



UNIMINUTO
Corporación Universitaria Minuto de Dios
Corporación universitaria minuto de Dios
Facultad de ingeniería

Asignatura:
Arquitectura de software

NRC: 8217

Docente
DIANA PATRICIA QUIROGA CAMACHO

Trabajo: “Documento final primer corte”

Presentado por:
Julian Alberto Peñuela Buitrago ID: 607452
Yeison Asdrubal Vela Santini ID: 597335
Andres Nicolas Tellez Calderón ID: 596017

Marzo 2021

Bogotá D.C

Contenido:

1. Introducción
- 2. Objetivo y Justificación**
3. Metodología de Desarrollo
4. Listado de Requerimientos
5. Diagrama de Arquitectura
6. Vista Lógica
7. Vista Proceso
8. Vista Despliegue
9. Vista Física
10. Requerimientos
11. Atributos de Calidad
12. Escenarios de Calidad
13. Estilo de arquitectura
14. Patrón de arquitectura

ITEM	DESCRIPCIÓN
Introducción	Breve descripción del proyecto
Objetivo y Justificación	Objetivo y Justificación del Proyecto
Metodología de Desarrollo	Descripción de la metodología seleccionada (Ágil o Tradicional)
Listado de Requerimientos	En caso de seleccionar tradicional se deben colocar los casos de Uso, en caso de seleccionar ágil Backlog con sus respectivas historias de usuario. DEFINIR EL MIVP y sobre éste realizar las historias de Usuario o Casos de Uso
Diagrama de Arquitectura	Seleccionó de arquitectura "SOA, CENTRADA A DATOS, MICROSERVICIOS, CLIENTE SERVIDOR" Se debe realizar el diagrama y la explicación del porqué
Vista Lógica	Diagramas de clases, secuencia y componentes
Vista Proceso	Diagrama de Actividades
Vista Despliegue	Diagrama de Despliegue
Vista Física	Diagrama Físico

1.Introducción

EL proyecto “Control de inventarios de un bar” generará un producto de software el cual se entregará en una versión 1.0 donde se digitalizarán las funciones principales que el bar “el bombillo rojo ” necesita para su control de inventarios, con ciertas características de la lógica del negocio implementadas de manera clara y sencilla facilitando el uso por parte del cliente. No es un documento exhaustivo, sino una primera versión, con requisitos iniciales del futuro sistema a implantar que deben ser refinados en sucesivas entrevistas apoyándose en el diseño de prototipos y modelos gráficos.

2. Objetivo y Justificación

El software a construir tiene como objetivo principal, apoyar en la gestión del inventario del bar “El bombillo rojo”. Se desea automatizar, fundamentalmente, la gestión de los productos que ingresan y salen de la bodega del bar teniendo en cuenta que se puede tanto como subir y bajar al sistema licores, los datos serán actualizados constantemente en la base de datos. , Nótese que el sistema solo se deberá utilizar para la gestión del control de inventarios, y no se incluirá algún manejo de dinero o comprar por parte del sistema.

Como alcance del producto se entregará un software funcional que supla las necesidades mencionadas anteriormente además de que este software beneficiará en gran manera el control de inventarios ya que brinda la herramienta que suple la necesidad del cliente que es tener un control sobre el inventario del bar “El bombillo rojo”.

3. Metodología de Desarrollo

Nuestro grupo de trabajo se regirá por la metodología tradicional imponiendo una disciplina de trabajo sobre el proceso de desarrollo del software, para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software.



4. Listado de Requerimientos

Requerimientos funcionales

1. Permitir el ingreso de licores al inventario del bar.
2. Se permite la actualización de cantidad de cada producto (Si entra o sale).
3. Se permite ver productos disponibles
4. Permitir la salida de licores al inventario del bar.
5. Formulario de entrada a cada usuario/rol.
6. Recibir la información subida por los usuarios a una base de datos (oracle).
7. Consultar licores
8. Borrar Licores
9. Modificar Licores

Requerimientos no Funcionales

1. Portabilidad.
2. Mantenibilidad.
3. fiabilidad.
4. Seguridad.
5. Usabilidad.
6. Compatibilidad.
7. Eficiencia de desempeño.
8. Adecuación funcional

Mínimo Viable:

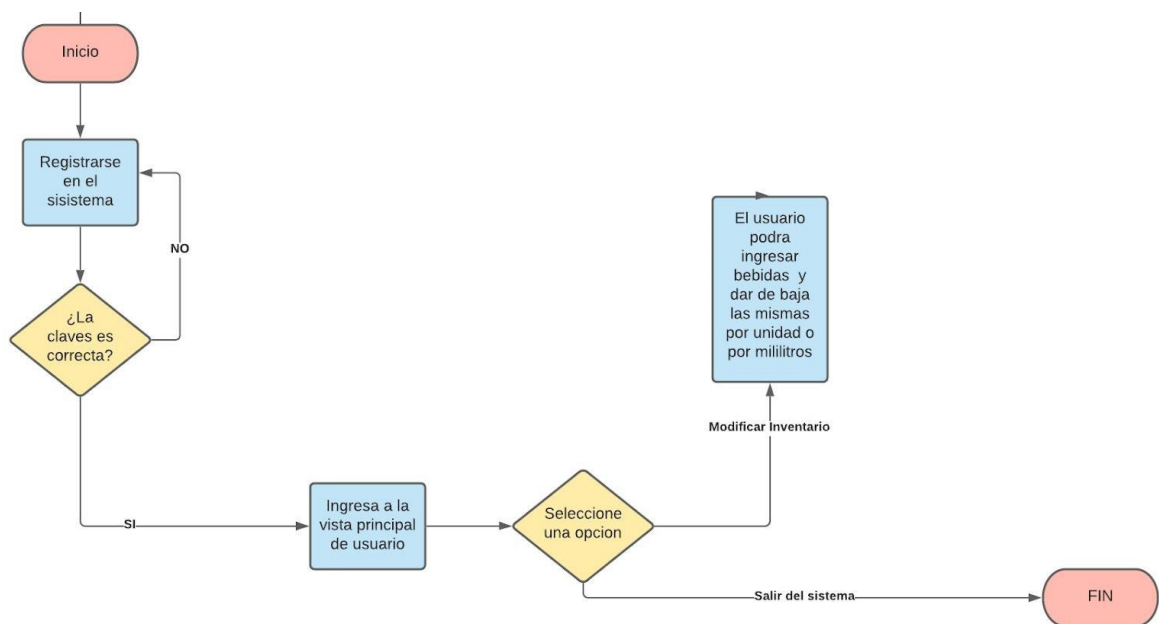
IDENTIFICADOR: R1		NOMBRE: Registro de licores	
Tipo: (necesario/deseable) Necesario		¿CRÍTICO? Si	
PRIORIDAD DE DESARROLLO: Alta		ROL AL QUE APLICA EL REQUERIMIENTO: Miembro y Administrador	
ENTRADA: <ul style="list-style-type: none">Nombre del licor.		SALIDA: Cantidad total de los productos ingresados.	
DESCRIPCIÓN: <p>Precondición: No debe estar vacío el espacio, cuando se registre la información.</p> <p>Descripción: Este requerimiento lo puede realizar el administrador tanto como el miembro, se debe introducir el nombre del producto para que quede registrado en la base de datos, si se cumplen las precondiciones se mostrará una alerta de que el producto fue creado correctamente.</p> <p>Postcondición: Se realizará la actualización de la base de datos con la cantidad de los productos ingresados.</p>			
MANEJO DE SITUACIONES ANORMALES			

