

Entrega final: Biblioteca

Análisis y Diseño de Sistemas de Información

**Ingeniería Informática de Gestión y Sistemas
de Información**

Grupo: **e³scape**
Curso: 2023-2024

Índice

1. ABSTRACT.....	3
2. DIAGRAMA DEL MODELO DE CASOS DE USO DEL SISTEMA.....	4
2.0. CUALQUIERA.....	4
2.1. ADMINISTRADOR.....	4
2.2. GENERAL.....	5
2.3. USUARIO.....	5
3. CASOS DE USO EXTENDIDOS DEL SISTEMA.....	6
3.0. COMIENZO.....	6
3.1. GESTIÓN RESERVAS.....	9
3.2. RESEÑAS.....	16
3.3. RED DE AMIGOS.....	19
3.4. ADMINISTRADOR.....	25
3.5. FORO.....	32
3.6. RECOMENDACIONES DEL SISTEMA.....	36
4. JERARQUÍA DE ACTORES.....	38
5. MODELO DE DOMINIO.....	39
5.0. ENTIDADES Y SUS ATRIBUTOS.....	39
5.1. FORO.....	40
5.2. LIBRO.....	41
5.3. ADMINISTRADOR.....	42
5.4. RED DE AMIGOS.....	42
6. PLAN DE PRUEBAS.....	43
7. DIAGRAMA DE BASE DE DATOS.....	50
8. DIAGRAMAS DE CLASES.....	51
9. DIAGRAMAS DE COMUNICACIÓN.....	52
10. DIAGRAMAS DE SECUENCIA.....	65
11. PROBLEMAS DE IMPLEMENTACIÓN Y SOLUCIONES.....	98
12. CONCLUSIONES.....	99
13. CHANGELOG (CAMBIOS).....	100

1. ABSTRACT

El presente trabajo, realizado por el equipo **e³scape**, entrega final del proyecto de la asignatura de Análisis y Diseño de Sistemas de Información, perteneciente al tercer curso del grado en Ingeniería Informática de Gestión y Sistemas de Información de la Universidad del País Vasco.

El grupo de trabajo está compuesto por Ander Gutiérrez Martín, Sergio Lusa Coria, Eneko Etxaniz Monge, Jon Marcos Mercadé, Adrián Mena Ruiz y Javier Arambarri Calvo.

El objetivo de la práctica completa es diseñar un sistema de información desde cero, empezando por la captura de requisitos (identificación de casos de uso y modelo de dominio) para ir transformándolo paso a paso en el código a implementar.

El documento está estructurado en nueve partes diferentes: diagrama completo de los casos de uso, explicación y detalles de los casos de uso extendidos, la jerarquía de actores, modelo de dominio, plan de pruebas y su implementación y diagramas de base de datos, clases, comunicación y secuencia.

En dicha estructura, con el fin de que haya una mayor coherencia, legibilidad y seguimiento del trabajo, se han agrupado los **casos de uso extendidos** por cada funcionalidad pedida en el enunciado.

Las ilustraciones han sido realizadas con draw.io y el software *Visual Paradigm*.

2. DIAGRAMA DEL MODELO DE CASOS DE USO DEL SISTEMA

2.0. CUALQUIERA

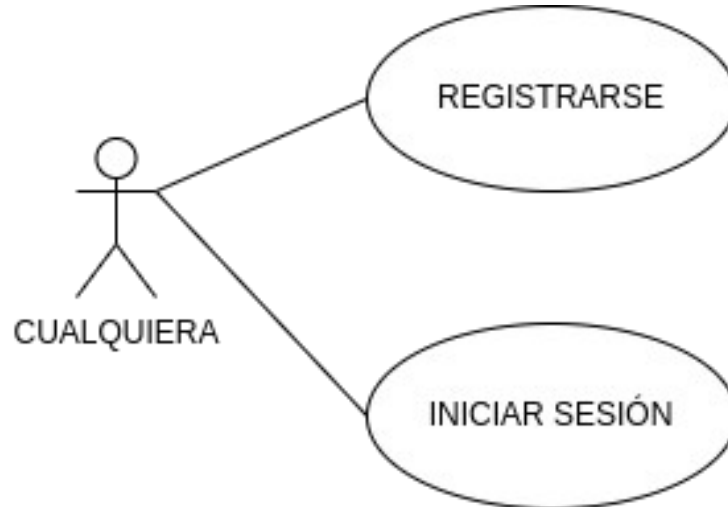


Figura 1: Casos de uso del actor "Cualquiera".

2.1. ADMINISTRADOR

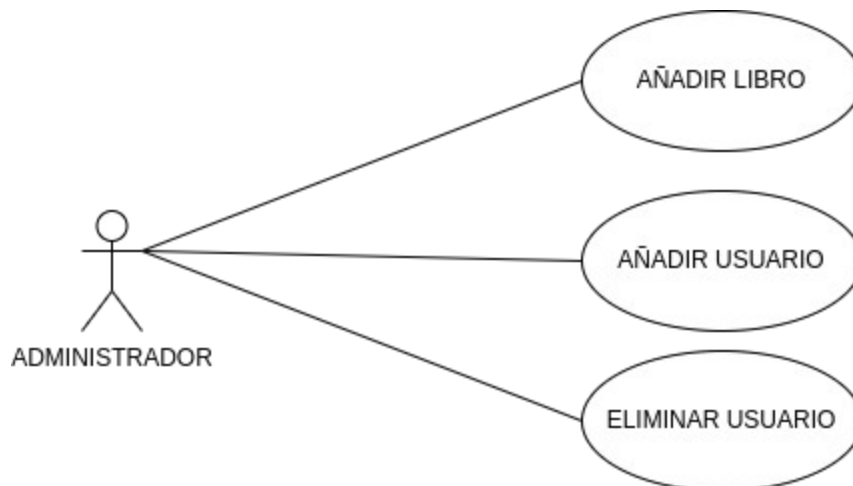


Figura 2: Casos de uso del actor "Administrador".

2.2. GENERAL

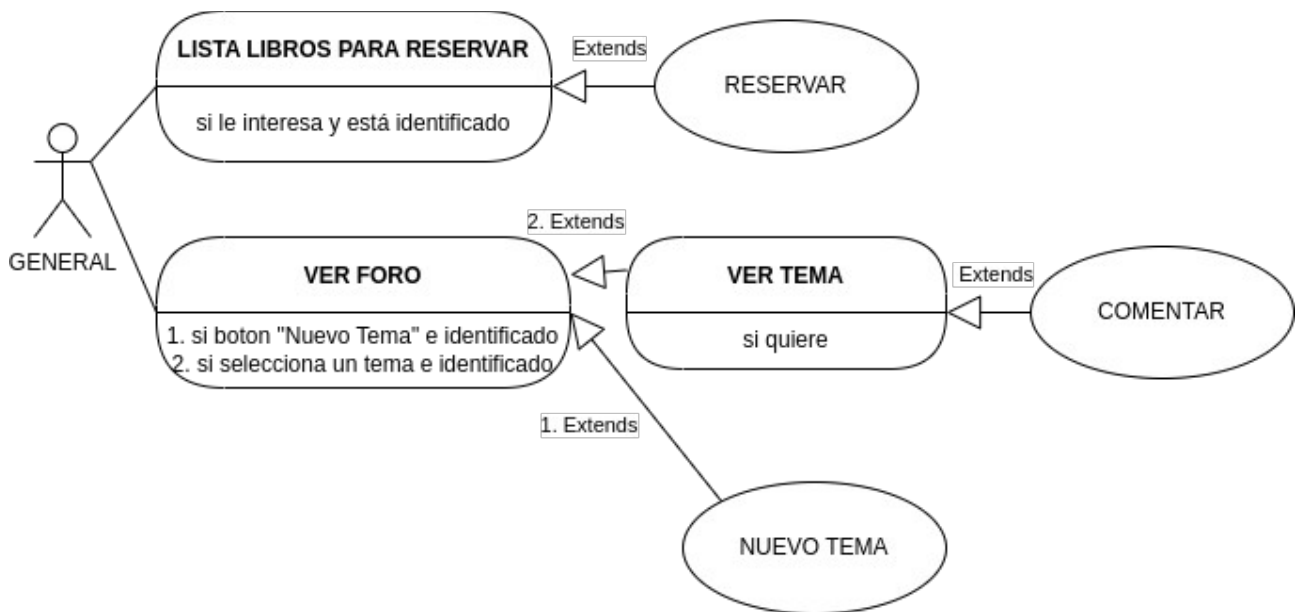


Figura 3: Casos de uso del actor "Usuario".

2.3. USUARIO



Figura 4: Casos de uso del actor "Usuario".

3. CASOS DE USO EXTENDIDOS DEL SISTEMA

3.0. COMIENZO



Figura 5: Casos de uso extendidos "Comienzo".

Nombre:	Registrarse
Descripción: Cualquier persona, independientemente del rol que tenga dentro del sistema, que acceda a la web del sistema podrá registrarse.	
Actores: Cualquiera	
Precondiciones: Ninguna.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: 1.- Se muestran en pantalla el botón de 'Registrarse'. [SI quiere registrarse] 2a.- Pulsa el botón registrarse. 2b.- Se redirige a la página de registro. 2c.- Introduce los datos solicitados en el formulario de registro. [SI los datos son correctos] 3a.- Se da de alta al usuario. 3b.- Se muestra la pantalla principal del usuario iniciado (home) [SI NO] 3c.- Se muestra un mensaje de error	
Poscondiciones: Si los datos son correctos el nuevo usuario queda registrado.	

Interfaz Gráfica:



A diagram of a login interface. At the top, the text "NOMBRE_SISTEMA" is centered. Below it are two input fields: "Usuario" and "Contraseña". Under these fields is a blue button labeled "Iniciar sesión". At the bottom, the text "Si no estás registrado:" is followed by a blue button labeled "Registrarse".

Figura 6: Página inicio.



A diagram of a registration form. At the top, the text "NOMBRE_SISTEMA" is centered. Below it are five input fields stacked vertically: "Nombre", "Apellido", "Email", "Usuario", and "Contraseña". At the bottom is a blue button labeled "Registrarse".

Figura 7: Formulario de registro al solicitar 'Registrarse'.



A diagram of a registration form showing an error state. At the top, the text "NOMBRE_SISTEMA" is centered. Below it are five input fields stacked vertically: "Nombre", "Apellido", "Email", "Usuario", and "Contraseña". Below the fields is a red error message: "Error: algún dato no cumple el formato". At the bottom is a blue button labeled "Registrarse".

Figura 8: Formulario de registro al solicitar 'Registrarse' e incumplir el formato de alguno de los campos.

Nombre:	Iniciar sesión
Descripción: Cualquier persona, podrá intentar iniciar sesión en el sistema. Si ya está registrado, accederá a su perfil. Si no está registrado, se mostrará un error.	
Actores: Cualquiera	
Precondiciones: Ninguna.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: 1.- Se muestra en pantalla el formulario de ‘Iniciar sesión’. [SI quiere iniciar sesión] 2a.- Escribe el usuario o mail. 2b.- Escribe la contraseña. 2c.- Pulsa el botón ‘Iniciar sesión’ [SI los datos son correctos y existe el usuario] 3a.- Se muestra la pantalla principal del usuario iniciado (home) [SI NO] 3b.- Se muestra un mensaje de error	
Poscondiciones: Si el usuario o mail y contraseña son correctos, se inicia sesión en el sistema. Y el actor pasa a ser ‘Usuario’, en caso de ser un usuario normal o ‘Administrador’, en caso de ser administrador.	
Interfaz Gráfica: <div style="display: flex; justify-content: space-around; align-items: flex-start;"> <div style="border: 1px solid black; padding: 10px; width: 45%; text-align: center;"> <p>NOMBRE_SISTEMA</p> <div style="margin-bottom: 5px;"><input type="text" value="Usuario"/></div> <div style="margin-bottom: 5px;"><input type="text" value="Contraseña"/></div> <div style="margin-bottom: 10px;"><input type="button" value="Iniciar sesión"/></div> <div>Si no estás registrado: <input type="button" value="Registrarse"/></div> </div> <div style="border: 1px solid black; padding: 10px; width: 45%; text-align: center;"> <p>NOMBRE_SISTEMA</p> <div style="margin-bottom: 5px;"><input type="text" value="Usuario"/></div> <div style="margin-bottom: 5px;"><input type="text" value="Contraseña"/></div> <div style="color: red; margin-bottom: 5px;">¡Datos incorrectos !</div> <div style="margin-bottom: 10px;"><input type="button" value="Iniciar sesión"/></div> <div>Si no estás registrado: <input type="button" value="Registrarse"/></div> </div> </div>	

Figura 9: Página inicio.

Figura 10: Introducir login incorrecto.

3.1. GESTIÓN RESERVAS

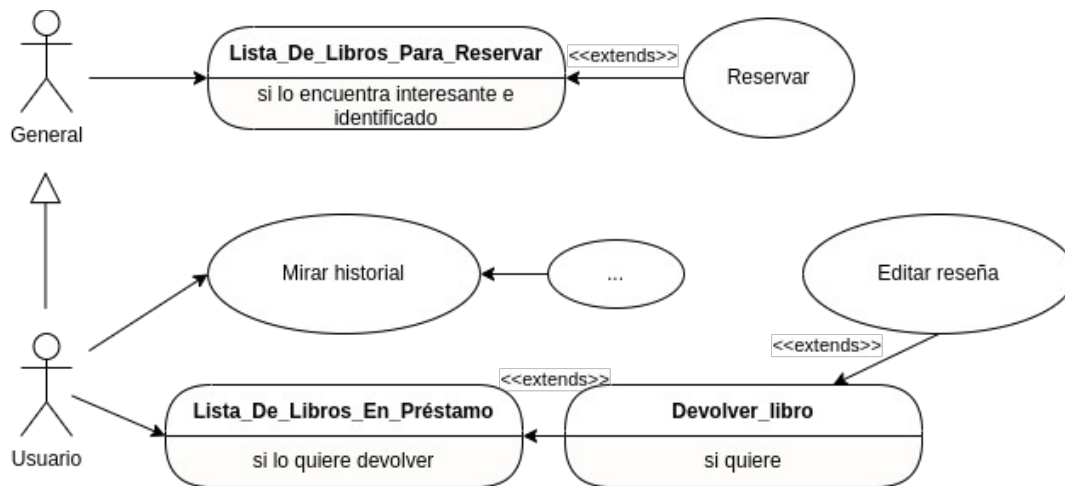


Figura 11: Casos de uso extendidos 'Gestión Reservas'.

Nombre:	Mirar historial (ampliado en el apartado "3.2 Reseñas").
Descripción:	Permite al usuario ver los libros que ha reservado.
Actores:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	1.- El Usuario entrará en "Gestión de Reserva" y pulsará en "Ver Historial de Reservas". (Figura 12) 2.- Se le muestra la lista de libros que ha adquirido junto con la fecha en la que fueron reservados. (Figura 13) [Si el usuario pulsa "Volver"] 3a.- Se vuelve al menú principal. (Figura 11)
Poscondiciones:	el usuario habrá visto el historial de las reservas.

Nombre:	Lista_De_Libros_Para_Reservar
Descripción:	Permite a los Usuarios ver la lista de libros que pueden ser reservados, y que en caso de desearlo, posibilita la reserva.
Actores:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario entrará en “Gestión de Reserva” y pulsará en “Lista De Libros Para Reservar”. (Figura 13)</p> <p>2.- Se le muestran los libros que puede reservar (si no hay libros en el sistema, se mostrará una lista vacía o pantalla en blanco) . (Figura 14)</p> <p> [Si el usuario pulsa en el libro]</p> <p> 3a.- Se accederá a la descripción del libro. (Figura 15)</p> <p> [Si el usuario pulsa “Reservar libro” y está identificado]</p> <p> 4aa.- <i>EXTENDS RESERVAR LIBRO</i></p> <p> [Si el usuario pulsa “Volver”]</p> <p> 4ab.- Se vuelve al menú principal. (Figura 12)</p>
Poscondiciones:	Si los datos eran correctos, se guardaran los cambios en el sistema, sino, dará error.

Nombre:	Reservar
Descripción: Cuando estemos leyendo los detalles de un libro, nos aparecerá un botón de reservar, que pinchando sobre él podremos reservar un libro para su lectura.	
Actores: Usuario	
Precondiciones: Estar identificado en el sistema como Usuario y el libro en cuestión tiene que existir.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: 1.- El Usuario pincha sobre el botón de “Reservar” (Figura 16). 2.- El libro quedará reservado siempre y cuando se cumplan los requisitos para poder reservar un libro, que en principio sólo serán “estar identificado en el sistema”, indicado en la precondición.	
Poscondiciones: El usuario habrá reservado un libro para su lectura.	

Nombre:	Lista_De_Libros_En_Préstamo
Descripción:	Permite a los Usuarios ver la lista de libros que tienen actualmente en préstamo, y que en caso de desearlo, posibilita la devolución.
Actores:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario entrará en “Gestión de Reserva” y pulsará en “Lista De Libros En Préstamo”. (Figura 13).</p> <p>2.- Se le muestran los libros que tiene en préstamos (si no hay libros en el sistema, se mostrará una lista vacía o pantalla en blanco) . (Figura 14)</p> <p> [Si el usuario pulsa en el Libro]</p> <p> 3a.- Se accederá a la descripción del libro. (Figura 15)</p> <p> [Si el usuario pulsa “Devolver libro”]</p> <p> 4aa.- <i>EXTENDS DEVOLVER LIBRO que INCLUDE EDITAR RESEÑA.</i></p> <p> [Si el usuario pulsa “Volver”]</p> <p> 4ab.- Se vuelve al menú principal. (Figura 12)</p>
Poscondiciones:	Si los datos eran correctos, se guardaran los cambios en el sistema, sino, dará error.

Nombre:	Devolver libro
Descripción:	Permite a los Usuarios devolver un libro que tienen actualmente en préstamo.
Actores:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario y que el libro exista.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	1.- El Usuario accede a en “Devolver libro” de la Figura 16. 2.- El libro en cuestión se devuelve automáticamente.
Poscondiciones:	El libro se devuelve al sistema.

Nombre:	Editar Reseña
Descripción: Permite al usuario editar una de las reseñas que ya tenga hechas o crear una nueva sobre algún libro que haya devuelto a la biblioteca.	
Actores: Usuario	
Precondiciones: Estar identificado en el sistema como Usuario y haber devuelto el libro a la biblioteca.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: 1.- El Usuario selecciona en su Menú Principal “Editar Reseña” (Figura 21) 2.- El usuario puede editar el comentario en la caja de texto y la puntuación dada al libro anteriormente. (Figura 22) [Si el usuario pulsa “Aceptar”] 3a.- Los cambios realizados se guardan y vuelve a las características del libro (Figura 22) [Si el usuario pulsa “Cancelar”] 3b.- Los cambios realizados no se guardan y vuelve a las características del libro (Figura 22)	
Poscondiciones: Si ha pulsado “Aceptar” la reseña se actualizará.	
Interfaz Gráfica: en el apartado “3.2 Reseñas”.	

Interfaz Gráfica:



Figura 12: Interfaz Gráfica.



Figura 13: Interfaz Gráfica.



Figura 14: Interfaz Gráfica.

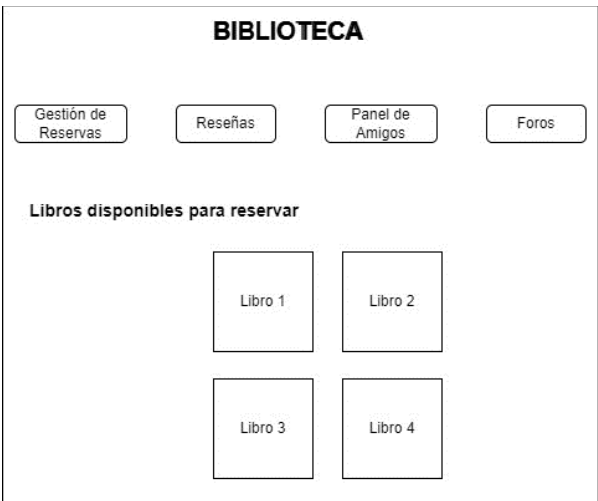


Figura 15: Interfaz Gráfica.

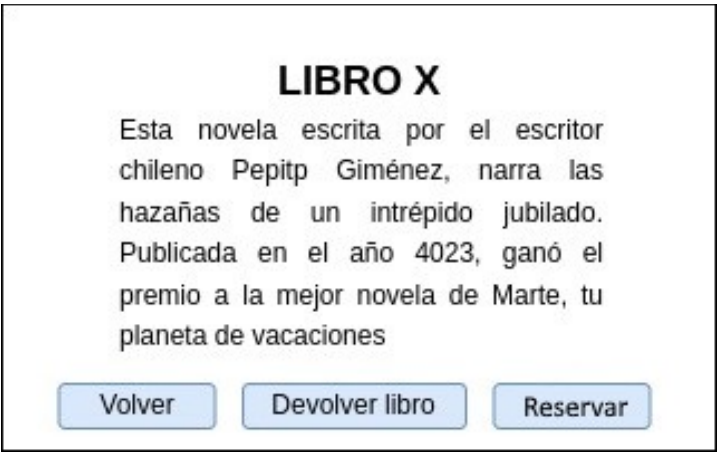


Figura 16: Interfaz Gráfica.

3.2. RESEÑAS



Figura 17: Casos de uso extendidos "Reseñas".

Nombre:	Mirar historial
Descripción:	Acceder al historial de los libros que ya han sido devueltos, puesto que desde aquí, los usuarios podrán puntuar y comentar un libro. Esta puntuación y comentario se podrá cambiar en el futuro.
Actores:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario y haber devuelto el libro sobre el que se va a hacer la reseña.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario selecciona en su Menú Principal “Reseñas” para ver los libros que ha leído y devuelto a la biblioteca. (Figura 18).</p> <p>2.- Se le muestran los libros que ha devuelto a la biblioteca. (Figura 19)</p> <p>[Si el usuario pulsa sobre el nombre del libro]</p> <p>3a.- Se accederá a las características de ese libro (Figura 20)</p> <p>[Si el usuario pulsa “Editar Reseña”]</p> <p>3aa.- <i>EXTENDS EDITAR_RESEÑA (Pág. 14)</i></p> <p>[Si el usuario pulsa “Volver”]</p> <p>3ab.- Se vuelve al historial (Figura 19)</p>
Poscondiciones:	se ha leído el historial de libros devueltos.

Interfaz Gráfica:

BIBLIOTECA

Gestión de Reservas

Reseñas

Panel de Amigos

Foros

Te gustaría ver:

Ejemplo

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Rol: USUARIO/ADMIN

Figura 18: Interfaz gráfica.

BIBLIOTECA

Gestión Reservas

Reseñas

Panel de amigos

Foros

LIBROS DEVUELTOS:

LIBRO

LIBRO

LIBRO

LIBRO

LIBRO

LIBRO

Rol: USUARIO/ADMIN

Figura 19: Interfaz gráfica.

BIBLIOTECA

Gestión Reservas

Reseñas

Panel de amigos

Foros

LIBRO:

FOTO LIBRO

VALORACIÓN DEL 1 AL 10
RESEÑA EJEMPLO:
Muy interesante el libro, recomendado a cualquier fan del género de ficción

VOLVER

EDITAR RESEÑA

Rol: USUARIO/ADMIN

Figura 20: Interfaz gráfica.

Nombre:	Editar Reseña
Descripción: Permite al usuario editar una de las reseñas que ya tenga hechas o crear una nueva sobre algún libro que haya devuelto a la biblioteca.	
Actores: Usuario	
Precondiciones: Estar identificado en el sistema como Usuario y haber devuelto el libro a la biblioteca.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: <ol style="list-style-type: none"> 1.- El Usuario selecciona en su Menú Principal “Editar Reseña” (Figura 21) 2.- El usuario puede editar el comentario en la caja de texto y la puntuación dada al libro anteriormente. (Figura 22) <ol style="list-style-type: none"> [Si el usuario pulsa “Aceptar”] 3a.- Los cambios realizados se guardan y vuelve a las características del libro (Figura 22) [Si el usuario pulsa “Cancelar”] 3b.- Los cambios realizados no se guardan y vuelve a las características del libro (Figura 22) 	
Poscondiciones: Si ha pulsado “Aceptar” la reseña se actualizará.	
Interfaz Gráfica: <div data-bbox="517 1055 1096 1449" data-label="Form"> </div> <p><i>Figura 21: Interfaz gráfica.</i></p> <div data-bbox="512 1527 1101 1926" data-label="Form"> </div> <p><i>Figura 22: Interfaz gráfica.</i></p>	

3.3. RED DE AMIGOS



Figura 23: Casos de uso extendidos "Red de amigos".

Nombre:	Solicitar Amigo
Descripción:	El usuario puede escribir un usuario de un amigo, y el programa solicitará a ese usuario ser su amigo. Cuando el otro usuario abra el programa lo verá, y puede decidir si aceptarlo o rechazarlo.
Actor:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario selecciona en su Menú Principal “Panel de Amigos” para ver el panel de amigos. (Figura 24)</p> <p>2.- Se le muestran varias opciones: Ver la lista de amigos, solicitar ser amigo a otro usuario, ver las solicitudes actuales. (Figura 25)</p> <p>[Si el usuario pulsa sobre “Solicitar Amigo”]</p> <p>3a.- Saldrá una casilla donde introducir el nombre del usuario para enviarle la solicitud.</p> <p>[Si el usuario escribe un nombre]</p> <p>[Si el usuario pulsa el botón de aceptar]</p> <p>[Si el nombre del usuario escrito existe]</p> <p>6a.- Se acepta el amigo.</p> <p>[Si el nombre escrito no existe]</p> <p>6b.- Se avisará que no existe el usuario.</p> <p>[Si el usuario pulsa fuera del pop-up]</p> <p>5b.- Se vuelve al menú principal. (Figura 25)</p> <p>[Si el usuario pulsa “salir”]</p> <p>3b.- Se saldrá de la interfaz. (Figura 24)</p>

Poscondición: Cuando se acepta una solicitud, el usuario solicitante y el usuario que acepta son automáticamente amigos, si se rechaza no lo serán.

Interfaz Gráfica:

The main menu interface is titled "BIBLIOTECA". It features four buttons: "Gestión de Reservas", "Reseñas", "Panel de Amigos", and "Foros". Below these buttons is a search bar with the placeholder text "Te gustaría ver:". Under the search bar, the word "Ejemplo" is displayed in a large font, followed by a paragraph of Lorem Ipsum text. In the bottom right corner, the role "Rol: USUARIO/ADMIN" is indicated.

BIBLIOTECA

Gestión de Reservas Reseñas Panel de Amigos

Foros

Te gustaría ver:

Ejemplo

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Rol: USUARIO/ADMIN

Figura 24: Menú Principal

The friends menu interface is titled "BIBLIOTECA" and "RED DE AMIGOS". It features three buttons: "SOLICITAR AMIGO" (which includes a sub-button "Introduzca Nombre"), "GESTIONAR SOLICITUDES", and "VER LISTA DE AMIGOS".

BIBLIOTECA
RED DE AMIGOS

SOLICITAR AMIGO

Introduzca Nombre

GESTIONAR SOLICITUDES

VER LISTA DE AMIGOS

Figura 25: Menú de amigos.

