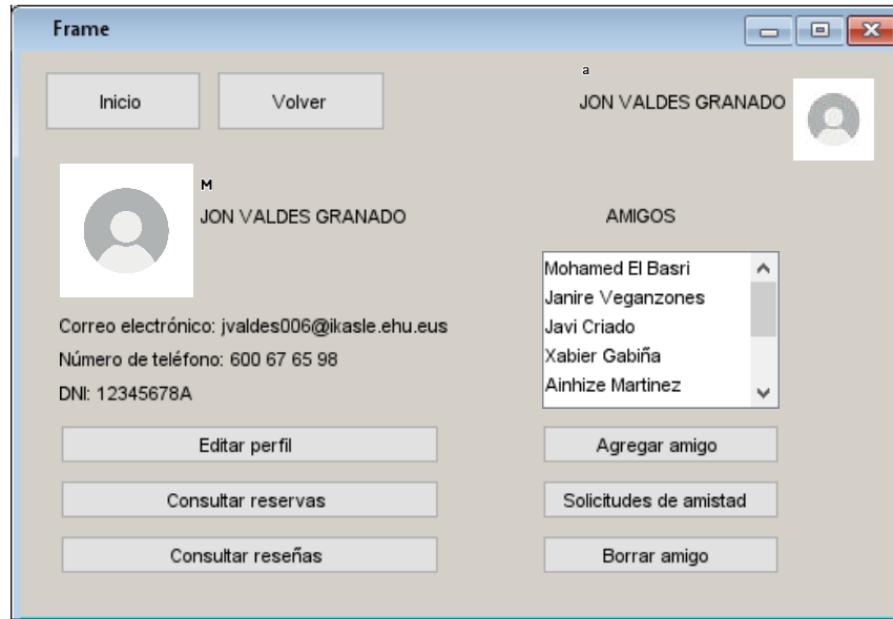


Figura 3.2.3.2: Interfaz del perfil de usuario.

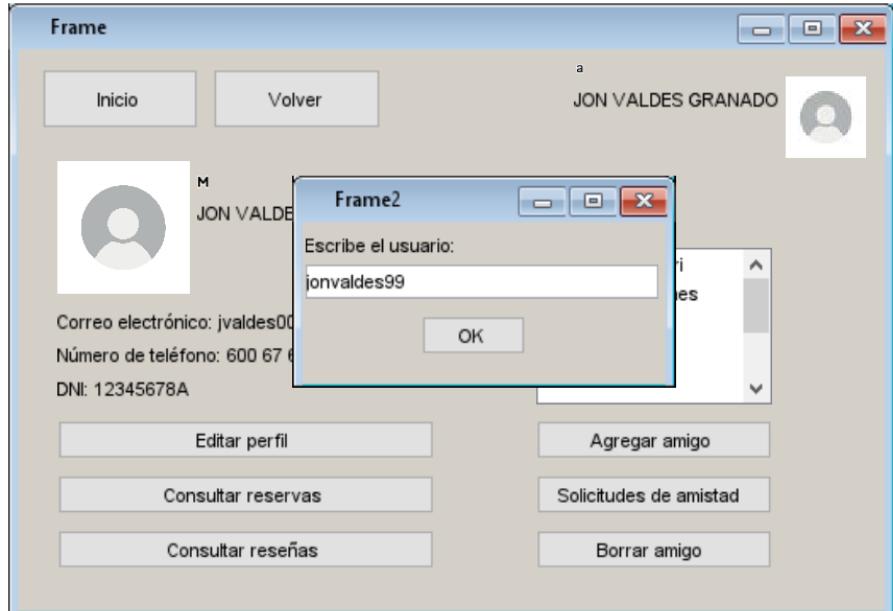


Figura 3.2.3.3: Interfaz para añadir un amigo.

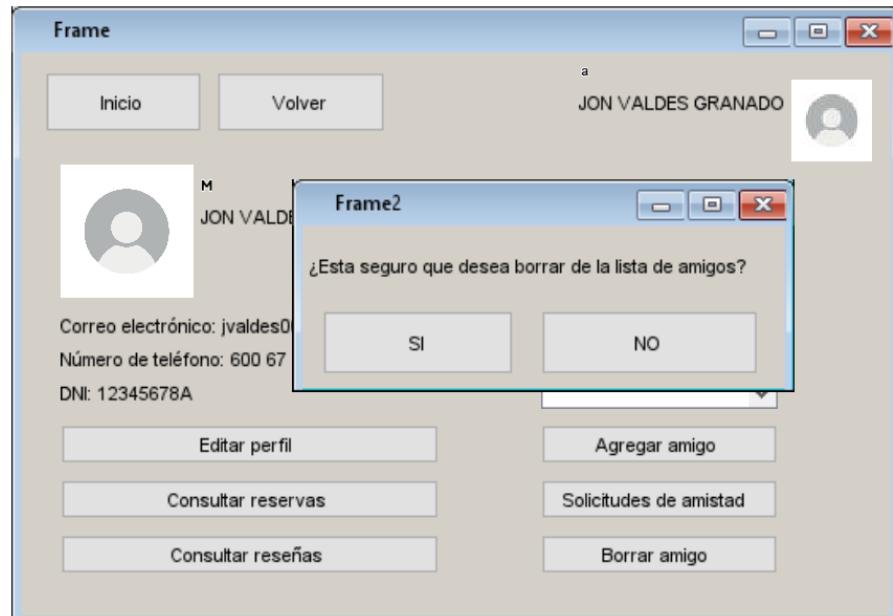


Figura 3.2.3.4: Interfaz de alerta para borrar un amigo.



Figura 3.2.3.5: Interfaz para editar el perfil.

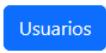
3.2.4. Administrador

Gestión de usuarios

Responsable: Janire Veganzones
<pre> graph TD AU1[Añadir usuario] -- "<<Extend>>" --> GU[Gestionar Usuarios] AU2[Eliminar usuario] -- "<<Extend>>" --> GU AU3[Modificar usuario] -- "<<Extend>>" --> GU GU -- "extension points
Si se quiere añadir usuario
Si se quiere eliminar usuario
Si se quiere modificar usuario" --> Admin[Admin] </pre>
Figura 3.2.4.1.1: Diagrama de casos de uso de la subfuncionalidad de gestión de usuarios
Nombre: Gestionar usuarios
Descripción: Permite crear, eliminar o modificar un usuario en el sistema.
Actores: Admin
Precondiciones: Estar identificado en el sistema y ser administrador.
Requisitos no funcionales: Ninguno
Flujo de Eventos:
<ol style="list-style-type: none"> 1. Se accede al menu de administrador. [Figura 3.2.4.1.2] <ol style="list-style-type: none"> a) Se selecciona la opción de gestionar usuarios. [Figura 3.2.4.1.3] <ol style="list-style-type: none"> 1) El administrador pulsa 'Crear' para crear un nuevo usuario. [Figura 3.2.4.1.4] <ol style="list-style-type: none"> a' Si hay un error al crear un usuario saldra un error mostrandolo. [Figura 3.2.4.1.5] 2) El administrador pulsa 'Borrar' para borrar un usuario. [Figura 3.2.4.1.6] <ol style="list-style-type: none"> a' Si el usuario a borrar no existe mostrara un error. [Figura 3.2.4.1.7] 3) El administrador pulsa 'Modificar' para modificar un usuario. [Figura 3.2.4.1.8] <ol style="list-style-type: none"> a' Si hay un error al modificar un usuario saldra un error mostrandolo. [Figura 3.2.4.1.9]
Poscondiciones: Ninguna
Interfaz Gráfica:

Library Home Catalogue Menu Admin Administrador 

Opciones de administrador

 Usuarios

 Libros

Figura 3.2.4.1.2: Interfaz del menú de administrador.

Library Home Catalogue Menu Admin Administrador 

Gestor de usuarios

 Crear

 Borrar

Figura 3.2.4.1.3: Interfaz del menu de gestión de usuarios.

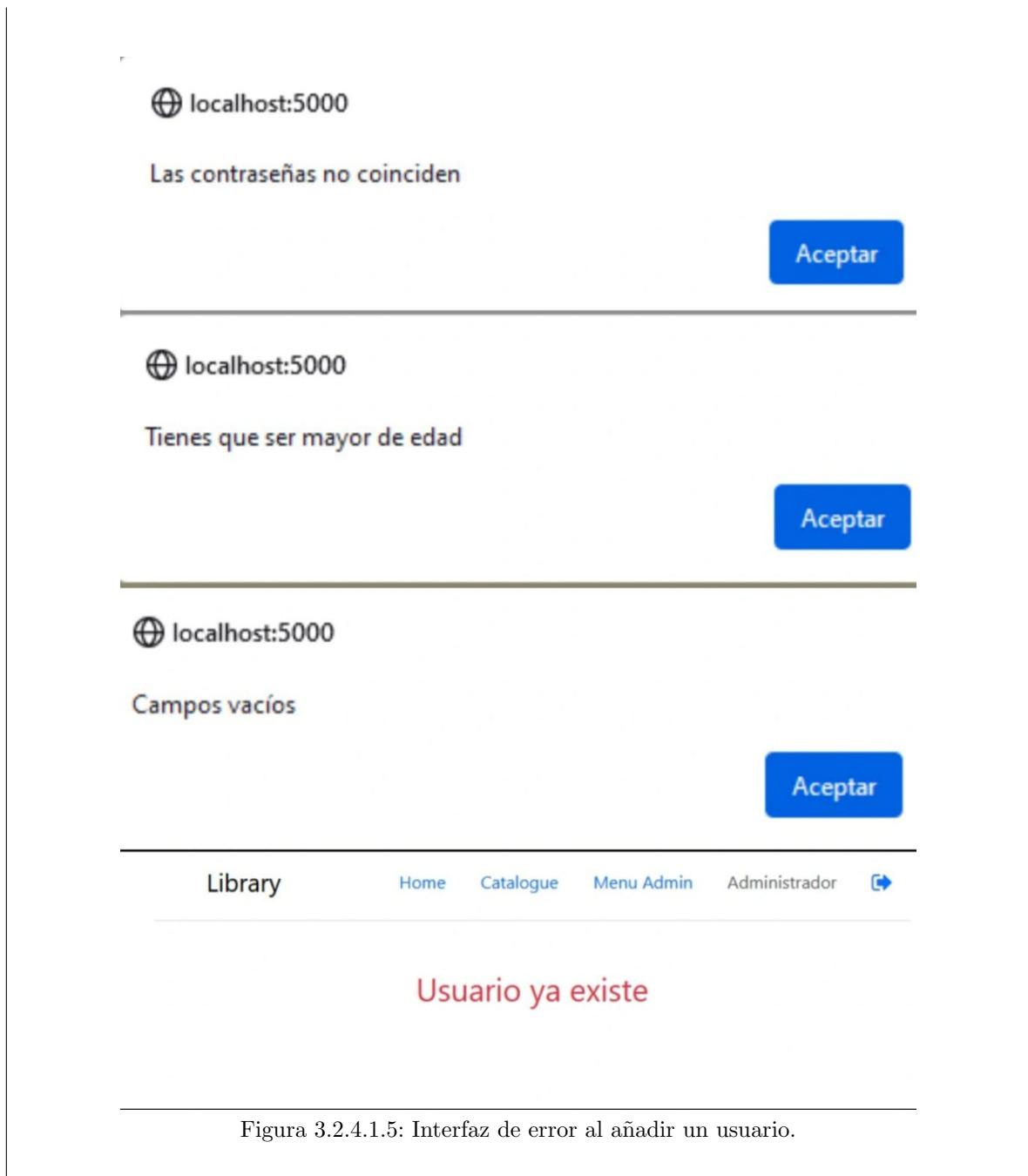
Library Home Catalogue Menu Admin Administrador 

Crear Usuario

Nombre	<input type="text" value="Nombre"/>
Apellidos	<input type="text" value="Apellidos"/>
Fecha de Nacimiento	<input type="text" value="dd / mm / aaaa"/> 
Email	<input type="text" value="Email"/>
Contraseña	<input type="text" value="Contraseña"/>
Repetir Contraseña	<input type="text" value="Repetir Contraseña"/>

 Crear

Figura 3.2.4.1.4: Interfaz para añadir un usuario.



Borrar Usuarios

Email

Email

Figura 3.2.4.1.6: Interfaz para borrar un usuario.

El email no existe o es admin

Figura 3.2.4.1.7: Interfaz de error al borrar un usuario.

Gestión de libros

Responsable: Janire Veganzones
<pre> graph TD A[Añadir libro] -- "<<Extend>>" --> B[Gestionar Libros] B -- "<<Extend>>" --> C[Eliminar libro] B -- "<<Extend>>" --> D[Modificar libro] subgraph ExtensionPoints [extension points] B SiA[Si se quiere añadir libro] SiE[Si se quiere eliminar libro] SiM[Si se quiere modificar libro] end Admin --- B </pre> <p>The diagram illustrates the 'Gestionar Libros' use case with three extension points: 'Si se quiere añadir libro', 'Si se quiere eliminar libro', and 'Si se quiere modificar libro'. These points are associated with the 'Gestionar Libros' use case via '<<Extend>>' relationships. The 'Admin' actor is shown interacting with the 'Gestionar Libros' use case.</p>
<p>Figura 3.2.4.2.1: Diagrama de casos de uso de la subfuncionalidad de gestión de libros</p>
<p>Nombre: Gestionar libros</p>
<p>Descripción: Permite añadir, eliminar o modificar un libro en el sistema.</p>
<p>Actores: Admin</p>
<p>Precondiciones: Estar identificado en el sistema y ser usuario.</p>
<p>Requisitos no funcionales: Ninguno</p>
<p>Flujo de Eventos:</p> <ol style="list-style-type: none"> 1. Se accede al menu de administrador. [Figura 3.2.4.2.2] a) Se selecciona la opción de gestionar libros. [Figura 3.2.4.2.3] <ol style="list-style-type: none"> 1) El administrador pulsa 'Crear' para crear un nuevo libro. [Figura 3.2.4.2.4] <ol style="list-style-type: none"> a' Si hay un error al crear un libro saldra un error mostrandolo. [Figura 3.2.4.2.5] 2) El administrador pulsa 'Borrar' para borrar un libro. [Figura 3.2.4.2.6] <ol style="list-style-type: none"> a' Si el libro a borrar no existe mostrara un error. [Figura 3.2.4.2.7] 3) El administrador pulsa 'Modificar' para modificar un libro. [Figura 3.2.4.2.8] <ol style="list-style-type: none"> a' Si hay un error al modificar un libro saldra un error mostrandolo. [Figura 3.2.4.2.9]
<p>Poscondiciones: Ninguna</p>
<p>Interfaz Gráfica:</p>

Library Home Catalogue Menu Admin Administrador 

Opciones de administrador

 Usuarios

 Libros

Figura 3.2.4.2.2: Interfaz del menú de administrador.