Para saber que el producto se creo correctamente se le mostrará una alerta de color verde, que diga que el producto se creo correctamente y esta alojado en la base de datos.

CRITERIOS DE ACEPTACIÓN

1. Para que el dato se cree correctamente, en el nombre del producto solo se pueden ingresar caracteres sin números.
2. Para que el contenido del producto sea correcto y ingrese a el sistema debe ser solamente un dato numérico.

Flujo Del Requerimiento



IDENTIFICADOR:

R2

NOMBRE:

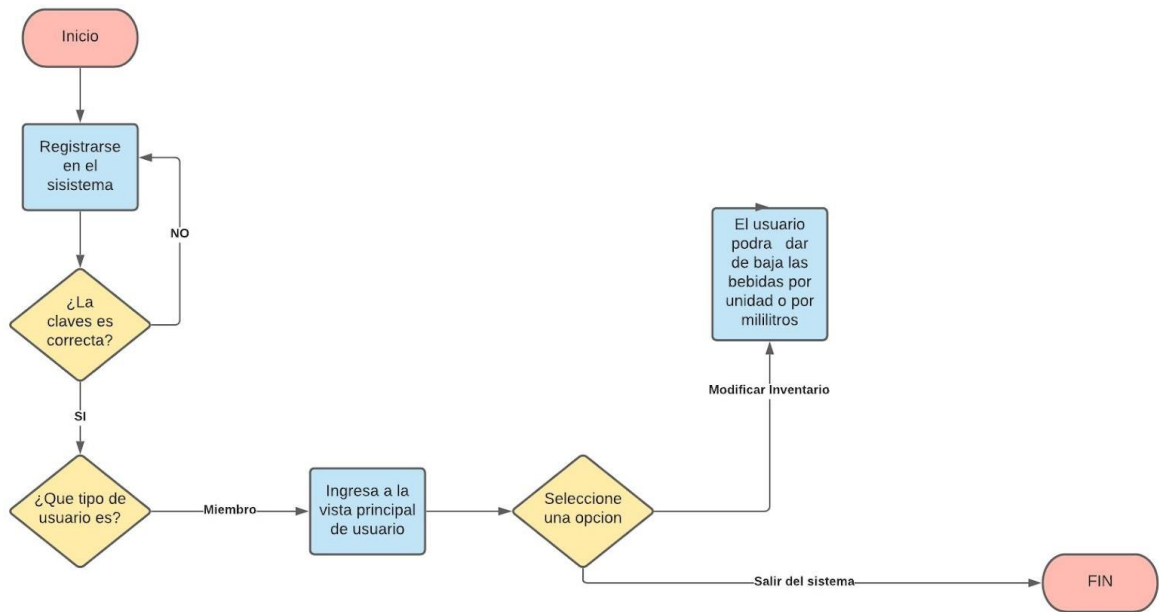
Salida de licores del inventario

Tipo:

(necesario/deseable)

Necesario		
PRIORIDAD DE DESARROLLO: Alta	ROL AL QUE APLICA EL REQUERIMIENTO: Miembro y Administrador	
ENTRADA: <ul style="list-style-type: none"> Nombre del licor. Cantidad de productos y litros o mililitros a salir. 	SALIDA: Actualización del inventario.	
DESCRIPCIÓN: <p>Precondición: No debe estar vacío el inventario del producto la cantidad de litros o mililitros tiene que ser de solo carácter numérico.</p> <p>Descripción: Este requerimiento lo puede realizar el administrador tanto como el miembro, se debe introducir el nombre del producto para que se pueda sacar de la base de datos, si se cumplen las precondiciones se actualizará la base de datos.</p> <p>Postcondición: Se realizará la actualización de la base de datos con la cantidad de los productos que se sacaron.</p>		
MANEJO DE SITUACIONES ANORMALES 1. Que el producto no tenga ninguna cantidad o no se encuentre disponible.		
CRITERIOS DE ACEPTACIÓN 1. Los datos ingresados al sistema en el momento de realizar la petición para sacar un producto sean correctos y establecidos para llevar a cabo su correcto despacho.		

Flujo Del Requerimiento



IDENTIFICADOR:

R3

NOMBRE:

Productos disponibles

Tipo:

(necesario/deseable)

Necesario

¿CRÍTICO?