Nombre:	Gestionar Solicitudes
Descripción: Permite poder gestionar las solicitudes entrantes de amistad, es decir, aceptar o rechazar, que como previamente se ha explicado, el usuario puede escribir un usuario de un amigo, y el programa solicitará a ese usuario ser su amigo. Cuando el otro usuario abra el programa lo verá, y puede decidir si aceptarlo o rechazarlo.	
Actor: Usuario	
Precondiciones: Estar identificado en el sistema como Usuario.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: 1.- El Usuario selecciona en su Menú Principal “Panel de Amigos” para ver el panel de amigos (Figura 26). 2.- Se le muestran varias opciones: Ver la lista de amigos, solicitar ser amigo a otro usuario, ver las solicitudes actuales. (Figura 27) [Si el usuario pulsa sobre “Gestionar Solicitudes”] 3a.- Se accederá a una pestaña con todas las solicitudes. (Figura 28) [Si el usuario acepta una solicitud] 4aa.- Se imprimirá por pantalla “Solicitud Aceptada”. [Si el usuario elimina una solicitud] 4ab.- Se imprimirá por pantalla “Solicitud Rechazada”. [Si el usuario pulsa “Volver”] 4ac.- Se volverá al menú principal. (Figura 27) [Si el usuario pulsa “Salir”] 3b.- Se saldrá de la interfaz. (Figura 26)	
Poscondición: El Usuario acepta o rechaza las solicitudes. Cuando se acepta una solicitud, el usuario solicitante y el usuario que acepta son automáticamente amigos, si se rechaza no lo serán.	
Interfaz Gráfica: <div style="border: 1px solid black; padding: 10px; text-align: center;"> <p>BIBLIOTECA</p> <div style="display: flex; justify-content: space-around; margin-bottom: 10px;"> <div style="border: 1px solid black; padding: 5px 10px;">Gestión de Reservas</div> <div style="border: 1px solid black; padding: 5px 10px;">Reseñas</div> <div style="border: 1px solid black; padding: 5px 10px;">Panel de Amigos</div> </div> <div style="border: 1px solid black; padding: 5px 10px; margin: 0 auto; width: 100px;">Foros</div> <div style="margin-top: 20px;"> <div style="border: 1px solid black; padding: 2px 5px; display: inline-block;">Te gustaría ver:</div> </div> <p style="font-size: 1.2em; font-weight: bold; margin-top: 10px;">Ejemplo</p> <p style="font-size: 0.8em; margin-top: 5px;">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p> <p style="text-align: right; font-size: 0.8em; margin-top: 10px;">Rol: USUARIO/ADMIN</p> </div>	

Figura 26: Menú Principal.

BIBLIOTECA
RED DE AMIGOS

SOLICITAR AMIGO

GESTIONAR SOLICITUDES

VER LISTA DE AMIGOS

Figura 27: Menú de amigos.

BIBLIOTECA
RED DE AMIGOS (GESTIONAR SOLICITUDES)

Usuario5	Usuario7
<input type="button" value="Aceptar"/>	<input type="button" value="Aceptar"/>
<input type="button" value="Rechazar"/>	<input type="button" value="Rechazar"/>
Usuario9	Usuario13
<input type="button" value="Aceptar"/>	<input type="button" value="Aceptar"/>
<input type="button" value="Rechazar"/>	<input type="button" value="Rechazar"/>

Figura 28: Gestionar Solicitudes.

Nombre:	Ver Lista Amigos
Descripción: Se ve la lista de todos los amigos de un usuario. Además, se puede entrar a un perfil de uno de ellos y se ve más en detalle.	
Actor: Usuario	
Precondiciones: Estar identificado en el sistema como Usuario.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: 1.- El Usuario selecciona en su Menú Principal “Panel de Amigos” para ver el panel de amigos.(Figura 29) 2.- Se le muestran varias opciones: Ver la lista de amigos, solicitar ser amigo a otro usuario, ver las solicitudes actuales. (Figura 30) [Si el usuario pulsa sobre “ver la lista de amigos”] 3a.- Se accederá a una pestaña con la lista de amigos. (Figura 31) [Si el usuario pulsa en un amigo] 4aa.- <i>EXTENDS Ver_Perfil_Amigo (se ve el perfil del amigo, en el que el sistema muestra sus libros y su usuario, pero no se puede realizar ninguna acción más que volver hacia atrás, por tanto no se va a desarrollar ese subcaso de uso ya que sería trivial).</i> [Si el usuario pulsa “Volver”] 4ab.- Se vuelve al menú principal. (Figura 30) [Si el usuario pulsa “Salir”] 3b.- Se saldrá de la interfaz. (Figura 29)	
Poscondiciones: Se ha accedido y visto los amigos que se tienen.	
Interfaz Gráfica: <div style="border: 1px solid black; padding: 10px; text-align: center; margin: 10px auto; width: 60%;"> <p>BIBLIOTECA</p> <div style="display: flex; justify-content: space-around; margin-bottom: 10px;"> Gestión de Reservas Reseñas Panel de Amigos </div> Foros <div style="margin-top: 10px;"> Te gustaría ver: </div> <p>Ejemplo</p> <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. </p> <p style="text-align: right;">Rol: USUARIO/ADMIN</p> </div>	

Figura 29: Menú Principal.

BIBLIOTECA
RED DE AMIGOS

SOLICITAR AMIGO

Introduzca Nombre

GESTIONAR SOLICITUDES

VER LISTA DE AMIGOS

Figura 30: Menú de amigos.

BIBLIOTECA
RED DE AMIGOS (LISTA AMIGOS)

Amigo1
Lorem ipsum dolor sit amet,
consectetur adipisicing elit,
sed do eiusmod tempor
incididunt ut labore et dolore
magna aliqua.

Amigo2
Lorem ipsum dolor sit amet,
consectetur adipisicing elit,
sed do eiusmod tempor
incididunt ut labore et dolore
magna aliqua.

Amigo3
Lorem ipsum dolor sit amet,
consectetur adipisicing elit,

Amigo4
Lorem ipsum dolor sit amet,
consectetur adipisicing elit,

Figura 31: Lista de Amigos.

3.4. ADMINISTRADOR

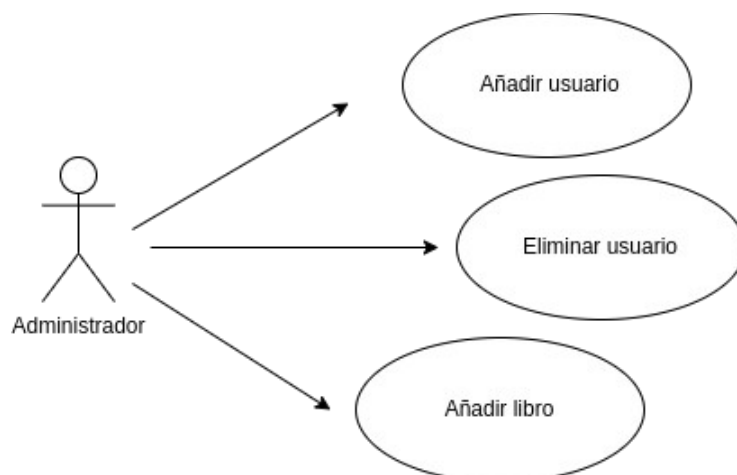


Figura 32: Casos de uso extendidos "Administrador".

Nombre:	Añadir usuario
Descripción:	Permite a los administradores crear usuarios.
Actores:	Administrador
Precondiciones:	Estar identificado en el sistema como Administrador y ha accedido al panel de administración clicando en “Rol: admin” en la pantalla de inicio (Figura 33).
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario con privilegios de administrador accede a su panel de control de administrador. (Figura 33)</p> <p>2.- Se le muestran los botones de añadir usuario, eliminar usuario y añadir libro junto a los campos de texto necesarios para realizar las acciones (Figura 34).</p> <p>[Si el administrador introduce un usuario]</p> <p>[Si el usuario existe]</p> <p>3a.- La cuenta será creada.</p> <p>[Si el usuario no existe]</p> <p>3b.- Saltará un mensaje de error</p> <p>[Si se clicca en “salir”]</p> <p>3c.- Se sale al menú principal. (Figura 34)</p>
Poscondiciones:	Si los datos son correctos, se guardan los cambios en el sistema y por tanto, un nuevo usuario queda añadido o dado de alta; si no, da error.

Interfaz Gráfica:

BIBLIOTECA

Gestión de Reservas

Reseñas

Panel de Amigos

Foros

Te gustaría ver:

Ejemplo

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Rol: USUARIO/ADMIN

Figura 33: Menú Principal.

Añadir usuario

Eliminar usuario

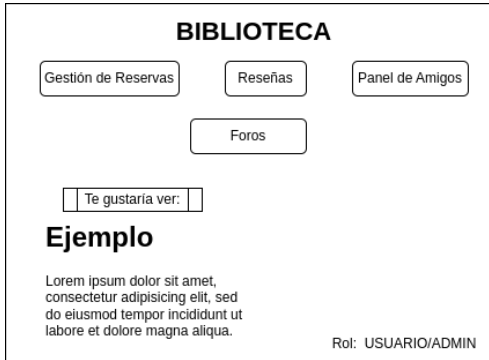
Añadir libro

usuario

contraseña

Título del libro


Figura 34: Panel de Administrador.

Nombre:	Eliminar usuario
Descripción: Permite a los administradores eliminar usuarios.	
Actores: Administrador	
Precondiciones: Estar identificado en el sistema como administrador y a accedido al panel de administración clicando en “Rol: admin” en la pantalla de inicio (Figura 35).	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: 1.- El Usuario con privilegios de administrador accede a su panel de control de administrador. (Figura 35) 2.- Se le muestran los botones de añadir usuario, eliminar usuario y añadir libro junto a los campos de texto necesarios para realizar las acciones. (Figura 36) [Si el administrador introduce un usuario] [Si la cuenta existe] 3a.- La cuenta será eliminada. [Si la cuenta no existe] 3b.- Sale un mensaje de error. [Si se clicla en “salir”] 3c.- Se sale al menú principal. (Figura 36)	
Poscondiciones: Si los datos son correctos, se guardarán los cambios en el sistema y el usuario quedará eliminado; si no, dará error.	
Interfaz Gráfica:  <p style="text-align: center;"><i>Figura 35: Menú Principal.</i></p>	

The diagram illustrates the Administrator Panel layout. It features a central container with the following elements:

- User Management:**
 - Buttons: "Añadir usuario" (Add user) and "Eliminar usuario" (Delete user).
 - Input fields: "usuario" (user) and "contraseña" (password).
- Book Management:**
 - Buttons: "Añadir libro" (Add book).
 - Input field: "Título del libro" (Book title).

Figura 36: Panel de Administrador.

Nombre:	Añadir libro
Descripción:	Permite a los administradores Añadir libros.
Actores:	Administrador
Precondiciones:	Estar identificado en el sistema como administrador y a accedido al panel de administración clicando en “Rol: admin” en la pantalla de inicio (Figura 36).
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario con privilegios de administrador accede a su panel de control de administrador. (Figura 37).</p> <p>2.- Se le muestran los botones de añadir usuario, eliminar usuario y añadir libro junto a los campos de texto necesarios para realizar las acciones (Figura 38).</p> <p>[Si el administrador introduce un libro] [Si el libro existe] 3a.- El libro se añade al sistema. [Si el libro no existe] 3b.- Sale un mensaje de error.</p> <p>[Si se clicla en “salir”] 3c.- Se sale al menú principal. (Figura 38).</p>
Poscondiciones:	Si los datos eran correctos, se guardaran los cambios en el sistema, sino, dará error
Interfaz Gráfica:	 <p><i>Figura 37: Menú principal.</i></p>

Añadir usuario	email
Eliminar usuario	contraseña
	rol
Añadir libro	id
	Título del libro
	Número de copias
	Descripción
	id Autor

Figura 38: Panel de Administrador.

3.5. FORO

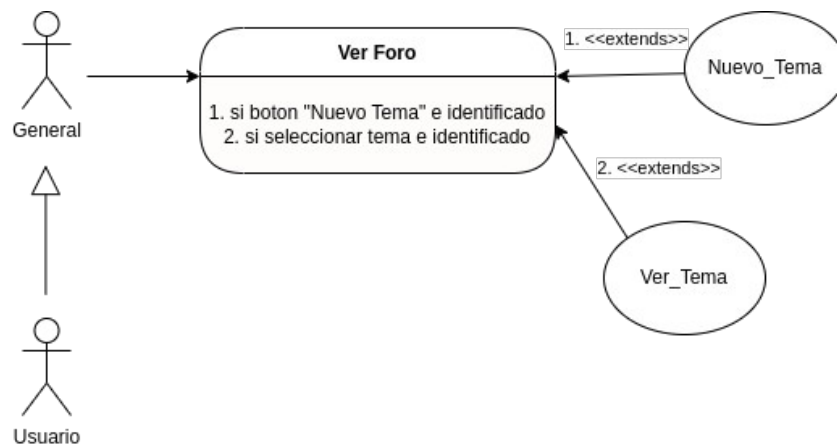


Figura 39: Casos de uso extendido "Foro".

Nombre:	VER FORO
Descripción:	Permite ver el foro, donde cualquier usuario puede crear un tema y puede ver cualquiera de los temas, ya sea para leerlo o comentarlo.
Actores:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario selecciona en su Menú Principal “Foro” para ver los diferentes temas. (Figura 40).</p> <p>2.- Se le muestran la lista con todos los temas.</p> <p>[Si el usuario pulsa sobre el botón “Nuevo tema” y está identificado]</p> <p>3a.- <i>EXTENDS NUEVO_TEMA</i></p> <p>[Si el usuario pulsa sobre un tema y está identificado]</p> <p>3b.- <i>EXTENDS VER_TEMA</i></p> <p>[Si el usuario pulsa “Volver”]</p> <p>3c.- Se vuelve a la lista de temas</p>
Poscondiciones:	Ninguna.
Interfaz Gráfica:	

Figura 40: Interfaz gráfica.

FORO

Menu principal
Nuevo tema

Tema de conversación 1

Tema de conversación 2

Tema de conversación 3

Rol: USUARIO/ADMIN

Figura 41: Interfaz gráfica

Nombre: NUEVO_TEMA

Descripción: Permite al cliente crear un nuevo tema para añadirlo al foro.

Actores: Usuario

Precondiciones: Estar identificado en el sistema como Usuario.

Requisitos no funcionales: Ninguno.

Flujo de Eventos:

- 1.- Se le pide que introduzca el tema del que hablar.
- 2.- Se publica el tema (Figura 42).

Poscondiciones: Si se publica el tema aparecerá en la pagina de foros.

Interfaz Gráfica:

FORO

Introduzca el nuevo tema

Publicar

Rol: USUARIO/ADMIN

Figura 42: Interfaz gráfica.

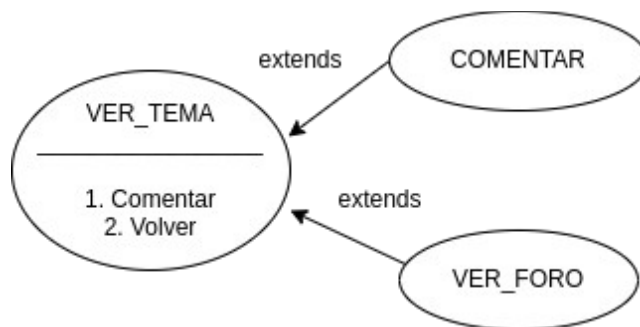


Figura 43: Ver_Tema

Nombre: VER_TEMA

Descripción: Permite al cliente ver el tema junto con las respuestas de la gente.

Actores: Usuario

Precondiciones: Estar identificado en el sistema como Usuario.

Requisitos no funcionales: Ninguno.

Flujo de Eventos:

1.- Se le muestra el tema junto con las respuestas a este. (Figura 44).

[Si el usuario selecciona “Comentar”]

2a.- Se le pide que introduzca respuesta al tema y luego esta se publica.

[Si el usuario selecciona “Foro”]

2b.- Se vuelve a la pagina del foro.

Poscondiciones: Ninguna.

Interfaz Gráfica:

FORO

Volver al foro

TEMA ORIGINAL

Respuesta 1

Respuesta 2

Respuesta 3

Añadir respuestas

Rol: USUARIO/ADMIN

Figura 44: Interfaz gráfica.

Nombre:	COMENTAR
Descripción: Permite al cliente opinar sobre un tema	
Actores: Usuario	
Precondiciones: Estar identificado en el sistema como Usuario.	
Requisitos no funcionales: Ninguno.	
Flujo de Eventos: <ol style="list-style-type: none"> 1.- Se le pide que introduzca su comentario (Figura 45). 2.- Se publica la respuesta. 3.- Se vuelve a la pagina del tema. 	
Poscondiciones: Si se comenta un tema el comentario se guardara y aparecerá en la pagina del tema.	
Interfaz Gráfica: <div data-bbox="454 884 1093 1355" data-label="Form"> </div>	
<i>Figura 45: Interfaz gráfica.</i>	

3.6. RECOMENDACIONES DEL SISTEMA

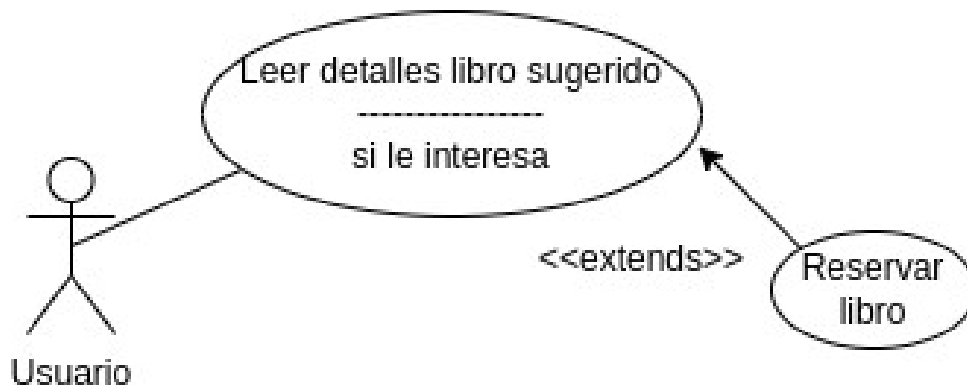


Figura 46: Casos de uso extendido "Recomendaciones del sistema".

Nombre:	Leer detalles libro sugerido
Descripción:	En el menú principal del usuario identificado nos aparecerán directamente los libros recomendados por el sistema. Pinchando en los libros leeremos más acerca del mismo y nos mostrará los detalles del libro. Además, en caso de estar interesados, este caso de uso podrá dar lugar a reservar libro.
Actores:	Usuario.
Precondiciones:	Estar identificado en el sistema y tiene que haber libros en el sistema y existir el libro sobre el que se quiere conocer más.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario observa en su menú principal ("Te gustaría ver") la lista de los libros recomendados por el sistema (Figura 47).</p> <p>2.- Si alguno de los libros le llama la atención, pinchará sobre él para acceder a la información del libro.</p> <p>[Si el usuario pulsa sobre el libro]</p> <p>3a.- Se accederá a la información de libro (Figura 48)</p> <p>[Si el usuario pulsa "Reservar libro"]</p> <p>4a.-<i>EXTENDS RESERVAR LIBRO</i> (en los casos de uso de <i>Gestión de Reservas</i>)</p> <p>[Si el usuario pulsa "Volver"]</p> <p>4b.- Se vuelve al menú principal (Figura 47)</p>
Poscondiciones:	El usuario habrá leído los detalles del libro sugerido que le ha interesado.

Nombre:	Reservar
Descripción:	Cuando estemos leyendo los detalles de un libro, nos aparecerá un botón de reservar, que pinchando sobre él podremos reservar un libro para su lectura.
Actores:	Usuario
Precondiciones:	Estar identificado en el sistema como Usuario y tiene que existir el libro.
Requisitos no funcionales:	Ninguno.
Flujo de Eventos:	<p>1.- El Usuario pincha sobre el botón de “Reservar” (Figura 48).</p> <p>2.- El libro quedará reservado siempre y cuando se cumplan los requisitos para poder reservar un libro, que en principio sólo serán “estar identificado en el sistema”, indicado en la precondición.</p>
Poscondiciones:	El usuario habrá reservado un libro para su lectura.

Interfaz Gráfica:

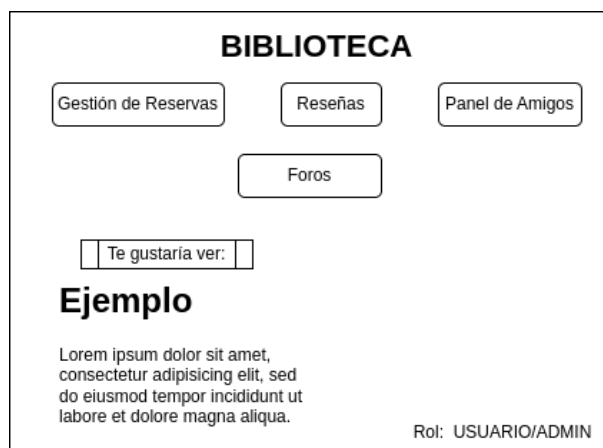


Figura 47: Menú Principal.



Figura 48: Detalles del libro.

4. JERARQUÍA DE ACTORES

Tendremos tres actores: cualquiera, usuario y administrador. Este último, además de sus funciones específicas también podrá realizar aquellas correspondientes al usuario, por lo que hereda de él.

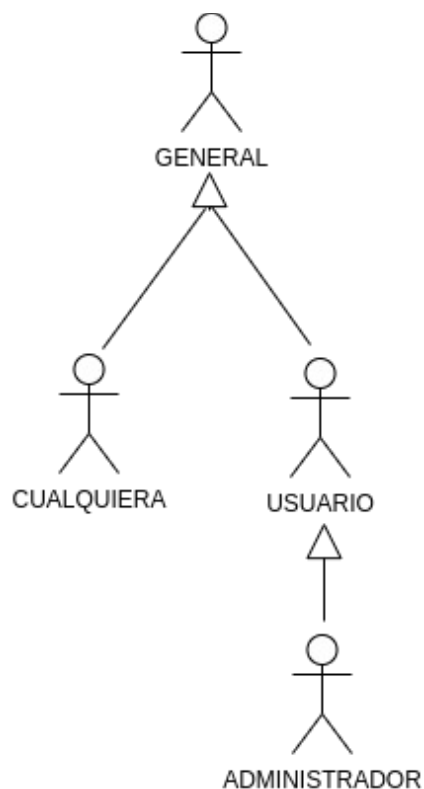


Figura 49: Jerarquía de actores.

5. MODELO DE DOMINIO

El diagrama del modelo de Dominio quedaría de esta manera, donde se pueden ver todas las relaciones y entidades. Más adelante se desarrollan todas las entidades, atributos y relaciones por separado, pero la vista general se puede observar en la siguiente figura:

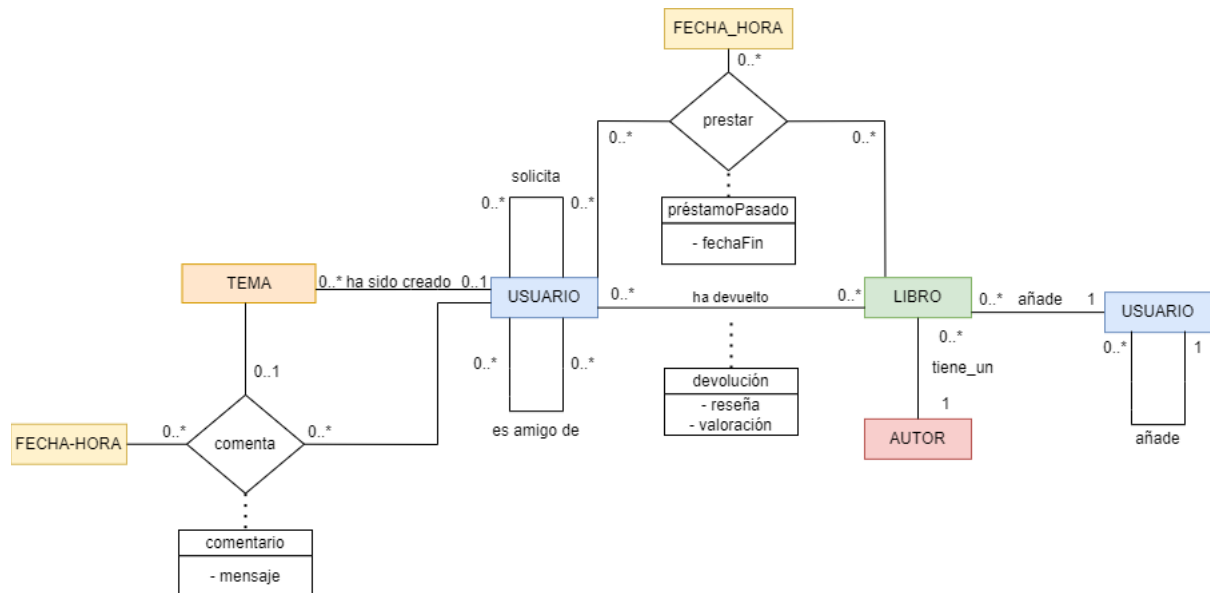


Figura 50: Diagrama completo modelo de dominio.

5.0. ENTIDADES Y SUS ATRIBUTOS

Tendremos varias entidades: usuario, fecha_hora, libro, tema y autor. Los atributos se muestran en la siguiente figura, y las relaciones se explicarán tanto en el modelo completo como en los apartados separados por funcionalidad.

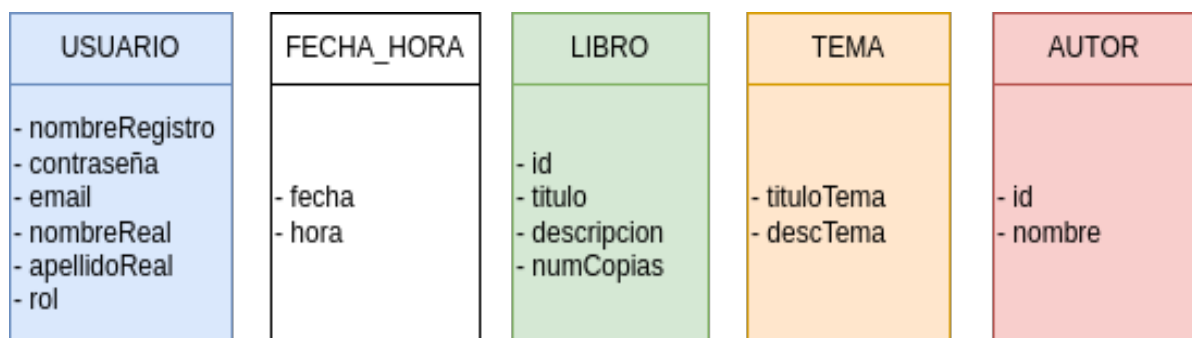


Figura 51: Entidades y atributos.

5.1. FORO

Un usuario, en el foro, puede o no haber creado un tema.

Un comentario es una relación entre un tema, un usuario y la hora en la que se escribió. Un usuario puede comentar un único tema en una fecha y hora determinada, y un tema puede ser comentado por múltiples usuarios, en diferentes horas.

Un comentario es a la vez un comentario y una respuesta, ya que el foro no tiene respuestas individuales.

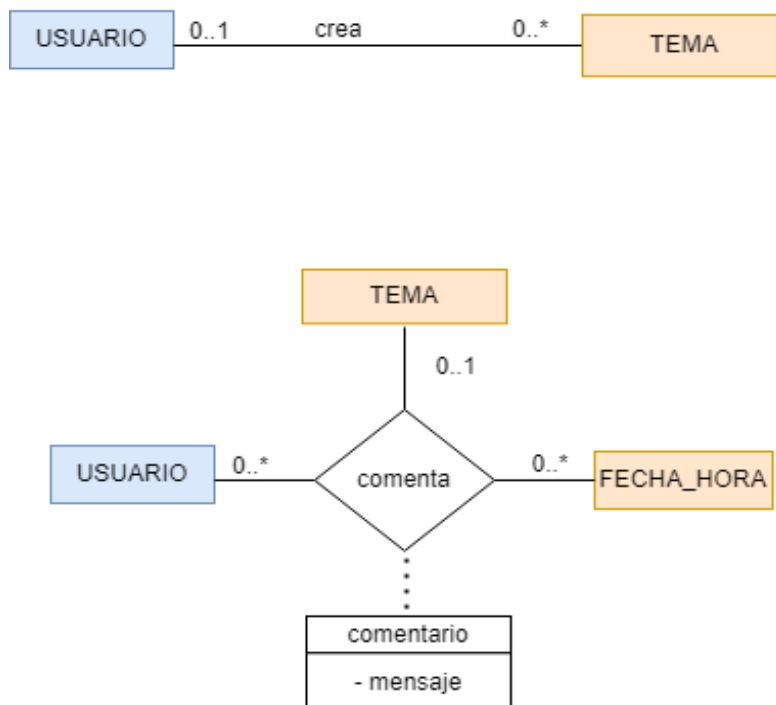


Figura 52: Crear tema y comentar.