Library Home Catalogue Menu Admin Administrador 

Gestor de libros

 Añadir

 Borrar

Figura 3.2.4.2.3: Interfaz del menu de gestión de libros.

Añadir Libros

Título	<input type="text" value="Título"/>
Autor	<input type="text" value="Autor"/>
Cover	<input type="text" value="Portada"/>
Descripción	<input type="text" value="Descripción"/>

[Añadir](#)

Figura 3.2.4.2.4: Interfaz para añadir un libro.

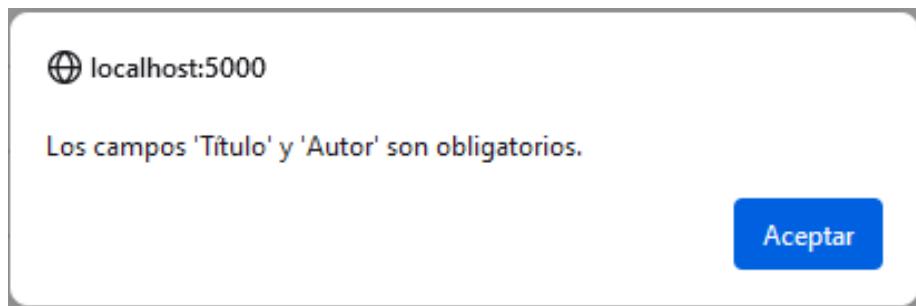


Figura 3.2.4.2.5: Interfaz de error al añadir un libro.

Borrar Usuarios

Título	<input type="text" value="Título"/>
Autor	<input type="text" value="Autor"/>

Borrar

Figura 3.2.4.2.6: Interfaz para borrar un libro.

localhost:5000

Faltan campos por rellenar

Aceptar

El libro no existe

Figura 3.2.4.2.7: Interfaz de error al borrar un libro.

3.2.5. Foros

Responsable: Ainhize Martinez
<pre> graph TD User --- VerHilo[Ver hilo] User --- VerForo[Ver foro] VerHilo -- "Si quiere responder hilo" --> ResponderHilo[Responder hilo] VerForo -- "Si quiere crear hillo Si quiere ver hilo" --> CrearHilo[Crear hilo] ResponderHilo -.-> VerHilo VerForo -.-> CrearHilo </pre> <p>The diagram illustrates the use cases for forum subfunctionalities. It starts with a User actor on the left. Two use cases, 'Ver hilo' and 'Ver foro', are connected to the User. 'Ver hilo' has an extension point 'Si quiere responder hilo' leading to the use case 'Responder hilo'. 'Ver foro' has two extension points: 'Si quiere crear hillo' and 'Si quiere ver hilo', both leading to the use case 'Crear hilo'. There are dashed arrows labeled '<<Extend>>' indicating the extension relationship between the base use cases and their respective extended use cases.</p>
Figura 3.2.5.1: Diagrama de casos de uso de la subfuncionalidad de foros
Nombre: Ver Foros Descripción: Permite al usuario acceder al foro, crear hilos nuevos, explorar todos los hilos, además de contestar hilos. Actores: Usuario Precondiciones: Estar identificado en el sistema. Requisitos no funcionales: Ninguno
Flujo de Eventos: <ol style="list-style-type: none"> 1. El usuario selecciona 'Foro' para acceder a él. <ol style="list-style-type: none"> a) Se le muestra toda la lista de hilos. [Figura 3.2.5.2] <ol style="list-style-type: none"> 1) Si el usuario pulsa en '+Hilo' se le abre una ventana para crear un hilo nuevo. [Figura 3.2.5.3] 2) Si el usuario selecciona un hilo se le mostraran las respuesta al hilo. [Figura 3.2.5.4] <ol style="list-style-type: none"> a' Si el usuario pulsa en '+Mensaje' se le abre una ventana para responder al hilo [Figura 3.2.5.5]
Poscondiciones: Ninguna Interfaz Gráfica:

[Library](#)

[Home](#)

[Catalogue](#)

[Foro](#)

[Login](#)

+ HILO

Nombre	Participantes	Mensajes
Hilo1	20	200
Hilo2	20	200
Hilo3	20	200
Hilo4	20	200
Hilo5	20	200
Hilo6	20	200
Hilo7	20	200

Figura 3.2.5.2: Interfaz del foro.

Library Home Catalogue **Foro** Login

Nombre:

Descripcion:

Primer mensaje:

Cancelar

+ HILO

Figura 3.2.5.3: Interfaz para crear un hilo del foro.

Library

Home

Catalogue

Foro

Login



Hilo1

Ainhize



Javi



+MENSAJE

Figura 3.2.5.4: Interfaz para ver un hilo del foro.

[Library](#)

[Home](#)

[Catalogue](#)

[Foro](#)

[Login](#)

◀ **Hilo1**

 Ainhize



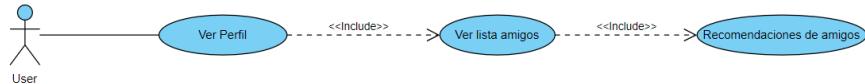
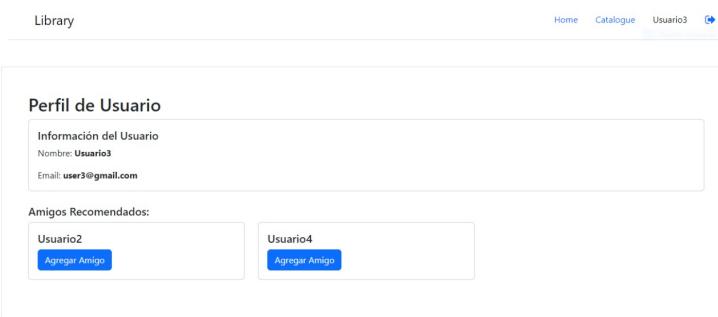
Enviar

Figura 3.2.5.5: Interfaz para responder a un hilo.

3.2.6. Recomendaciones del sistema

Responsable: Xabier Gabiña
<pre> classDiagram actor General useCase VerCatalogo useCase Recomendaciones VerCatalogo --> extensionPoint SiUsuario extensionPoint <<--Extend-->> Recomendaciones </pre>
Figura 3.2.6.1: Diagrama de casos de uso de la subfuncionalidad de recomendaciones de libros
Nombre: Recomendaciones del sistema Descripción: Muestra una lista en la parte superior del catálogo mostrando los libros con más préstamos del sistema en función de los gustos del usuario. Actores: Usuario Precondiciones: Estar identificado en el sistema y haber tenido un libro reservado Requisitos no funcionales: Ninguno Flujo de Eventos: 1. El usuario entra en el catalogo para ver los libros disponibles. a) Se le muestra en la parte superior de la pantalla una serie de libros basadas en los generos y peliculas que consume el usuario. [Figura 3.2.6.2]
Poscondiciones: Ninguna Interfaz Gráfica:
Figura 3.2.6.2: Interfaz de recomendaciones de libros.

3.2.7. Recomendaciones de amigos

Responsable: Kepa Reche

Figura 3.2.7.1: Diagrama de casos de uso de la subfuncionalidad de recomendaciones de amigos
Nombre: Ver amigos recomendados Descripción: Muestra una lista en la parte superior de la lista de amigos mostrando posibles nuevos amigos basados en amigos de tus amigos o personas con las que tengas lecturas relacionadas. Actores: Usuario Precondiciones: Estar identificado en el sistema y tener algun amigo o libro guardado en tu perfil Requisitos no funcionales: Ninguno
Flujo de Eventos: <ol style="list-style-type: none">1. El Usuario entra en su perfil.<ol style="list-style-type: none">a) Se le muestra en la parte superior de la lista de amigos varias personas recomendadas segun las lecturas en comun y amigos de amigos. [Figura 3.2.7.2]
Poscondiciones: Ninguna
Interfaz Gráfica: 
Figura 3.2.7.2: Interfaz de recomendaciones de amigos.

Modelo de Dominio

4.1. Diagrama de modelo de dominio

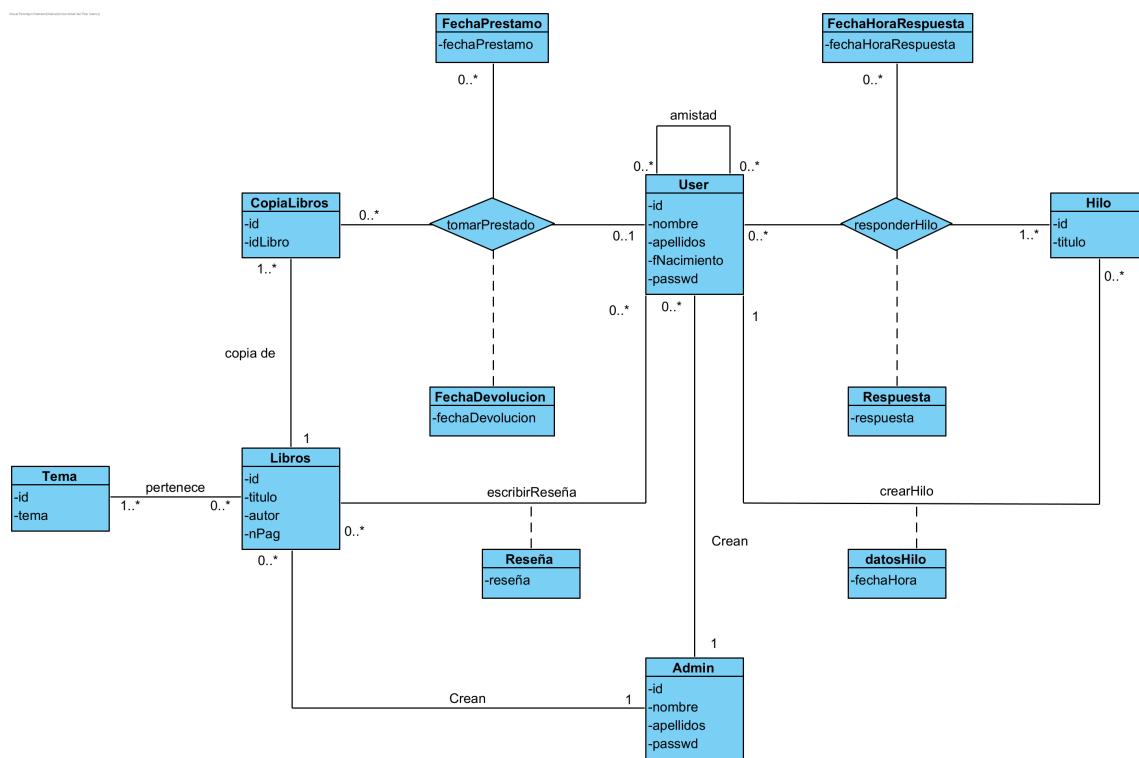


Figura 4.1: Diagrama de modelo de dominio

4.2. Desarrollo del modelo de dominio

4.2.1. Entidades

User

La entidad usuario representa a los usuarios de la aplicación. Dentro almacena sus datos personales como su id, nombre, apellidos, fecha de nacimiento...

Admin

La entidad admin representa a los administradores de la aplicación. Dentro almacena su id, nombre, apellidos y su contraseña.

Libros

La entidad libro representa a los libros de la aplicación. Dentro almacena sus datos como su id, título, autor, número de páginas y género.

CopiaLibros

La entidad copiaLibros representa a las copias de los libros que hay en la aplicación. Dentro almacena su id y el id del libro al que pertenece.

Tema

La entidad tema representa a los temas de los libros. Dentro almacena su id y el nombre del tema.

FechaPrestamo

La entidad fechaPrestamo representa a las fechas de préstamo de los libros. Sirve como clave para diferenciar los diferentes préstamos de un libro que puede hacer un usuario.

Hilo

La entidad hilo representa a los hilos del foro. Dentro almacena la id del hilo y un título.

FechaHoraRespuestas

La entidad fechaHoraRespuesta representa la fecha y hora en la que alguien ha dado una respuesta en el foro y sirve como clave para diferenciar del resto de respuestas que pueda dar un mismo usuario en un mismo hilo.

4.2.2. Relaciones

tomarPrestado

Esta relación es la que almacena los datos de los libros que se han tomado prestados y lo usuarios que los han tomado. La entidad fechaPrestamo es la encargada de diferenciar las dos claves de usuario y libro y permitir así que un usuario pueda tomar un mismo libro mas de una vez. El atributo de FechaDevolucion únicamente almacena como dato en la relación la fecha en la que se devuelve el libro.