Si

PRIORIDAD DE

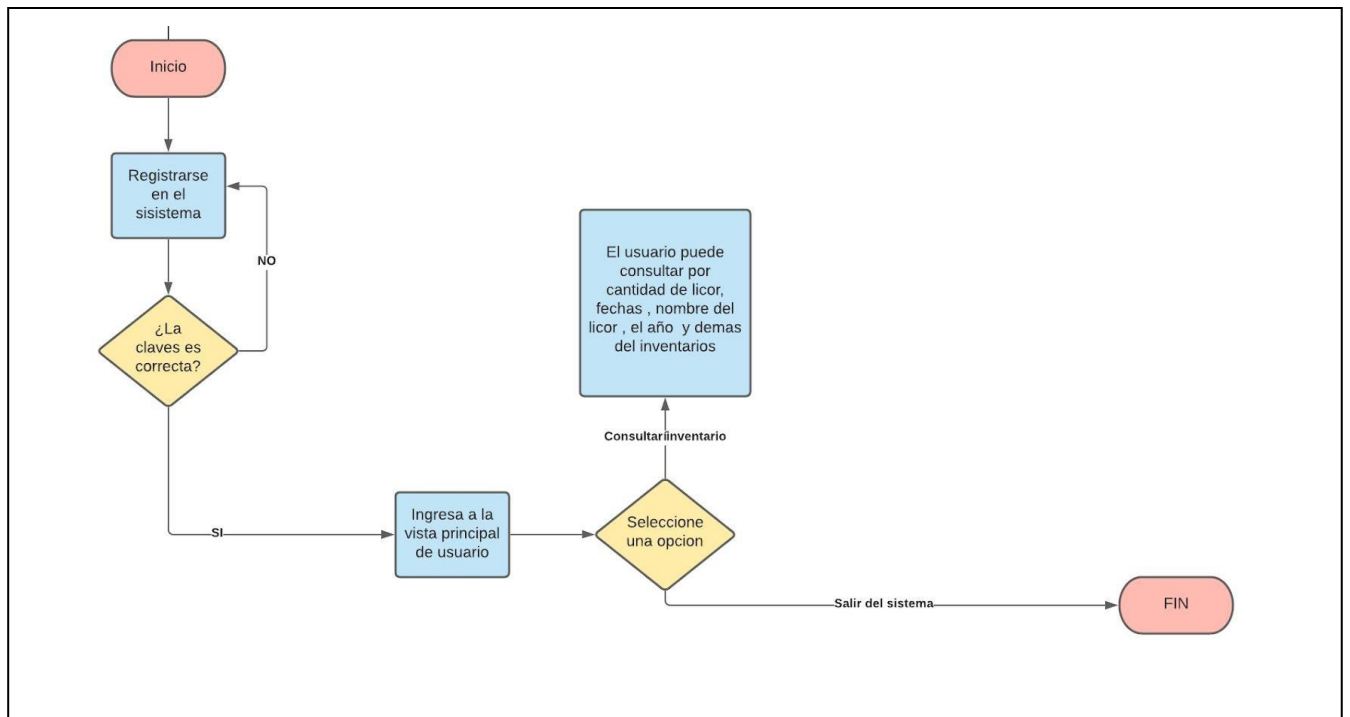
DESARROLLO:

Alta

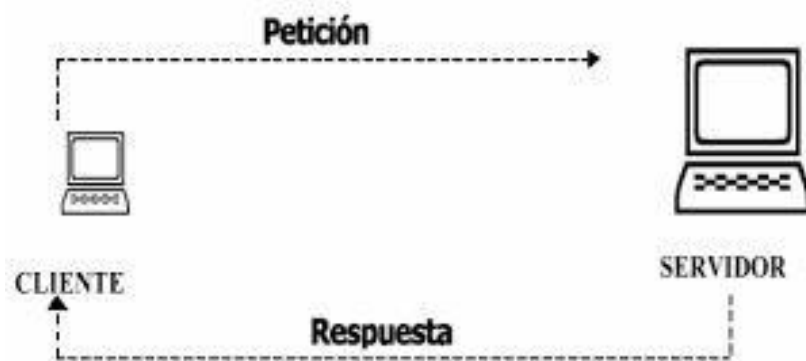
ROL AL QUE APLICA EL REQUERIMIENTO:

Miembro y Administrador

<p>ENTRADA:</p> <ul style="list-style-type: none"> ● Nombre del licor. ● Cantidad de productos y litros o mililitros. 	<p>SALIDA:</p> <p>Cantidad total de los productos que se encuentran en la base de datos con el total de litros o mililitros.</p>
<p>DESCRIPCIÓN:</p> <p>Precondición: La base de datos debe estar actualizada con el nombre del producto, la cantidad de productos y los litros o mililitros equivalentes.</p> <p>Descripción: Este requerimiento lo puede realizar el administrador tanto como el miembro, se debe introducir el nombre del producto para poder visualizar el estado actual del producto.</p> <p>Postcondición: Se mostrará el artículo seleccionado.</p>	
<p>MANEJO DE SITUACIONES ANORMALES</p> <p>1. Dado que el sistema cause anomalías deberá alertarse que hay un error con la BD y deberá contactarse con el Administrador</p>	
<p>CRITERIOS DE ACEPTACIÓN</p> <p>1. Los datos ingresados al sistema en el momento de realizar la petición para sacar un producto sean correctos y establecidos para llevar a cabo su correcto despacho.</p>	
<p>Flujo Del Requerimiento</p>	

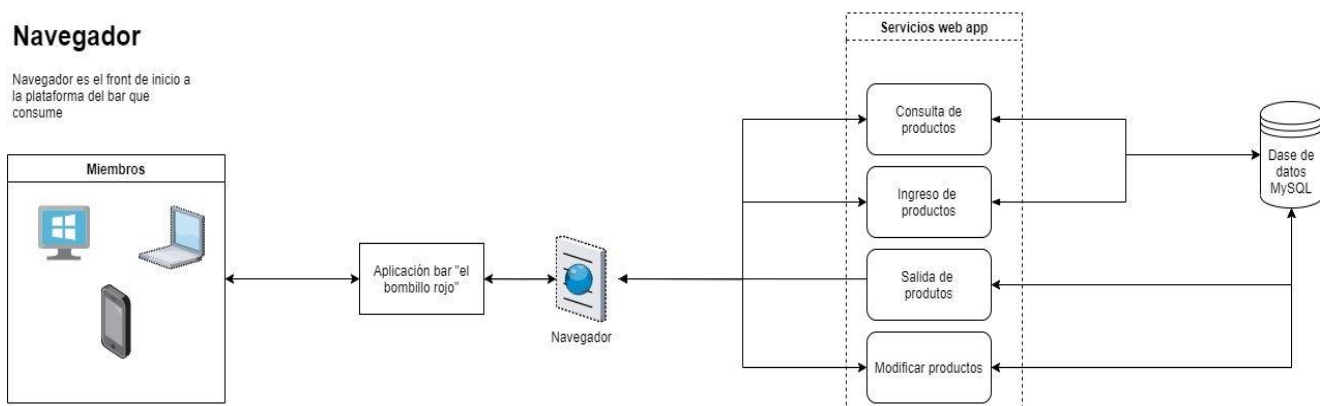


5. Diagrama de arquitectura



Navegador

Navegador es el front de inicio a la plataforma del bar que consume



La aplicación se maneja por medio de una arquitectura cliente servidor , siendo la aplicación del bar una aplicación web que se encuentra en el servidor apache que a su vez se relaciona con una base de datos MySQL