5.2. LIBRO

Un libro se puede prestar tantas veces a un mismo usuario como se quiera (siempre y cuando haya suficientes copias disponibles para prestar) pero en diferentes fechas y horas. Asimismo, en la relación se guardará el atributo *fechaFin*, que permitirá saber si el libro está actualmente en préstamo (*fechaFin* es vacía) o si se ha devuelto.

Por tanto, el libro se podrá prestar más de una vez y el elemento diferenciador será la entidad FECHA_HORA, que marca la fecha y la hora del préstamo (Figura 50). El atributo *fechaFin* se establecerá solo cuando se devuelva el libro.

Además, por cada tupla USUARIO-LIBRO que se haya devuelto, se podrá almacenar una reseña, una única reseña por libro, es decir, que si se reserva varias veces, sólo habrá una reseña.

En adición a ello, existe una última relación AUTOR-LIBRO, ya que al ser una entidad añadida recientemente, es necesaria una relación entre los dos. Un libro tiene un autor y un autor puede ser autor de varios libros.

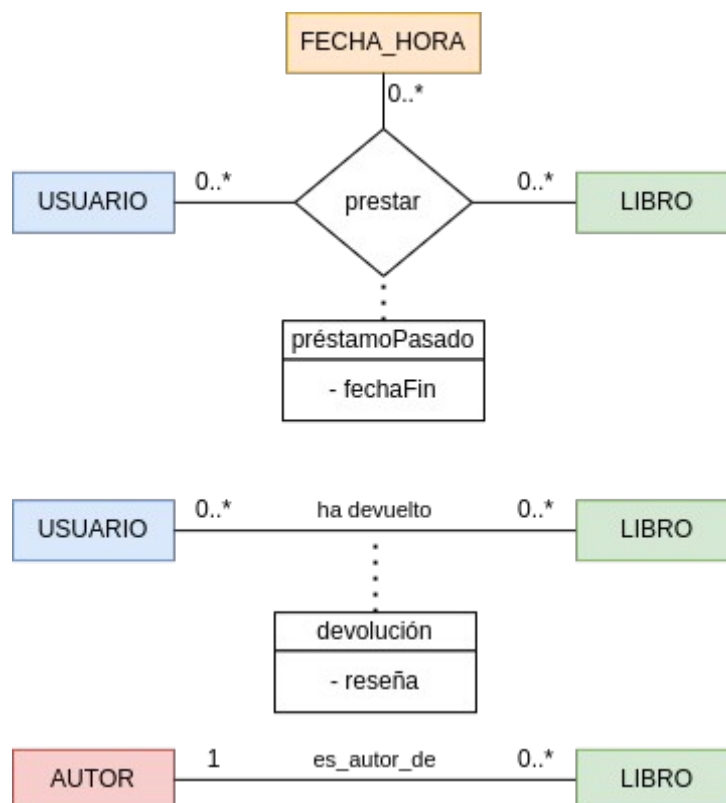


Figura 53: Prestar y devolver libro, reseña, autor del libro.

5.3. ADMINISTRADOR

Un usuario con rol de administrador podrá añadir tanto nuevos usuarios como nuevos libros. Se podrán añadir tantos usuarios y libros como desee el administrador, pero cada libro y usuario sólo podrá ser añadido por un único administrador.

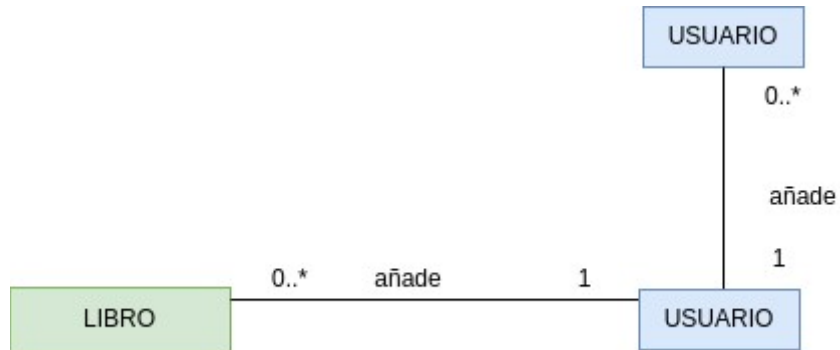


Figura 54: Añadir usuario y libro.

5.4. RED DE AMIGOS

Un usuario puede tener múltiples amigos, y esos amigos pueden tener a su vez múltiples amigos. Como el amigo de un usuario es un usuario también, la relación es de usuario a usuario.

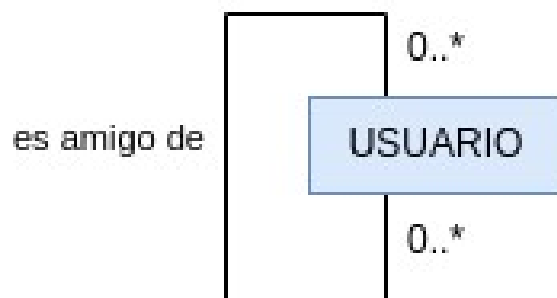


Figura 55: Red de amigos.

6. PLAN DE PRUEBAS

A continuación se desarrollan las pruebas de caja negra por cada caso de uso descrito en el apartado dos y tres de este documento. En total son doce casos de uso.

Las pruebas de caja negra consisten en verificar los “outputs” del sistema apartir de unos “inputs”, sin tener en cuenta el código. Nos permiten buscar y corregir los errores que pueda contener el propio software.

aa) REGISTRARSE (JAVIER)

<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
1.1	Ha introducido los datos en el formato correcto y se ha hecho “submit” correctamente.	Alta de usuario y página de inicio de sesión.
1.2	Ha introducido los datos en el formato correcto y se ha producido algún error durante el “submit”.	Mensaje de error y “vuelve a intentarlo” o “inténtalo más tarde”, nunca el código del error interno (por seguridad).
1.3	Ha introducido el nombre en formato incorrecto.	Mensaje de error: “nombre incorrecto”.
1.4	Ha introducido los apellidos en formato incorrecto.	Mensaje de error: “apellidos incorrectos”.
1.5	Ha introducido caracteres no permitidos en el campo de usuario.	Mensaje de error: “usuario no válido”.
1.6	Ha introducido caracteres no permitidos en el campo de contraseña.	Mensaje de error: “contraseña no válida”.
1.7	Ha introducido sólo caracteres permitidos en el campo de contraseña, pero no cumple los requisitos de seguridad.	Mensaje de error: “contraseña insegura, prueba a...”

ab) INICIAR SESIÓN (JAVIER)

<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
2.1	El usuario y contraseña son válidos.	Acceso al sistema.
2.2	Usuario correcto y contraseña incorrecta.	Mensaje de error.
2.3	Usuario incorrecto y contraseña correcta.	Mensaje de error.

En los apartados aa) y ab) los casos de uso son los títulos de los mismos.

b) GESTIÓN RESERVAS (SERGIO)

<u>CASO DE USO</u>	<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
-----	1a	Pulsar botón Lista De Libros Para Reservar estando identificado	Te muestra los libros que se pueden reservar.
	1b	Pulsar botón Lista de Libros para Reservar no estando identificado	Te muestra los libros que se pueden reservar.
ListaLibrosPara Reservar/ MirarHistorial/ VerListaLibros EnPréstamo	2a	Pulsar en libro y no se encuentra en la BD	Muestra un mensaje de error
	2b	Pulsar libro y se encuentra en la BD	Muestra las características del libro
Reservar	3a	Pulsar botón reservar y se tiene el libro reservado	Te muestra un mensaje indicando que ya se tiene el libro reservado/en préstamo
	3b	Pulsar botón reservar y no se tiene el libro reservado	Reserva el libro
Mirar Historial	4a	Pulsar botón Historial	Muestra la lista de libros que ha adquirido junto con la fecha en la que fueron reservados
VerListaLibros EnPréstamo	5b	Pulsar botón Ver Lista De Libros En Préstamo	Muestra la lista de Libros que se tienen en préstamo
Devolver libro	6	Estando en un libro pulsar el botón Devolver	Devuelve el libro

c) RESEÑAS (ADRIÁN)

<u>CASO DE USO</u>	<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
-----	1a	Pulsar botón de reseñas estando autenticado	Que el sistema acceda al apartado de reseñas. Automáticamente se muestran los libros que el usuario ha devuelto al sistema.
	1b	Pulsar botón de reseñas sin estar autenticado	Mensaje de error o aviso, indicando que el usuario tiene que estar identificado para acceder.
Mirar Historial	2a	Pulsar sobre un libro que exista en la BD	Se accederá a las características de ese libro.
	2b	Pulsar sobre un libro que ya no exista en la BD	Mensaje de error o aviso, donde se indique que el libro ya no está en la BD. Enviar un aviso al administrador para que lo gestione.
Mirar Historial / Editar Reseña	3a	Entrar en 'Editar Reseña' y la reseña existe	Se accederá al menú de 'Editar Reseña'.
	3b	Entrar en 'Editar Reseña' y la reseña no existe	Mensaje de error, donde se indique que la reseña no existe o se ha borrado. Enviar un aviso al administrador para que lo gestione.
	4a	Pulsar en 'Aceptar Cambios' y la descripción o valoración es NULL/Vacía	Mensaje de error, que no ejecute el botón, sino que vuelva al menú de 'Editar Reseña'.
	4b	Pulsar en 'Aceptar Cambios' y la descripción o valoración es la misma que la original	Mensaje de aviso, '¡La descripción/valoración no ha cambiado!', se ejecuta 'Cancelar Cambios'.
	4c	Pulsar en 'Aceptar Cambios' y la descripción y valoración no son nulas ni las mismas que las originales	Se aceptarán los cambios, y estos cambios se enviarán a la BD ya que son correctos.
	5	Pulsar en 'Cancelar Cambios'	Sea como sea la descripción y la valoración no se tienen en cuenta, se vuelve a la pantalla de las características del libro.
	6	Pulsar en 'Volver'	Se vuelve al historial de libros que el usuario ha devuelto al sistema.

d) ADMINISTRADOR (JON)

<u>CASO DE USO</u>	<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
Añadir usuario	1a	Añadir un nuevo usuario y que no se encuentre en la BD	Nuevo usuario en la BD.
	1b	Añadir un nuevo usuario pero que este ya se encuentre añadido a la BD de la Biblioteca	Mensaje de error diciendo que ya existe un usuario con esa clave primaria en la BD.
Añadir libro	2a	Añadir un nuevo libro y que no se encuentre en la BD	Nuevo libro en la BD.
	2b	Añadir un nuevo libro pero que este ya se encuentre añadido a la BD de la Biblioteca	Mensaje de error diciendo que ya existe un libro con esa clave primaria en la BD.
Eliminar usuario	3a	Eliminar un usuario que existe en la BD de la Biblioteca, introduciendo su identificador de forma correcta	Mensaje avisando de que el usuario ha sido eliminado correctamente.
	3b	Eliminar un usuario que existe en la BD de la Biblioteca, introduciendo mal su identificador	Mensaje avisando de que el identificador no existe.
	3c	Eliminar un usuario que no existe en la BD	Mensaje de error diciendo que no existe nadie con ese identificador.

e) RED DE AMIGOS (ENeko)

<u>CASO DE USO</u>	<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
Solicitar Amigo	1a	Pulsar solicitar amigo y el usuario actual existe	Aparece un pop up.
	1b	Pulsar solicitar amigo y el usuario actual no existe	No pasa nada
	1c	Pulsar solicitar amigo e introducir un nombre correcto	Se envía una solicitud de amistad.
	1d	Pulsar solicitar amigo e introducir un nombre que no existe	Da un aviso de que no existe el usuario.
	1e	Pulsar solicitar amigo y luego pulsar fuera del pop up	Se cierra el pop up de solicitud.
Gestionar Solicitudes	2a	Pulsa gestionar solicitudes y el usuario existe	Se abre la página de solicitudes.
	2b	Pulsa gestionar solicitudes y el usuario no existe	No ocurre nada
	2c	Aceptar una solicitud	Se acepta la solicitud
	2d	Rechazar una solicitud	Se rechaza la solicitud
	2e	Aceptar o rechazar una solicitud de un usuario inexistente	No se puede dar por que si se borra un usuario también se borran las solicitudes que haya enviado
Ver Lista Amigos	3a	Acceder a la lista de amigos y el usuario existe	Se muestra la lista de amigos
	3b	Acceder a la lista de amigos y el usuario no existe	No ocurre nada
	3c	Ver perfil de un amigo que existe	Muestra el perfil del amigo
	3d	Ver perfil de un amigo que no existe	No se puede dar por que cuando un administrador borra un usuario se borra de todas las listas de amigo
-----	4	Volver	Vuelve al menú

f) FOROS (ANDER)

<u>CASO DE USO</u>	<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
Ver Foro	1a	Usuario no identificado intenta ver foro	Accede al foro
	1b	Usuario identificado intenta ver foro	Accede al foro
Ver Foro/Nuevo tema	2a	Usuario no identificado intenta añadir tema	Mensaje de error, no está identificado y no puede llevar a cabo la acción.
	2b	Usuario identificado intenta añadir tema que no existe	Añade el nuevo tema.
	2c	Usuario identificado intenta añadir un tema ya existente	Mensaje de error, ya existe un tema con el mismo nombre.
Ver Foro/ Ver Tema	3a	Usuario no identificado intenta ver tema	Accede al tema.
	3b	Usuario identificado intenta ver tema	Accede al tema.
Ver Foro/ Ver Tema/Comentar	4a	Usuario no identificado intenta comentar	Mensaje de error, no está identificado y no puede llevar a cabo la acción.
	4b	Usuario identificado intenta comentar un comentario distinto a los demás	Se publica el comentario.
	4c	Usuario identificado intenta comentar un comentario igual a otro dentro del mismo tema	Se publica el comentario
Ver Foro/ Ver Tema/ Ver Foro	5a	Usuario no identificado intenta volver de tema a foro	Vuelve al foro.
	5b	Usuario identificado intenta volver de tema a foro	Vuelve al foro.

g) RECOMENDACIONES DEL SISTEMA (JAVIER)

Las pruebas identificadas con ‘-’ como número de prueba corresponden a otros módulo, y no son específicos de esta funcionalidad.

<u>CASO DE USO</u>	<u>NÚMERO DE PRUEBA</u>	<u>DESCRIPCIÓN</u>	<u>RESULTADO ESPERADO</u>
-	-	Usuario no está identificado	Menú inicial sin sección de recomendaciones.
	1.1	Usuario está identificado y hay sugerencias.	Se muestran las sugerencias.
Leer detalles libro sugerido	2.1	Usuario está identificado, hay sugerencias y pincha sobre un libro.	Se muestran las sugerencias sin repetirse. Se muestran los detalles del libro sugerido, se cambia de interfaz.
	2.2	Usuario está identificado y no hay sugerencias (no se ha leído ningún libro con el que poder recomendar otros).	Se muestra una lista vacía o un mensaje: “no hay sugerencias”, o, en caso de que se hayan leído libros, se muestran los diez libros más leídos. Pinchando sobre cualquiera de ellos, se muestran los detalles del libro sugerido, se cambia de interfaz.
	2.3	Usuario está identificado y no hay sugerencias (no hay ningún usuario que haya leído algún libro nuestro).	Se muestran los detalles del libro sugerido, se cambia de interfaz y el libro queda reservado, visible en el caso de uso “Libros En Préstamo” de la parte Gestión de Reservas.
Leer detalles libro sugerido/ Reservar	-	Usuario está identificado, hay sugerencias, pincha sobre un libro y clic en reservar.	Se muestran los detalles del libro sugerido, se cambia de interfaz y el libro queda reservado, visible en el caso de uso “Libros En Préstamo” de la parte Gestión de Reservas.

Para la implementación de pruebas de caja negra, se han tenido en cuenta los siguientes casos:

- Sin sesión iniciada
- Con sesión iniciada y sin recomendaciones (se sugerirán los diez libros más leídos):
 - sin libros leídos
 - sin usuarios que hayan leído los mismos libros
- Con sesión iniciada y con recomendaciones:
 - sin sugerencias repetidas
 - con sugerencias repetidas

Aquellas prueba subrayadas hacen referencia a clases equivalentes entre sí, motivo por el que se consideran subcasos de la prueba general.

La ejecución se ha llevado a cabo eliminando la información de la tabla *Prestar* de la base de datos (de manera virtual, gracias a los módulos de *unit test*) e introduciendo valores que generen esas situaciones, por ejemplo: leer un libro que nadie a leído, no leer ningún libro...

Se ha seguido la documentación de eGela para ejecutar las pruebas unitarias.

7. DIAGRAMA DE BASE DE DATOS

USUARIO	
PK	<u>email</u>
	rol
	contraseña
FK	emailAdmin

LIBRO	
PK	<u>id</u>
	titulo
	descripción
	numCopias
FK	idAutor
FK	emailAdmin

TEMA	
PK	<u>tituloTema</u>
FK	emailUsuario
	descTema
	fechaHora

AUTOR	
PK	<u>id</u>
	nombre

RESEÑA	
PK,FK1	<u>emailUsuario</u>
PK,FK2	<u>idLibro</u>
	reseña
	valoracion

PRESTAR	
PK,FK1	<u>emailUsuario</u>
PK,FK2	<u>idLibro</u>
PK	<u>fechaHora</u>
	fechaFin

SON AMIGOS	
PK	<u>emailUsuario1</u>
PK	<u>emailUsuario2</u>

COMENTA	
PK,FK1	<u>emailUsuario</u>
PK,FK2	<u>tituloTema</u>
PK	<u>fechaHora</u>
	mensaje

SOLICITA	
PK	<u>emailUsuario1</u>
PK	<u>emailUsuario2</u>

Figure 56: Tablas de la Base de Datos

8. DIAGRAMAS DE CLASES

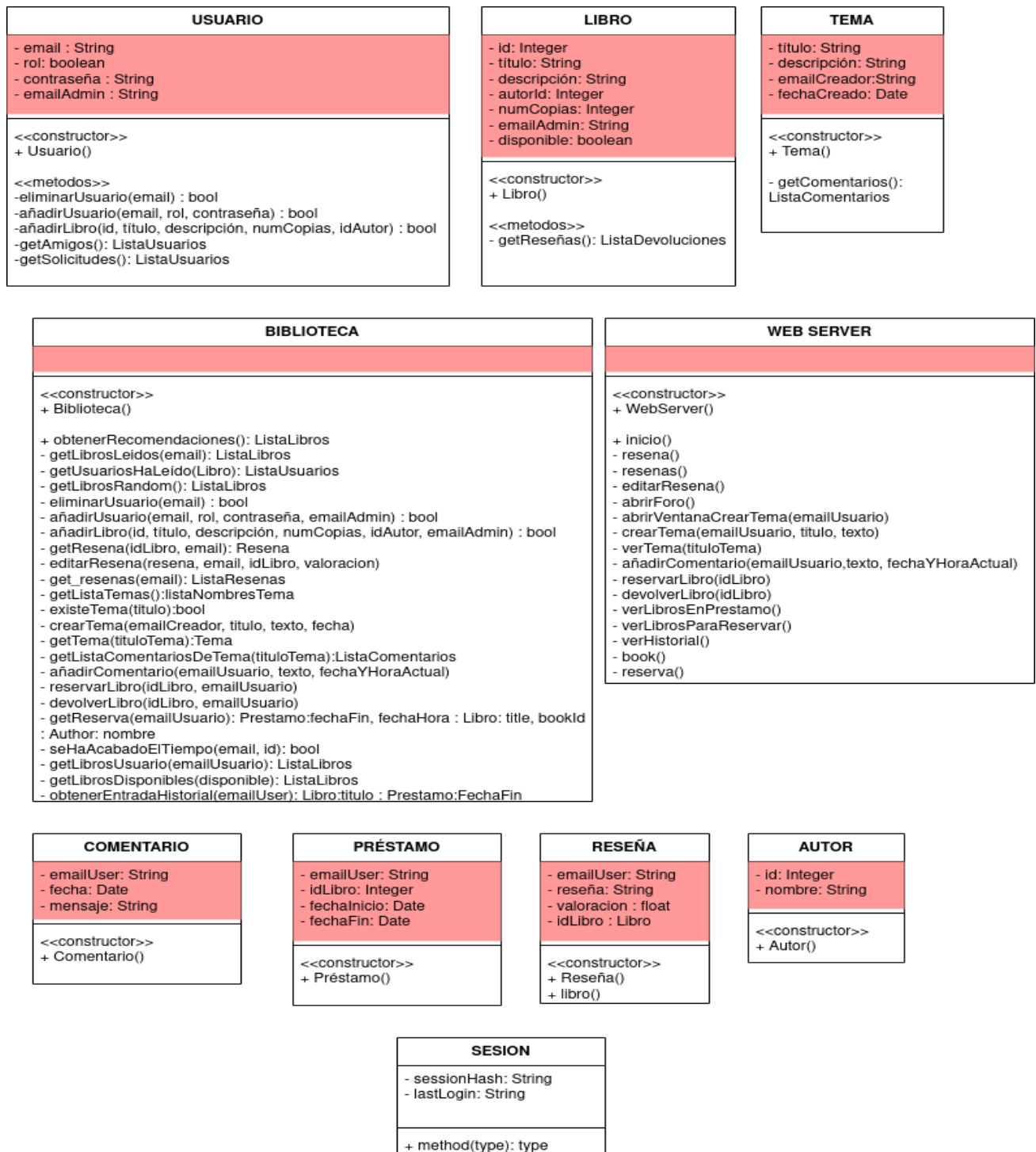


Figure 57: Diagrama de Clases

9. DIAGRAMAS DE COMUNICACIÓN

a) GESTIÓN DE RESERVAS (SERGIO)

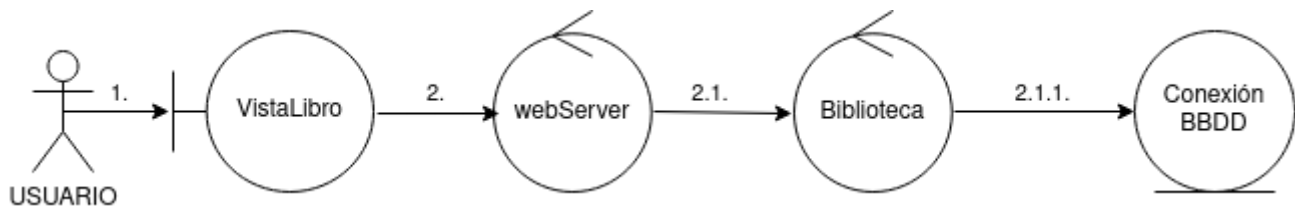


Figura 58: Diagrama de comunicación de reservar

1. El usuario pulsa en un libro.
2. reservarLibro(idLibro)
 - 2.1. reservarLibro(idLibro, emailUsuario)
 - 2.1.1. executeQuery(Insertar libro en el usuario)

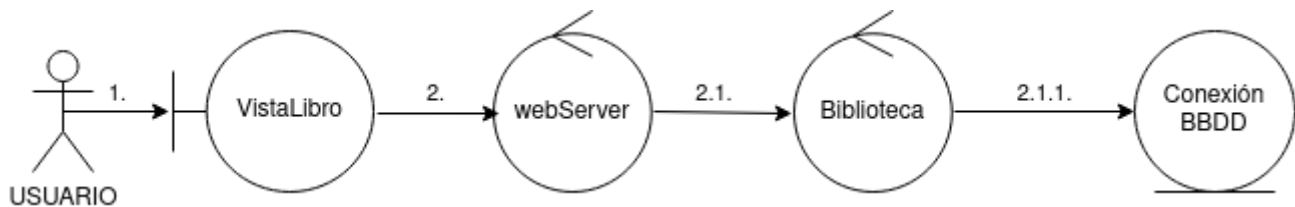


Figura 59: Diagrama de comunicación de devolver

1. El usuario pulsa en un libro que tiene.
2. devolverLibro(idLibro)
 - 2.1. devolverLibro(idLibro, emailUsuario)
 - 2.1.1. executeQuery(Quitar libro del usuario)
 - 2.1.2. executeQuery(Añadir indicacion de que lo ha teniedo el usuario)

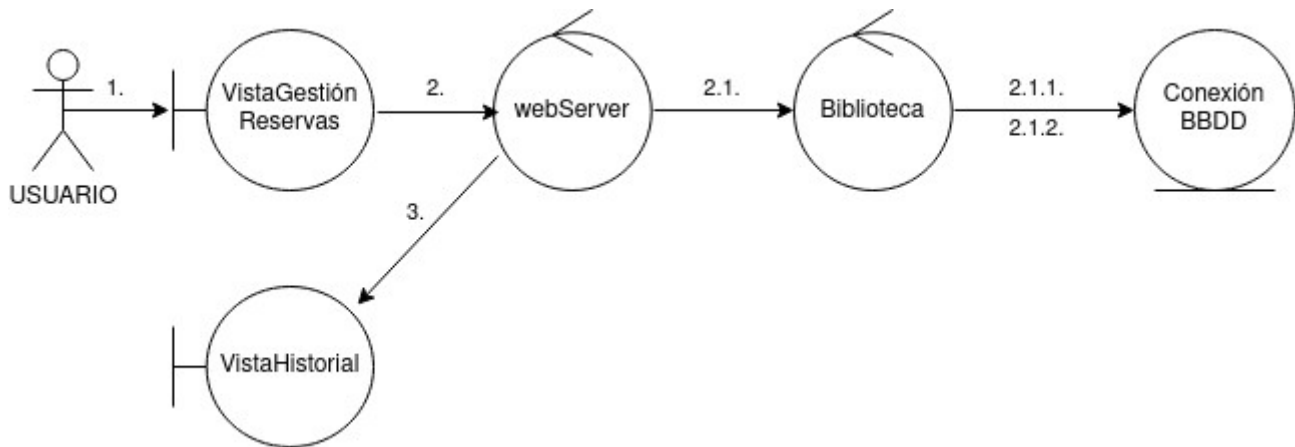


Figura 62: Diagrama de comunicación de verHistorial

1. El usuario pulsa en el botón VerLibrosParaReservar
2. verHistorial()
 - 2.1. obtenerEntradaHistorial(emailUsuario): (Libro:titulo,Prestamo: fechaFin)
 - 2.1.1. executeQuery(select de los titulos de los libros que ha tenido y tiene el usuario)
 - 2.1.2. executeQuery(select de las fechasFin de los libros que ha tenido el usuario)
 - 2.1.3 executeQuery(select de las fechas para devolver de los libros que tiene usuario)
 - 2.1.4 executeQuery(select del nombre del autor del respectivo libro)
3. new VistaHistorial(Libro:titulo,Prestamo: fechaFin)

b) RESEÑAS (ADRIÁN)

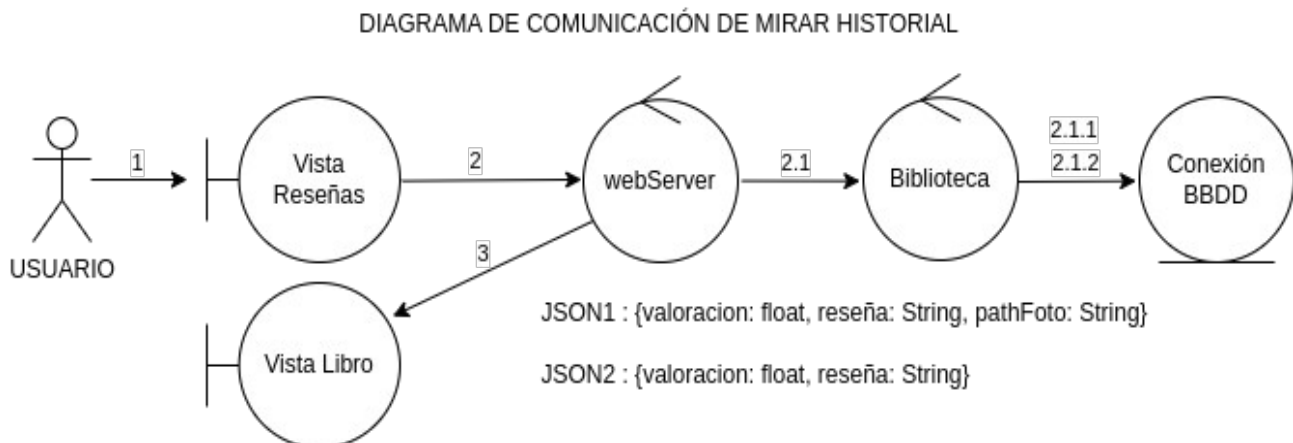


Figura 63: Diagrama de comunicación de mirar el historial.