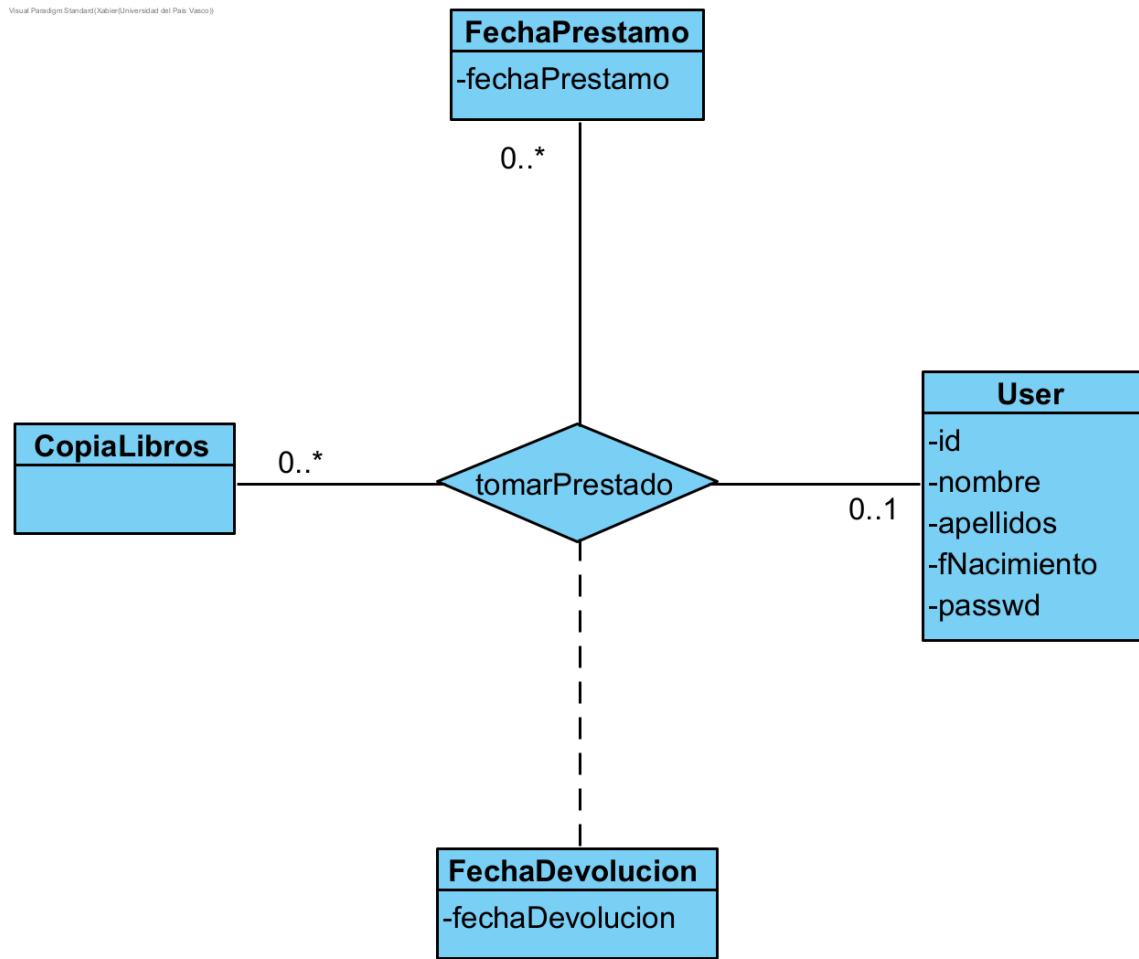


Figura 4.2: Diagrama de modelo de dominio de la relación tomarPrestado

escribirReseña

Esta relación es la que permite guardar la reseña que un usuario pueda escribir en un libro. Al solo tener el atributo Reseña el usuario solo puede escribir una única reseña sobre un libro.

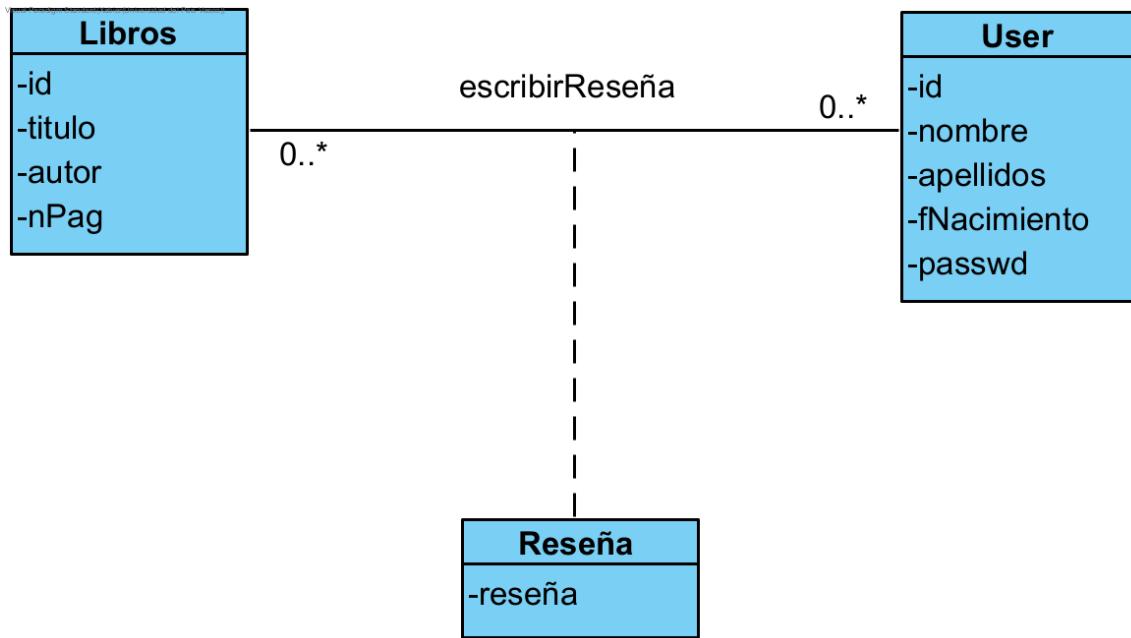


Figura 4.3: Diagrama de modelo de dominio de la relación escribirReseña

amistad

La relacion amistad es la que permite que los usuarios puedan tener amigos en el sistema y guardar dichas amistades.

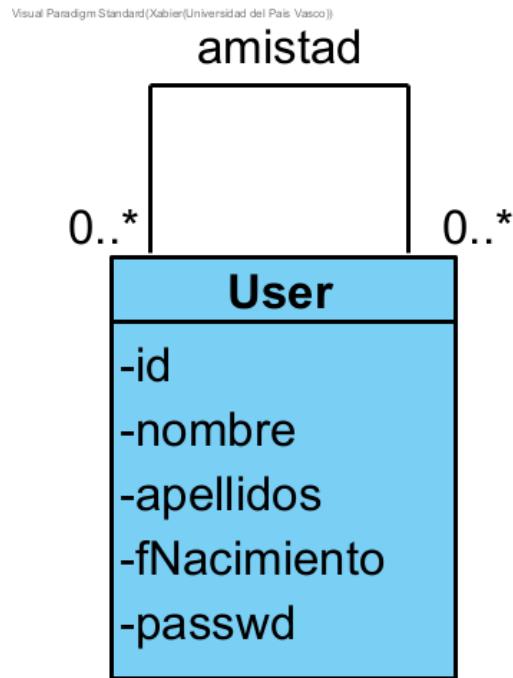


Figura 4.4: Diagrama de modelo de dominio de la relación amistad

responderHilo

Esta relación permite que los usuarios puedan responder a los hilos del foro. La entidad FechaHoraRespuesta es la encargada de diferenciar las respuestas de un mismo usuario en un mismo hilo.

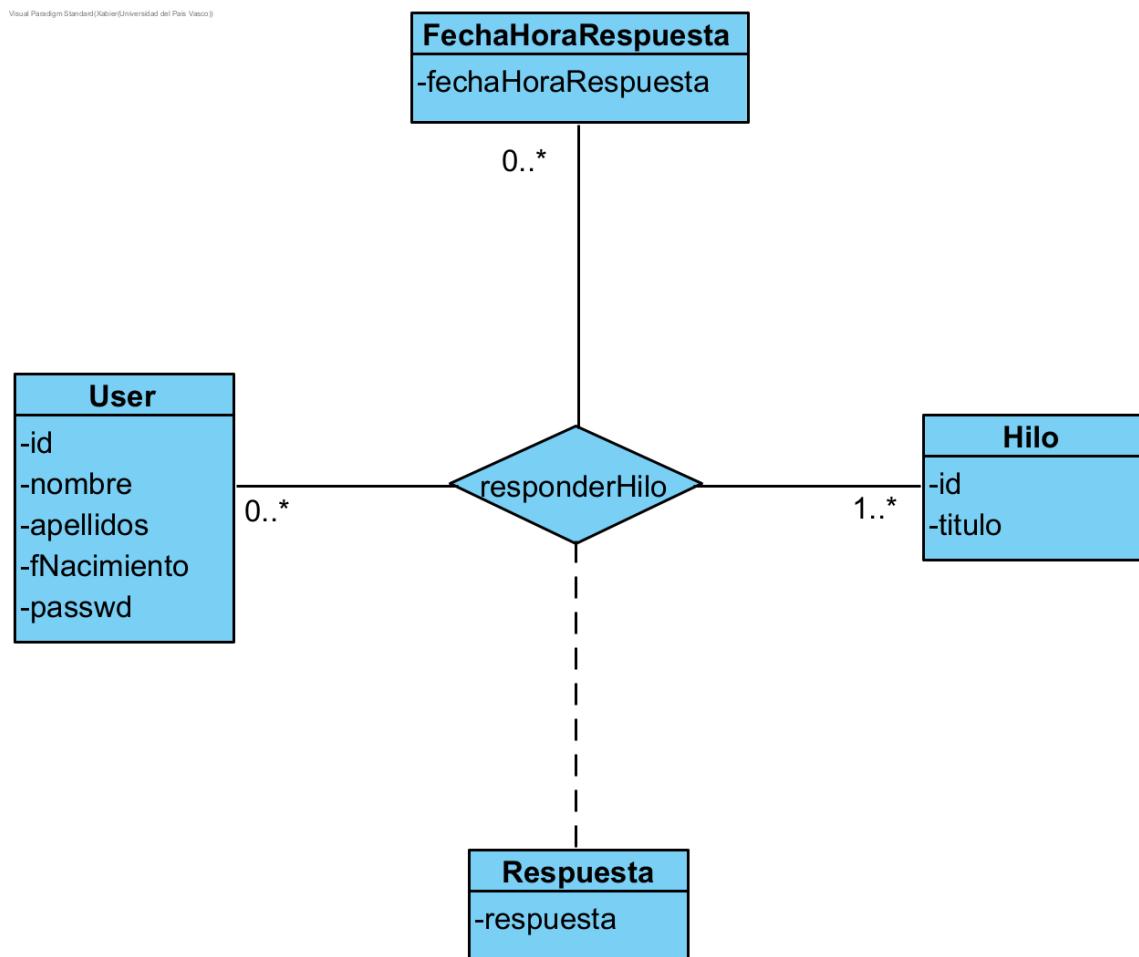


Figura 4.5: Diagrama de modelo de dominio de la relación responderHilo

crearHilo

Esta relacion almacena los datos de creacion de un hilo en el sistema. El sistema guardara el usuario que crea el hilo y en que fecha lo crea. Los hilos son unicos y para evitar repeticiones hacemos que sean unicos por lo que no guardamos la fecha como entidad si no como atributo de la relacion.

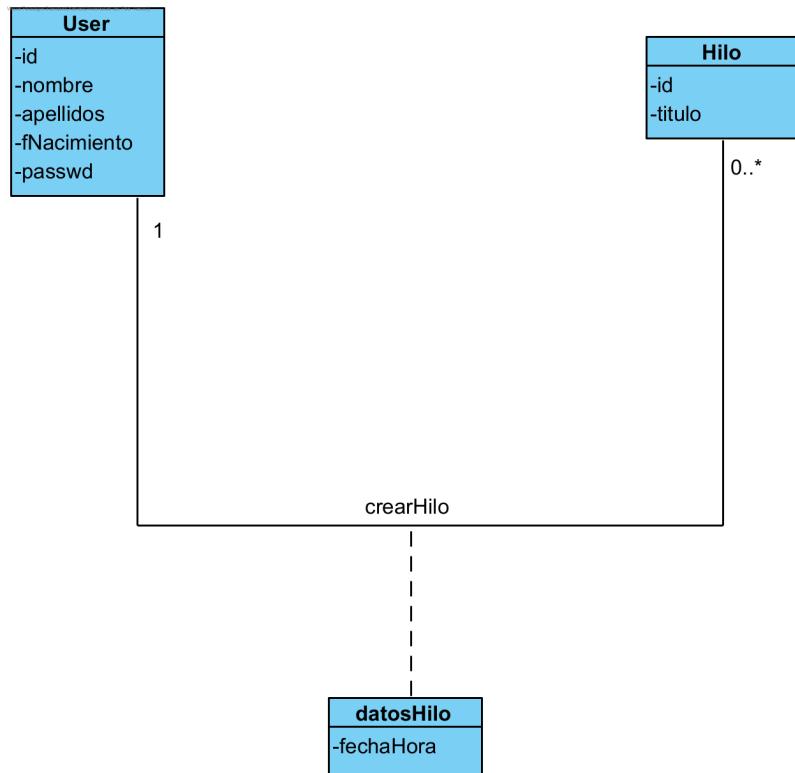


Figura 4.6: Diagrama de modelo de dominio de la relación crearHilo

crean

Esta relacion almacena los datos de creacion de los usuarios y de los libros por parte de un administrados.

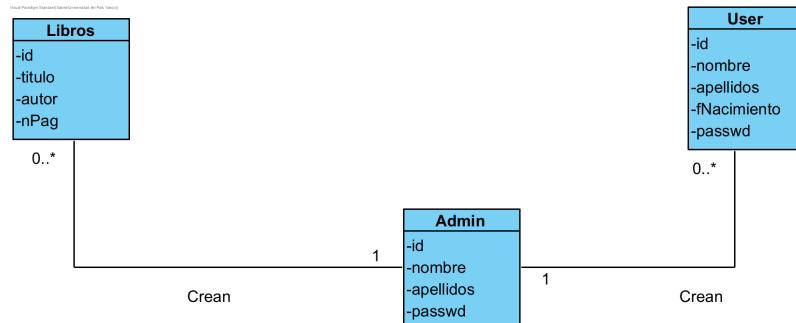


Figura 4.7: Diagrama de modelo de dominio de la relación crean

copiaDe

Esta relación almacena los datos de las copias de los libros que hay en el sistema. La entidad copiaLibros es la encargada de diferenciar las copias de un mismo libro.