A continuación encontraremos los diagramas que corresponden al proyecto en el siguiente orden de manera consecutiva garantizando que sean legibles y de fácil comprensión.

Vista Lógica (3 diagramas) Vista

Proceso (1 diagramas)

Vista Despliegue(1 diagramas)

Vista Física (1 diagramas)

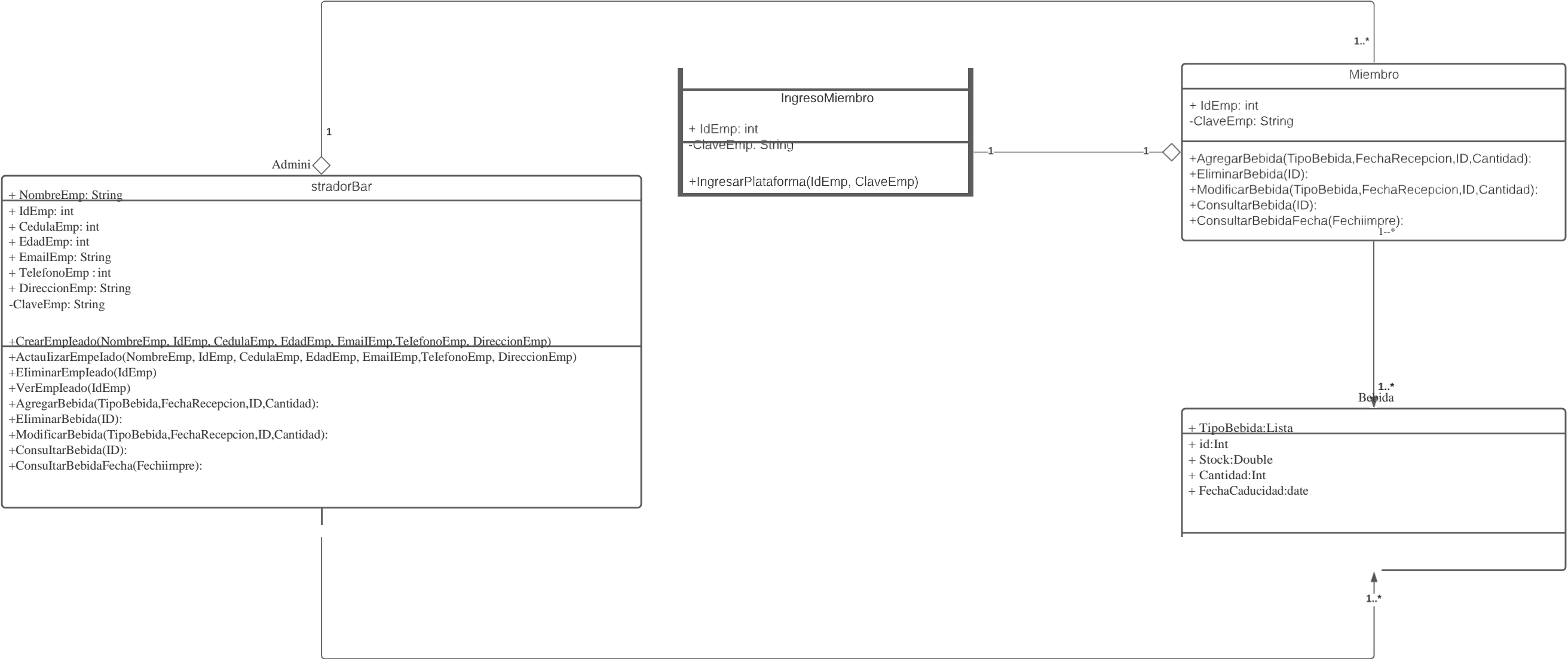