- 1.- El usuario pulsa en un libro.
- 2.- obtenerReseña(idElLibro) : JSON1
 - 2.1.- obtenerReseña(idElLibro, emailElUsuario) : JSON1
 - 2.1.1.- select("SELECT valoracion, reseña FROM devolucion WHERE emailUsuario= emailElUsuario AND idLibro= idElLibro ") : JSON2
 - 2.1.2.- select("SELECT pathLibro FROM libro WHERE idLibro = idElLibro"): String
- 3.- new VistaLibro(idElLibro, JSON1) [Una reseña en concreto]

DIAGRAMA DE COMUNICACIÓN DE CAMBIO A VISTA DE EDITAR RESEÑA

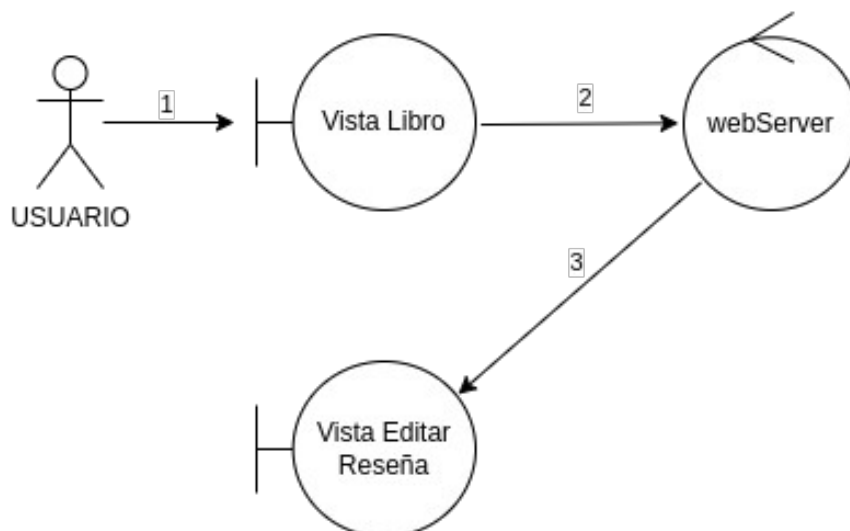


Figura 64: Diagrama de comunicación, cambio de vista a editar reseña.

- 1.- El usuario pulsa en "Editar Reseña".
 - 2.- pasarAEditarReseña(JSON1)
 - 3.- new VistaEditarReseña(JSON1)
- JSON1: {valoracion: float, reseña:String, pathFoto: String}

DIAGRAMA DE COMUNICACIÓN DE EDITAR RESEÑA

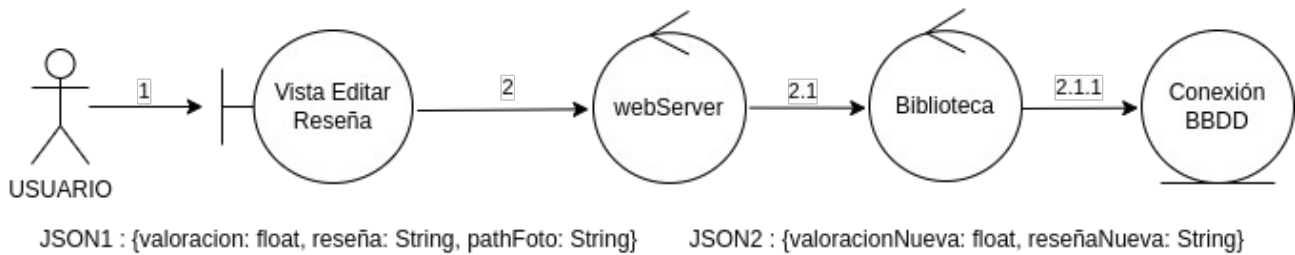


Figura 65: Diagrama de comunicación de editar reseña

1.- El usuario introduce los cambios (JSON2, la reseña y la valoración nueva) en la reseña y da a aceptar.

[si la valoración y reseña no son NULL ni datos incorrectos]

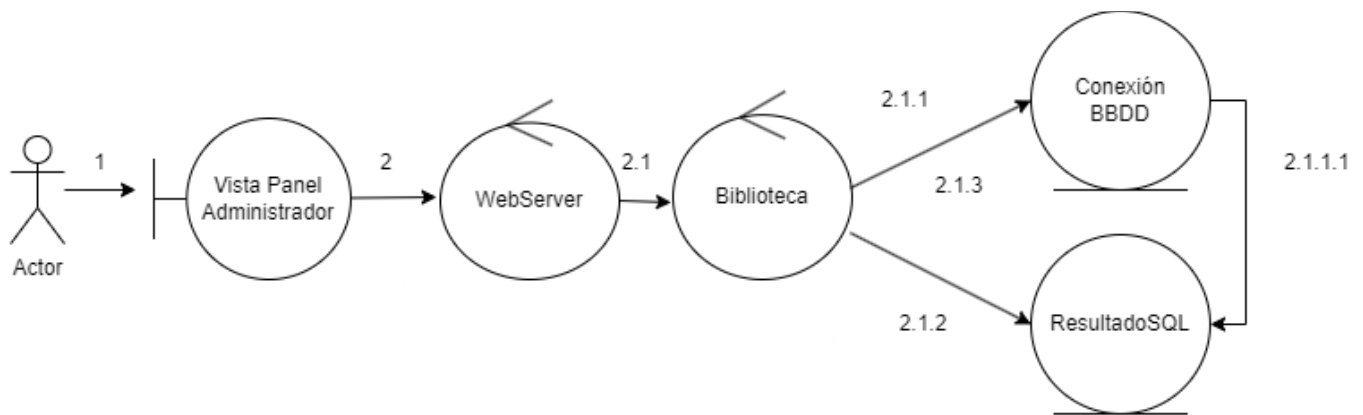
2.- editarReseña(emailUser, idElLibro, JSON2)

2.1.- editarReseña(idLibro, email, JSON2)

2.1.1.- update("UPDATE Reseña SET valoracion = %valoracion%, resena = %resena % WHERE idLibro = %idLibro% AND emailUser = %email% ")

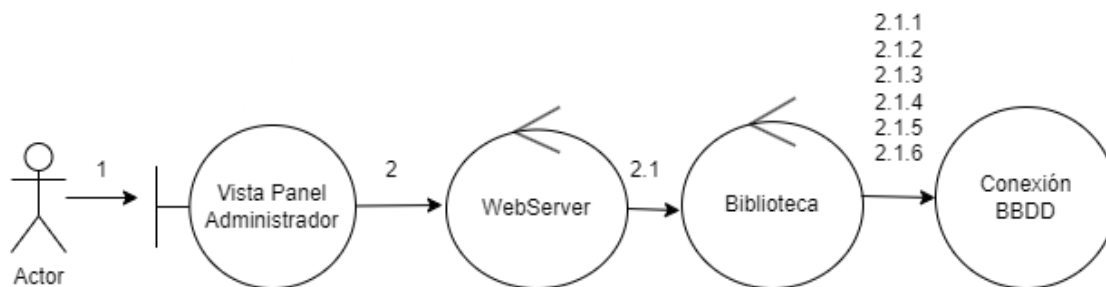
[si la valoración y reseña son o NULL o incorrectos no se entra a editarReseña en la biblioteca]

c) ADMINISTRADOR (JON)



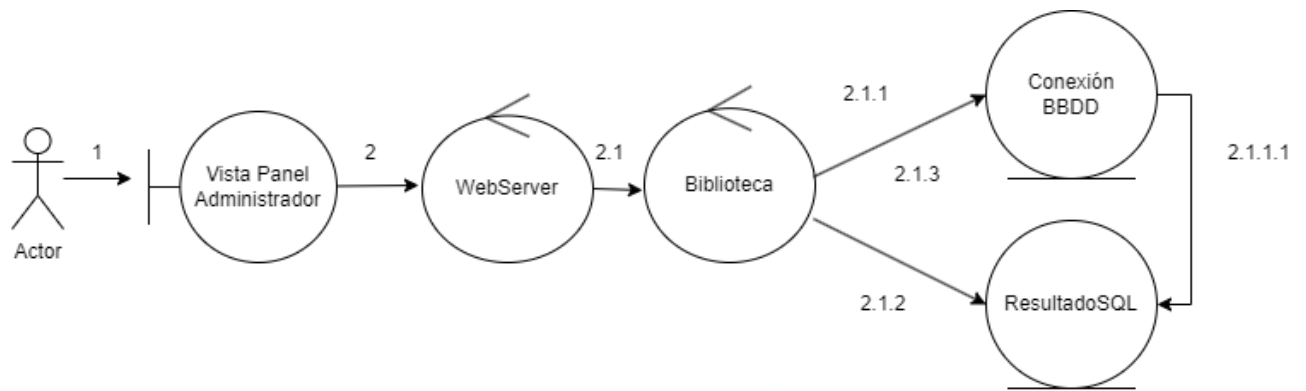
1. El administrador accede al apartado "Panel de Administrador", introduce los datos del nuevo libro y le da al botón "Añadir Libro"
2. añadirLibro(id, título, descripción, numCopias, idAutor) : bool
 - 2.1 añadirLibro(id, título, descripción, numCopias, idAutor, emailAdmin) : bool
 - 2.1.1 execSQL("SELECT FROM libro WHERE id=%") : ResultadoSQL
 - 2.1.1.1 new ResultadoSQL()
 - 2.1.2 next()
 - 2.1.3 execSQL(INSERT INTO libro VALUES...)

Figura 66: Diagrama de comunicación de Añadir libro.



1. El administrador accede al apartado "Panel de Administrador", introducir el email del usuario en el campo y hacer clic en eliminar usuario
2. eliminarUsuario(email) : bool
 - 2.1 eliminarUsuario(email) : bool
 - 2.1.1 executeQuery("Eliminar libros prestados")
 - 2.1.2 executeQuery("Eliminar libros devueltos")
 - 2.1.3 executeQuery("Eliminar amigos")
 - 2.1.4 executeQuery("Eliminar solicitudes de amistad enviadas/recibidas")
 - 2.1.5 executeQuery("Eliminar los comentarios")
 - 2.1.6 executeQuery("Eliminar usuario")

Figura 67: Diagrama de comunicación de Eliminar usuario.



1. El administrador accede al apartado "Panel de Administrador", introduce los datos del nuevo usuario y le da al botón "Añadir Usuario"
2. añadirUsuario(email, rol, contraseña) : bool
 - 2.1 añadirUsuario(email, rol, contraseña, emailAdmin) : bool
 - 2.1.1 execSQL("SELECT FROM usuario WHERE email=%email%") : ResultadoSQL
 - 2.1.1.1 new ResultadoSQL()
 - 2.1.2 next()
 - 2.1.3 execSQL(INSERT INTO usuario VALUES...)

Figura 68: Diagrama de comunicación de Añadir usuario.

d) RED DE AMIGOS (ENeko)

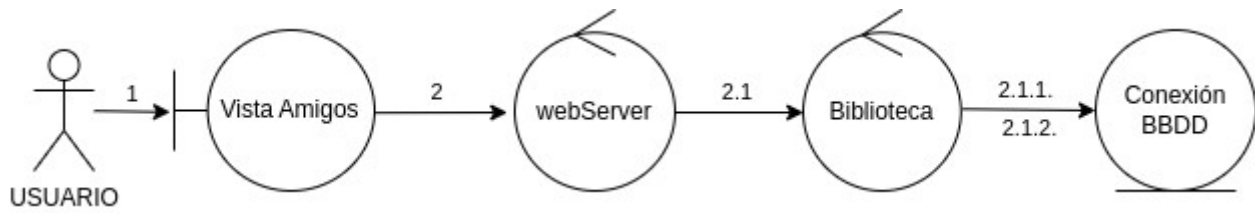


Figure 69: Diagrama de comunicación de Solicitar Amigo

1. El usuario pulsa "Solicitar amigo" e introduce un correo
2. enviarSolicitud(correoUsuarioActual, correoUsuarioObjetivo) : Bool
 - 2.1. enviarSolicitud(correoUsuarioActual, correoUsuarioObjetivo) : Bool
 - 2.1.1. select("SELECT email FROM usuario WHERE email = correoUsuarioObjetivos")
 - 2.1.2. insert("INSERT INTO solicitud VALUES (correoUsuarioActual, correoUsuarioObjetivo)")

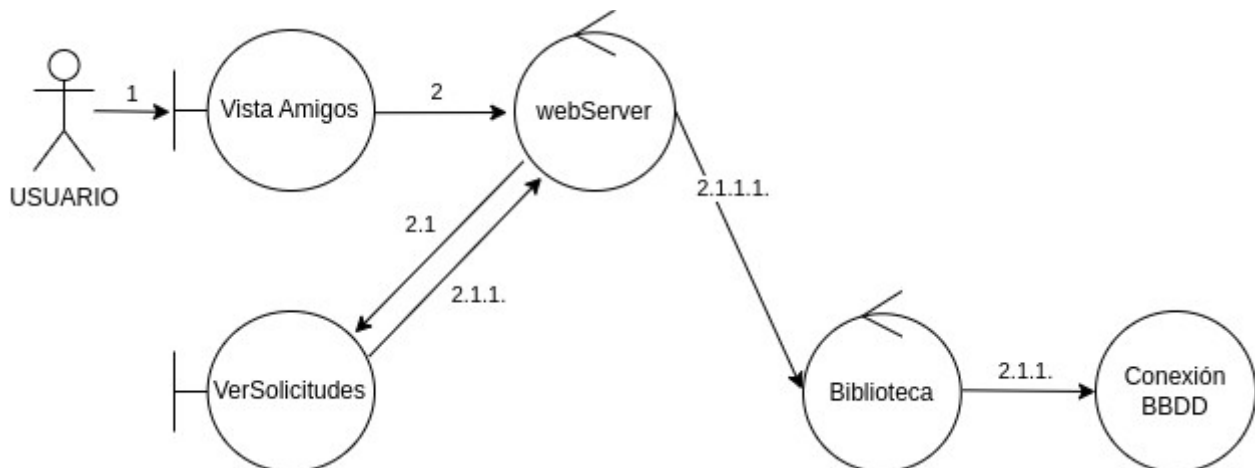


Figure 70: Diagrama de comunicación de Ver Solicitudes

1. El usuario pulsa "Ver solicitudes"
2. abrirSolicitudes()
 - 2.1. new VerSolicitudes()
 - 2.1.1. getSolicitudes()
 - 2.1.1.1. getSolicitudes(correoUsuarioActual)
 - 2.1.1.1.1. select("SELECT email FROM solicita WHERE emailObjetivo = correoUsuarioActual")

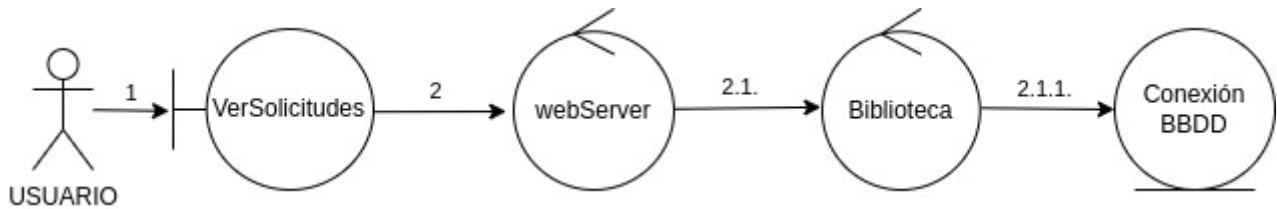


Figure 71: Diagrama de comunicación de Aceptar Solicitud

1. El usuario pulsa aceptar sobre una solicitud
2. aceptarSolicitud(correoUsuarioSolicitud)
 - 2.1. aceptarSolicitud(correoUsuarioSolicitud, correoUsuarioActual)
 - 2.1.1.insert("INSERT INTO sonAmigos VALUES (correoUsuarioActual, correoUsuarioSolicitud)")

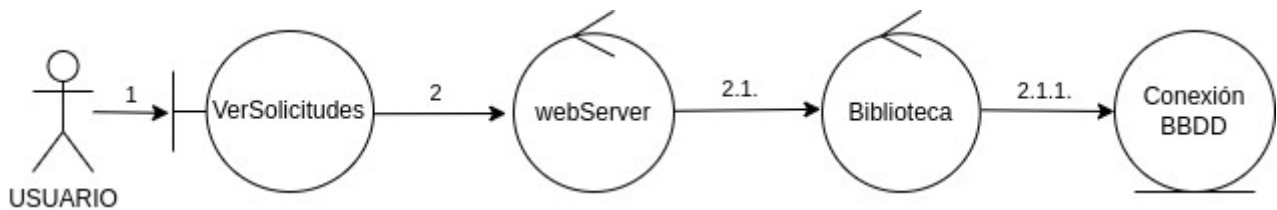


Figure 72: Diagrama de comunicación de Rechazar Solicitud

1. El usuario pulsa rechazar sobre una solicitud
2. rechazarSolicitud(correoUsuarioSolicitud)
 - 2.1. rechazarSolicitud(correoUsuarioSolicitud, correoUsuarioActual)
 - 2.1.1. delete("DELETE FROM solicita WHERE emailObjetivo = correoUsuarioActual")

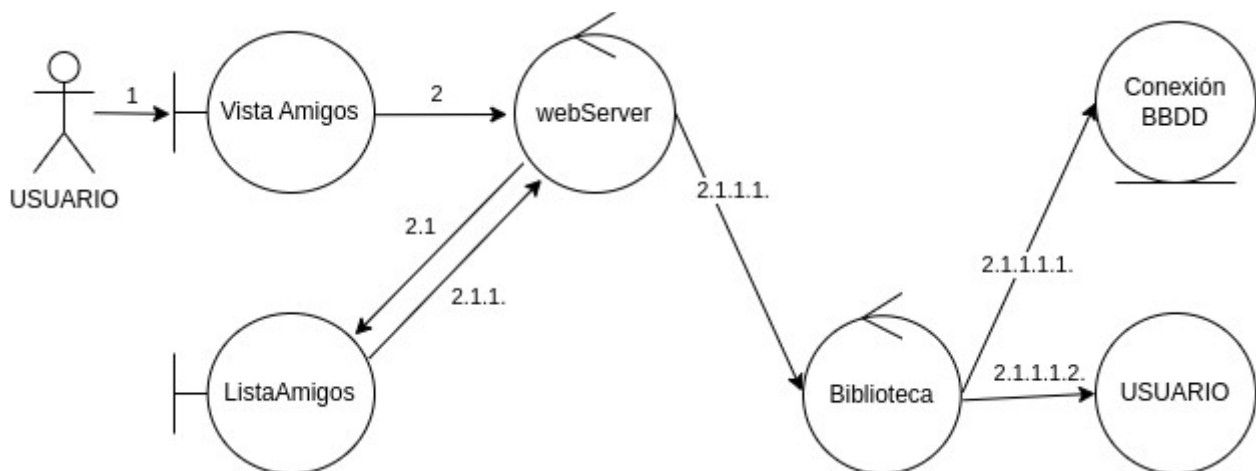


Figure 73: Diagrama de comunicación de Ver Solicitudes

1. El usuario pulsa "Ver solicitudes"

2. verListaAmigos()

2.1. new ListaAmigos()

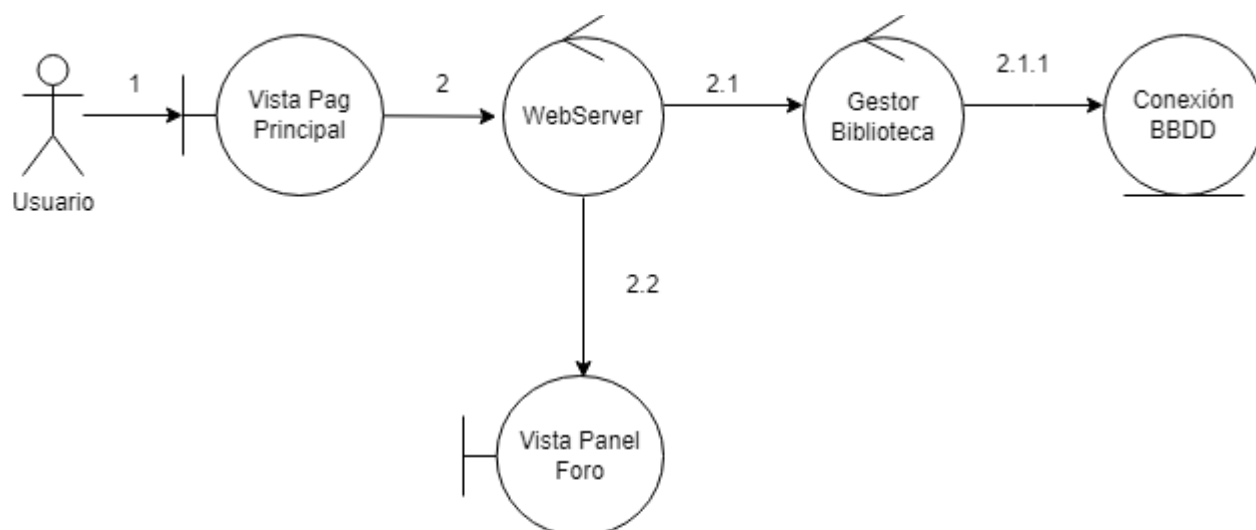
2.1.1. getAmigos()

2.1.1.1. getAmigos(correoUsuaroiActual)

2.1.1.1.1. select("SELECT * FROM sonAmigos WHERE email1= correoUsuaroiActual OR email2 = correoUsuarioActual")

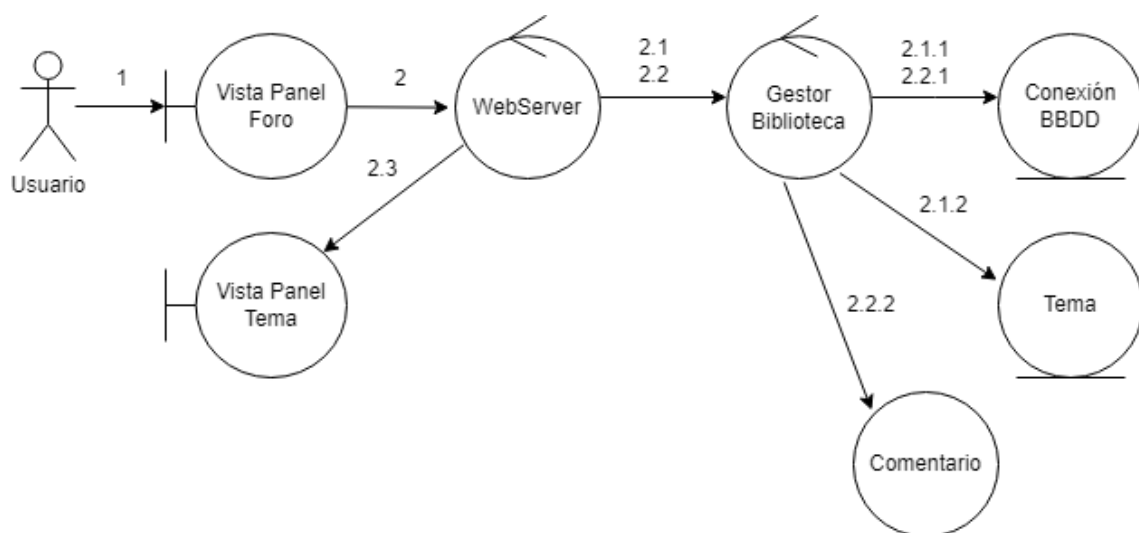
2.1.1.1.2 new Usuario(email, rol, contraseña, emailAdmin)

e) FOROS (ANDER)



1. El usuario pulsa en "Foro" para acceder al foro
2. abrirForo()
 - 2.1 getListTemas():listaNombresTema
 - 2.1.1 - executeQuery ("Select de todos los nombres de temas del foro")
 - 2.2 new vistaPanelForo(listaNombresTema)

Figura 74: Diagrama de comunicación de acceder al foro



1. - El usuario pulsa en el tema que desea ver

2. - verTema(tituloTema)

2.1 - getTema(tituloTema): Tema

2.1.1 - executeQuery("select buscando los datos del tema que coincide con el titulo y obtiene todos sus atributos)

2.1.2 - elTema = new Tema (tituloTema, descripcion, emailAdmin, fechaCreado)

2.2 - getListaComentariosDeTema(tituloTema):ListaComentarios

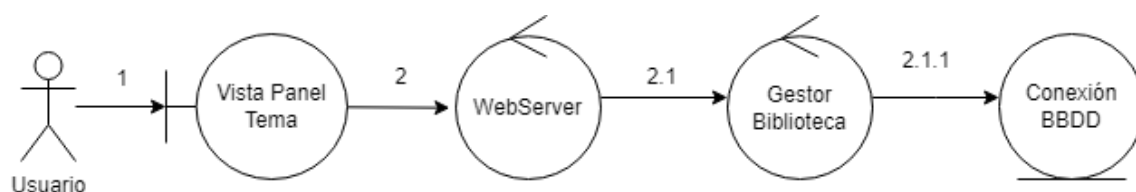
2.2.1 - executeQuery("Select buscando todos los comentarios que hay en el tema)

[Por cada comentario encontrado]

2.2.2 - new Comentario(emailUsuario, fecha, mensaje)

2.3 - new VistaPanelTema (Tema,ListaComentarios)

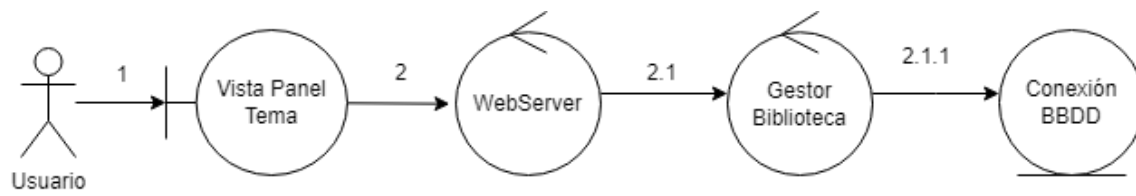
Figura 75: Diagrama de comunicación de ver Tema



1. - El usuario pulsa en comentar e introduce el texto del comentario
2. - añadirComentario(emailUsuario, texto, fechaYHoraActual)
 - 2.1 - añadirComentario(emailUsuario, texto, fechaYHoraActual)
 - 2.1.1 - executeQuery("añade un nuevo comentario a la BD")

- 1 - El usuario pulsa en crear tema
- 2 - abrirVentanaCrearTema(emailUsuario)
 - 2.1 - new VistaPanelCrearTema (emailUsuario)
- 3 - El usuario introduce el título del tema y su contenido
- 4 - crearTema (emailUsuario, título, texto)
 - 4.1 - existeTema (título) : bool
 - 4.1.1 - executeQuery ("Comprobar si ya existe")
 [Si false (no existe ningún tema con el mismo título)]
 - 4.2 - crearTema(emailCreador, título, texto, fecha)
 - 4.2.1 - executeQuery ("Añade nuevo tema")

Figura 76: Diagrama de comunicación de crear Tema



- 1. - El usuario pulsa en comentar e introduce el texto del comentario
- 2. - añadirComentario(emailUsuario, texto, fechaYHoraActual)
- 2.1 - añadirComentario(emailUsuario, texto, fechaYHoraActual)
- 2.1.1 - executeQuery("añade un nuevo comentario a la BD")

Figura 77: Diagrama de comunicación de crear Comentario

f) RECOMENDACIONES DEL SISTEMA (JAVIER)