Figura 4.8: Diagrama de modelo de dominio de la relación copiaDe

Plan de Pruebas

5.1. Gestión de reservas

Responsable: Kepa Reches Urrutia

Codigo	Descripción	Resultado esperado
F1P1	El usuario selecciona un libro y pulsa reservar	Muestra la interfaz de reserva
F1P1.2	El usuario selecciona un libro y NO pulsa reservar	Se mantiene en la interfaz del catálogo
F1P2	El usuario selecciona una fecha menor de 2 meses	La reserva es correcta
F1P2.2	El usuario selecciona una fecha mayor de 2 meses	Se abre una interfaz de error en la reserva
F1P2.3	El usuario no selecciona ninguna fecha	Se abre una interfaz de error en la reserva

5.2. Reseñas

Responsable: Mohamed El Basri Dehbi

Codigo	Descripción	Resultado esperado
F2P1	Agregar reseña y puntuación a un libro	La reseña y puntuación se registran correctamente
F2P1.1	Modificar reseña y puntuación de un libro	La reseña y puntuación se actualizan correctamente
F2P2	Intentar agregar reseña y puntuación sin devolver el libro	Se muestra un mensaje de error
F2P3	Ver reseña y puntuación de un libro	La reseña y puntuación del libro son visibles
F2P4	Cambiar reseña y puntuación después de devolver el libro	La reseña y puntuación se pueden cambiar
F2P5	Ver reseña y puntuación de otro usuario	La reseña y puntuación del usuario son visibles
F2P6	Ver reseña y puntuación de un libro sin reseñas	Sistema muestra un mensaje indicando la ausencia de reseñas

5.3. Red de amigos

Responsable: Javier Criado Garcia

Codigo	Descripción	Resultado esperado
F3P1	Ver perfil con sesión iniciada	Muestra el perfil
F3P2	El usuario pulsa el botón “Editar perfil”	Se habilitan los campos para editar el perfil
F3P3	El usuario pulsa el botón “Consultar reservas”	Se abre una interfaz donde ver las reservas
F3P4	El usuario pulsa el botón “Consultar reseñas”	Se abre una interfaz donde ver las reseñas
F3P5	El usuario pulsa el botón “Añadir amigo”	Se abre una interfaz donde añadir un amigo
F3P5.1	El usuario pulsa “OK”	El amigo introducido es añadido
F3P6	El usuario pulsa el botón “Solicitudes de amistad”	Se abre una interfaz donde ver las solicitudes de amistad
F3P7	El usuario pulsa el botón “Borrar amigo”	Se abre una interfaz donde eliminar un amigo
F3P7.1	El usuario pulsa ”SI”	El amigo seleccionado es eliminado
F3P7.2	El usuario pulsa ”NO”	El amigo seleccionado no es eliminado

5.4. Administrador

Responsable: Jon Valdes Granado

Codigo	Descripción	Resultado esperado	Resultado obtenido	Observaciones
F4P1	El usuario accede como administrador al sistema	Muestra la interfaz de menú del administrador	Muestra la interfaz de menú del administrador	Funciona
F4P2	El usuario pulsa el botón “Usuarios”	Muestra la interfaz de gestión de usuario	Muestra la interfaz de gestión de usuario	Funciona
F4P3	El usuario pulsa el botón “Crear”	Muestra la interfaz de creación de usuario	Muestra la interfaz de creación de usuario	Funciona
F4P3.1	El usuario introduce todos los campos correctamente y pulsa el botón “Crear”	Se crea un nuevo usuario correctamente	Se crea un nuevo usuario correctamente	Funciona
F4P3.2	El usuario introduce algún campo erróneo o no introduce alguno	Se mostrará un error por pantalla	Se mostrará un error por pantalla	Funciona
F4P4	El usuario pulsa el botón “Editar”	Muestra la interfaz de edición de usuario	Muestra la interfaz de edición de usuario	Funciona
F4P4.1	El usuario introduce todos los campos correctamente y pulsa el botón “Crear”	Se edita el usuario correctamente	Se edita el usuario correctamente	Funciona
F4P4.2	El usuario introduce algún campo erróneo o no introduce alguno	Se mostrará un error por pantalla	Se mostrará un error por pantalla	Funciona

Codigo	Descripción	Resultado esperado	Resultado obtenido	Observaciones
F4P5	El usuario pulsa el botón “Borrar”	Muestra la interfaz de eliminación de usuario	Muestra la interfaz de eliminación de usuario	Funciona
F4P5.1	El usuario introduce el correo electrónico del usuario correctamente y pulsa el botón ”Borrar”	Se borra el usuario correctamente	Se borra el usuario correctamente	Funciona
F4P5.2	El usuario introduce un correo electrónico inexistente en el sistema	Se mostrará un error por pantalla	Se mostrará un error por pantalla	Funciona

5.5. Foros

Responsable: Janire Veganzones García

Codigo	Descripción	Resultado esperado
F5P1	El usuario pulsa la opción “Foro” de la cabecera	Se muestra la interfaz principal del foro con todo los hilos creados y con el botón para poder crear uno nuevo
F5P2	El usuario pulsa el botón “+Hilo”	Se muestra la interfaz de crear un hilo
F5P2.1	El usuario pulsa el botón “Cancelar”	No se crea ningún hilo y te redirecciona a la pantalla principal del foro
F5P2.2	El usuario introduce todos los campos correctamente y pulsa el botón “+Hilo”	Se crea un Hilo correctamente y te redirecciona a la pantalla principal del foro donde lo podrás comprobar
F5P2.3	El usuario introduce todos los campos excepto nombre	Se mostrará un mensaje de error por pantalla
F5P2.4	El usuario introduce todos los campos excepto descripción	Se mostrará un mensaje de error por pantalla
F5P2.5	El usuario introduce todos los campos excepto mensaje	Se mostrará un mensaje de error por pantalla
F5P3	EL usuario pulsa sobre el hilo que quiere leer	Se mostrará la interfaz con los mensajes escritos por cada persona hasta ahora y con un botón para poder escribir un mensaje
F5P3.1	El usuario pulsa el botón “+Mensaje”	Se mostrará una interfaz donde podrás escribir tú mensaje para posteriormente publicarlo en el hilo
F5P3.1.1	El usuario escribe un mensaje y pulsa el botón “Enviar”	Te redirecciona a la pantalla anterior donde se pueden ver los mensaje y comprobar que tu mensaje ha sido publicado correctamente
F5P3.1.2	El usuario pulsa el botón “Enviar” sin escribir ningún mensaje	Se mostrará un mensaje de error por pantalla

5.6. Recomendaciones del sistema

Responsable: Ainhize Martinez Duran

Codigo	Descripción	Resultado esperado	Resultado obtenido	Observaciones
F6.P1	Entrar en el catálogo sin haber iniciado sesión	Muestra el catálogo pero no mostrará ninguna recomendación	Muestra el catálogo pero no mostrará ninguna recomendación	Funciona
F6.P2	Entrar en el catálogo habiendo iniciado sesión pero sin haber hecho ninguna reserva	Muestra el catálogo pero no mostrará ninguna recomendación	Muestra el catálogo pero no mostrará ninguna recomendación	Funciona
F6.P3	Entrar en el catálogo habiendo iniciado sesión y reservado un único libro	Muestra el catálogo y las recomendaciones asociadas al único libro que ha reservado	Muestra el catálogo y las recomendaciones asociadas al único libro que ha reservado	Funciona
F6.P3.1	Entrar en el catálogo habiendo iniciado sesión y reservado el único libro que existe de un tema	Muestra el catálogo pero no mostrará ninguna recomendación	Muestra el catálogo pero no mostrará ninguna recomendación	Funciona
F6.P4	Entrar en el catálogo habiendo iniciado sesión y reservado varios libros	Muestra el catálogo y las recomendaciones asociadas a los libros que ha reservado	Muestra el catálogo y las recomendaciones asociadas a los libros que ha reservado	Funciona

5.7. Recomendaciones de amigos

Responsable: Xabier Gabiña Barañano

Codigo	Descripción	Resultado esperado	Resultado obtenido	Observaciones
F7P1	Ver perfil sin iniciar sesion	Muestra el perfil sin recomendaciones	Muestra el perfil sin recomendaciones	Funciona
F7P1.1	Ver otro perfil al de la sesion	Muestra el otro perfil sin recomendaciones	Muestra el otro perfil sin recomendaciones	Funciona
F7P2	Ver perfil con sesion pero sin amigos ni libros alquilados previamente/actualmente	Muestra el perfil sin recomendaciones	Muestra el perfil sin recomendaciones	Funciona
F7P3	Ver perfil con sesion y amigos pero sin libros alquilados previamente/actualmente	Muestra el perfil sin recomendaciones con recomendaciones basadas en los amigos	Muestra el perfil sin recomendaciones con recomendaciones basadas en los amigos	Funciona
F7P3.1	Ver perfil con sesion y amigos pero sin libros alquilados previamente/actualmente y tus amigos no tienen otros amigos	Muestra el perfil sin recomendaciones	Muestra el perfil sin recomendaciones	Funciona
F7P4	Ver perfil con sesion y libros alquilados previamente/actualmente pero sin amigos	Muestra las recomendaciones en base a los libros alquilados previamente/actualmente	Muestra las recomendaciones en base a los libros alquilados previamente/actualmente	Funciona

Codigo	Descripción	Resultado esperado	Resultado obtenido	Observaciones
F7P4.1	Ver perfil con sesion y libros alquilados previamente/actualmente pero sin amigos y el libro es el unico de su tema	Muestra el perfil sin recomendaciones	Muestra el perfil sin recomendaciones	Funciona
F7P5	Ver perfil con sesion, amigos y libros alquilados previamente/actualmente	Muestra las recomendaciones en base a los libros alquilados previamente/actualmente y los amigos	Muestra las recomendaciones en base a los libros alquilados previamente/actualmente y los amigos	Funciona

Diagrama relacional

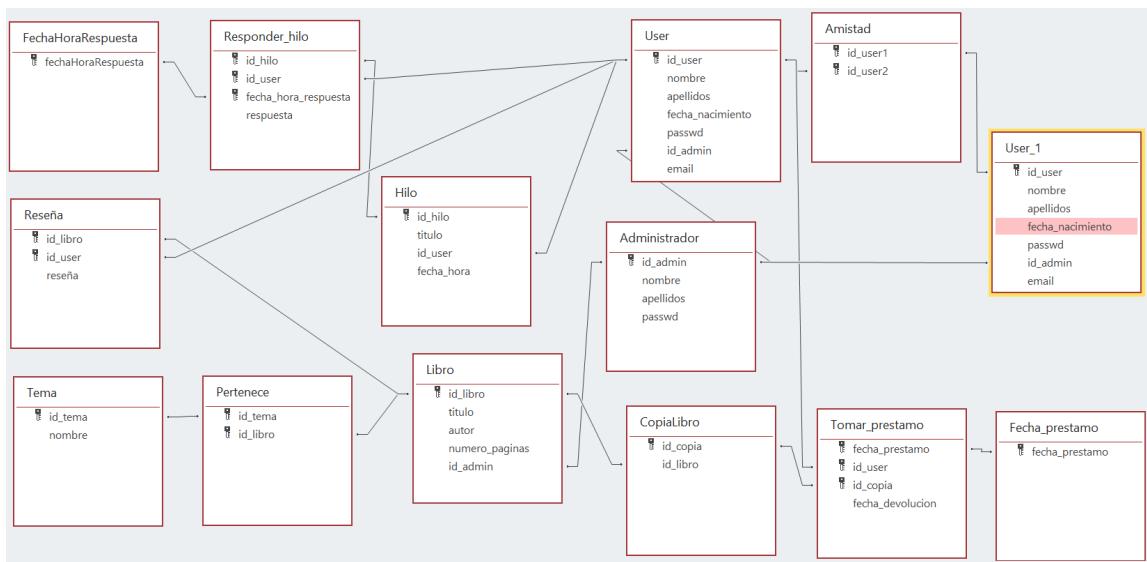


Figura 6.1: Diagrama relacional

Diagrama de clases

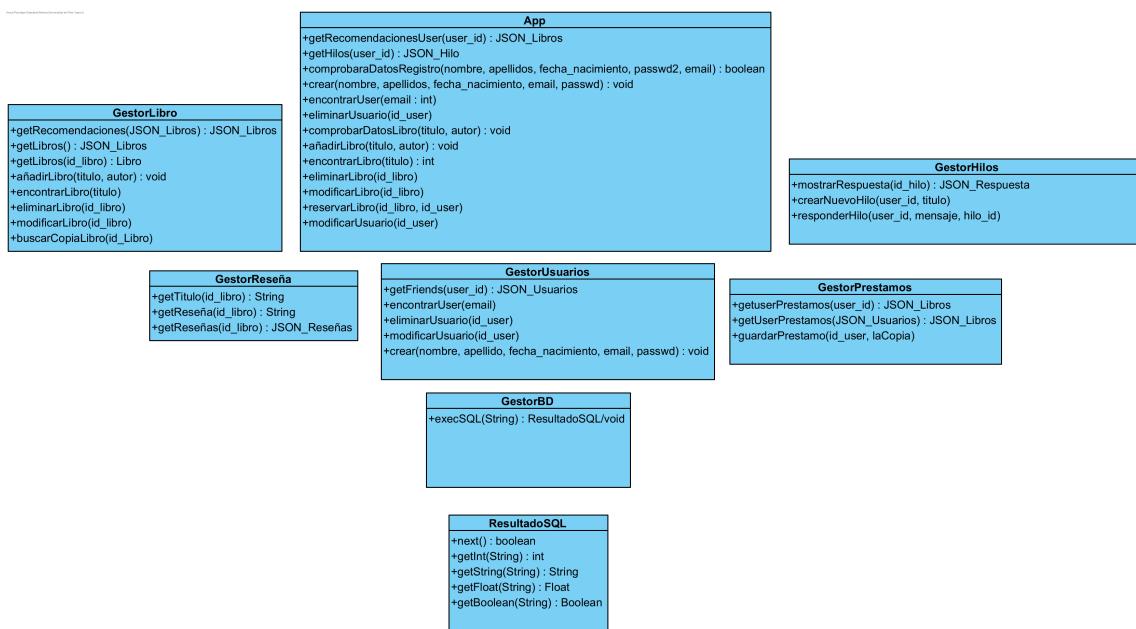


Figura 7.1: Diagrama de clases

Diagramas de comunicación

8.1. Gestión de reservas

Mohamed El Basri

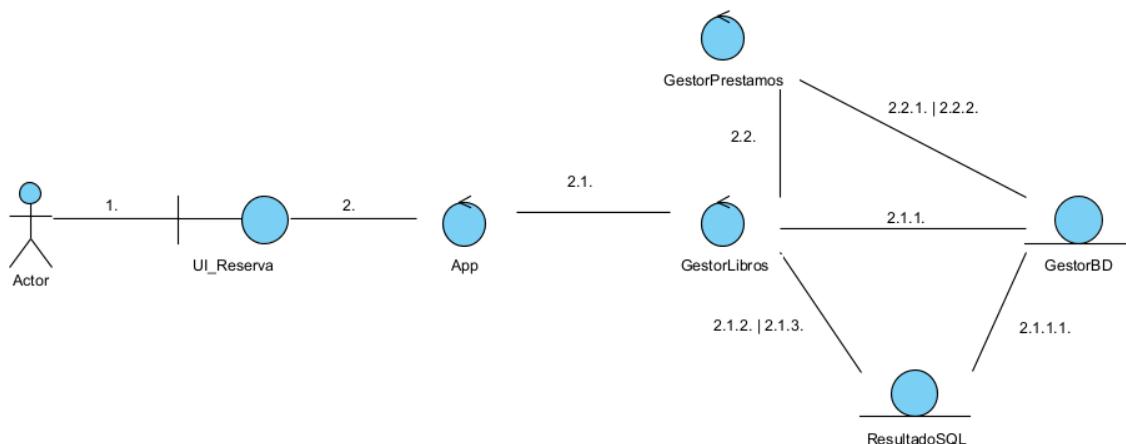


Figura 8.1: Diagrama de comunicación de la gestión de reservas

- 1 Selecciona un libro y pulsa el botón Reservar.
- 2 reserverLibro(id_libro, id_user)
 - 2.1 laCopia=buscarCopiaLibro(id_libro)
 - 2.1.1 execSQL(SELECT id_copia FROM CopiaLibro WHERE id_libro = %id_libro% AND Libre=1) : ResultadoSQL
 - 2.1.1.1 new ResultadoSQL()
 - 2.1.2 next()
 - 2.1.3 getInt(id_copia): int
 - 2.2 guardarPrestamo(id_user, laCopia)
 - 2.2.1 execSQL(INSERT INTO Tomar_prestamo (fecha_prestamo, id_user, id_copia) VA-LUES (%fecha%, %id_user%, %laCopia%))
 - 2.2.2 execSQL(UPDATE CopiaLibro SET Libre=0 WHERE id_copia= %laCopia%)