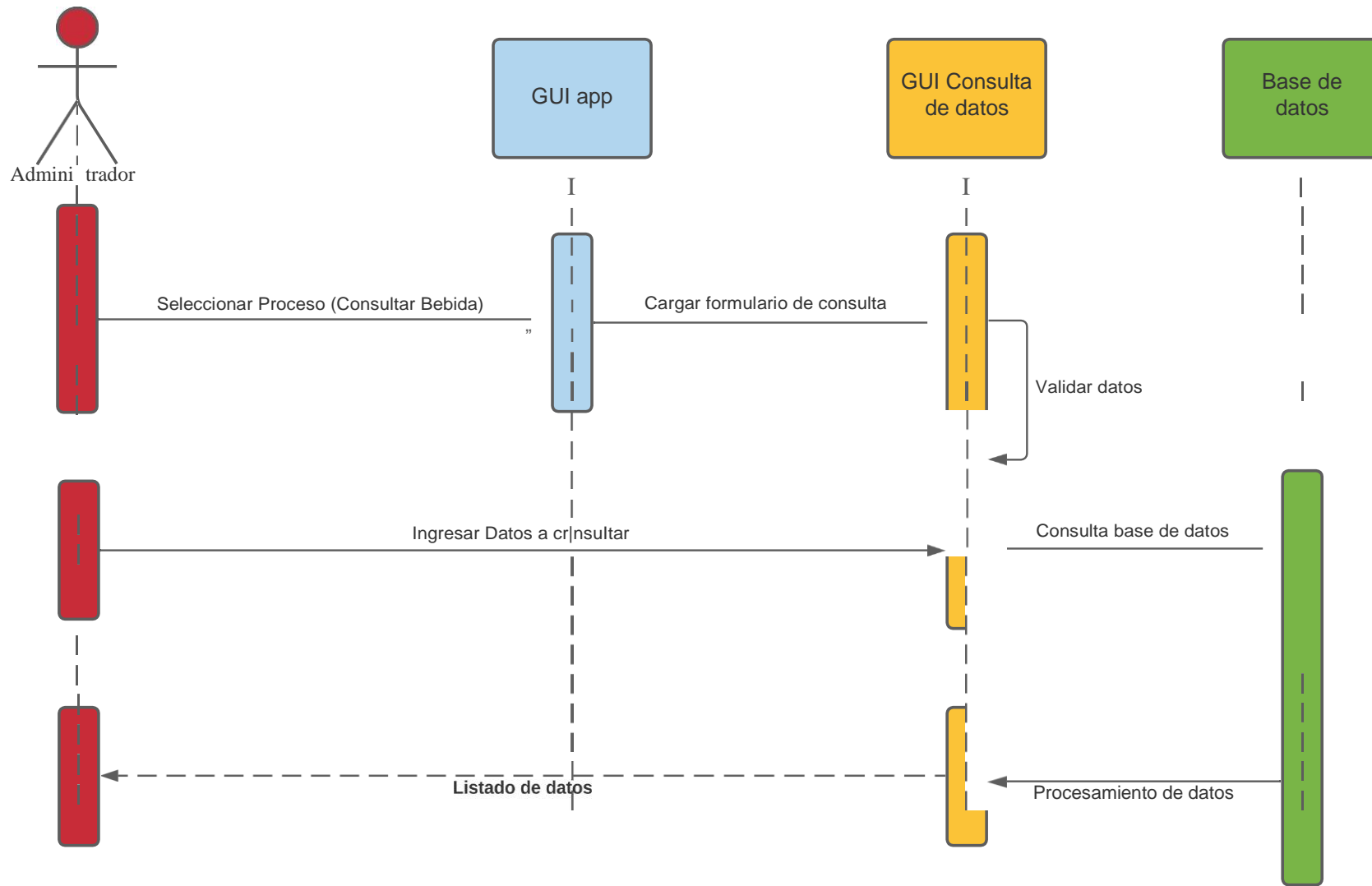
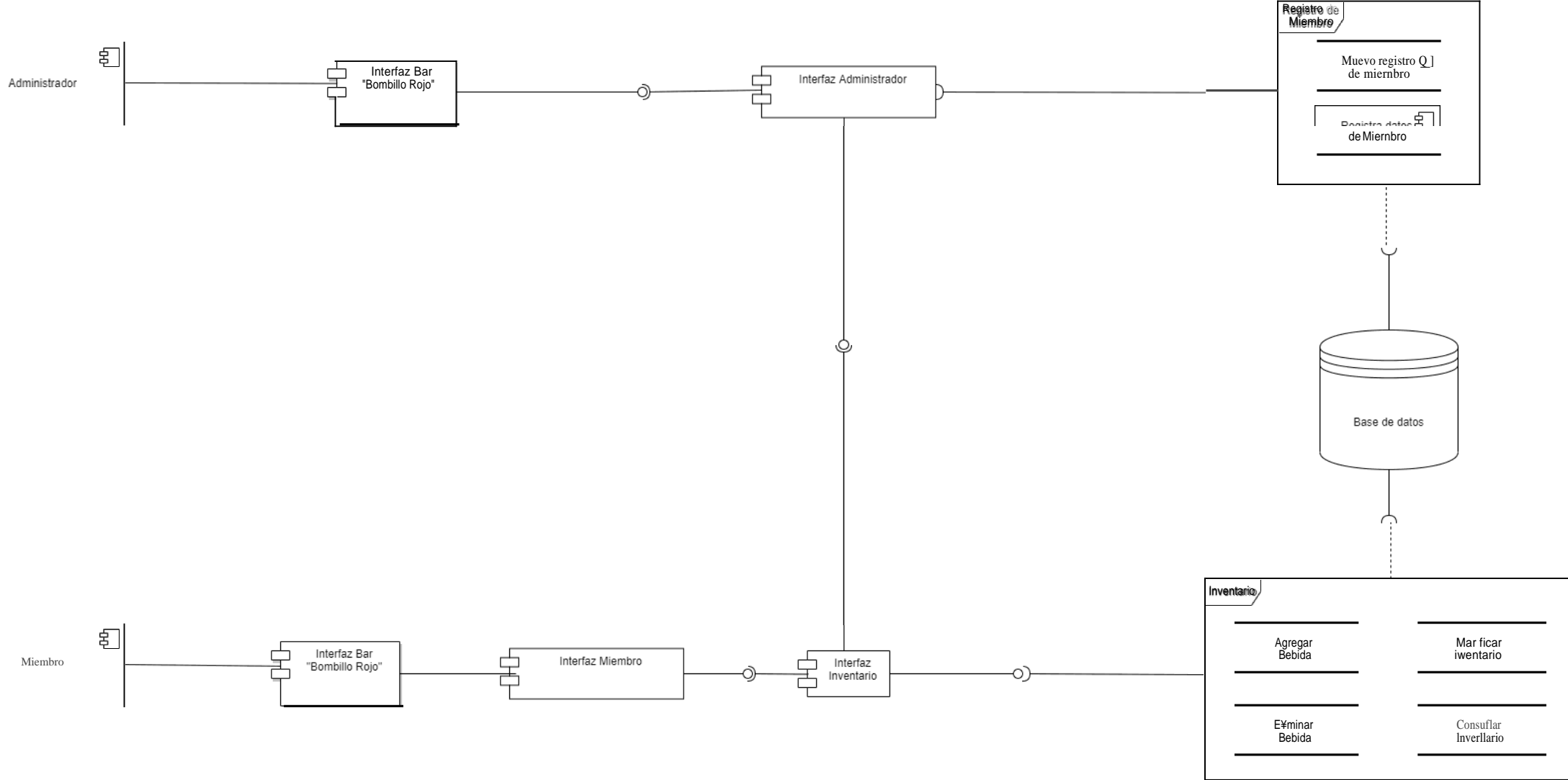
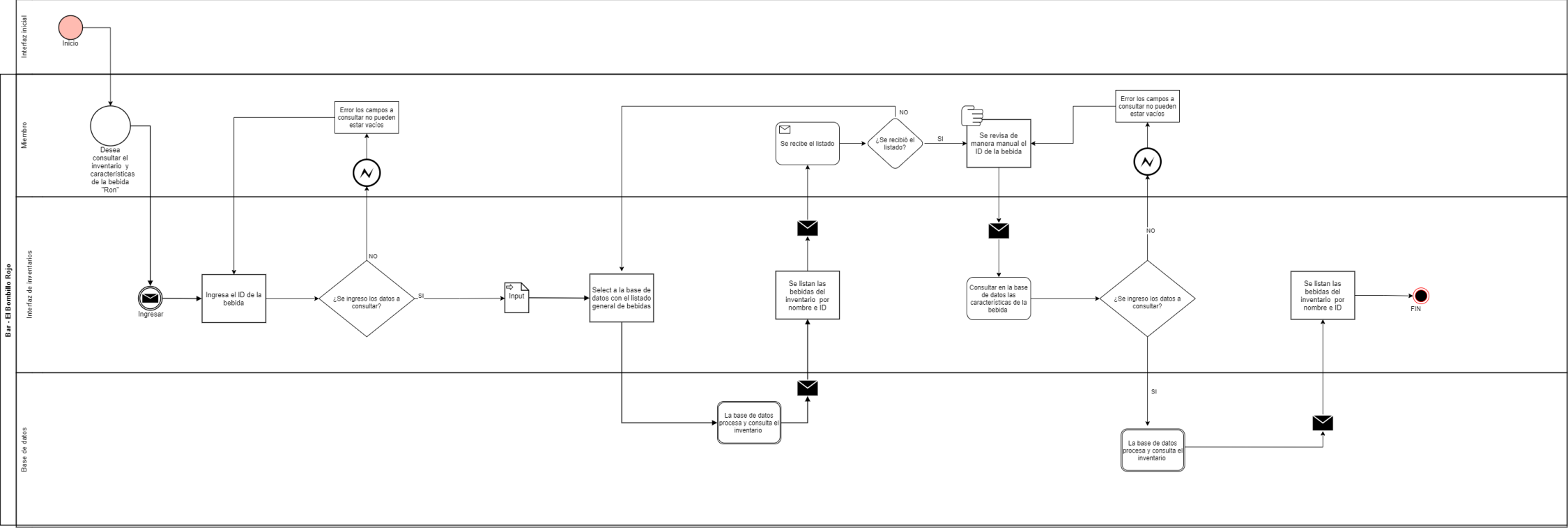
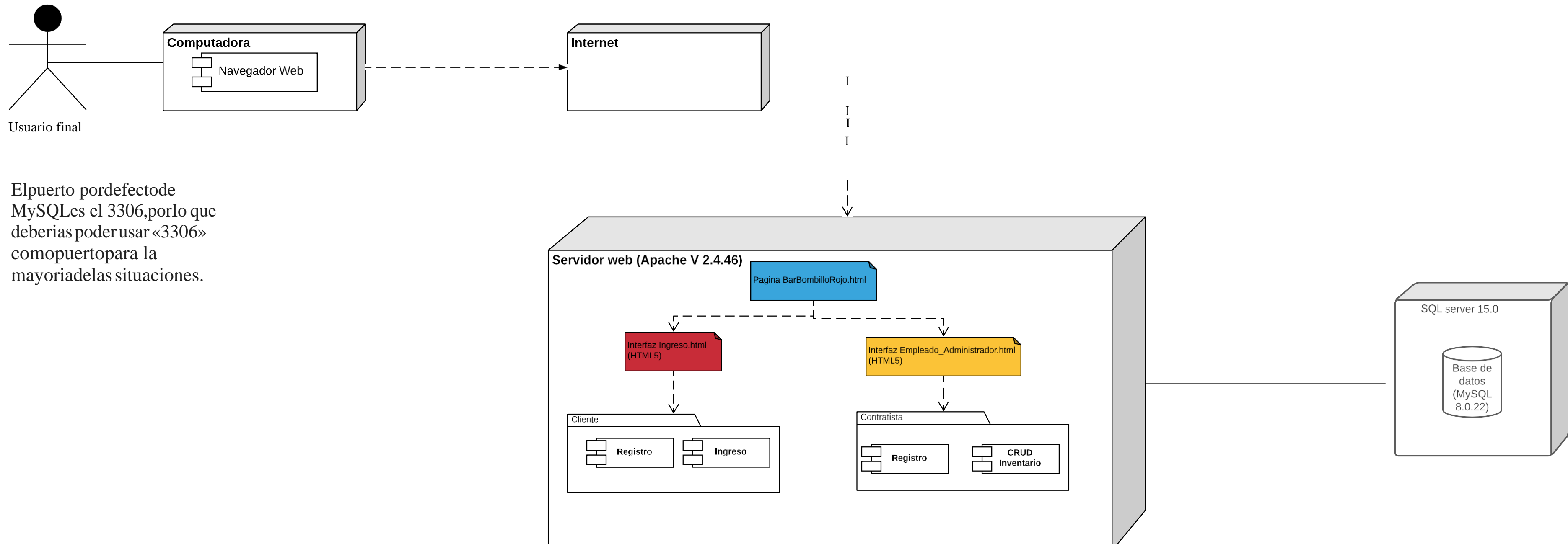