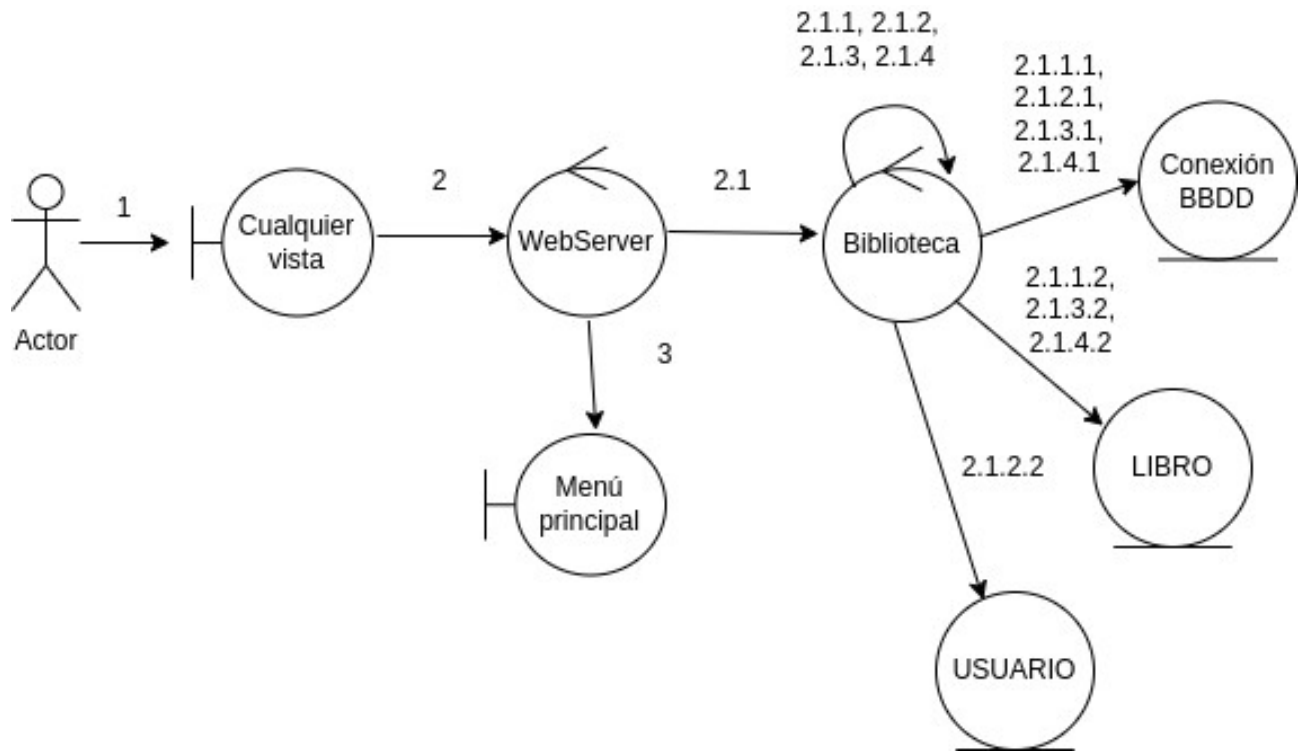


Figura 78: Diagrama de comunicación de recomendaciones

1. El usuario clicka "Inicio".

2. Inicio()

2.1 obtenerRecomendaciones(email)

2.1.1 getLibrosLeídos(email)

2.1.1.1 executeQuery("Devolver libros leídos por el usuario")

2.1.1.2 new Libro(id, título, descripción, idAutor, numCopias, emailAdmin)

2.1.2.getUsuariosHaLeído(idLibro): ListaUsuarios

2.1.2.1 executeQuery("Usuarios que han leído el libro")

2.1.2.2 new Usuario(email, rol, contraseña, emailAdmin)

//Por cada usuario que ha leído el libro

2.1.3 getLibrosLeídos(emailOtro)

[Desarrollado en el paso 2.1.1]

//Cuando no hay libros leídos

2.1.4.getLibrosRandom()

2.1.4.1 executeQuery("Devolver libros top 10 leídos")

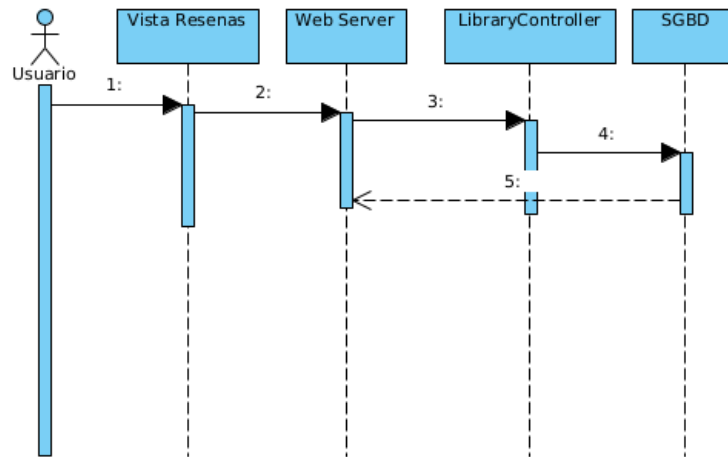
2.1.4.2 new Libro(id, título, descripción, idAutor, numCopias, emailAdmin)

3. new MenúPrincipal(ListaRecomendaciones)

10. DIAGRAMAS DE SECUENCIA

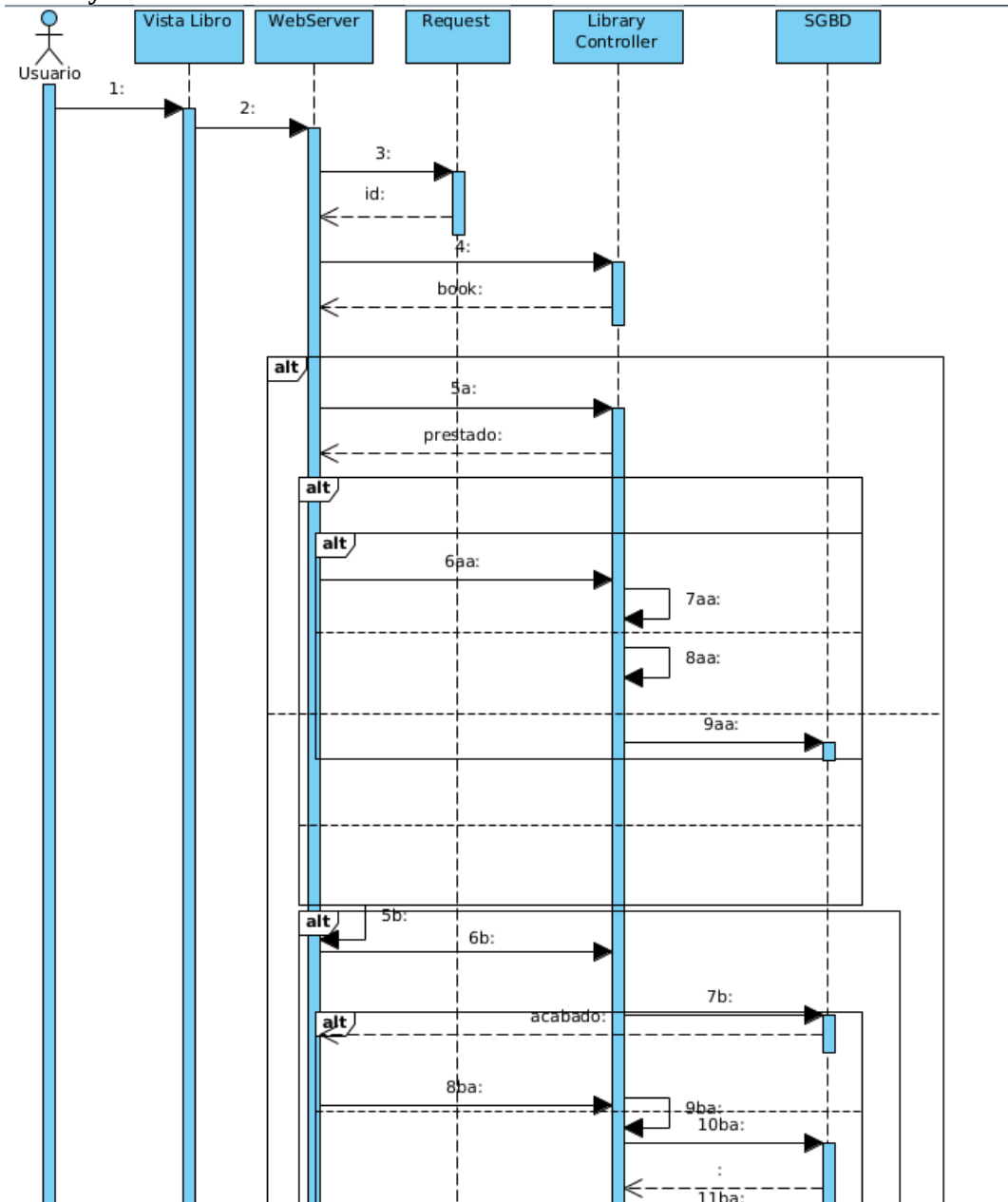
a) RESERVAS (SERGIO)

A1.- Cargar Vista del Historial de Reservas (Nombre vista: Reservas)



1: Usuario pulsa en "Reservas"
2: reserva()
3: datos= library.getReserva(request.user.email)
4: res= execSQL("SELECT Prestar.fechaFin, Book.title, Author.name, Book.id, Prestar.fechaHora FROM Prestar INNER JOIN Book ON Prestar.idLibro=Book.id INNER JOIN Author ON Book.author= Author.id WHERE Prestar.emailUser LIKE email");
RespuestaSQL

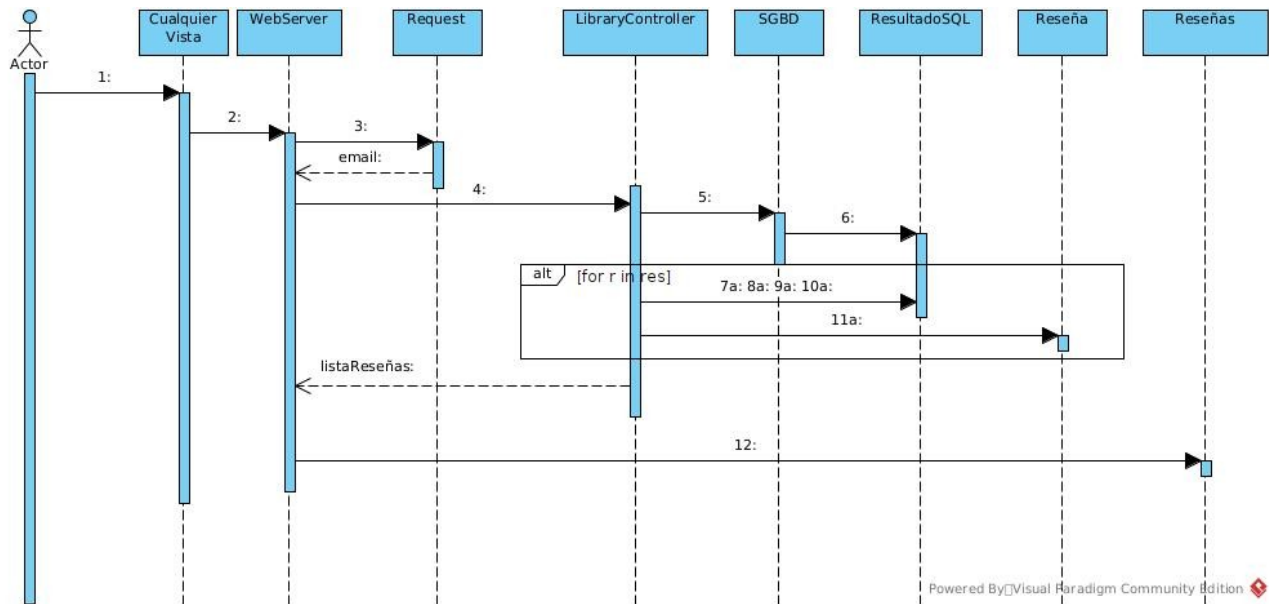
A2.- Reservar y Devolver Libros



1: Usuario pulsa en el título de un libro en el catálogo
2: book()
3: id = request.values.get("id","")
4: book = library.get_book(id): Book
[si se está registrado]
5a: prestado = library.isOnLoan(request.user.email, id): bool
[si no se tiene ya el libro]
[si se quiere reservar]
6aa: library.reservarLibro(request.user.email, id)
7aa: fecha_actual= datetime.now()
8aa: fechaHFin = fecha_actual + relativedelta(month=2)
fechaHFinal: (se pone el formato con el que tenemos la entidad en la bd)
9aa: execSQL ("INSERT INTO Prestar (emailUser, idLibro, fechaHora, fechaFin) VALUES (emailUsuario, id, fechaHFinal, NULL)")
[si ya tiene el libro]
5b: fechaDevuelto = datetime.now().strftime('%Y-%m-%d')
6b: library.seHaAcabadoElTiempo(request.user.email, id): acabado
7b: execSQL("SELECT fechaHora FROM Prestar WHERE emailUser LIKE email AND idLibro = id AND fechaFin IS NULL"): ResultadoSQL
(si se ha pasado la fecha para devolver) -> True
(sino) -> False
[si se clicka en el botón devolver ó si se acaba el tiempo límite]
8ba: library.devolverLibro(request.user.email, id)
9ba: fechaDevuelto = datetime.now().strftime('%Y-%m-%d')
10ba: execSQL("SELECT fechaHora FROM Prestar WHERE emailUser LIKE emailUsuario AND idLibro=id AND fechaFin IS NULL"): ResultadoSQL
11ba: execSQL("UPDATE Prestar SET fechaFin = fechaDevuelto WHERE idLibro=id AND emailUser= emailUsuario AND fechaHora= fechaS")

b) RESEÑAS (JON)

B1.- Pestaña donde salen todas las reseñas (Nombre vista: Reseñas)



1: Usuario hace click en Reseñas

2: resenas()

3: email = request.user.email

4: get_resenas(email: String): ListaResenas

5: execSQL("SELECT * from Reseña WHERE emailUser = %email%")

6: new ResultadoSQL()

[mientras haya reseñas]

7a: emailUser = get(0)

8a: idLibro = get(1)

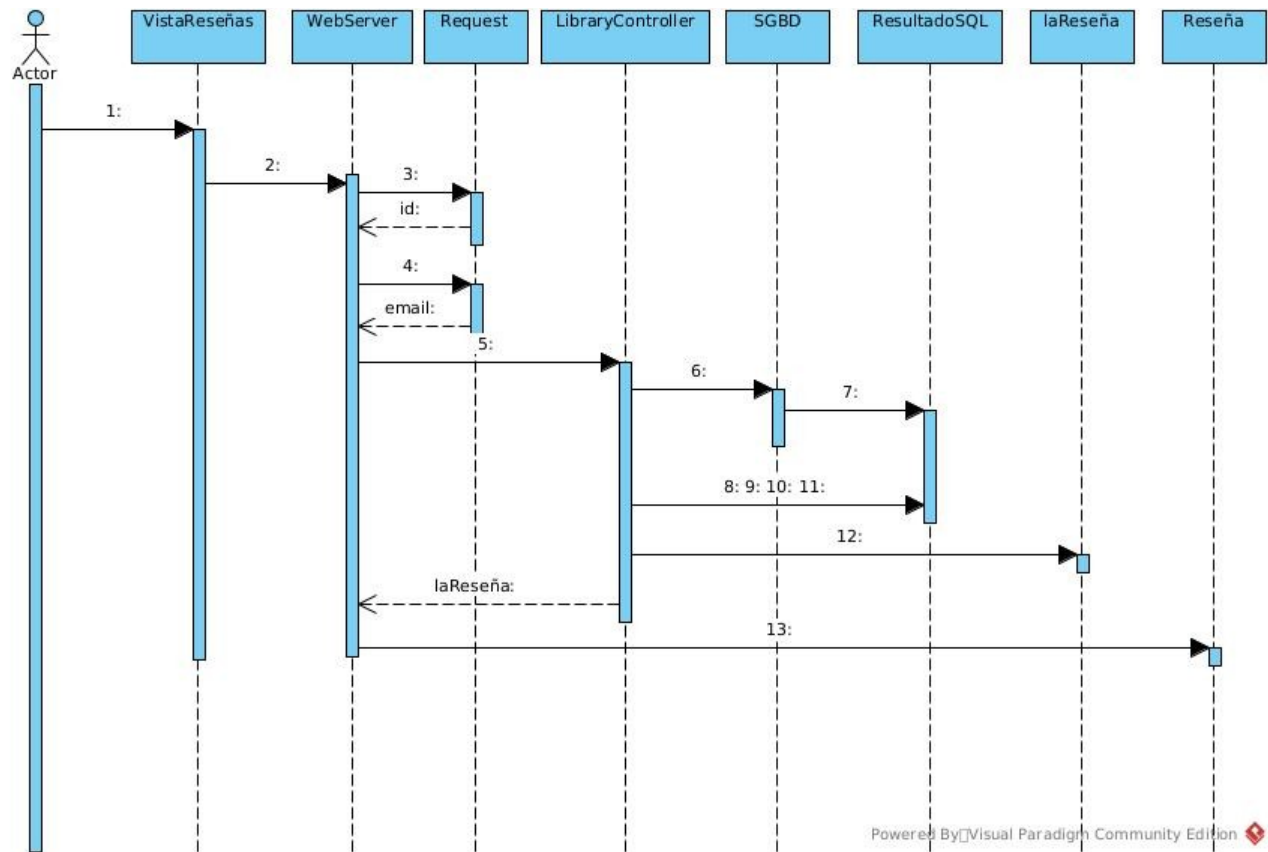
9a: resena = get(3)

10a: valoracion = get(4)

11a: new Reseña(emailUser, idLibro, resena, valoracion)

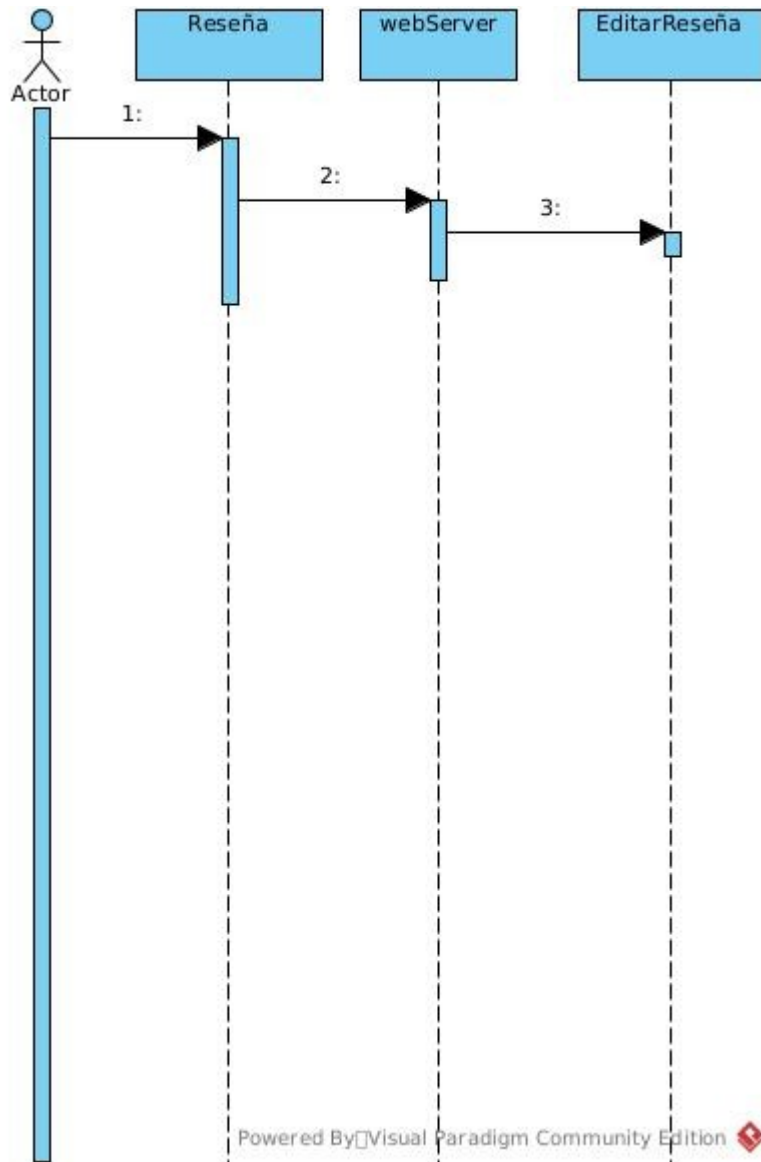
12: new Reseñas(listaReseñas)

B2.- Pestaña donde se ve una única reseña (Nombre vista: Reseña)



- 1: El usuario hace click en una reseña en concreto desde la pestaña de Reseñas
- 2: resena()
- 3: id = request.values.get("idLibro", "")
- 4: email = request.user.email
- 5: getResena(id: int, email: String): Resena
- 6: execSQL("SELECT * from Reseña WHERE emailUser = %email% AND idLibro = %id%")
- 7: new ResultadoSQL()
- 8: emailUser = get(0)
- 9: idLibro = get(1)
- 10: resena = get(3)
- 11: valoracion = get(4)
- 12: new Reseña(emailUser, idLibro, resena, valoracion)
- 13: new Reseña(laResena)

B3.- Cargar pestaña para editar reseña (Nombre vista: EditarReseña)

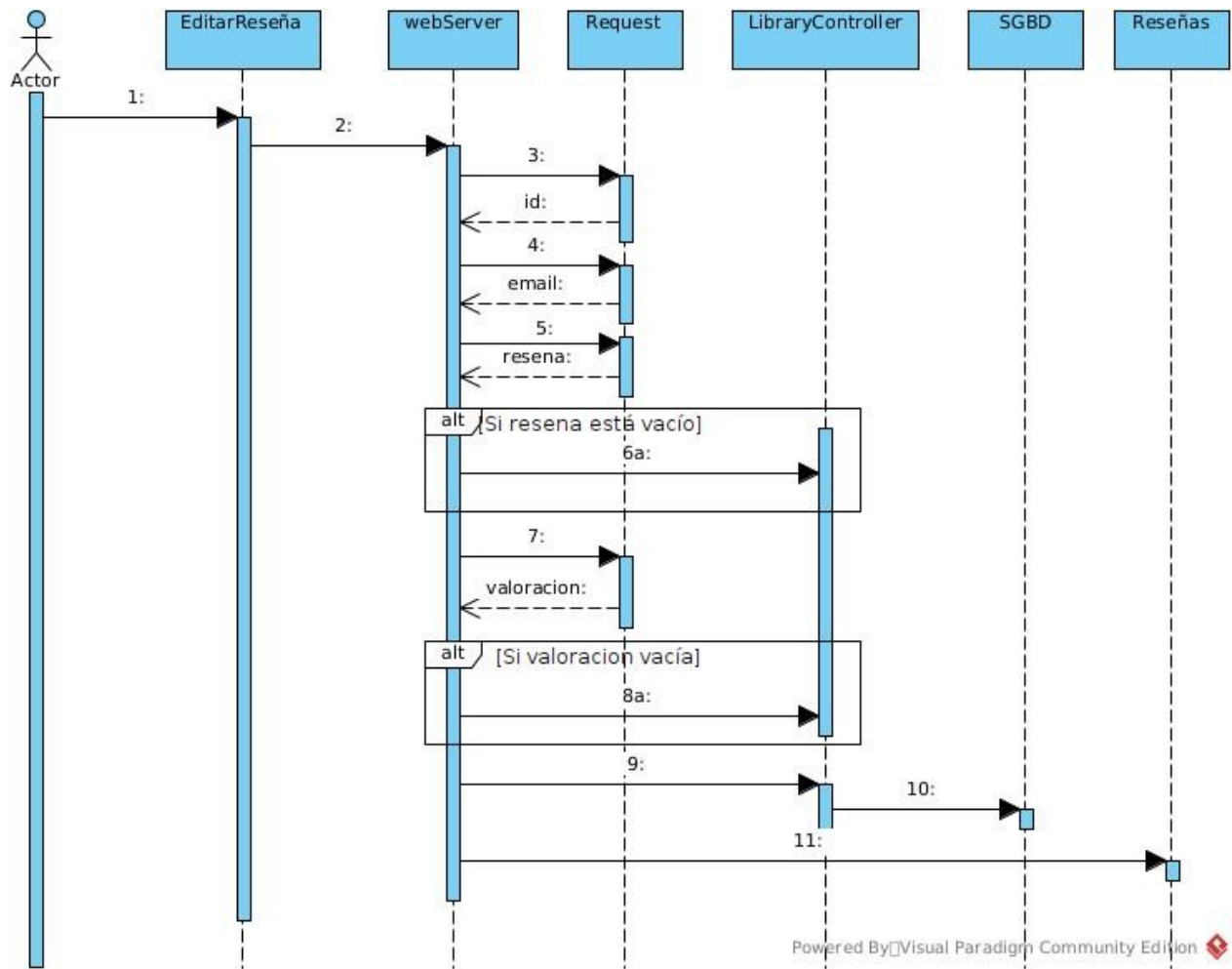


1: El usuario le da click a "Editar Reseña"

2: editarResena()

3: new EditarReseña()

B4.- Cargar pestaña para editar reseña (Nombre vista: EditarReseña)



1: El usuario le da a "Guardar cambios"

2: editarReseña()

3: id = request.values.get("id", "")

4: email = request.user.email

5: resena = request.form['resena']

[si resena está vacía]

6a: resena = getResena(id, email).resena

7: valoracion = request.form['valoracion']

[si valoracion está vacía]

8a: valoracion = getResena(id, email).valoracion

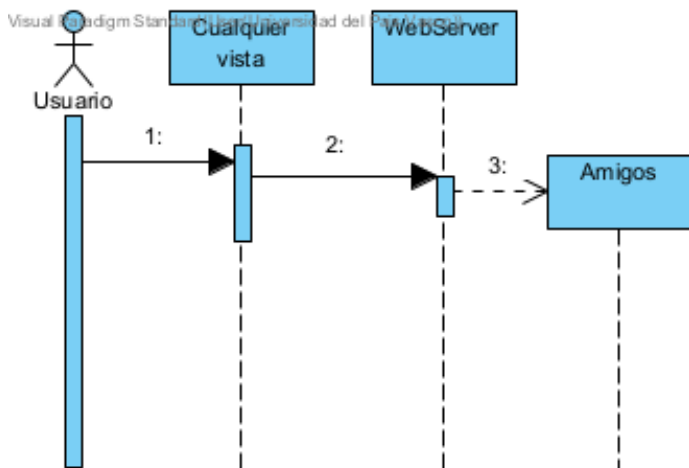
9: editarResena(resena, email, id, valoracion)

10: execSQL("UPDATE Reseña SET valoracion = %valoracion%, resena = %resena% WHERE idLibro = %id% AND emailUser = %email%")

11: new Reseñas()