8.2. Reseñas

Javier Criado Garcia

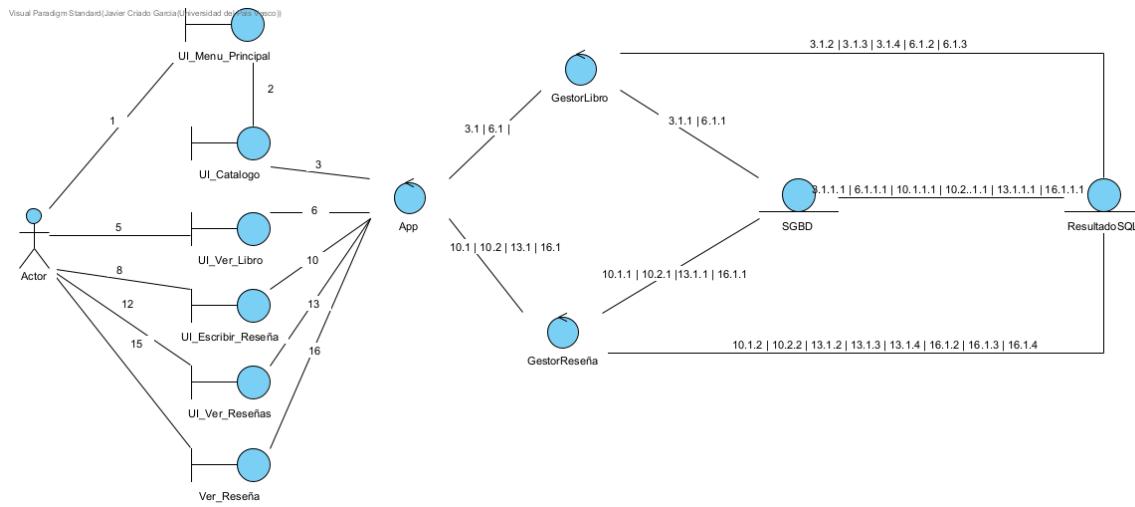


Figura 8.2: Diagrama de comunicación de las reseñas

```

JSON_Libros{
    [
        {
            "id_libro": int,
            "nombre":String
        }
    ]
}

JSON_Reseñas{
    [
        {
            "titulo": String,
            "reseña":String
        }
    ]
}
    
```

1. El usuario selecciona el Catalogo en el menú principal.
2. New UI_CATALOGO(user_id).
3. getLibrosUsuario(user_id):JSON_Libros.

- 3.1 getLibros():JSON.Libros
 - 3.1.1 execSQL(SELECT Libro From Libros)
 - 3.1.1.1 new ResultadoSQL
 - 3.1.2 next()
 - 3.1.3 getInt('id_libro'):int
 - 3.1.4 getString('nombre'):String
- 4. El usuario selecciona un libro.
- 5. New UI_Ver_Libro(user_id, id_libro, nombre).
- 6. getLibro(id_libro): Libro.
 - 6.1 getLibros():JSON.Libros
 - 6.1.1 execSQL(SELECT id_libro FROM Libros WHERE Libro.id_libro= %id_libro %): ResultadoSQL
 - 6.1.1.1 new ResultadoSQL
 - 6.1.2 next()
 - 6.1.3 getString('nombre'):String
- 7. El usuario selecciona escribir reseña.
- 8. New UI_Escribir_Reseña(user_id, id_libro, nombre).
- 9. El usuario pulsa el botón enviar.
- 10. EscribirReseña(user_id):void.
 - 10.1 elTitulo: getTitulo(): String
 - 10.1.1 execSQL(SELECT id_libro FROM Libros WHERE Libro.id_libro= %id_libro %): ResultadoSQL
 - 10.1.1.1 new ResultadoSQL
 - 10.1.2 next()
 - 10.2 laReseña: getReseña(id_libro):String
 - 10.2.1 execSQL(INSERT INTO Reseñas.reseña SELECT laReseña where User.id_user= %user_id AND Libro.id_libro= %id_libro %): ResultadoSQL
 - 10.2.1.1 new ResultadoSQL
 - 10.2.2 next()
- 11. El usuario selecciona ver reseñas
- 12. New UI_Ver_Reseñas(id_libro)
- 13. getReseñas(id_Libro): JSONReseñas
 - 13.1 getReseña(id_libro):String
 - 13.1.1 execSQL(SELECT Reseña From Reseñas where Libro.id_libro= %id_libro %): ResultadoSQL
 - 13.1.1.1 new ResultadoSQL
 - 13.1.2 next()
 - 13.1.3 getTitulo(' id_titulo'):String
 - 13.1.4 getReseña('id_reseña'):String
- 14. El usuario selecciona una reseña

15. New UI_Ver_Reseña(id_libro,id_reseña)
16. getReseña():JSONReseña
 - 16.1 getReseña(id_libro):String
 - 16.1.1 execSQL(SELECT Reseña From Reseñas where Libro.id_libro= %id_libro% AND Reseña.reseña= %id_reseña%): ResultadoSQL
 - 16.1.1.1 new ResultadoSQL
 - 16.1.2 next()
 - 16.1.3 getTitle(' id_titulo'):String
 - 16.1.4 getReseña('id_reseña'):String

8.3. Red de amigos

Jon Valdes Granado

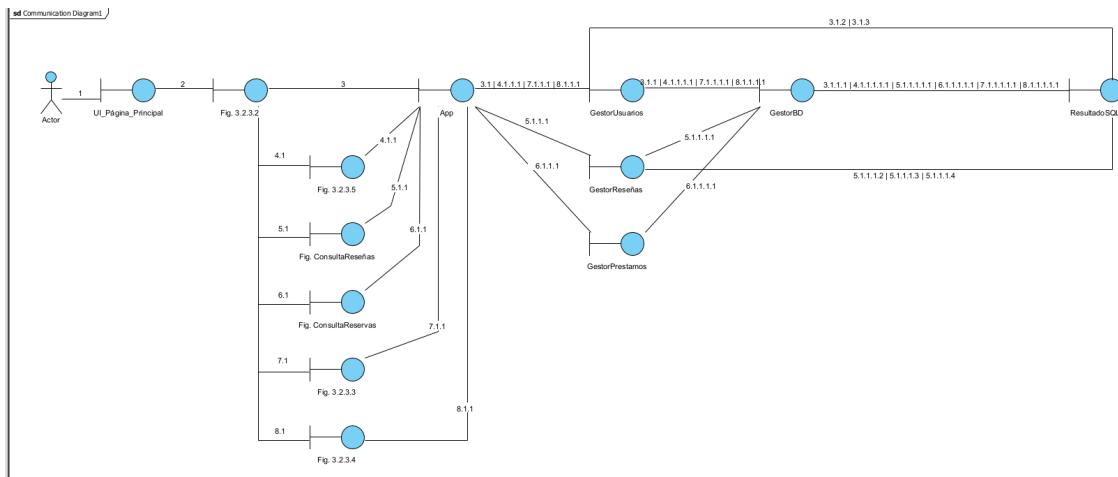


Figura 8.3: Diagrama de comunicación de la red de amigos

```

JSON_Libros{
    [
        {
            "id_libro":int,
            "nombre":String
        }
    ]
}

JSON_Usuarios{
    [
        {
            "id_user": int
        }
    ]
}

JSON_Reseñas{
    [
        {
            "titulo": String,
            "reseña": String
        }
    ]
}

```

8.4. Administrador

Janire Veganzones García

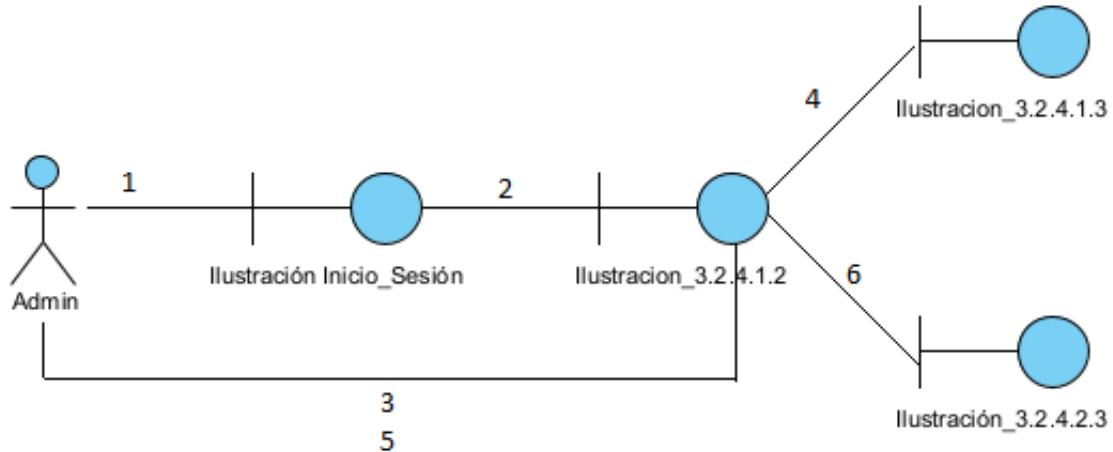


Figura 8.4: Diagrama de comunicación general del administrador

1. El usuario inicia sesión como Admin.
2. new `ilustration_3.2.4.1.2()`.
3. Pulsa “Usuarios” .
4. new `ilustration_3.2.4.1.3()`.
5. Pulsa “Libros” .
6. new `ilustración_3.2.4.2.3()`.

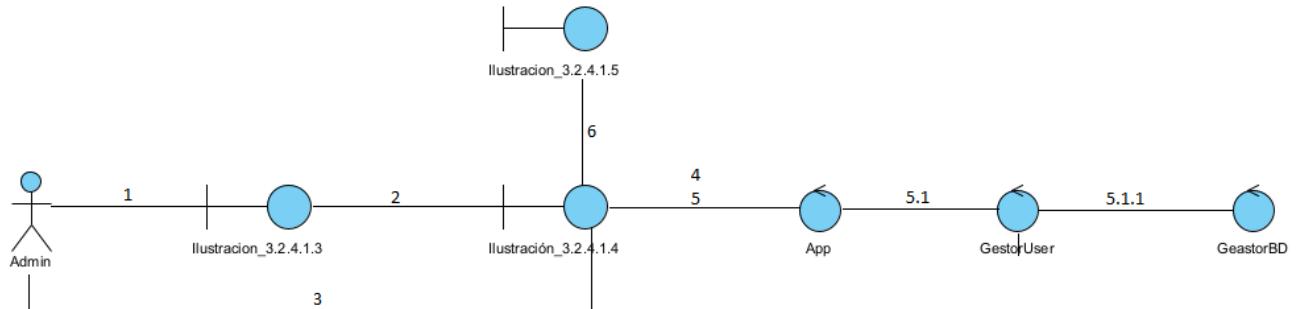


Figura 8.5: Diagrama de comunicación Crear un Usuario

1. El usuario pulsa “Crear”.
2. new `ilustracion_3.2.4.1.4()`.
3. Introducir nombre, apellidos, fecha nacimiento, email, contraseña y contraseña2 y pulsar “Crear”.
4. `comprobarDatosRegistro(nombre, apellidos, fecha_nacimiento, passwd, passwd2, email) : boolean`
[Si `comprobarDatosRegistro == TRUE`]
5. `crear(nombre, apellidos, fecha_nacimiento, email, passwd) : void`
 - 5.1 `crear(nombre, apellidos, fecha_nacimiento, email, passwd) : void`
 - 5.1.1 `execSQL (INSERT INTO User VALUES (%nombre %, %apellidos %, %fecha_nacimiento %, %email %, %pa)) : ResultadoSQL`

[Si No]
6. new `ilustracion_3.2.4.1.5()`.