Diagrama de clases - Arquitectura de software
Juliana Peñuela ID: 607452
Yeison Vela ID: 597335
Andres Tellez ID: 596017







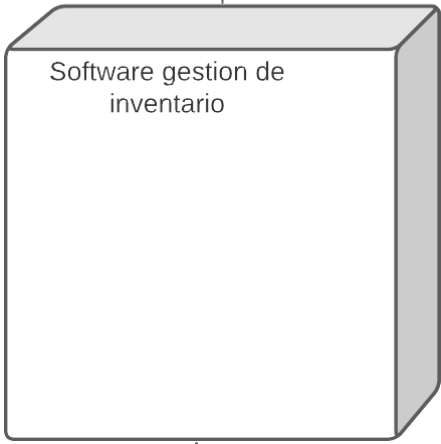
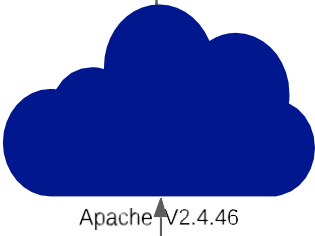




El puerto por defecto de MySQL es el 3306, por lo que deberías poder usar «3306» como puerto para la mayoría de las situaciones.



modules			
Service	Module	PID(s)	Port(s)
	Apache	11180 13328	80, 443
	MySQL	12804	3306



10. Escenarios de calidad

<i>Requerimientos</i>	<i>Tipo de requerimiento</i>	
	<i>Funcional</i>	<i>Tecnico</i>
Permitir el ingreso de licores al inventario del bar.	<i>X</i>	
Se permite la actualización de cantidad de cada producto(Si entra o sale).	<i>X</i>	
Se permite ver productos disponibles	<i>X</i>	
Permitir la salida de licores al inventario del bar.	<i>X</i>	
Formulario de entrada a cada usuario/ro	<i>X</i>	
Recibir la información subida por los usuarios a una base de datos (oracle)		<i>X</i>
Consultar licores	<i>X</i>	
Borrar Licores	<i>X</i>	
Modificar Licores	<i>X</i>	
Portabilidad.		<i>X</i>
Mantenibilidad.		<i>X</i>
Fiabilidad.		<i>X</i>
Seguridad.		<i>X</i>
Usabilidad.		<i>X</i>
Compatibilidad.		<i>X</i>

Eficiencia de desempeño		X
El admin o el miembro del bar debe realizar la consulta de una bebida en específico por medio del ID único de la misma.	X	
El tiempo mínimo de recuperación de la conexión de la aplicación con la base de datos ante una falla no debe superar 30 minutos.		X
Se garantiza que el software disponga de todas las funcionalidades que se establecieron en las historias de usuario teniendo en cuenta sus privilegios sin que se produzcan errores lógicos que puedan vulnerar o trastornar los procesos de la aplicación		X
Las búsquedas que se realicen a la base de datos no deben tardar más de 3 segundos		X
La aplicación debe funcionar con total normalidad sin importar que se realicen modificaciones del código fuente o de la base de datos		X

11. Atributos de calidad

De acuerdo al proyecto seleccionado por el grupo, listar dos atributos de calidad por:

1. Funcionalidad

- ***Idoneidad:***

La aplicación web se basa en el registro y manejo de inventario del bar el bombillo rojo, cada requerimiento se establece de manera concreta a lo que se

quiere llegar especificando sus atributos y restricciones que se pueden evidenciar al hacer uso de estas. por otro lado y teniendo en cuenta el listado de requerimientos y sus especificaciones validar uno a uno al final si se implementan dichos procesos en el producto final.

- Exactitud :

La aplicación web en su respectivo análisis de requerimientos al final de su desarrollo deberá establecer cada funcionalidad donde se pueda evidenciar el resultado y necesidades del cliente, además se deberá contemplar posibles casos de redundancia para evitar que los requerimientos se contradigan entre sí y así evitar ambigüedades en el funcionamiento del producto final.

2. Fiabilidad

- Tolerante a errores:

La aplicación web se realizará con un servidor que consumirá servicios de calidad y que se eviten ciertas falencias, para que esta tenga un uso confiable, además se tendrán bien estructurados los procesos establecidos y desarrollados para que sea tolerante a eventos de falla.

Por otro lado se deberán contemplar excepciones para cada uno de los procesos para poder realizar un buen manejo de las situaciones anormales del sistema, teniendo así una ruta alterna en caso de que se presenten fallos robusteciendo un diseño tolerante a fallos.

- Recuperación:

Se deberá contemplar que a la aplicación web se le realizarán backups para la restaurar la aplicación web o partes concretas que presentaron errores, a su vez deberá contemplar escenarios de calidad en dado caso que el sistema falle así poniendo una pausa en el sistema dándoles a conocer a los usuarios lo anterior.

3. Usabilidad:

- **Operatividad:**

Se deberá garantizar que el software disponga de todas las funcionalidades que se establecieron en las historias de usuario teniendo en cuenta sus privilegios sin que

se produzcan errores lógicos que puedan vulnerar o trastornar los procesos de la aplicación.

- ***Facilidad de Aprender:***

Nuestro software al contemplarse como una aplicación web deberá contar con una interfaz minimalista con colores agradables que reflejen la lógica de negocio, por otro lado deberá estar estructurado de tal manera que sea muy intuitivo para la persona siempre priorizando la fácil navegación del usuario en el sistema.

4. Eficiencia:

- ***Comportamiento en el tiempo:***

Se deberán optimizar los procesos de la aplicación teniendo en cuenta los modelos y patrones según el diseño de algoritmos garantizando que el tiempo de respuesta de cada operación sea lo más bajo posible y que aun así siga funcionando de manera óptima.

- ***Comportamiento de recursos:***

Se debe diseñar y configurar bien el sistema para poder optimizar las consultas y operaciones de la base de datos para no sobrecargar de procesos el servidor y la base de datos por otro lado se deberá configurar la base de datos de tal manera que soporte la gran cantidad de usuarios que podría tener el sistema.

5. Mantenimiento:

- ***Estabilidad:***

Se deberá garantizar que la aplicación siga funcionando con total normalidad sin importar que se realicen modificaciones del código fuente o de la base de datos, por ello se debe contemplar tener excepciones para cada una de las situaciones anormales que se presenten.

- ***Facilidad de pruebas:***

La aplicación web deberá estar diseñada para que al momento de realizar las pruebas correspondientes podamos detectar errores que en el momento de la construcción del software no se detectaron, se debe contemplar lógicamente como funciona cada operación para así suponer en qué casos se puede producir un error y a su vez mitigarlos de la mejor manera posible.

6. Portabilidad:

- ***Adaptabilidad:***

La aplicación web deberá tener una adaptabilidad para el correcto funcionamiento por ello se contempla un diseño responsive que garantice la correcta visualización y funcionamiento en diferentes dispositivos.