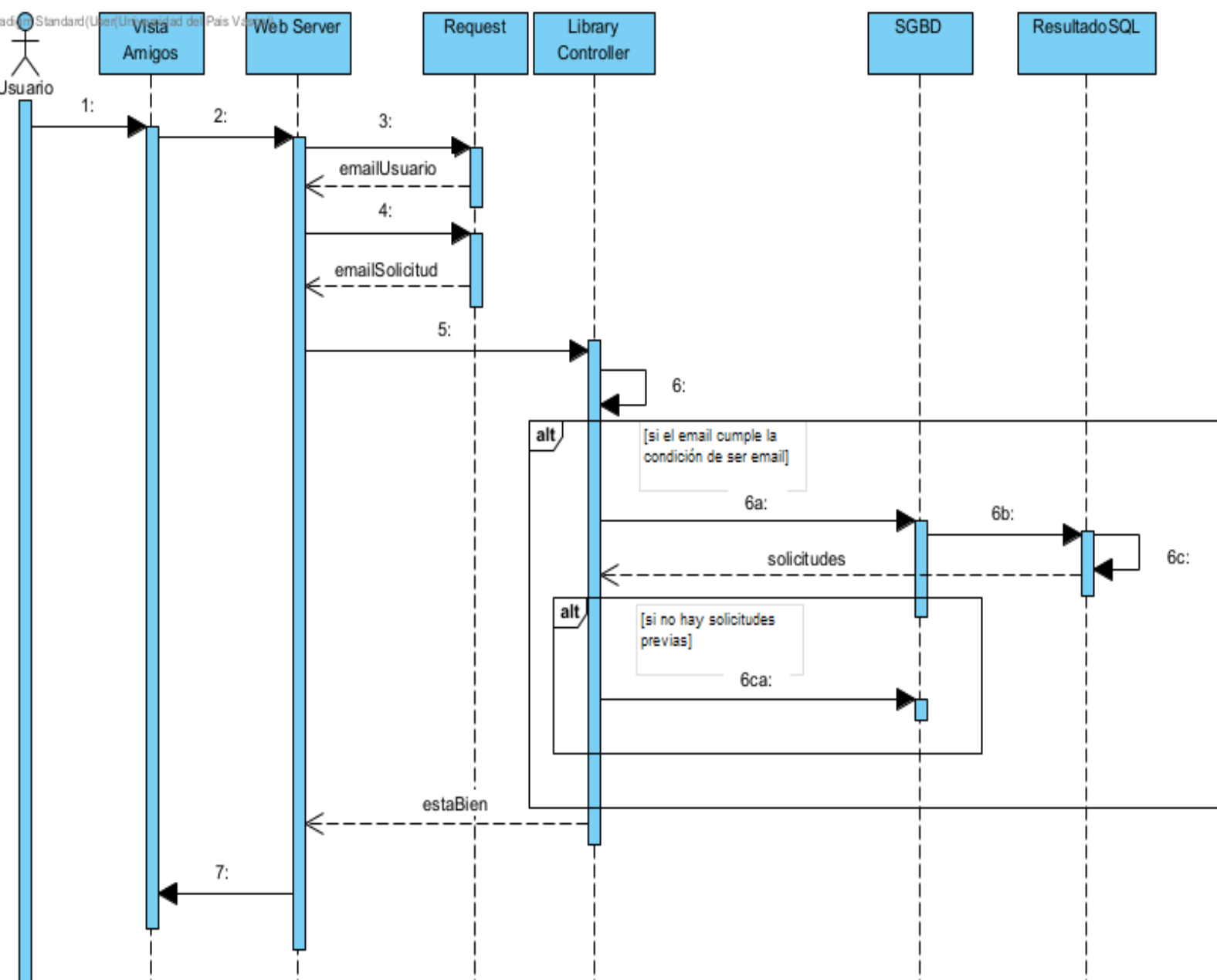
c) AMIGOS (ADRIAN)

C1.- Cargar pestaña de Amigos desde cualquier otra pestaña



1: Usuario hace click en Amigos (para ello tiene que estar logeado previamente)
2: amigos()
3: new Amigos()

C2.- Solicitar Amigo

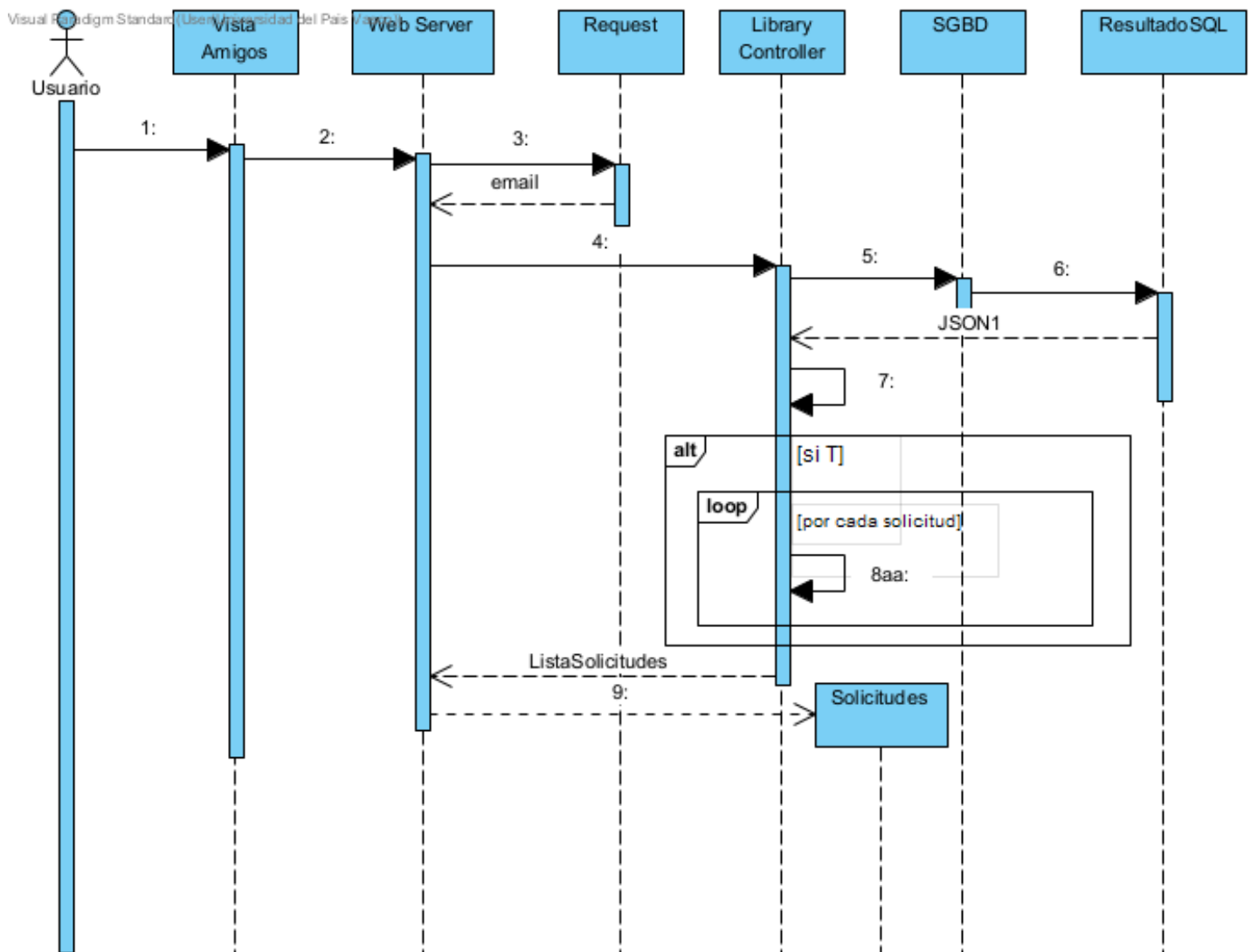



```

1: Usuario introduce un correo y pulsa en "Solicitar"
2: amigos()
3: emailUsuario = request.user.email
4: emailSolicitud = request.form["solicitarAmigo"]
5: enviarSolicitud(emailUsuario, emailSolicitud) : bool
6: re.match(patronEmail, emailSolicitud): bool
[si el email cumple la condición de ser mail]
6a: execSQL("SELECT emailUser1 FROM solicita WHERE emailUser2=emailObjetivo") : ResultadoSQL
6b: newResultadoSQL : ResultadoSQL
6c: next(): bool
[si no hay solicitudes previas]
6ca: execSQL("INSERT INTO solicita VALUES emailUsuario, emailObjetivo")
7: new Amigos()

```

C3.- Gestionar Solicitudes



```

1: Usuario clic en "Gestionar Solicitudes"
2: solicitudes()
3: email = request.user.email
4: getSolicitudes(email) : ListaSolicitudes
5: execSQL("SELECT * FROM solicita WHERE emailUser2 = email") :
ResultadoSQL
6: new ResultadoSQL : ResultadoSQL
7: next(): bool
[si T]
    [por cada solicitud]
        8aa: listaSolic.append(solicitud)
9: new Solicitudes()

```

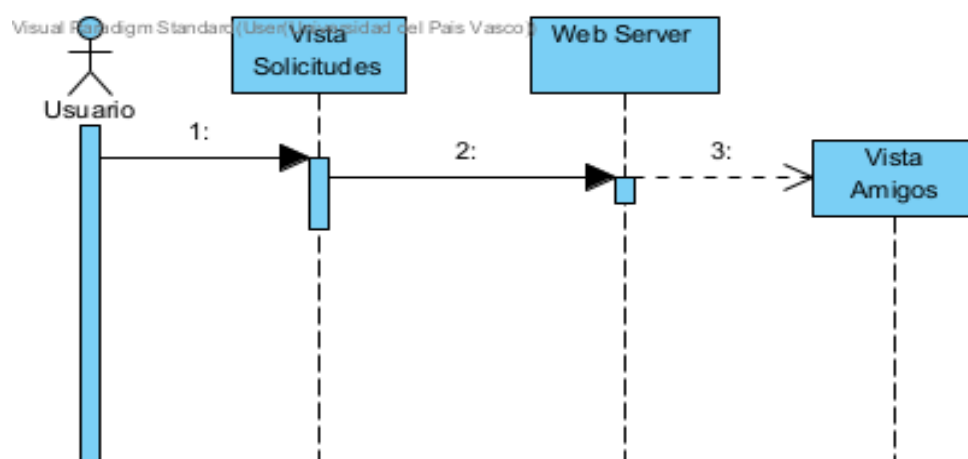
```

ResultadoSQL: {
  "emails": [
    {emailUser1: String, "emailUser2": String}
  ]
}

```

C4a.- Volver al menú de Amigos (desde gestionarSolicitudes)

Nota, esto se podría haber reciclado para otro diagrama posterior, pero se ha decidido mantener aunque sea algo redundante, para ayudar a la claridad del mismo.

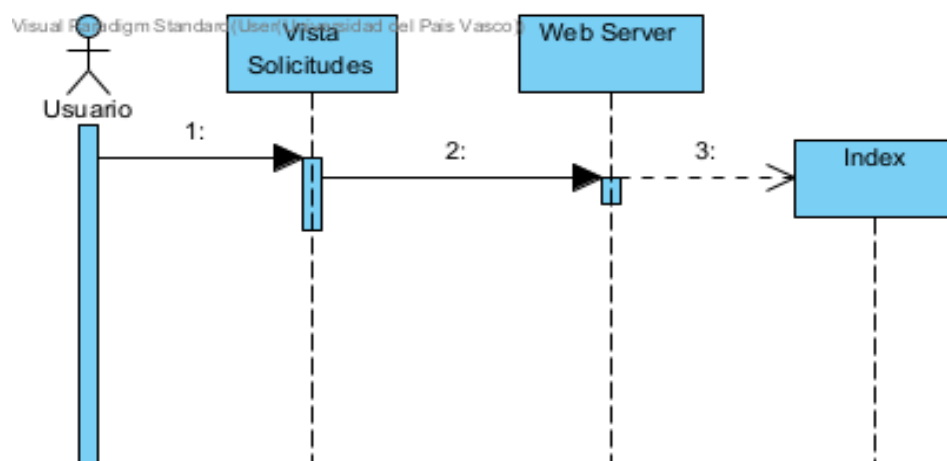


```

1: Usuario pulsa en "Volver al Menú de amigos"
2: amigos()
3: new Amigos()

```

C4b.- Volver al menú Principal (desde gestionarSolicitudes)

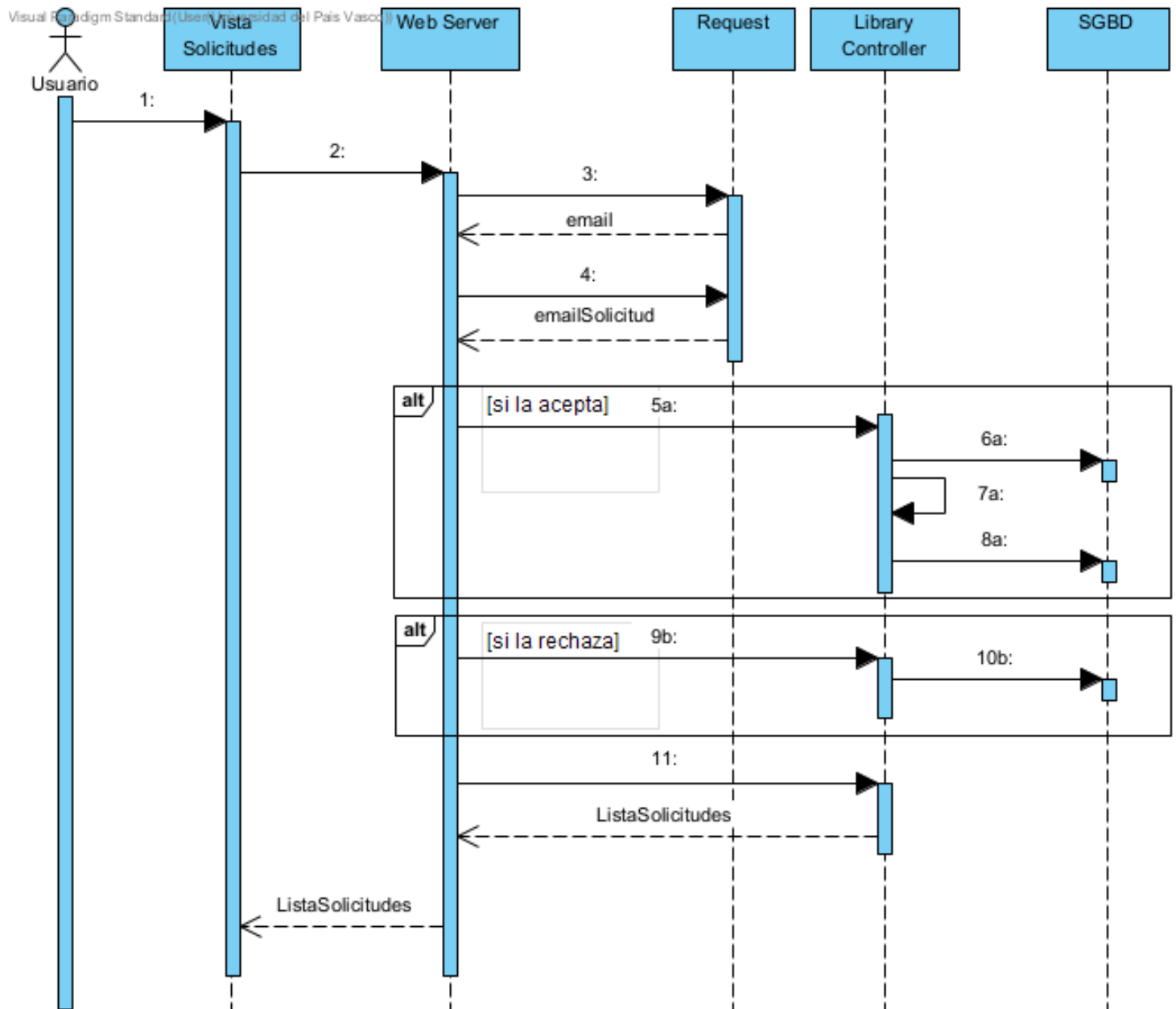


```

1: Usuario pulsa en "Volver al Menú de amigos"
2: index()
3: new Index()

```

C5.- Aceptar o Rechazar una solicitud



1: Usuario acepta o rechaza una solicitud.

2: solicitudes()

3: email = request.user.email

4: emailSolicitud = request.form.get("EmailSolicitud")

[si la acepta]

5a: aceptarSolicitud(email, emailSolicitud)

6a: execSQL("INSERT INTO SonAmigos VALUES (email, emailSolicitud)")

7a: rechazarSolicitud(email, emailSolicitud)

8a: execSQL("DELETE FROM solicita WHERE emailUser1 = emailSolicitud AND emailUser2 = email")

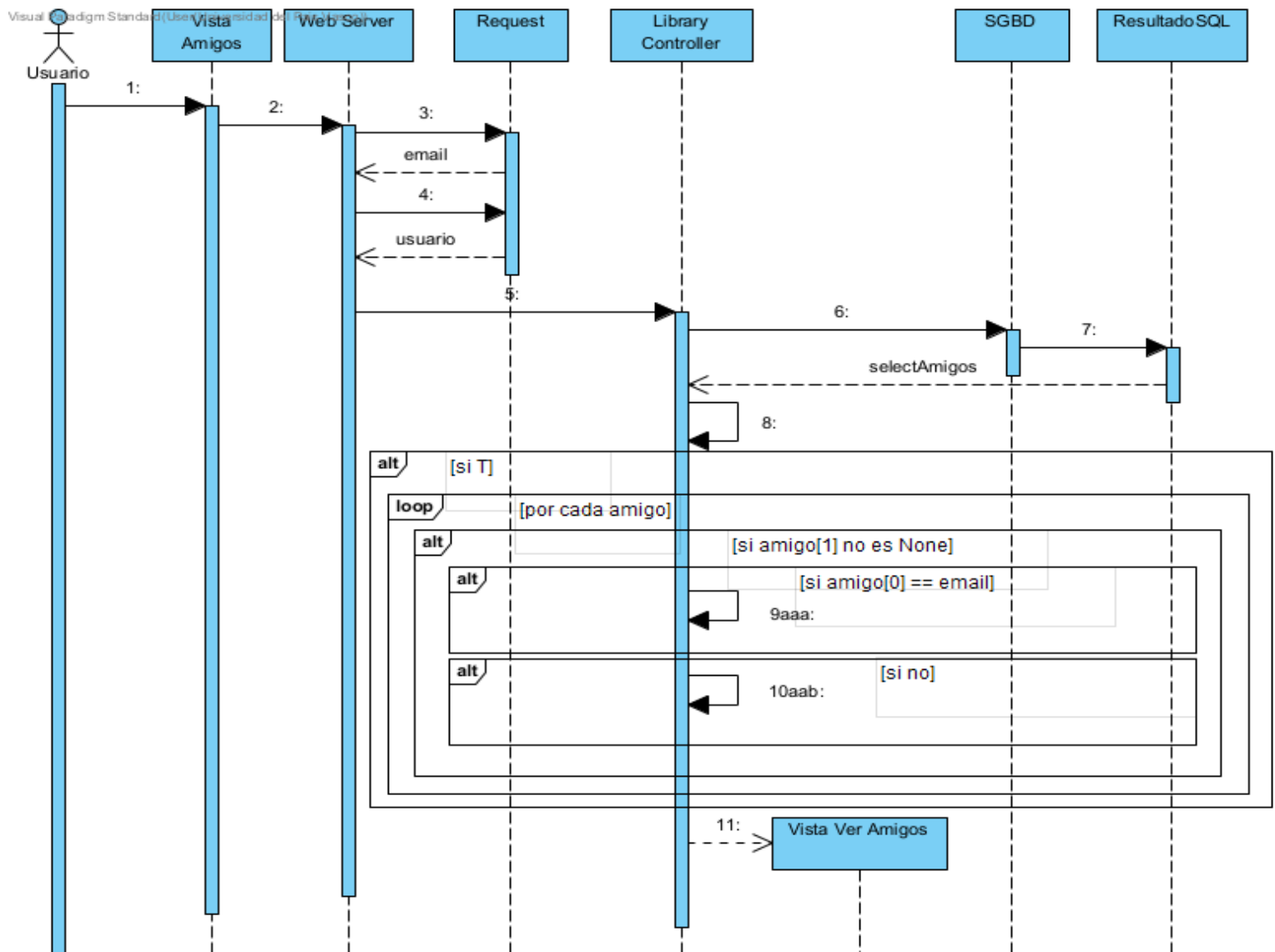
[si la rechaza]

9b: rechazarSolicitud(email, emailSolicitud)

10b: execSQL("DELETE FROM solicita WHERE emailUser1 = emailSolicitud AND emailUser2 = email")

11: getSolicitudes(email): ListaSolicitudes # Desarrollado anteriormente

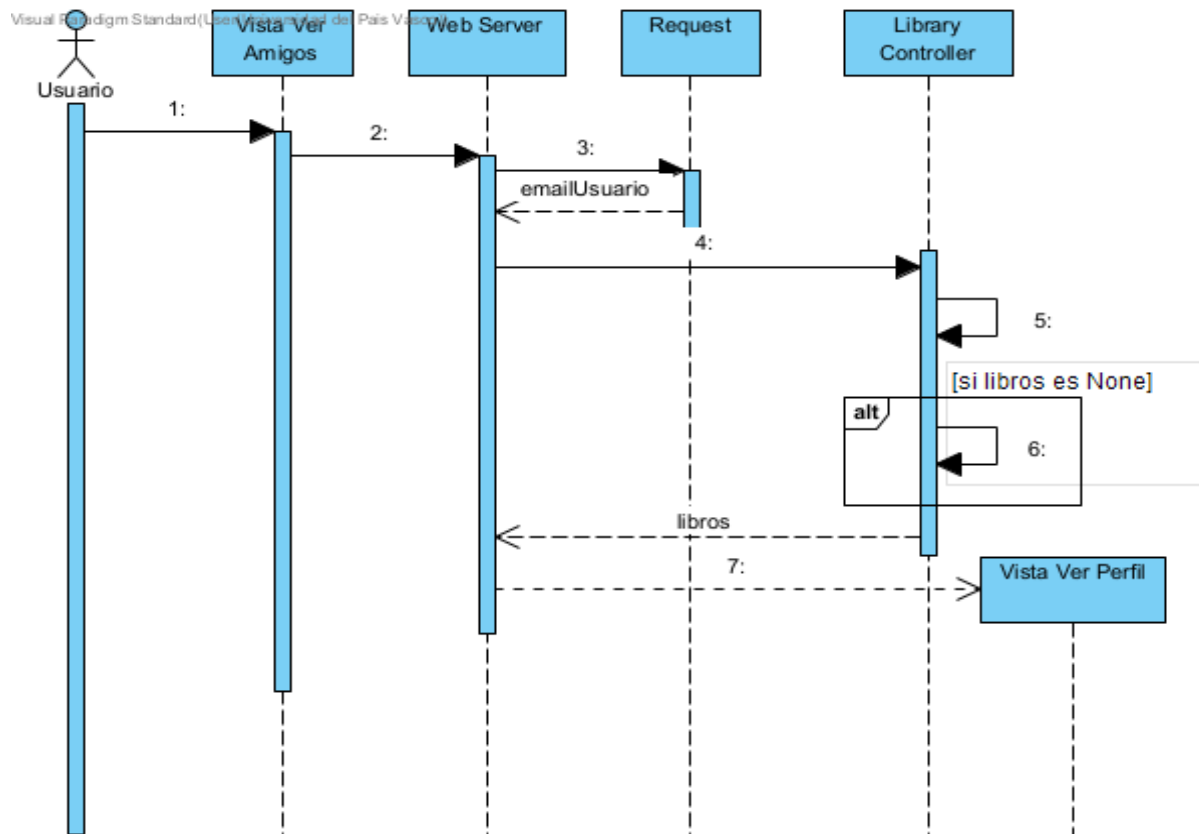
C6.- Ver Amigos



```

1: Usuario pulsa en "Ver Amigos"
2: verAmigos()
3: email = request.user.email
4: usuario = request.user.username
5: amigos = getAmigos(email): ListaAmigos
6: execSQL("SELECT * FROM SonAmigos WHERE
emailUser1 = email OR emailUser2 = email") : ResultadoSQL
7: new ResultadoSQL : ResultadoSQL
8: next(): bool
[si T]
    [por cada amigo]
        [si emailUser2 no es None]
            [si emailUser1 == email]
                9aaa: amigos.append(amigo[1])
            [si no]
                10aab: amigos.append(amigo[0])
            end
        end
    end
11: new verAmigos()
  
```

C7.- Ver Perfil



1: Usuario pincha en "Ver Perfil" de un usuario

2: verPerfil()

3: emailUsuario = request.args.get("emailUsuario")

4: obtenerDatosPerfil(emailUsuario) : JSON1

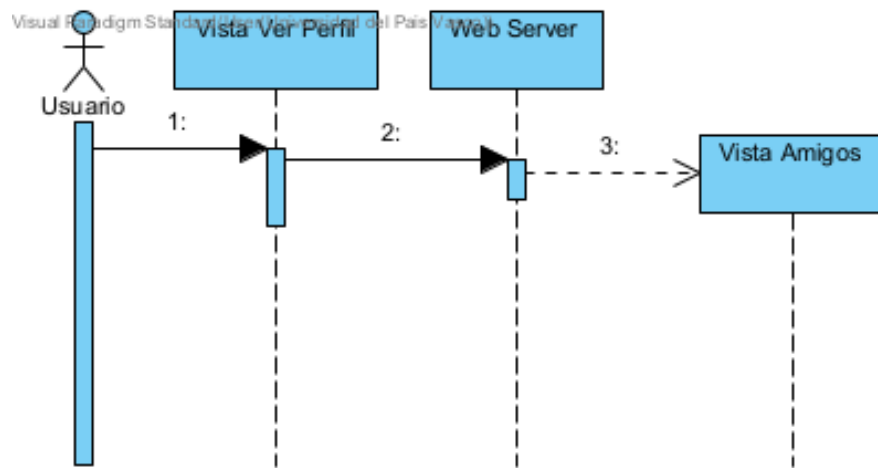
5: libros = getLibrosLeidos(emailUsuario): ListaBooks #Desarrollado en recomendaciones del sistema

[si libros es None]

6a: libros = []

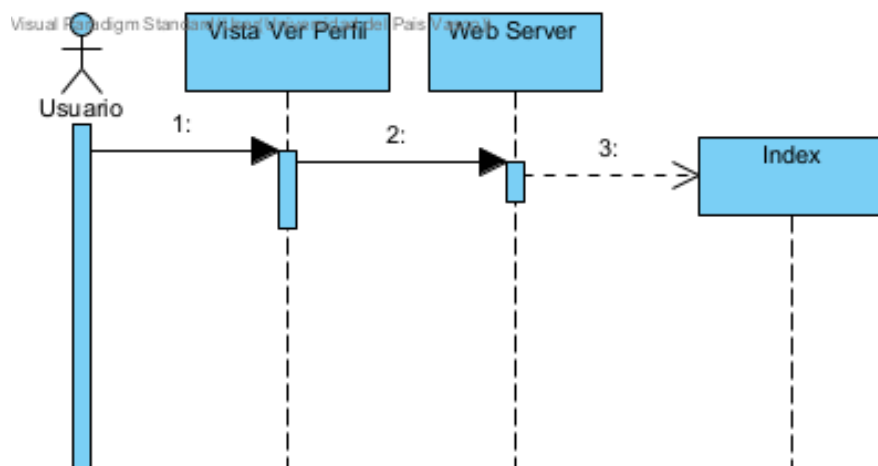
7: new verPerfil()

C8a.- Volver al menú de Amigos (desde verPerfil)



1: Usuario pulsa en "Ver Amigos"
2: verAmigos()
3: new verAmigos()

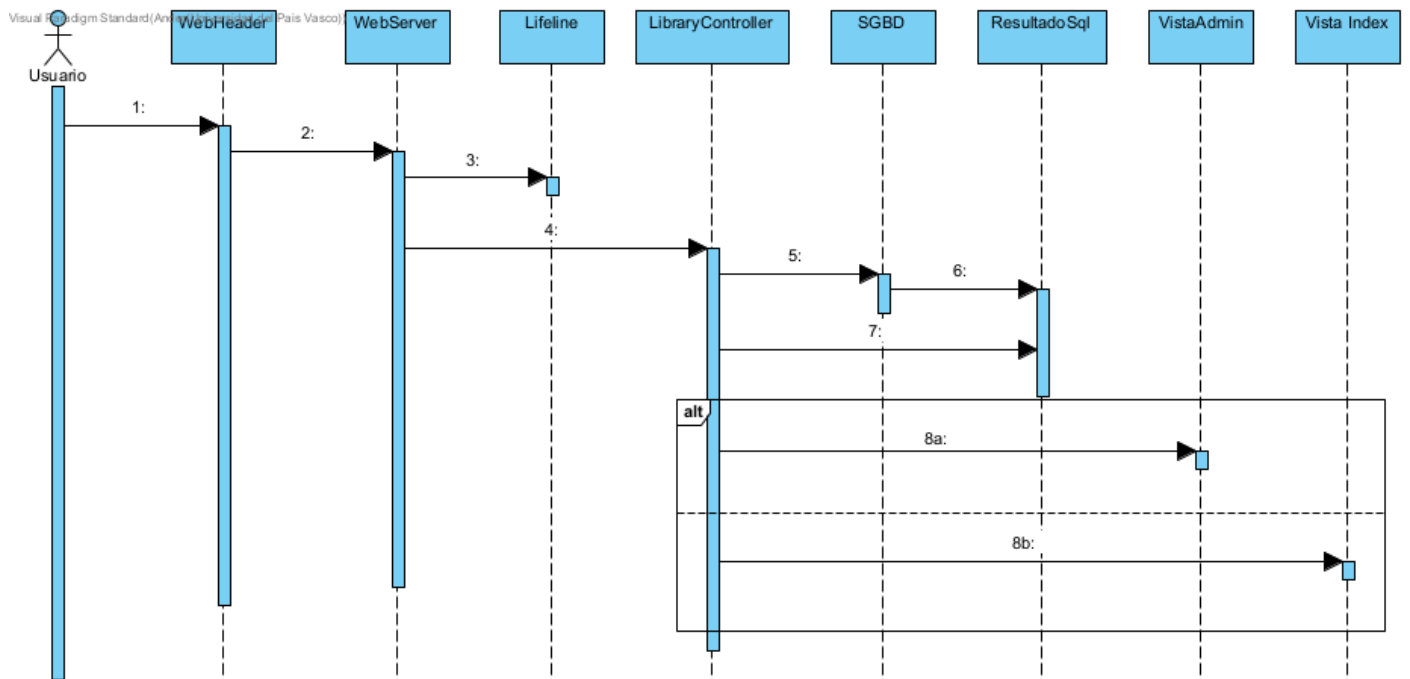
C8b.- Volver al menú Principal (desde gestionarSolicitudes)