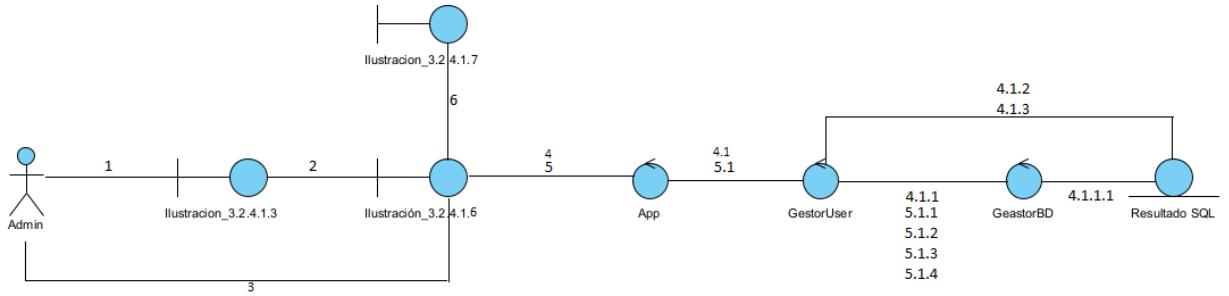


Figura 8.6: Diagrama de comunicación Borrar un Usuario

1. El usuario pulsa “Borrar”.
2. new `ilustration_3.2.4.1.6()`.
3. Escribir el email del usuario que deseas eliminar y pulsar “Borrar”.
4. `usuarioBuscado = encontrarUser(email): int`
 - 4.1 `encontrarUser(email)`
 - 4.1.1 `execSQL(SELECT id_user FROM User WHERE email = %email %): ResultadoSQL`
 - 4.1.2 `new ResultadoSQL()`
 - 4.1.3 `getInt(id_user): int`

[SI `usuarioBuscado != NULL`]
 5. `eliminarUsuario(id_user)`
 - 5.1 `eliminarUsuario(id_user)`
 - 5.1.1 `execSQL(DELETE * FROM Reseña WHERE id_user= usuarioBuscado)`
 - 5.1.2 `execSQL(DELETE * FROM Responder_hilo WHERE id_user= usuarioBuscado)`
 - 5.1.3 `execSQL(DELETE * FROM Tomar_prestamo WHERE id_user = usuarioBuscado)`
 - 5.1.4 `execSQL(DELETE * FROM User WHERE id_user= UsuarioAEliminar)`

[Si No]
 6. new `ilustration_3.2.4.1.7()`.

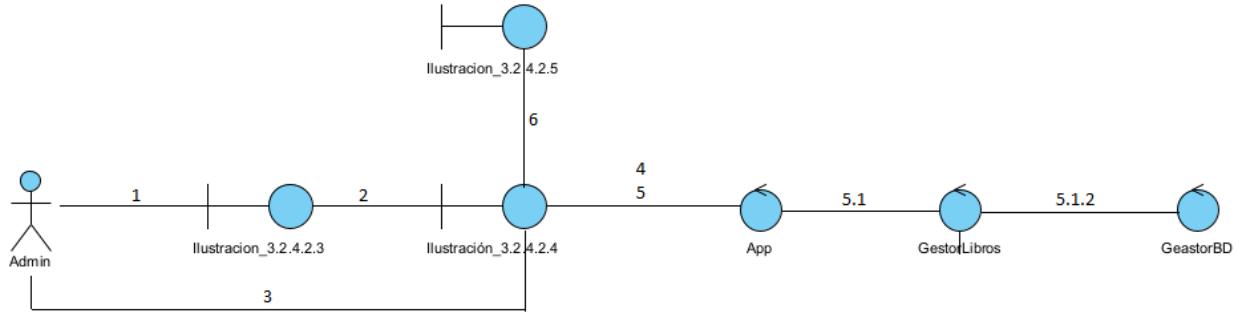


Figura 8.7: Diagrama de comunicación Insertar un Libro

1. El usuario pulsa “Añadir”.
2. new `ilustracion_3.2.4.2.3()`.
3. Rellenar los campos Título y autor y pulsar “Insertar”.
4. `comprobarDatosLibro(título, autor): boolean`
[Si `comprobarDatosLibro == TRUE`]
5. `añadirLibro(título, autor): void`
 - 5.1 `añadirLibro(título, autor): void`
 - 5.1.1 `execSQL (INSERT INTO Libros (%título%, %autor%)): ResultadoSQL`
[Si No]
6. new `ilustracion_3.2.4.2.5()`.

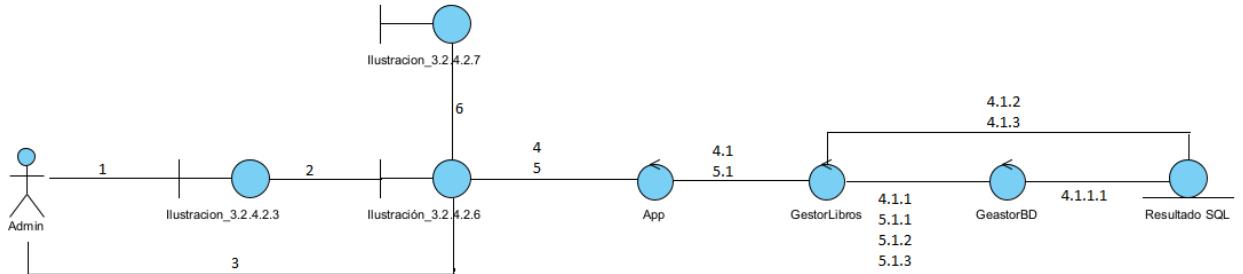


Figura 8.8: Diagrama de comunicación Borrar un Libro

1. El usuario pulsa “Eliminar”.
2. new ilustration_3.2.4.2.6().
3. Escribir el título y autor del libro que deseas eliminar y pulsar “Eliminar”.
4. libroBuscado= encontrarLibro(título): int
 - 4.1 encontrarLibro(título)
 - 4.1.1 execSQL(SELECT id_libro FROM Libro WHERE título = %título%): ResultadoSQL
 - 4.1.1.1 new ResultadoSQL()
 - 4.1.1.2 next()
 - 4.1.1.3 getInt(id.libro): int

[SI libroBuscado != NULL]
 5. eliminarLibro(id.libro)
 - 5.1 eliminarLibro(id.libro)
 - 5.1.1 execSQL(DELETE * FROM Reseña WHERE id.libro= libroBuscado)
 - 5.1.2 execSQL(DELETE * FROM CopiaLibro WHERE id.libro= libroBuscado)
 - 5.1.3 execSQL(DELETE * FROM Libro WHERE id.libro= libroBuscado)

[Si No]
 6. new ilustration_3.2.4.2.7().

8.5. Foros

Ainhize Martinez Duran

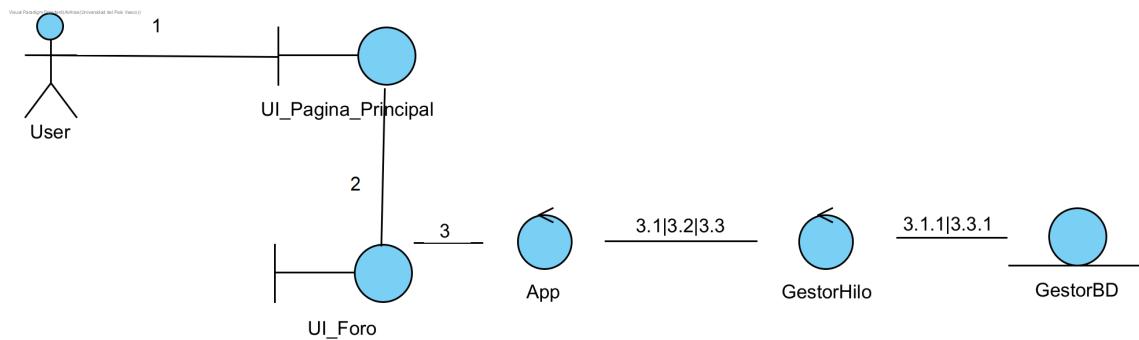


Figura 8.9: Diagrama de comunicación para Foro

```

JSON_Hilo{
    [
        {
            "id_hilo":int,
            "titulo":String
        }
    ]
}

JSON_Respuesta{
    [
        {
            "mensaje":String,
            "user":String
        }
    ]
}
  
```

1. El usuario selecciona Foro en el menú principal.
2. new UI_Foro(user_id)
3. getHilos(user_id):JSON_Hilo
 - 3.1 crearNuevoHilo(user_id,titulo)
 - 3.1.1 execSQL(INSERT INTO Hilo (id_user,titulo) VALUES(%user_id %, %titulo %))
 - 3.2 mostrarRespuesta(id_hilo):JSON_Respuesta
 - 3.3 responderHilo(user_id,mensaje,hilo_id)
 - 3.3.1 execSQL(INSERT INTO Responder_Hilo (id_user,respuesta,id_hilo) VALUES (%user_id %, %mensaje %, %hilo_id %))

8.6. Recomendaciones del sistema

Xabier Gabiña Barañano

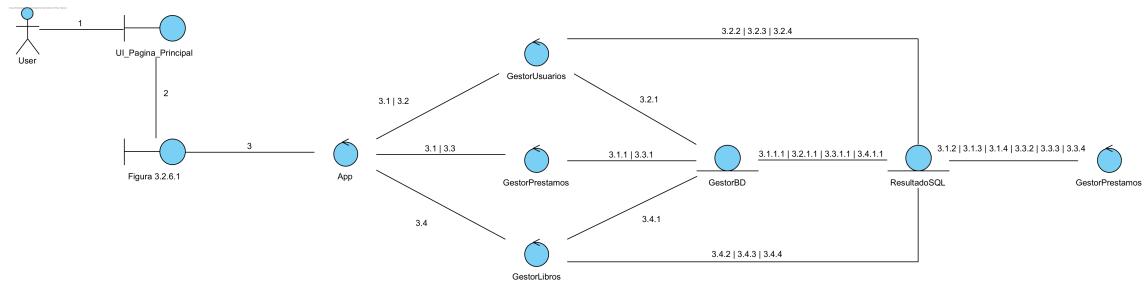


Figura 8.10: Diagrama de comunicación de las recomendaciones del sistema

```

JSON_Libros{
    [
        {
            "id_libro":int,
            "nombre":String
        }
    ]
}

JSON_Usuarios{
    [
        {
            "id_user":int
        }
    ]
}
    
```

- 1 El usuario selecciona el catálogo en el menú principal.
- 2 new Figura_3.2.6.1(user_id).
- 3 getRecomendacionesUser(user_id):JSON_Libros
 - 3.1 listaLibros=getUserPrestamos(user_id):JSON_Libros
 - 3.1.1 execSQL(" SELECT Libro.* FROM Tomar_prestamo INNER JOIN CopiaLibro ON Tomar_prestamo.id_copia = CopiaLibro.id_copia INNER JOIN Libro ON CopiaLibro.id_libro = Libro.id_libro WHERE Tomar_prestamo.id_user = %user_id %; "): new ResultadoSQL()
 - 3.1.1.1 new ResultadoSQL()
 - 3.1.2 next()
 - 3.1.3 getInt('id_libro'):int
 - 3.1.4 getString('nombre'):String
 - 3.2 listaAmigos=getFriends(user_id):JSON_Usuarios
 - 3.2.1 execSQL(" SELECT U1.id_user AS id_amigo, U1.nombre AS nombre_amigo FROM User AS U JOIN Amistad AS A ON U.id_user = A.id_user1 OR U.id_user = A.id_user2 JOIN User AS U1 ON (U.id_user = A.id_user1 AND U1.id_user = A.id_user2) OR (U.id_user = A.id_user2 AND U1.id_user = A.id_user1) WHERE U.id_user = %user_id %; "): new ResultadoSQL()
 - 3.2.1.1 new ResultadoSQL()
 - 3.2.2 next()
 - 3.2.3 getInt('id_amigo'):int
 - 3.3 listaLibros+=getUserPrestamos(listaAmigos):JSON_Libros
 - 3.3.1 execSQL(" SELECT Libro.* FROM Tomar_prestamo INNER JOIN CopiaLibro ON Tomar_prestamo.id_copia = CopiaLibro.id_copia INNER JOIN Libro ON CopiaLibro.id_libro = Libro.id_libro WHERE Tomar_prestamo.id_user = %user_id %; "): new ResultadoSQL()
 - 3.3.1.1 new ResultadoSQL()
 - 3.3.2 next()
 - 3.3.3 getInt('id_libro'):int
 - 3.3.4 getString('nombre'):String
 - 3.4 getRecomendaciones(listaLibros):JSON_Libros
 - 3.4.1 execSQL(" SELECT Libro.* FROM Libro INNER JOIN Pertenece ON Libro.id_libro = Pertenece.id_libro INNER JOIN Tema ON Pertenece.id_tema = Tema.id_tema WHERE Tema.id_tema IN (SELECT Pertenece.id_tema FROM Pertenece WHERE Pertenece.id_libro = %libro_id %); "): new ResultadoSQL()
 - 3.4.1.1 new ResultadoSQL()
 - 3.4.2 next()
 - 3.4.3 getInt('id_libro'):int
 - 3.4.4 getString('nombre'):String

8.7. Recomendaciones de amigos

Kepa Reche Urrutia

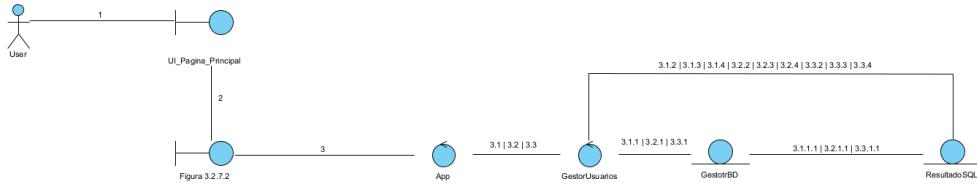


Figura 8.11: Diagrama de comunicación de las recomendaciones de amigos

```
JSON_Libros{  
    [  
        {  
            "id_libro":int,  
            "nombre":String  
        }  
    ]  
}  
JSON_Usuarios{  
    [  
        {  
            "user_id":int,  
            "nombre":String  
        }  
    ]  
}
```