- ***Facilidad de instalación:***

Debido a la arquitectura y desarrollo futuro de nuestra aplicación como resultado se espera tener una aplicación web que permita la facilidad de instalación en cualquier equipo debido a que no necesitará recursos en específico, sin embargo sí es indispensable un equipo que cuente un navegador de red.

12.Escenarios de Calidad

<https://drive.google.com/file/d/1nwg-ITLN4sukDoTo1y89gr1EOE1eHhn3/view?usp=sharing> ***13.Estilo***

de arquitectura

Estilo de arquitectura

Se debe dar una breve explicación de qué estilos de arquitectura se acopla a su solución y porque son los más adecuados

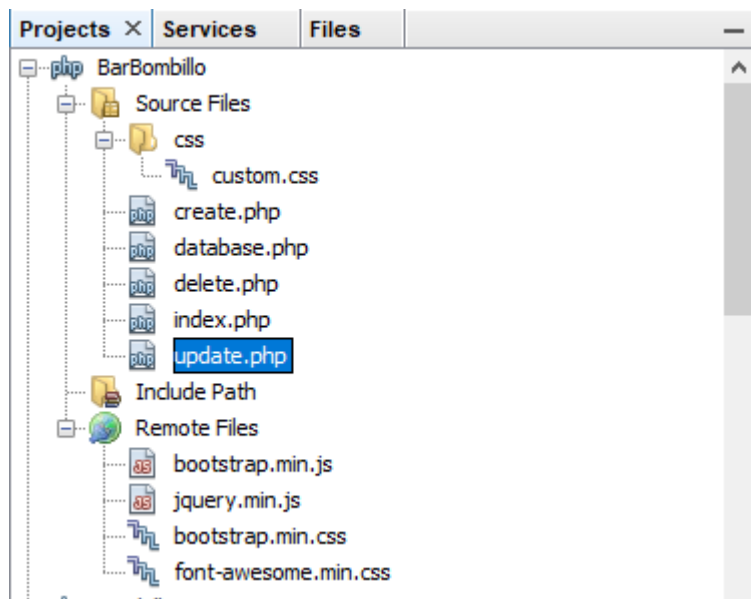
Analizando los distintos tipos de estilos que se acoplan a nuestra arquitectura cliente servidor se optó por usar el estilo de arquitectura llamada y retorno ya que hace énfasis a el paradigma de programación orientada a objetos, se facilitará la implementación de este estilo de arquitectura ya que se hacen uso de funciones y objetos que se pueden reutilizar mediante el desarrollo y ejecución del software planteado.

Teniendo en cuenta los componentes que conforman nuestra solución se determina que son elementos independientes entre sí los cuales cumplen la características principal de este estilo arquitectónico.

Además, este estilo debido a que también se tienen seleccionadas las capas nos permite particionar problemas de consulta complejos dentro del contexto del software. En conclusión nos favorece la reusabilidad y el tipo de estilo arquitectónico llamada y retorno por las características y soluciones que llega a dar este software.

Esqueleto de nuestro proyecto

Para el proyecto planteado se propuso la creación de un software que facilitara la gestion de un bar, en este caso el bar tiene como nombre “El bombillo rojo”, para la creación de este se implementó el paradigma de programación orientada a objetos con un patrón estilo MVC, en el cual se establecen varias clases como se puede observar en la siguiente imagen, estas permiten el correcto funcionamiento del software desarrollado.



Como podemos apreciar en la imagen anterior se encontrarán los elementos que pertenecen al MVC solo que debido a la madurez que se tiene hasta el momento podemos ver que se tienen definidos elementos propios de este patrón , además se tiene un componente de estilos el cual nos provee los estilos del framework bootstrap.

1. Carpeta css:

En esta carpeta se tendrán todos los componentes referentes a los estilos implementados en el software , se eligió bootstrap debido a que es el framework popular diseñado para crear sitios con capacidad de respuesta, primero en dispositivos móviles, con jsDelivr y una página de inicio de plantilla.

En esta carpeta tendremos la siguiente sentencia

<link

```
href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbZKgwra6" crossorigin="anonymous">
```

Al agregar esto tendríamos incluido los servicios de este framework y restaría aplicarlo a los elementos gráficos e interactivos que se trabajen a lo largo del desarrollo del mismo.

2. Clase create:

En esta clase se establece la captura de los datos ingresados por el usuario, y se almacenan en la base de datos establecida además se realizan validaciones.

3. Clase database:

En esta clase se establece las variables con las que podemos realizar la conexión con la base de datos en esta clase también encontraremos más funciones como una función read con la cual leemos los datos que tengamos en ese momento, se tiene una función create con la cual insertamos los valores que el usuario escribió, tenemos una función llamada update con la cual realizamos una consulta por medio del ID y con ello localizar el dato que buscamos para actualizar su información y por último tenemos la función de delete la cual se encarga de eliminar un registro por medio de su ID correspondiente

4. Clase delete:

En esta clase se implementa el método eliminar que básicamente lo que realiza es eliminar datos establecidos por el usuario, y se eliminan de la base de datos de esta manera se realiza el correcto funcionamiento de la clase.

5. Clase index:

En esta clase se establecen todas las etiquetas que conforman la interfaz de usuario, además de los espacios de captura de datos, botones de acción, etc. en esta clase se realiza el respectivo llamado de los datos que se encuentran en la base de datos.

6. Carpeta Remote files

Debido a que muchos de nuestros componentes de bootstrap requieren el uso de JavaScript para funcionar. Específicamente, requieren complementos de JavaScript y Popper . Por ello en nuestro software se tendrá un componente donde se colocan cada uno de los siguientes `<script>`s cerca del final de las páginas, teniendo en cuenta que deben ser justo antes de la `</body>` etiqueta de cierre , para habilitarlos.

Por último en esta carpeta de archivos remotos se deberán incluir todos los complementos y dependencias de Bootstrap JavaScript con uno de los dos paquetes disponibles . Ambos `bootstrap.bundle.js` `bootstrap.bundle.min.js` incluyen Popper para importar la información sobre herramientas y ventanas emergentes.