1: Usuario pulsa en "Menú Principal"
2: index()
3: new Index()

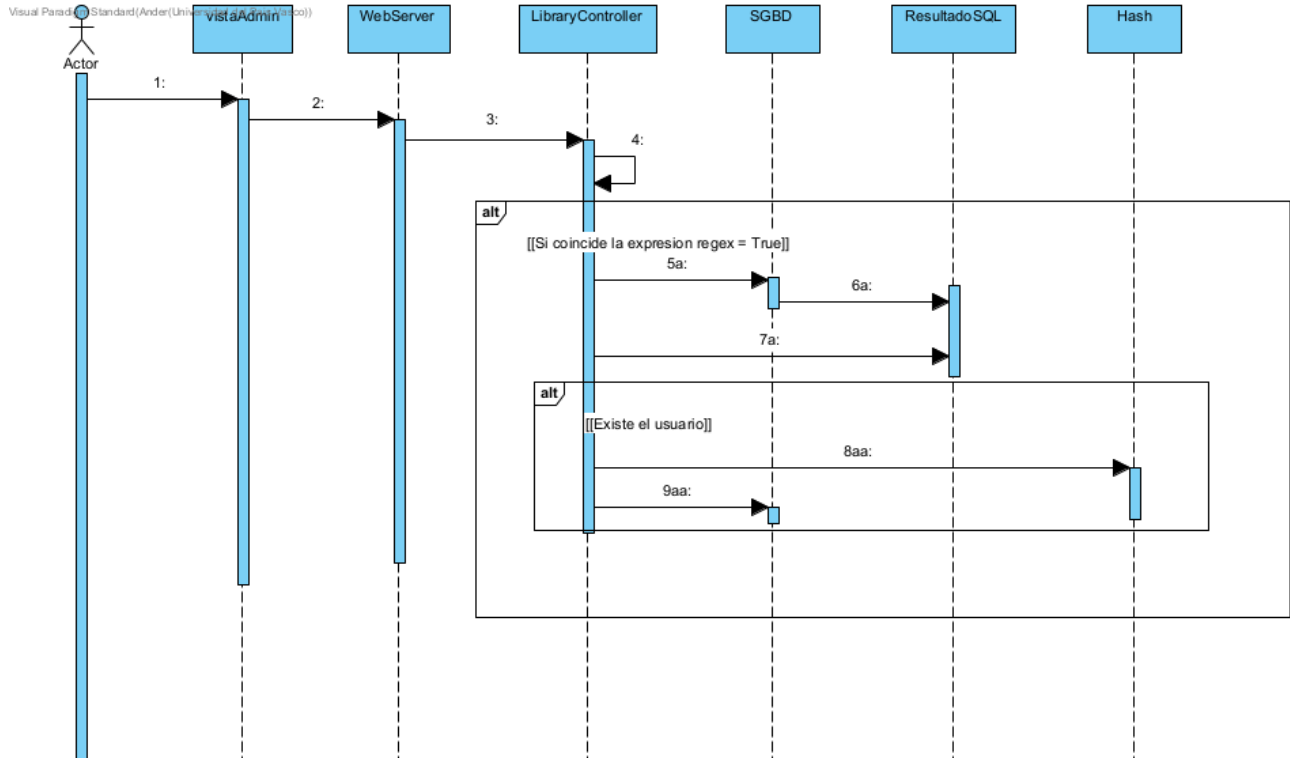
D) FORO (ANDER)

D1.- Acceder a administrador



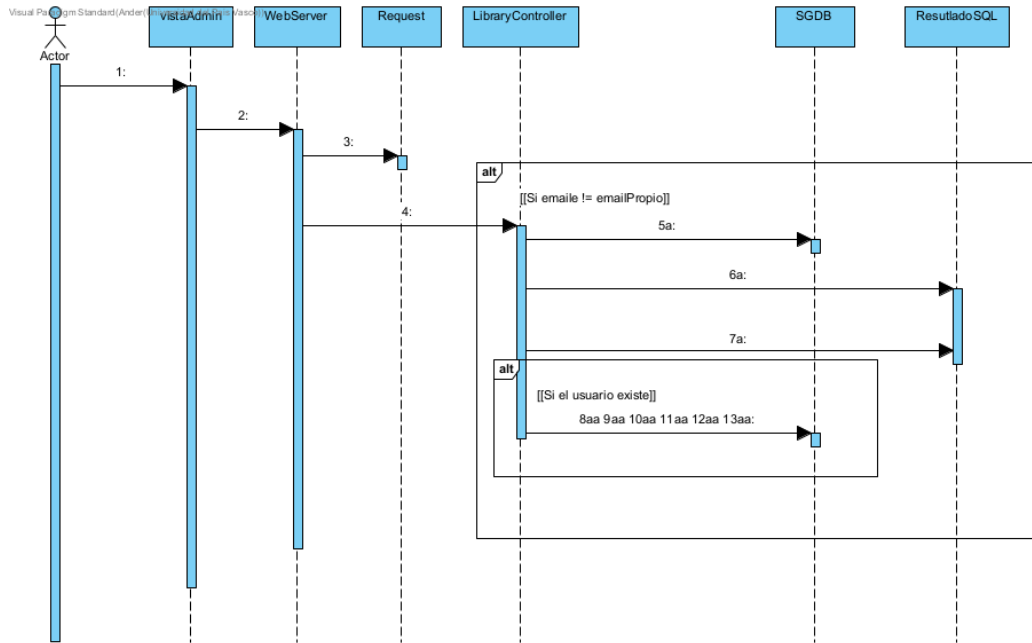
1: El usuario logeado como admin clicka en el boton de admin en el header desde cualquier vista
2: admin() : void
3: email = request.user.email
4: admin = esAdmin(email) : boolean
5: res = db.select(SELECT admin from USER WHERE email = %email%)
6: new ResultadoSQL()
7: getAdmin() : int
[If admin = 1]
 8a: new VistaAdmin()
[If admin = 0]
 8b: new VistaIndex()

D2.-CrearUsuario



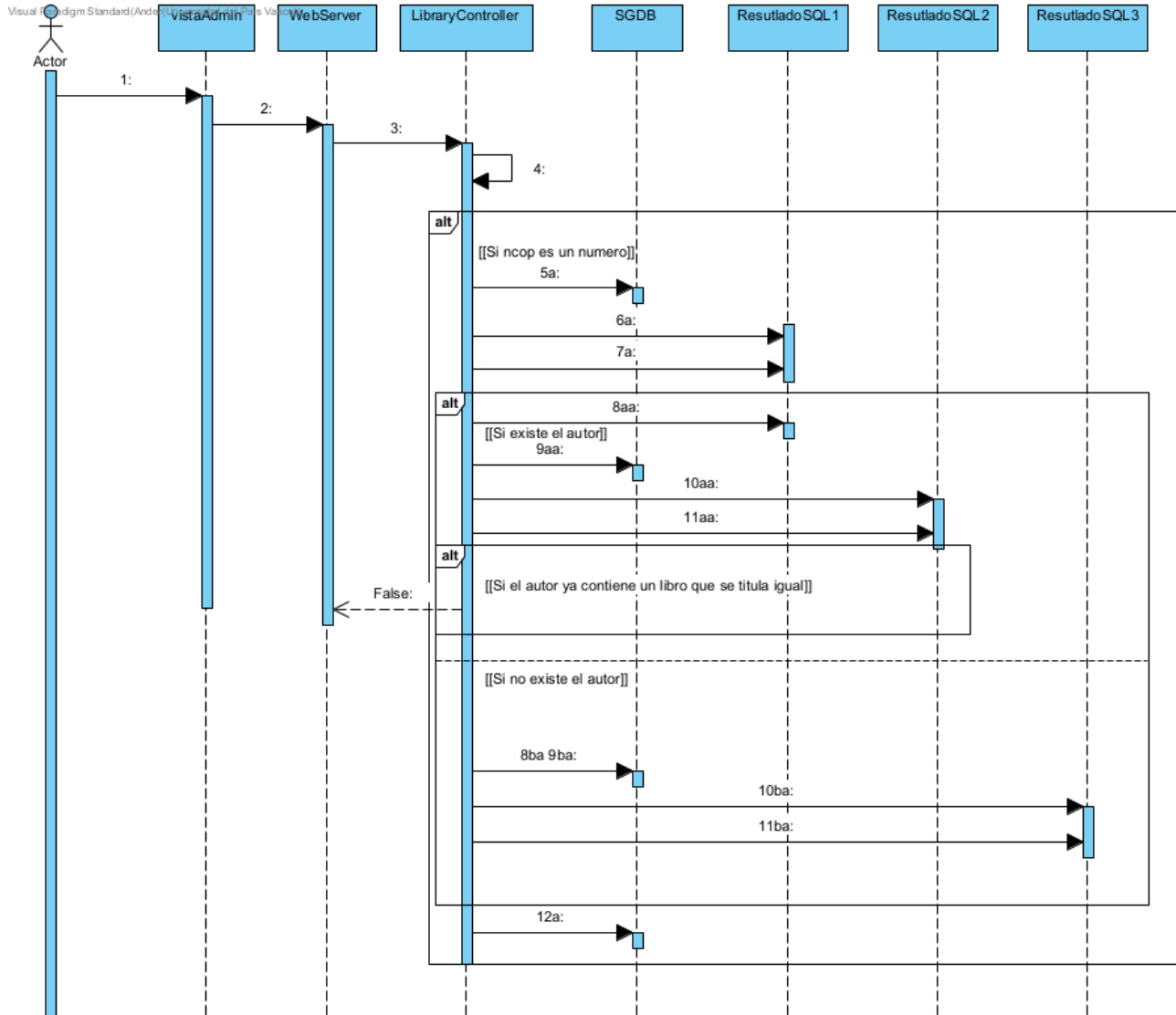
1: Rellena el formulario para crear un usuario y lo envía
2: admin()
3: nuevo_usuario(email,nombre,contraseña,admin) : void
4: validar_email(email): boolean
[Si coincide con la expresion regex = True]
5a: existe = db.select("SELECT * from USER WHERE email = %email%")
6a: new ResultadoSQL()
7a: getTamañoResultado() : int
[Si no existe el usuario]
8aa: md5(claveSalteada) : string
9aa: db.insert("INSERT INTO USER VALUES (email,nombre,claveSalteada,esUnAdmin)")
[Si existe el usuario]
Fin
[Si no coincide la expresion regex]
Fin

D3. Eliminar Usuario



1: El admin rellena el formulario para eliminar un usuario y lo envia
2: admin()
3: request.user.email
4: eliminarUsuario(email, emailPropio) : void
[si email != emailPropio]
5a: db.select("SELECT * FROM User WHERE email = emailElim")
6a: new ResultadoSQL()
7a: getTamano() : int
[Si el usuario existe]
8aa: db.update("UPDATE Tema SET emailUser = 'deleted@user.com' WHERE emailUser = emailElim")
9aa: db.update("UPDATE Comenta SET emailUser = 'deleted@user.com' WHERE emailUser = emailElim")
10aa: db.update("UPDATE Reseña SET emailUser = 'deleted@user.com' WHERE emailUser = emailElim")
11aa: db.delete("DELETE FROM SonAmigos WHERE emailUser1 = emailElim OR emailUser2 = emailElim")
12aa: db.delete("DELETE FROM Prestar WHERE emailUser = emailElim")
13aa: db.delete("DELETE FROM user WHERE email = emailElim")
[Si el usuario no existe]
Fin
[Si email == emailPropio]
Fin

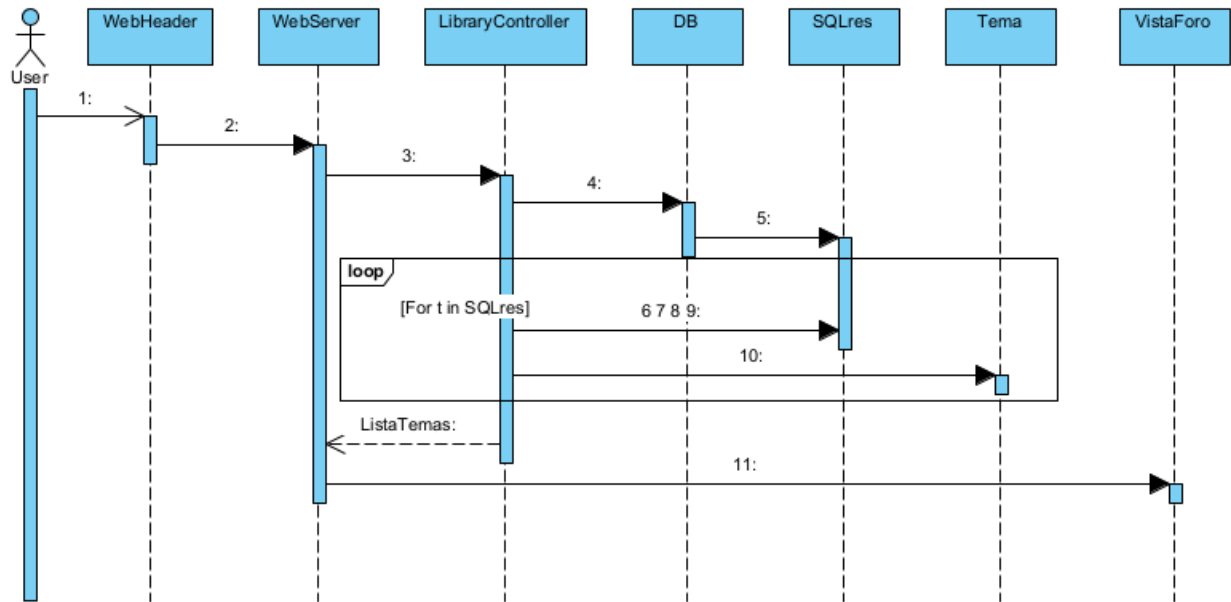
D4. Añadir Libro



1: El admin rellena el formulario para añadir un libro y lo envía
 2: admin() : void
 3: nuevo_libro(titulo,autor,ncop,desc,portada) : void
 4: es_numero(ncop) : boolean
 [[Si ncop es un numero]
 5a: db.select("SELECT id from AUTHOR WHERE name = autor")
 6a: new ResultadoSQL()
 7a: getTamano() : int
 [[Si existe el autor]
 8aa: getIdAutor() : int
 9aa: db.select("SELECT * from BOOK WHERE title = titulo AND author = getIdAutor()")
 10aa: new ResultadoSQL()
 11aa: getTamano() : int
 [[Si el autor ya contiene un libro que se titula igual]
 Fin (Return False)
 [[Si no existe el autor]
 8ab: db.insert("INSERT INTO AUTHOR (name) VALUES (autor)")
 9ab: db.select("SELECT id from AUTHOR WHERE name = autor")
 10ab: new ResultadoSQL()
 11ab: getIdAutor() : int
 12a: db.insert("INSERT INTO BOOK (title,author,cover,description) VALUES (titulo,getIdAutor,portada,desc,)"

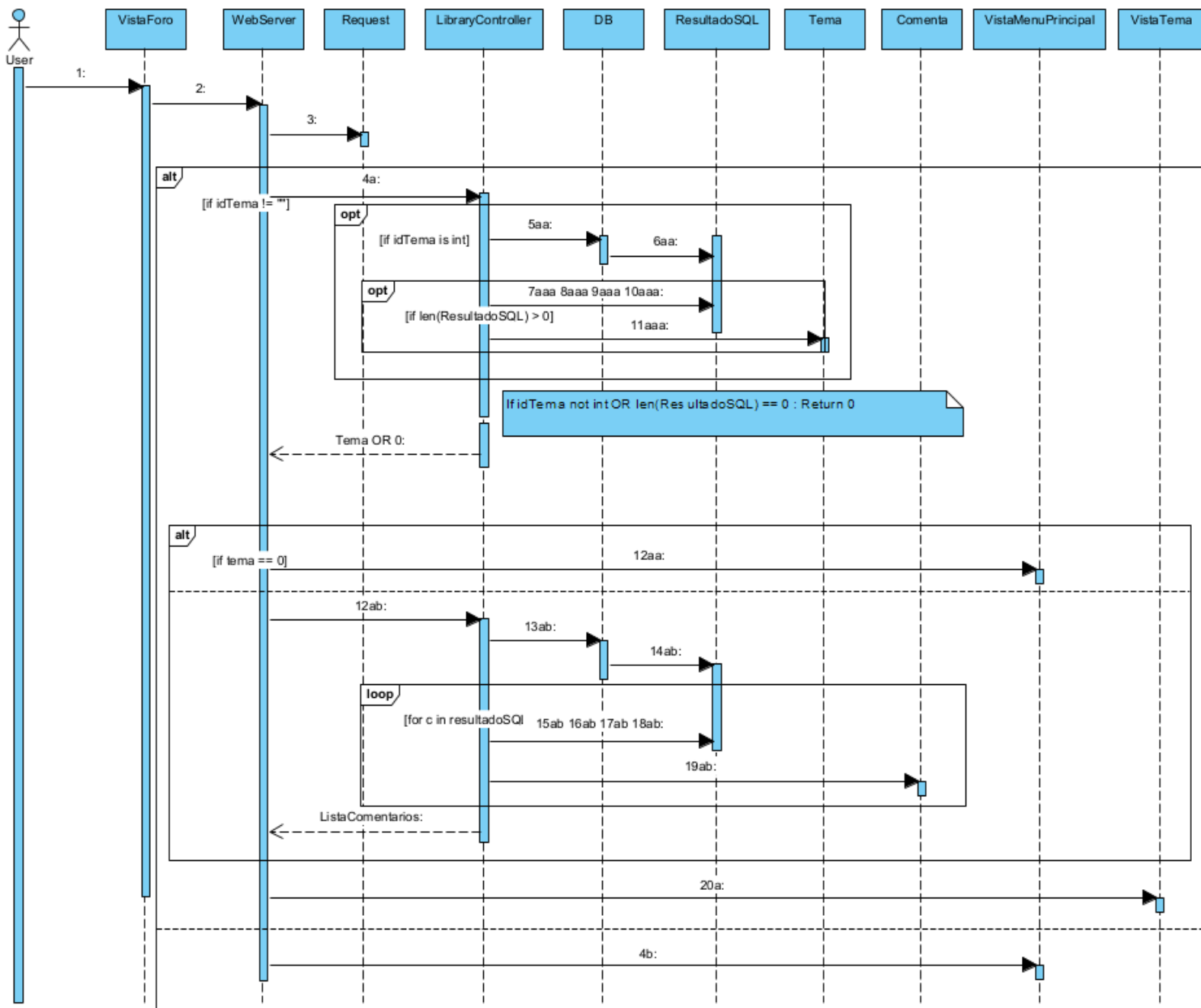
e) FORO (ENEKO)

e1.- Usuario accede al foro



1. User o cualquiera pulsa "foro" en el header
 2. forum()
 3. get_temas() : ListaTemas
 4. execSQL("SELECT * FROM Tema") : ResultadoSQL
 5. new ResultadoSQL() : ResultadoSQL
 6a. id : get(0) : int
 7a. titulo : get(1) : string
 8a. emailUser : get(2) : string
 9a. descTema : get(3) : string
 10a. new Tema(id, titulo, emailUser, descTema) : Tema
 11. new VistaForo(listaTemas)

e2.- Usuario accede a un tema desde el foro



```

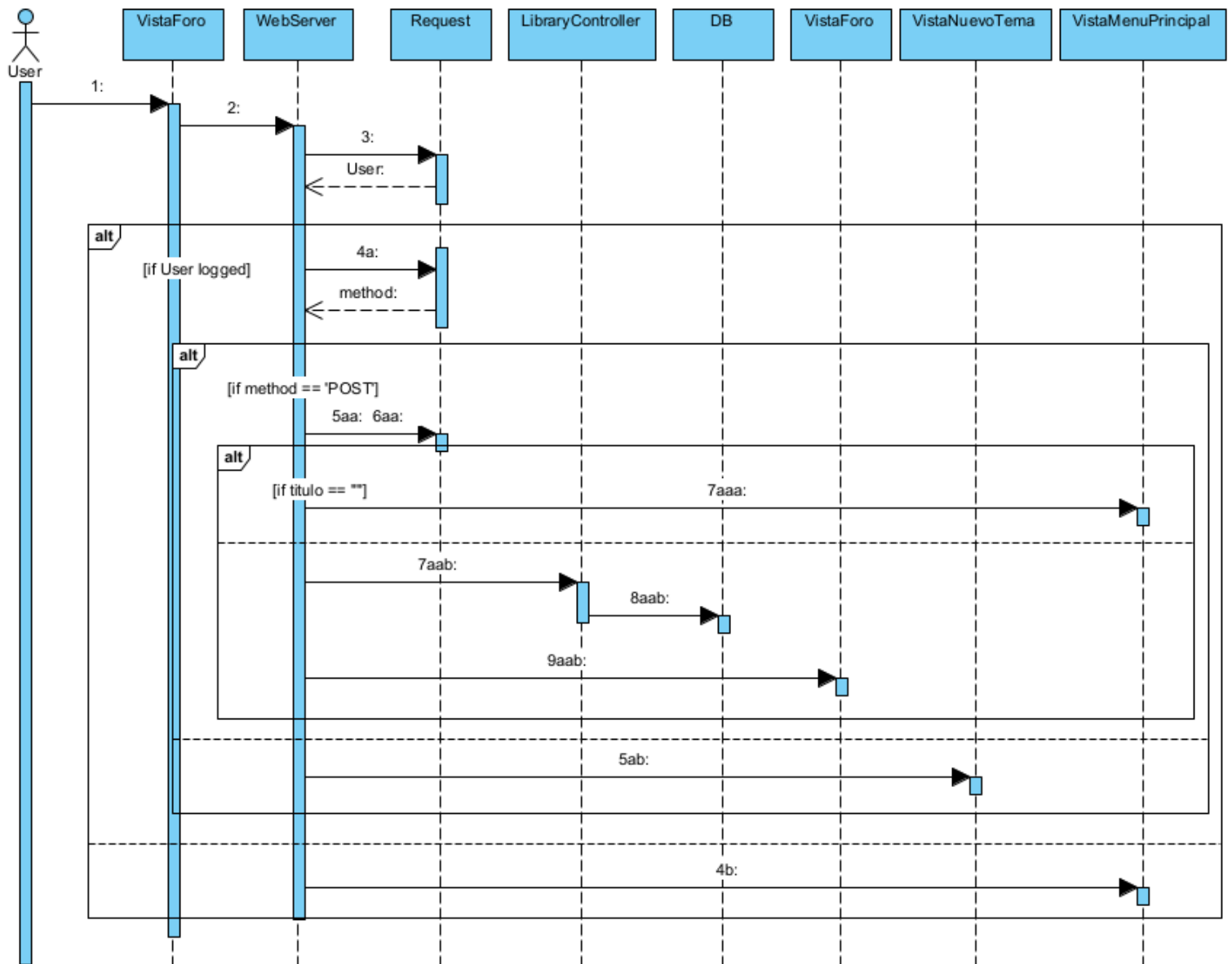
1. User o cualquiera pulsa sobre un tema de los mostrados
2. tema()
3. values.get("id", "")
[if idTema != ""]
    4a. eITema : get_tema(idTema) : Tema OR 0
    [if idTema is int]
        5aa. execSQL("SELECT * FROM Tema WHERE id = %idTema%") : ResultadoSQL
        6aa. new ResultadoSQL() : ResultadoSQL
        [if len(ResultadoSQL) > 0]
            7aaa. id : get(0) : int
            8aaa. titulo : get(1) : string
            9aaa. emailUser : get(2) : string
            10aaa. descTema : get(3) : string
            11aaa. new Tema(id, titulo, emailUser, descTema) : Tema

    [if eITema == 0]
        12aa. new MenuPrincipal()
    [else]
        12ab. getComentarios(idTema) : listaComentarios
        13ab. execSQL("SELECT * FROM Comenta WHERE idTema = %idTema% ORDER BY fechaHora") : ResultadoSQL
        14ab. new ResultadoSQL() : ResultadoSQL
        15ab. emailUser : get(0) : string
        16ab. idTema : get(1) : int
        17ab. mensaje : get(2) : string
        18ab. fechaHora : get(3) : string
        19ab. new Comenta(emailUser, idTema, mensaje, fechaHora) : Comenta

    20a. new VistaTema(listaComentarios, listaTemas)
[else]
    4b. new VistaMenuPrincipal()