1. El usuario selecciona el perfil en el menú principal.
2. new UI_PERFIL(user_id).
3. getRecomendacionesAmigos(user_id):JSON_Usuarios
 - 3.1 listaLibros=getLibros():JSON_Libros
 - 3.1.1 execSQL(" SELECT Libro.* FROM User INNER JOIN Tomar_prestamo ON User.id_user = Tomar_prestamo.id_user INNER JOIN CopiaLibro ON Tomar_prestamo.id_copia = CopiaLibro.id_copia INNER JOIN Libro ON CopiaLibro.id_libro = Libro.id_libro WHERE User.id_user = %user_id%; "): ResultadoSQL
 - 3.1.1.1 new ResultadoSQL()
 - 3.1.1.2 next()
 - 3.1.1.3 getInt('id_libro'):int
 - 3.1.1.4 getString('nombre'):String
 - 3.2 listaUsuarios=getFriendsFriends(user_id):JSON_Usuarios
 - 3.2.1 execSQL(" SELECT DISTINCT a2.usuario2 AS amigodeamigo FROM amigos a1 JOIN amigos a2 ON a1.usuario2 = a2.usuario1 WHERE a1.usuario1 = a1.id_user AND a2.usuario2 = %user_id%; "
 - 3.2.1.1 new ResultadoSQL()
 - 3.2.1.2 next()
 - 3.2.1.3 getInt('user_id'):int
 - 3.2.1.4 getString('nombre'):String
 - 3.3 getAmigosPorRecomendaciones(listaLibros):JSON_Usuarios
 - 3.3.1 execSQL(" SELECT DISTINCT a2.usuario2 AS amigodeamigo, l.id_libro FROM amigos a1 JOIN amigos a2 ON a1.usuario2 = a2.usuario1 JOIN lecturas l ON a2.usuario2 = l.user_id WHERE a1.usuario1 = 'user_id' AND a2.usuario2 = 'user_id'; "
 - 3.3.1.1 new ResultadoSQL()
 - 3.3.1.2 next()
 - 3.3.1.3 getInt('user_id'):int
 - 3.3.1.4 getString('nombre'):String