```

e3.- Usuario crea un nuevo tema



```
1. Usuario pulsa nuevo tema en foro
2. nuevoTema()
3. getUser() : User

[if User logged]
  4a. method : getRequestMethod() : string

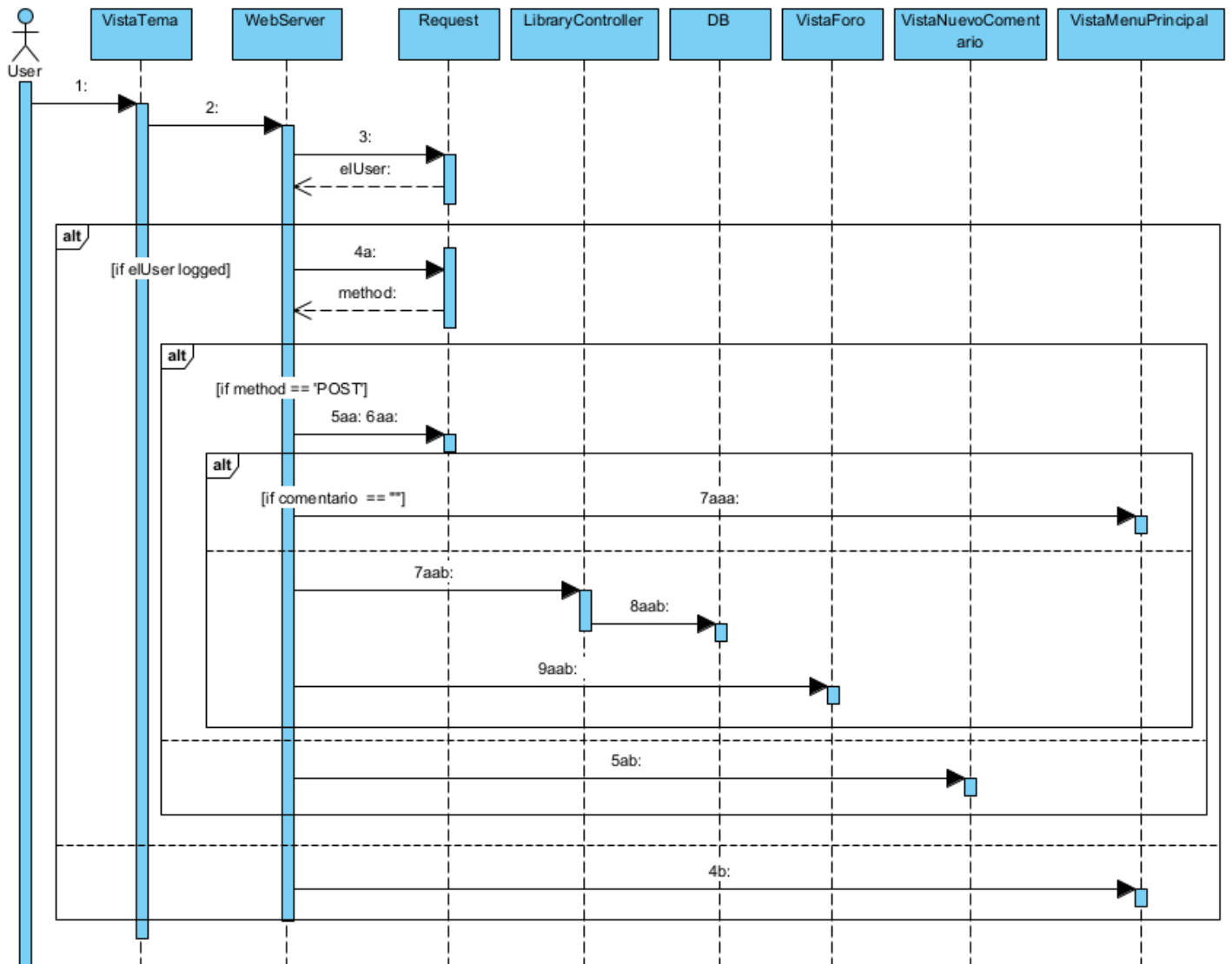
  [if method == 'POST']
    5aa. titulo : form['titulo'] : string
    6aa. descripcion : form['descripcion'] : string

    [if titulo == ""]
      7aaa. new MenuPrincipal()

    [else]
      7aab. nuevoTema(titulo, descripcion, request.user.email)
      8aab. insert("INSERT INTO Tema (titulo, emailUser, descTema) VALUES (%titulo%, %descripcion%, %request.user.email%)")
      9aab. new VistaForo()

  [else]
    5ab. new VistaNuevoTema()
[else]
  4b. new VistaMenuPrincipal()
```

e4.- Usuario comenta en un tema




```
1. Usuario pulsa "Comentar"
2. nuevoComentario()
3. elUser : getUser() : User

[if elUser logger]
  4a. method : getRequestMethod() : String

  [if method == 'POST']
    5aa. comentario : form['comentario'] : string
    6aa. id : values.get("id")

    [if comentario == ""]
      7aaa. new VistaMenuPrincipal()

    [else]
      7aab. nuevoComentario(comentario, elUser.email, id)
      8aab. insert(INSERT INTO Comenta (mensaje, emailUser, idTema, fechaHora) VALUES (%comentario%, %elUser.email%, %id%, datetime('now')))
      9aab. new VistaForo()

  [else]
    5ab. new VistaNuevoComentario()

[else]
  4b. new VistaMenuPrincipal()
```


f) RECOMENDACIONES DEL SISTEMA (JAVIER)

f1.- General

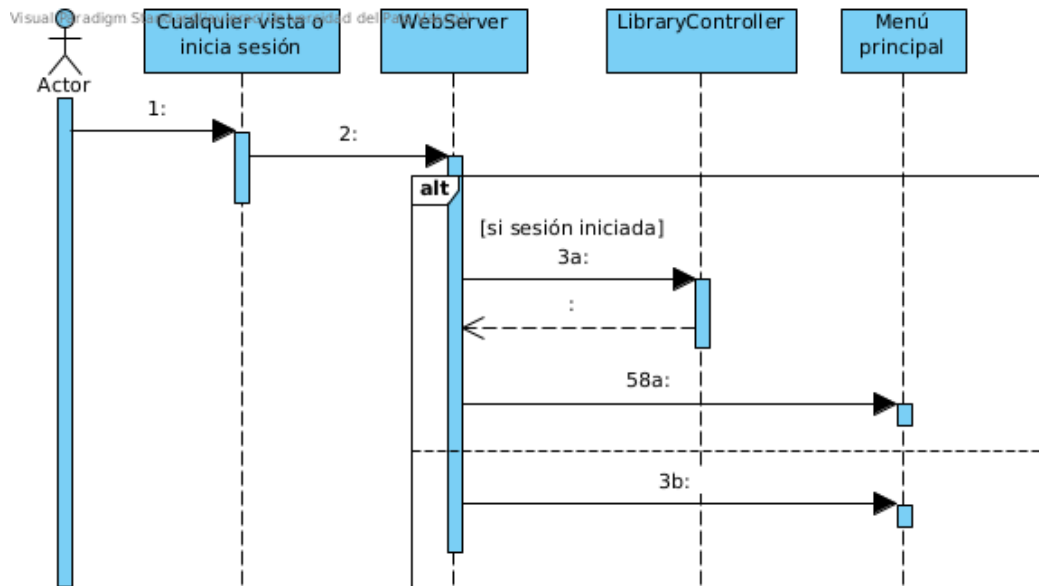


Figura 79: Diagrama de Secuancia de Recomendaciones: General

1: Usuario hace click en "Inicio" o inicia sesión

2: index(): new Inicio -menú principal-

[si hay una sesión iniciada]

3a: recomendaciones = obtenerRecomendaciones(email: String): ListaBooks

...

58a: new Menú Principal(recomendaciones = sugeridos: ListaBooks)

[si no]

3b: new Menú Principal()

f2.- General con sesión iniciada

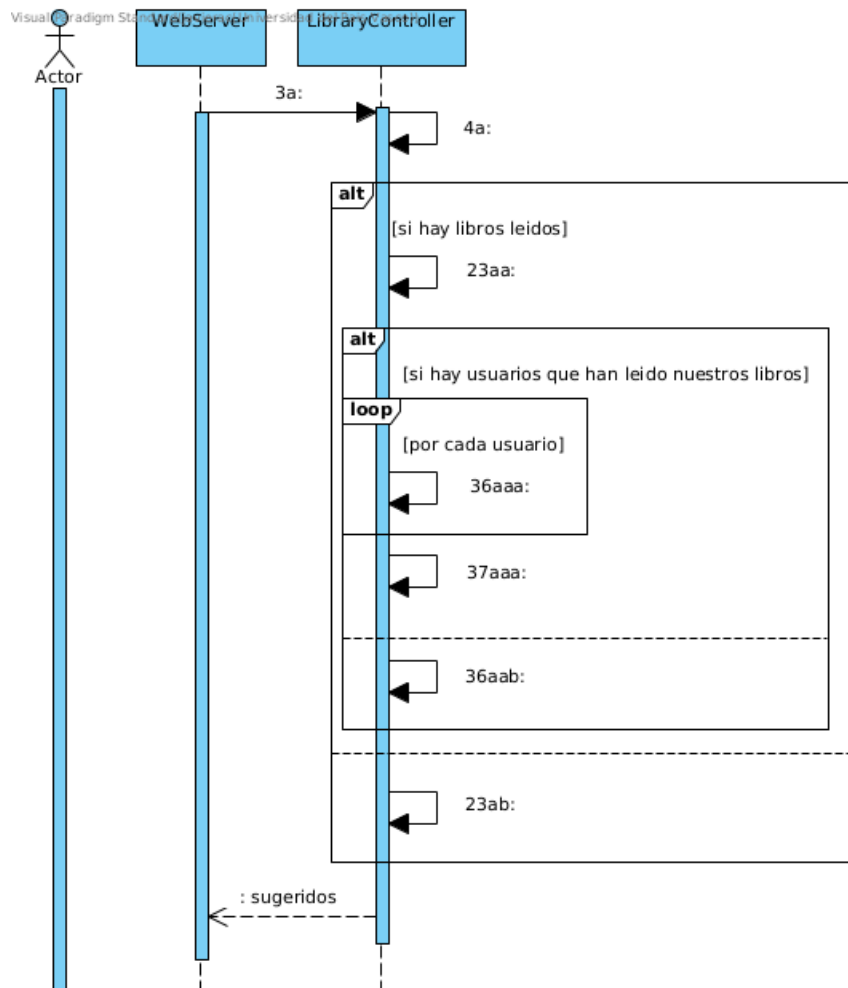


Figura 80: Diagrama de Secuencia de Recomendaciones:
General con sesión iniciada

3a: recomendaciones = obtenerRecomendaciones(email: String): ListaBooks

4a: leídos = getLibrosLeídos(email: String): ListaBooks

[si hay libros leídos]

23aa: getUsersHaLeído(leídos: ListaBooks, email: String): ListaUsers

[si hay usuarios que han leído nuestros libros]

36aaa: sugeridos = getLibrosLeídos(usuarioEmail: String): ListaBooks

37aaa: deleteRepeated(leídos: ListaBooks, sugeridos: ListaBooks): ListaBooks

[si no]

36aab: sugeridos = getLibrosRandom(email: String): ListaBooks

[si no]

23ab: sugeridos = getLibrosRandom(email: String): ListaBooks

f3.- Obtener los libros que se han leído (getLibrosLeidos)

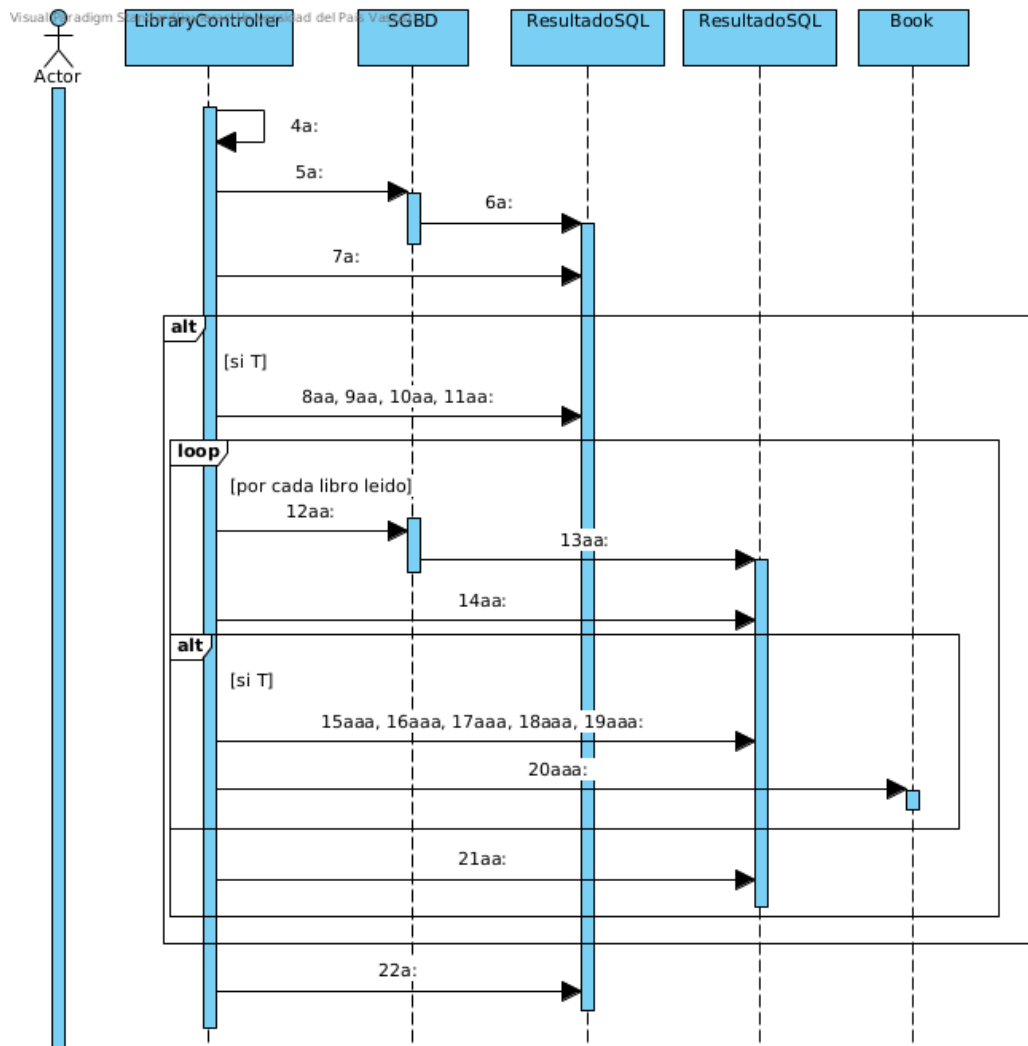


Figura 81: Diagrama de Secuencia de Recomendaciones: getLibrosLeidos

4a: leidos = getLibrosLeidos(email: String): ListaBooks

5a: execSQL("SELECT * from Prestar WHERE emailUser LIKE %email%")

6a: new ResultadoSQL()

7a: next(): bool

[si hay libros leídos]

8aa: getString("emailUser")

9aa: getInt("idLibro")

10aa: getString("fechaHora")

11aa: getString("fechaFin")

12aa: execSQL("SELECT * from Book WHERE id = %idLibro%")

13aa: new ResultadoSQL()

14aa: next(): bool

[si T]

15aaa: getInt("id")

16aaa: getString("title")

17aaa: getInt("author")

18aaa: getString("cover")

19aaa: getString("description")

20aaa: new Book(id: int, title: String, author: int, cover: String, description: String)

21aa: close()

22a: close()

f4.- Obtener usuarios que han leído los mismos libros (getUsuariosHaLeido)

Qual Paradigm Standard(javierac(Universidad del País Vasco))

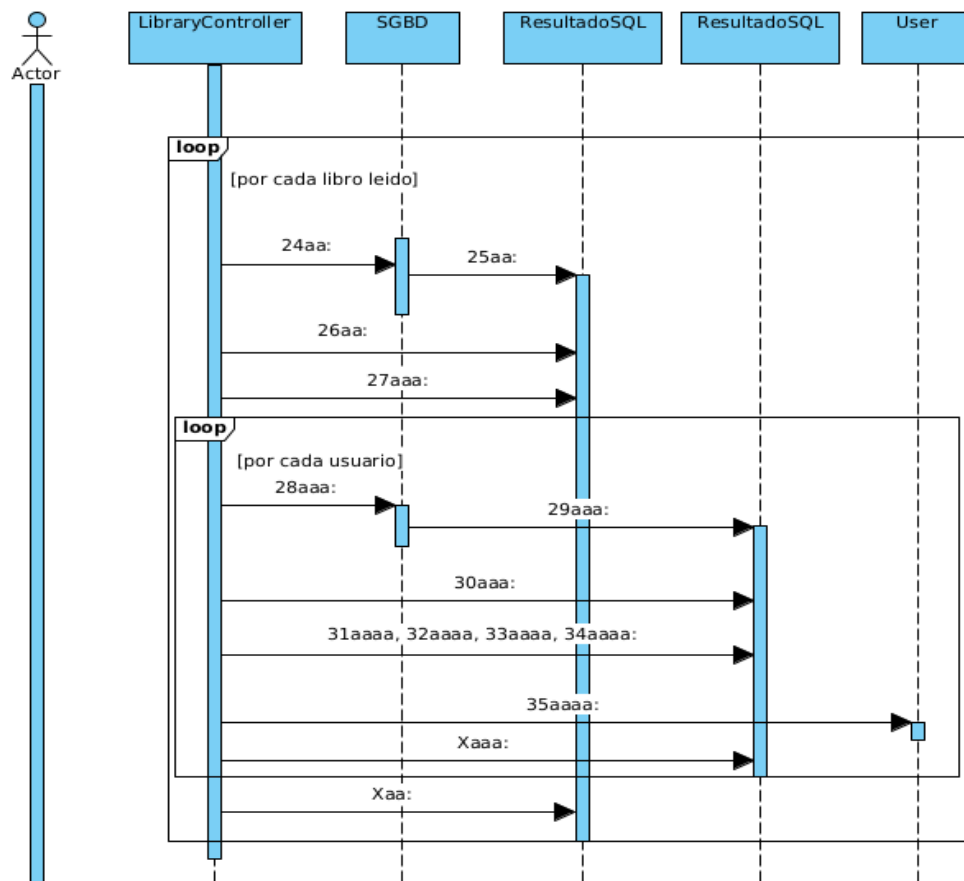


Figura 82: Diagrama de Secuencia de Recomendaciones:
getUsuariosHaLeido

24aa: `execSQL("SELECT emailUser from Prestar WHERE idLibro %idLibroLibroLeido%
AND emailUser NOT LIKE %email%")`

25aa: `new ResultadoSQL()`

26aa: `next(): bool`

[si T]

27aaa: `getString("emailUser")`

28aaa: `execSQL("SELECT * from User WHERE email =%emailUser%")`

29aaa: `new ResultadoSQL()`

30aaa: `next(): bool`

[si T]

31aaaa: `getString("email")`

32aaaa: `getString("name")`

33aaaa: `getString("password")`

34aaaa: `getBool("admin")`

35aaaa: `new User(email: String, name: String, password: String, admin: Bool)`

Xaaa: `close()`

Xaa: `close()`

f5.- Eliminar libros repetidos entre los sugeridos (deleteRepeated)

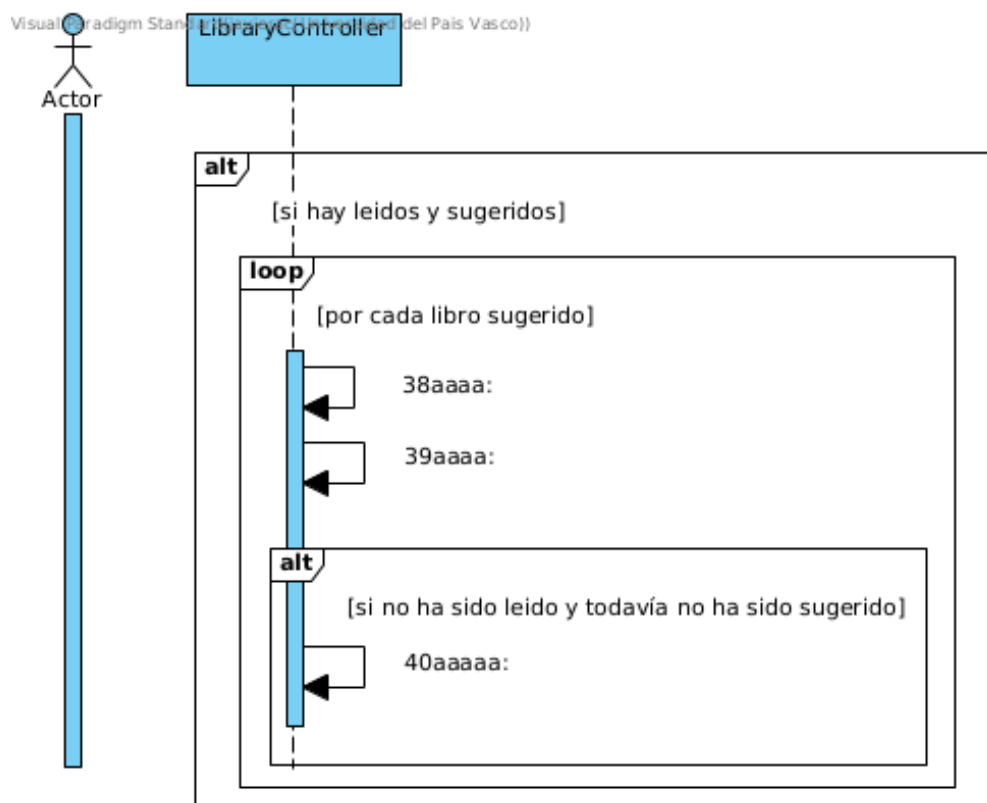


Figura 83: Diagrama de Secuencia de Recomendaciones: *deleteRepeated*

[si hay leidos y sugeridos]

38aaaa: `isRead(leidos: ListaBooks, idLibroSugerido: int)`

39aaaa: `isRead(sugeridos: ListaBooks, idLibroSugerido: int)`

[si ambos son False (no ha sido leído ni sugerido)]

40aaaaa: `añadir(libroSugerido: Book)`

f6.- Obtener top 10 libros más leídos (getLibrosRandom)

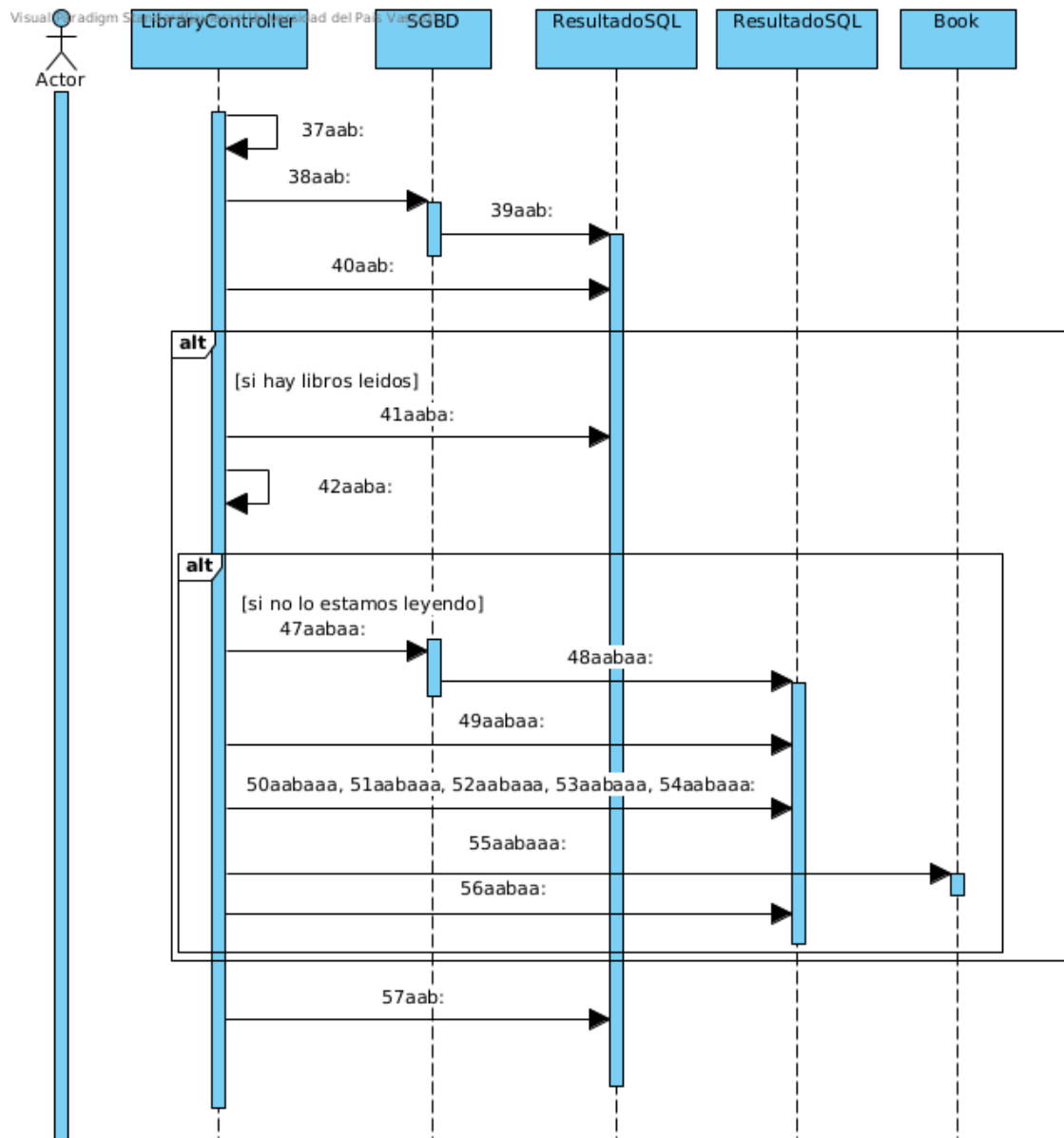


Figure 85: Diagrama de Secuencia de Recomendaciones: getLibrosRandom

37aab: sugeridos = getLibrosRandom(email: String): ListaBooks

38aab: execSQL("SELECT idLibro,COUNT(idLibro) FROM Prestar
GROUP BY idLibro ORDER BY COUNT(idLibro) DESC")

39aab: new ResultadoSQL()

40aab: next(): bool

[si hay libros leídos]

41aaba: getInt("idLibro")

42aaba: isOnLoan(email: String, idLibro: int): bool

```
[si no lo estamos leyendo]
    47aabaa: execSQL("SELECT * from Book WHERE id =
                        %idLibro%")
    48aabaa: new ResultadoSQL()
    49aabaa: next(): bool

[si T]
    50aabaaa: getInt("id")
    51aabaaa: getString("title")
    52aabaaa: getInt("author")
    53aabaaa: getString("cover")
    54aabaaa: getString("description")

    55aabaaa: new Book(id: int, title: String, author: int, cover: String,
                        description: String)
    56aabaa: close()

57aab: close()
```

f7.- isOnLoan()

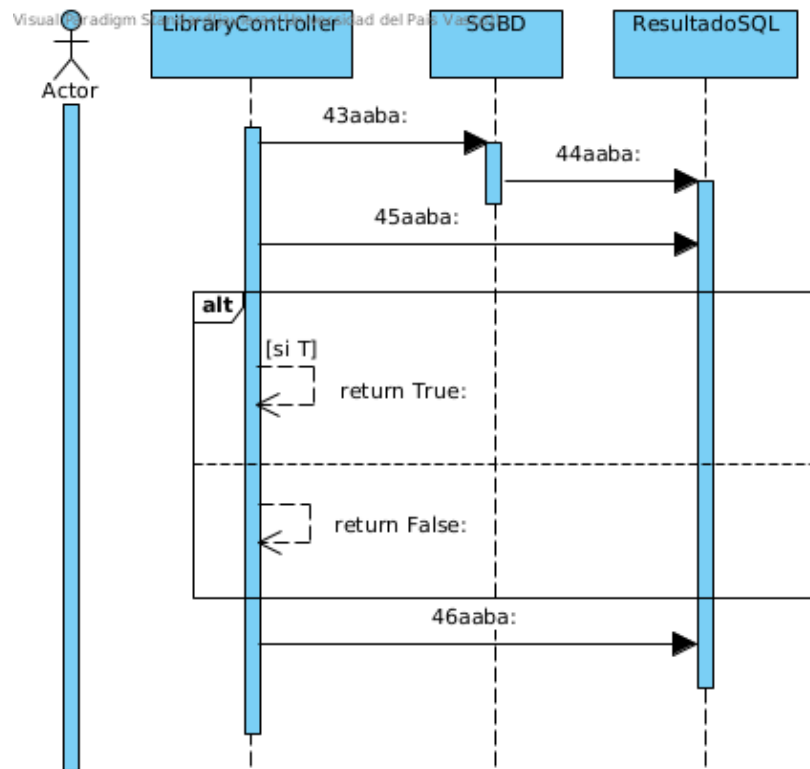


Figure 84: Diagrama de Secuencia de Recomendaciones: *isOnLoan*

43aaba: `execSQL("SELECT * FROM Prestar WHERE emailUser
LIKE %email% AND idLibro = %idLibro% AND fechaFin =")`

44aaba: `new ResultadoSQL()`

45aaba: `next(): bool`

[si T]

`return True`

[si no]

`return False`

46aaba: `close()`

11. PROBLEMAS DE IMPLEMENTACIÓN Y SOLUCIONES

12. CONCLUSIONES

13. CHANGELOG (CAMBIOS)

25/10/2023

El foro ya no funciona de la misma manera. Ahora en el foro solo se puede responder al tema, no responder a un comentario.

Actualizada la jerarquía de actores.

Actualizados los diagramas del modelo de casos de uso.

Actualizado el primer diagrama del punto 3.3 (Gestión de Reservas).

Actualizado el primer diagrama del punto 3.5 (Foro), y el diagrama Ver_Tema, diagrama Ver_Foro.

Introducido todo el modelo del dominio.

Se han cambiado las figuras 16 y 47, añadiendo a la interfaz la opción de reservar libro.

8/11/2023

Se han añadido las tablas respectivas a los planes de pruebas de los casos de uso.

15/11/2023

Incluida entidad AUTOR, añadida al diagrama de dominio y en el punto 5.2 su relación con LIBRO.

22/11/2023

Se ha modificado la relación de crear tema en el modelo de dominio.

Se ha añadido una interfaz al CUE de Foro.

Se ha modificado la interfaz del Administrador.

24/11/2023

Se han añadido los diagramas de comunicación y los diagramas de clases y BD.

21/12/2023

Se ha corregido un error en el paso 2.1 (faltaba el paso del parámetro) del diagrama de comunicación de Recomendaciones. También se ha eliminado el desarrollo del paso 2.1.3 por ser el mismo que el del 2.1.1.

22/12/2023

Se han corregido las prueba de recomendaciones y se han añadido los diagramas de secuencia de recomendaciones.

26/12/2023

Se ha corregido un pequeño error en el diagrama de comunicación de la red de Amigos.