Diagrama de secuencias

9.1. Gestión de reservas

Mohamed El Basri

9.2. Reseñas

Javier Criado Garcia

9.3. Red de amigos

Jon Valdes Granado

9.4. Administrador

Janire Veganzones García

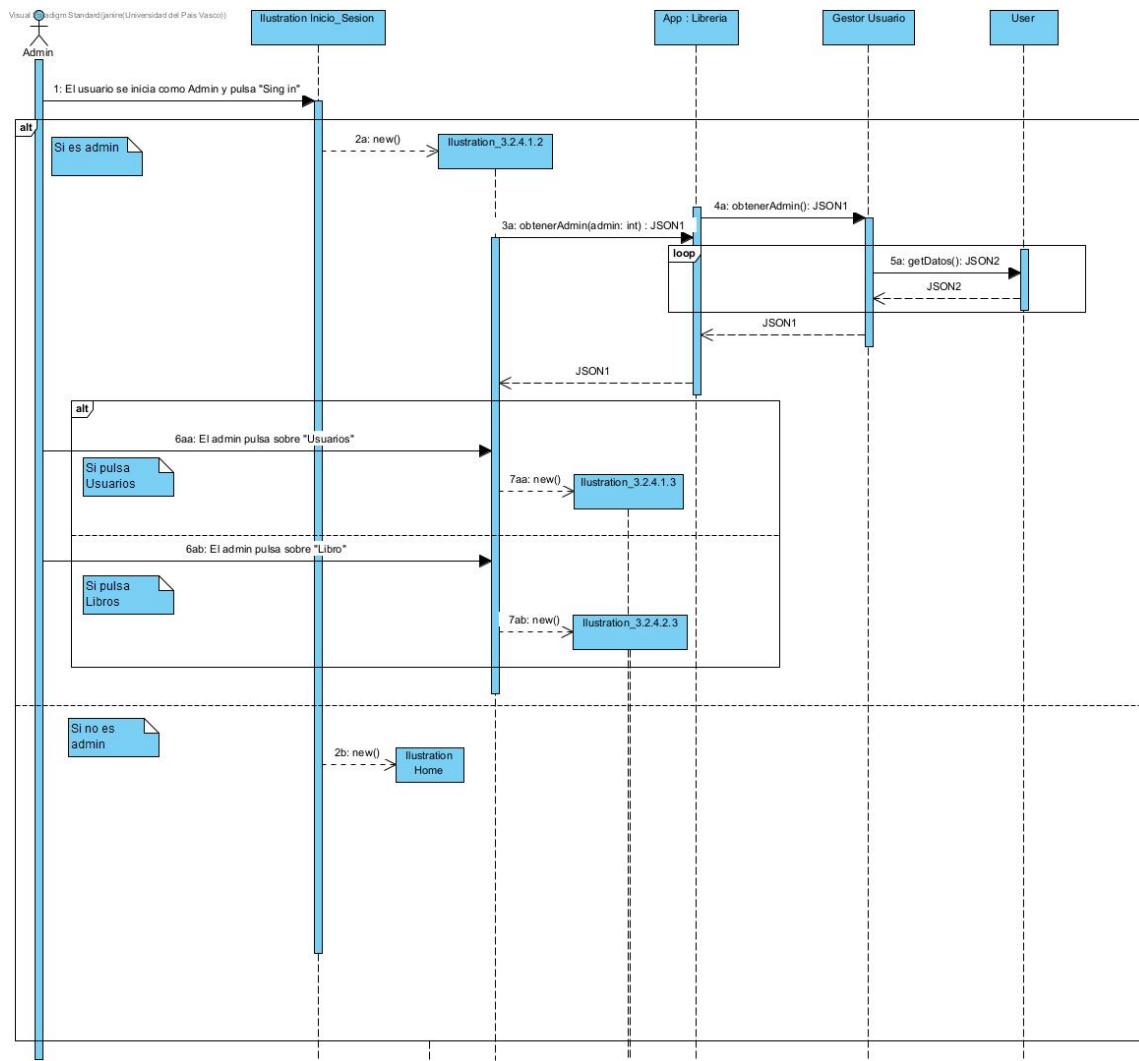


Figura 9.1: Diagrama de secuencia de Administracion general

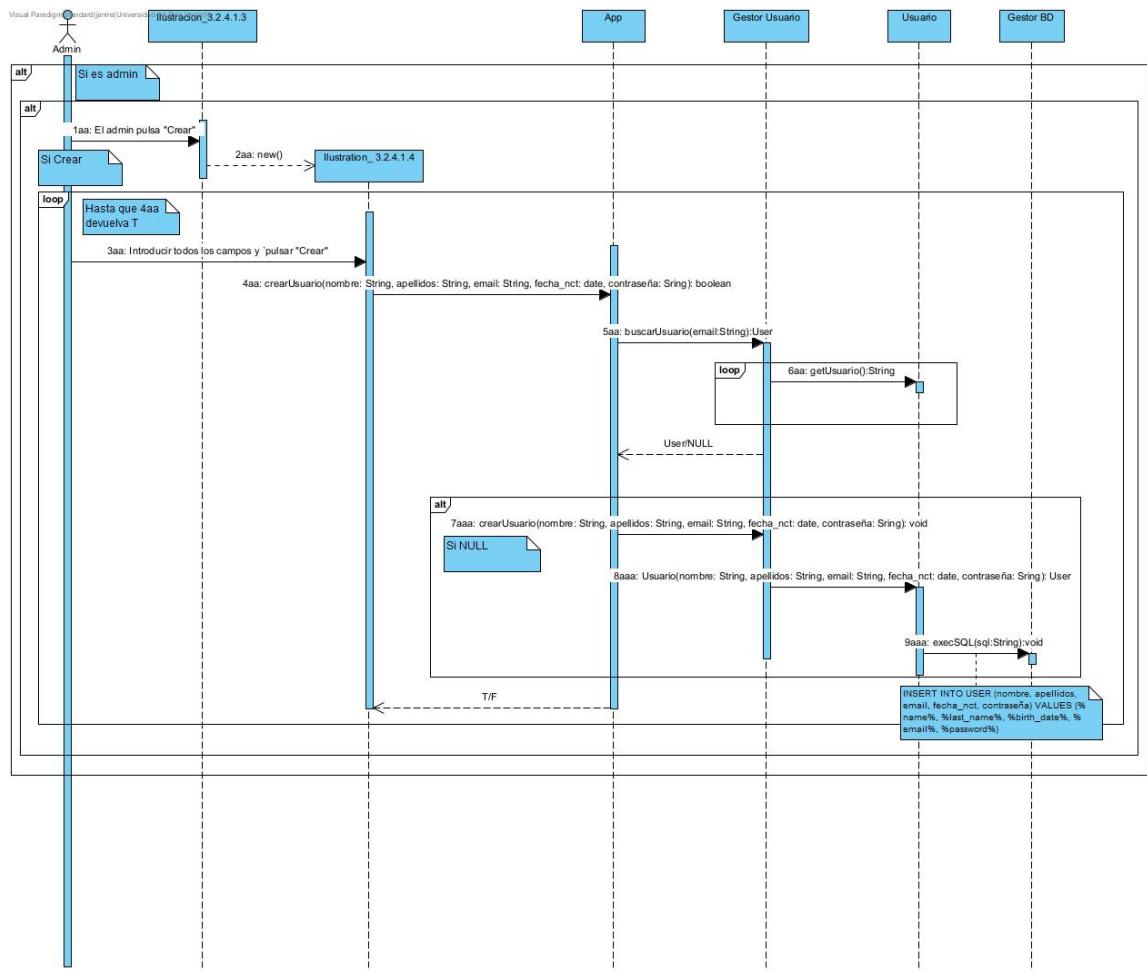


Figura 9.2: Diagrama de secuencia de Crear Usuario

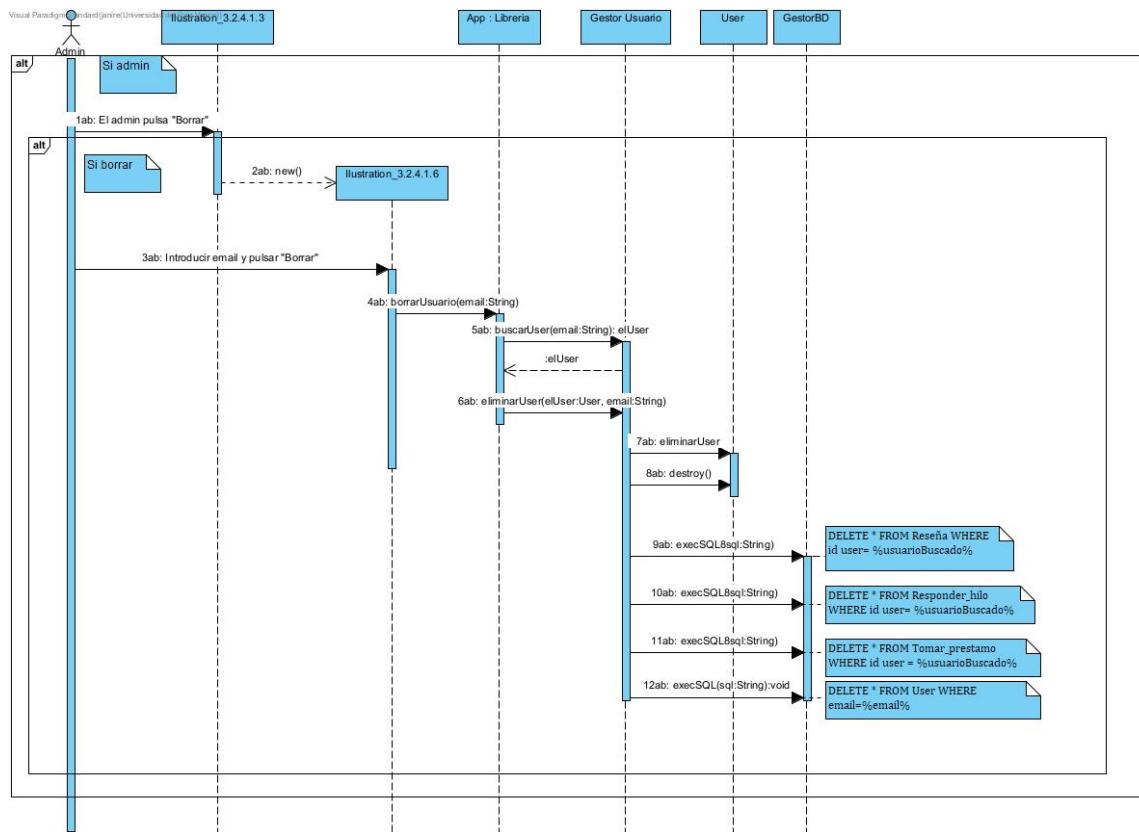


Figura 9.3: Diagrama de secuencia de Borrar Usuario

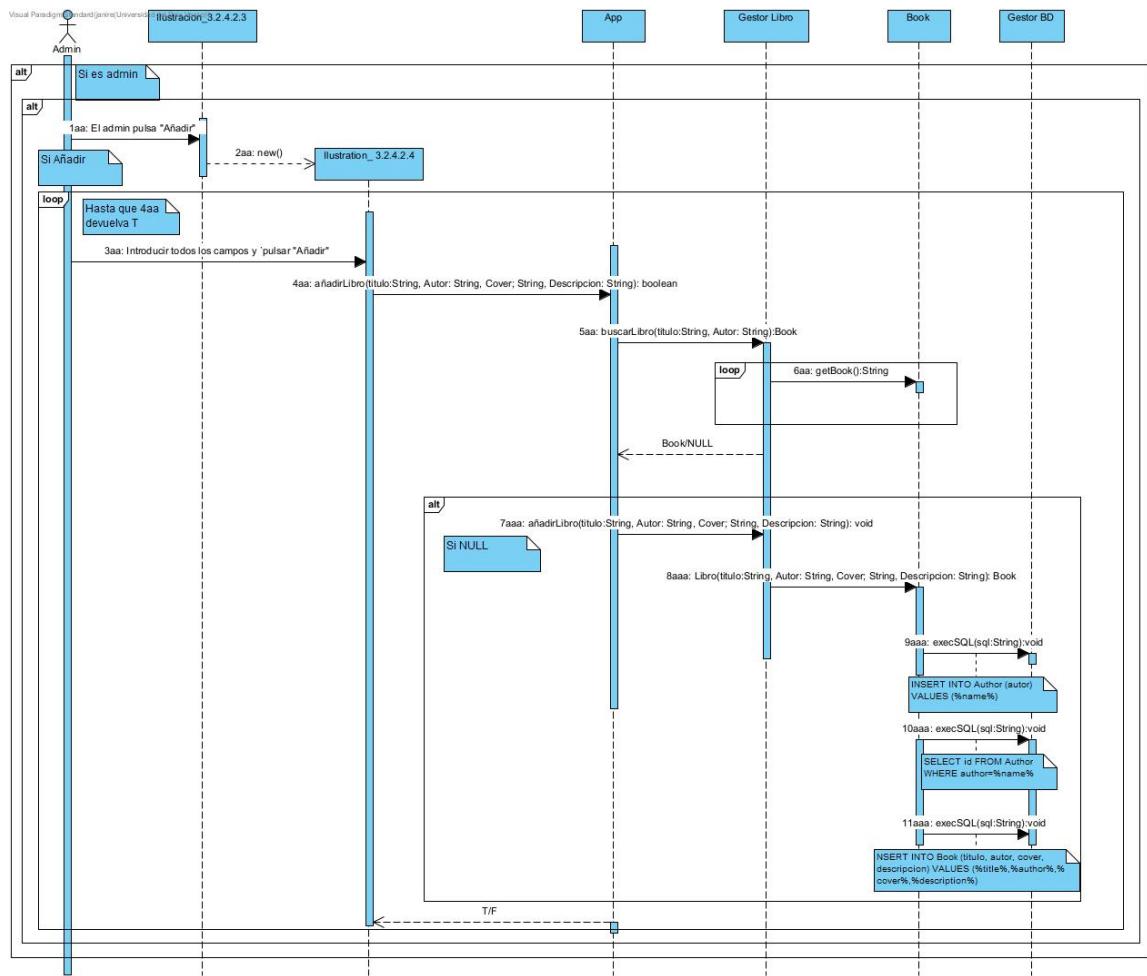


Figura 9.4: Diagrama de secuencia de Añadir Libro

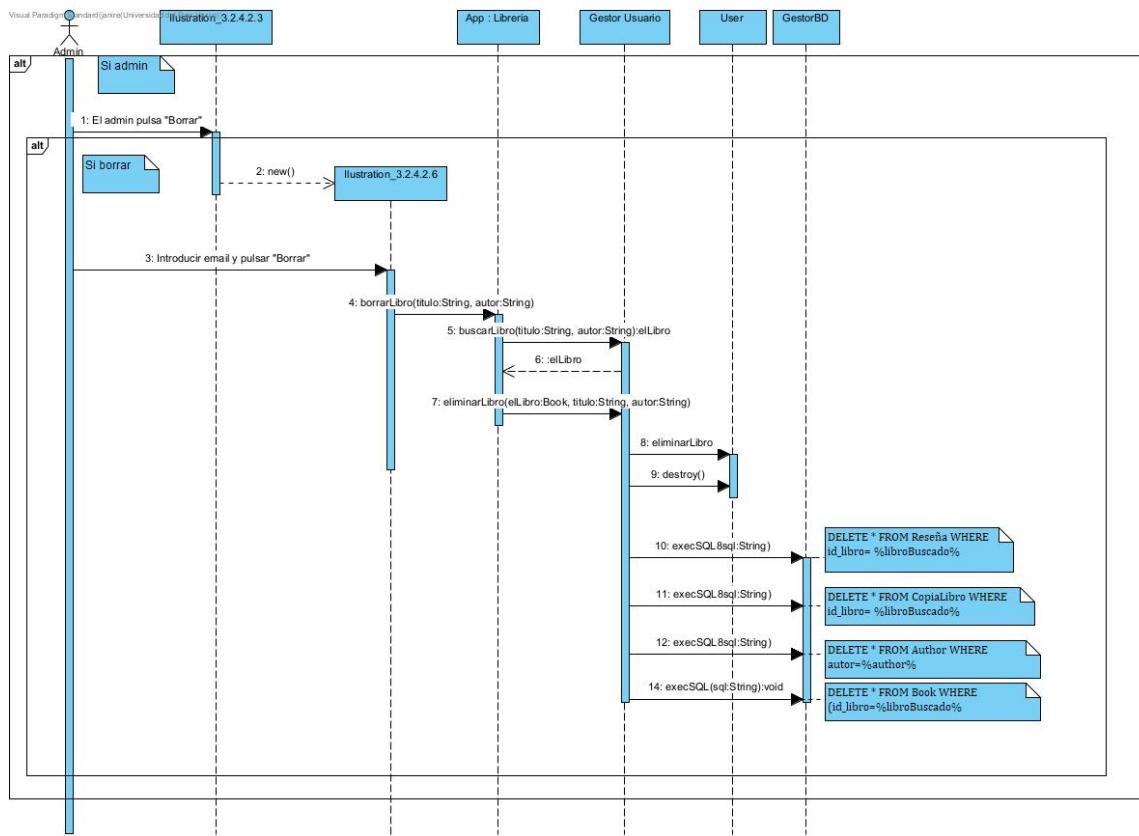


Figura 9.5: Diagrama de secuencia de Borrar Libro

9.5. Foros

Ainhize Martinez Duran

9.6. Recomendaciones del sistema

Xabier Gabiña Barañano

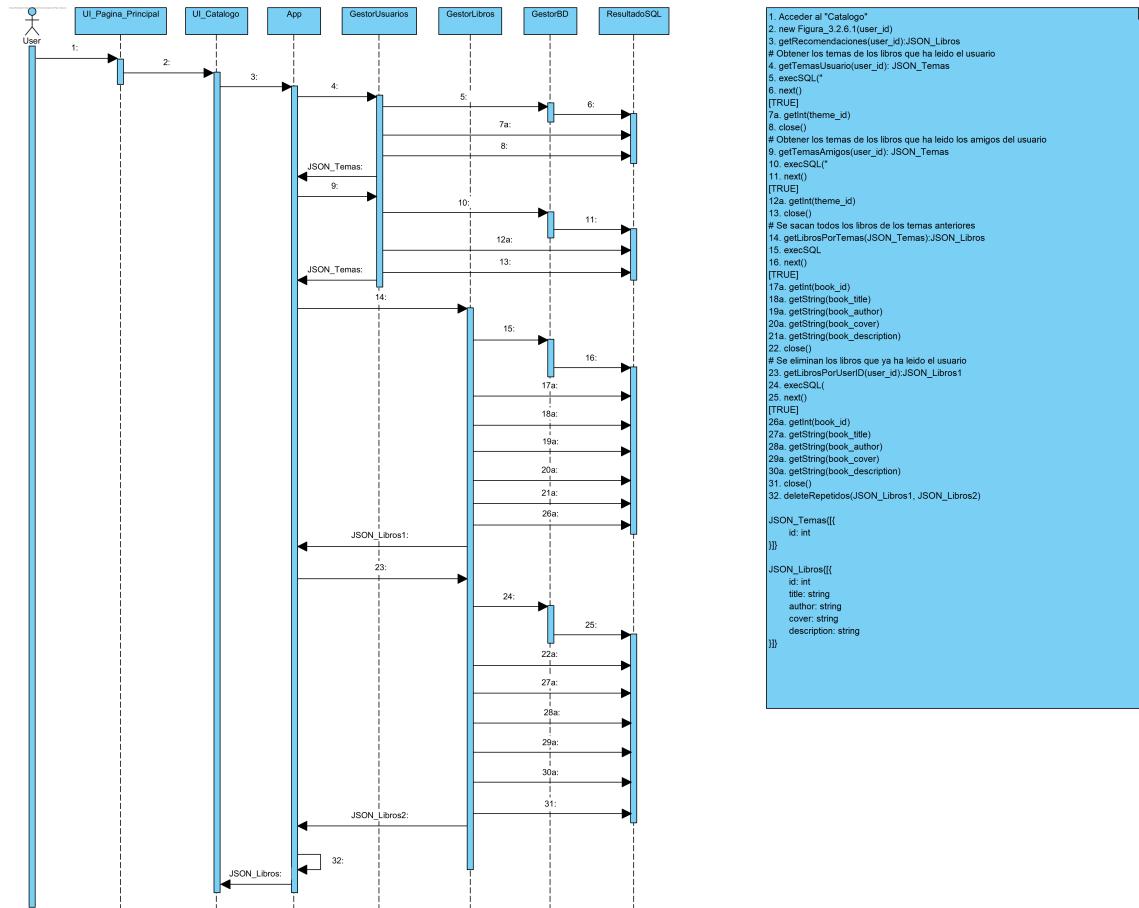


Figura 9.6: Diagrama de secuencia de las recomendaciones del sistema

9.7. Recomendaciones de amigos

Kepa Reche Urrutia

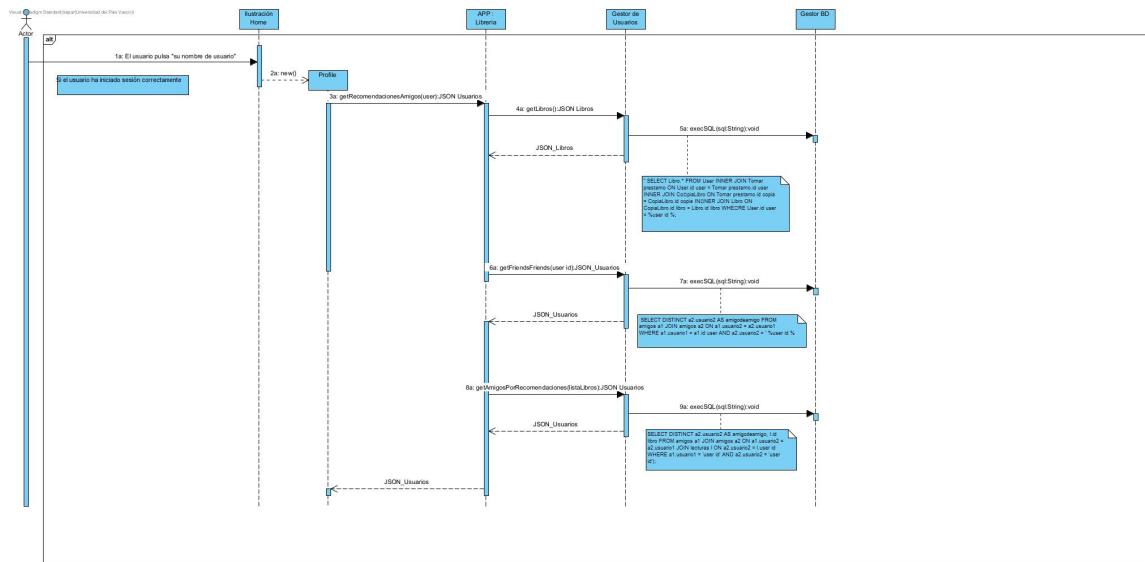


Figura 9.7: Diagrama de secuencia de las Recomendaciones de amigos

Problemas de implementación

10.1. Gestión de reservas

10.2. Reseñas

10.3. Red de amigos

10.4. Administrador

Janire Veganzones García

A la hora de programar el caso de uso de administrador he tenido varios problemas. El que mas tiempo me ha llevado ha sido el de mostrar las alertas, no me las mostraba entonces he tenido que crear una clase html de mensaje para que saliese un mensaje. Otro error que he tenido ha sido a la hora de crear la bd ya que la he tenido que cambiar para crear mas campos. Por ultimo a la hora de fusionar las ramas con mis compañeros me dio un problema y estuvimos varias horas intentando fusionarlo.

Algo para comentar de mi implementación es que a la hora de crear libros se pueden crear varios con el mismo titulo y autor ya que entiendo que puede haber varias versiones de ese mismo libro.

10.5. Foros

10.6. Recomendaciones del sistema

Xabier Gabiña Barañano

Principalmente los problemas y limitaciones vienen dada del hecho de que varios integrantes del grupo han abandonado el proyecto haciendo que funciones como el tomar prestamos o añadir amigos no esten implementadas por lo que han tenido que se hechas "hardcodeadas." en la base de datos. Esto ha realentizado sobre todo la parte de las pruebas unitarias ya que no se podian realizar pruebas de las funciones que no estaban implementadas.

De la parte de recomendaciones del sistema, la mayor dificultad la he encontrado en las SELECT que he implementado ya que buscando hacer las minimas busquedas posibles he tenido que hacer consultas complejas que ha llevado mas tiempo en implementarse.

10.7. Recomendaciones de amigos

Kepa Reche Urrutia

En el desarrollo de la funcionalidad de recomendación de amigos he tenido varios problemas que citare a continuación:

A la hora de crear amistades por ejemplo el User1 era amigo del User2 y eso deberia servir para que el User2 tambien fuese del User1, pero no era asi. Del problema anterior, derivaba el de la obtencion de los amigos de tus amigos, intentaba obtenerlos utilizando una unica Select pero no conseguia el resultado esperado. La solucion que encontré era un poco liosa ya que habia estructuras con tuplas anidadas y resultaba tedioso seguir el orden. Una vez conseguido el resultado, no sabia muy bien como enviar el resultado al HTML y mostrarlo de forma atractiva visualmente.

Con la ayuda de mis compañeros Xabi y Janire y consultando informacion en foros y herramientas como Copilot he conseguido resolver todos los problemas mencionados.

Conclusiones

Por último, se va a profundizar en las conclusiones obtenidas durante la realización de la práctica.

Cómo se desarrolla en el apartado introductorio del trabajo, nos encontrábamos con una biblioteca en proceso de transformación digital y expansión de sus servicios en línea. En la web de esta biblioteca, se han integrado nuevas funcionalidades para mejorar la experiencia digital de los usuarios como la gestión de reservas, la red de amistades, la capacidad de dejar reseñas y comentarios y la implementación de sistemas de recomendación.

Del mismo modo, también se ha integrado la inclusión de un rol de administrador con capacidades de gestión, los foros para discusión y la consideración de sistemas de recomendación basados en afinidades de intereses entre usuarios.

Para poder implementar las funcionalidades anteriormente mencionadas, se ha seguido la siguiente metodología: primero se desarrolló un caso de uso general que dio lugar a los casos de uso extendidos, donde se pretendía definir y describir los requerimientos funcionales del sistema; posteriormente, se realizó el modelo de dominio para capturar los elementos del contexto y sus relaciones, donde había que tener en cuenta que datos tendría que almacenar el sistema; después, se ideó un plan de pruebas para cada funcionalidad, teniendo en cuenta los puntos críticos de cada una; a continuación, se creó el diagrama relacional utilizando como base el modelo de dominio mencionado; más tarde, se desarrollaron los diagramas de clases y de comunicación; finalmente, se realizaron los diagramas de secuencia, y se implementaron tanto el código correspondiente como las pruebas.

En resumen, este proyecto ha sido una gran oportunidad para aprender sobre el desarrollo software, ofreciendo una experiencia práctica y enriquecedora al aplicar una metodología estructurada en la creación de una plataforma web funcional. De hecho, no solo nos ha permitido aprender sobre los procesos y técnicas involucradas en el desarrollo de software, sino también aplicar esos conocimientos en un entorno práctico. Además, a través de este proyecto, hemos podido comprender la importancia de la planificación detallada, el diseño cuidadoso y la implementación.

Repositorio

Link al repositorio