



C.P.R. Liceo “La Paz”

Proyecto Fin de Ciclo

Desarrollo de Aplicaciones Multiplataforma Restaurante Bloski's.

Autor: Pablo Rodríguez Rodríguez
Tutor: Jesús Ángel Pérez-Roca Fernández

Resumen

Restaurante Bloski's es un proyecto de desarrollo de una aplicación dedicada al sector hostelero. Su objetivo es brindar ayuda para la gestión de ciertas partes de un restaurante como pueden ser la gestión del personal durante los turnos de servicio, facilitar la formación de nuevos miembros de la plantilla, realización de facturas durante el servicio.

Abstract

Restaurante Bloski's is a development project for an application dedicated to the hospitality sector. Its goal is to provide assistance in managing specific aspects of a restaurant, such as overseeing staff during service shifts, facilitating the training of new team members, and generating invoices during service.

Palabras Clave

- JavaFX: Conjunto de bibliotecas y herramientas para el desarrollo de la interfaz de usuario en java.
- Hibernate: Tecnología de mapeado de clases para trabajar, entre otras cosas con base de datos de manera eficiente y rápida.
- Base de datos (MySQL): Se emplea una base de datos para diferentes partes de la aplicación como son la recogida de información, la inserción o la modificación.
- JavaMail: Dependencia usada para el envío de correos electrónicos.
- ITextPdf :Dependencia usada para la generación de PDFs.
- Java: Lenguaje de programación empleado.
- CSS: Lenguaje de programación utilizado para el diseño de estilos de los elementos fxml.
- SMTP: Simple Mail Transfer Protocol , es el protocolo utilizado para el envío de correos electrónicos.

A mis profesores, que tuvieron la paciencia de enseñarme las tecnologías necesarias para la implementación de este trabajo.

A Teresa que me enseñó sobre el sector de la hostelería.

Sumario

Resumen	3
Abstract.....	4
Palabras Clave	5
Introducción/motivación.	11
Objetivos.....	12
Estado del arte.	13
Caso de estudio.	14
Diagramas.	15
Desarrollo del proyecto	16
Manual Administrador	17
Manual Usuario	18
Viabilidad tecno-económica.	19
Trabajo futuro.	20
Conclusiones.....	21
Biblioteca de recursos web y referencias.....	22
Anexos.....	23

Motivación/Objetivos.

La idea detrás del desarrollo de la aplicación de Restaurante Bloski's , fue para brindar dentro del sector de la hostelería, (más concretamente en el ámbito de la restauración) una solución óptima y sencilla de utilizar a la hora de gestionar diversos apartados de la organización de tareas y de personal.

Al haber trabajado en el sector, identifiqué que se modularizaba el software para distintas cosas como las siguientes: el horario, se utilizaba un Excel para gestionarlo; para aprender en el apartado de cocina directamente se hacía mediante la enseñanza clásica de boca en boca y para marcar los pedidos, que posteriormente se servían e imprimían durante el servicio, se usaba una aplicación a mayores.

En cuanto al enfoque del desarrollo, planteo todos estos requisitos y busco darles una solución unificada en mi aplicación para mejorar la productividad a la plantilla y la comodidad de usar una interfaz de usuario agradable para sus tareas.

Estado del arte.

◇ **Excel.**

- ❖ Puntos fuertes: Aplicación multiusos que te sirve para guardar información útil para la gestión de personal y llevar registros de pedidos y demás funcionalidades deseadas en el sector de la hostelería.
- ❖ Puntos débiles: Esta aplicación por si sola, no se ajusta al 100% de las funcionalidades que se prevén en un sistema de hostelería. [OBJ]

◇ **Aplicación Foster Hollywood:**

- ❖ Puntos fuertes: A la hora de ver los productos disponibles, era muy intuitiva y su interfaz gráfica, era más amigable, además implementaba un sistema de roles para tener control de los usuarios. [OBJ]
- ❖ Puntos débiles: Estaba limitada a una única funcionalidad, que era la de pasar productos a cocina, y llevar registro de los productos en mesa. No contemplaba la gestión de horario y otras cosas.

En cuanto a las tecnologías usadas para el desarrollo, principalmente son dos: JavaFX la cual es una librería que permite realizar interfaces de usuario más fácilmente, y el scene Builder , que es una herramienta que permite distribuir los elementos de JavaFX de manera visual.

Caso de estudio.

En Restaurante Bloski's se ha buscado solventar distintos apartados a la hora de gestionar un local de restauración sencillo o una franquicia.

En este punto desglosaremos las partes más importantes, así como las funcionalidades principales.

Registro de Usuarios:

En nuestra aplicación, se contempla gestionar más de un restaurante o establecimiento, por eso, se guarda un registro de los usuarios, almacenando el restaurante en el que trabajan, considerando que no pueden darse de alta en dos restaurantes a la vez.

Horarios:

Para el cómodo acceso de los usuarios a su portal de horarios, se ha implementado una sección que puede crear horarios y actualizar los de los trabajadores, consultarlos visual y fácilmente.

Pedidos:

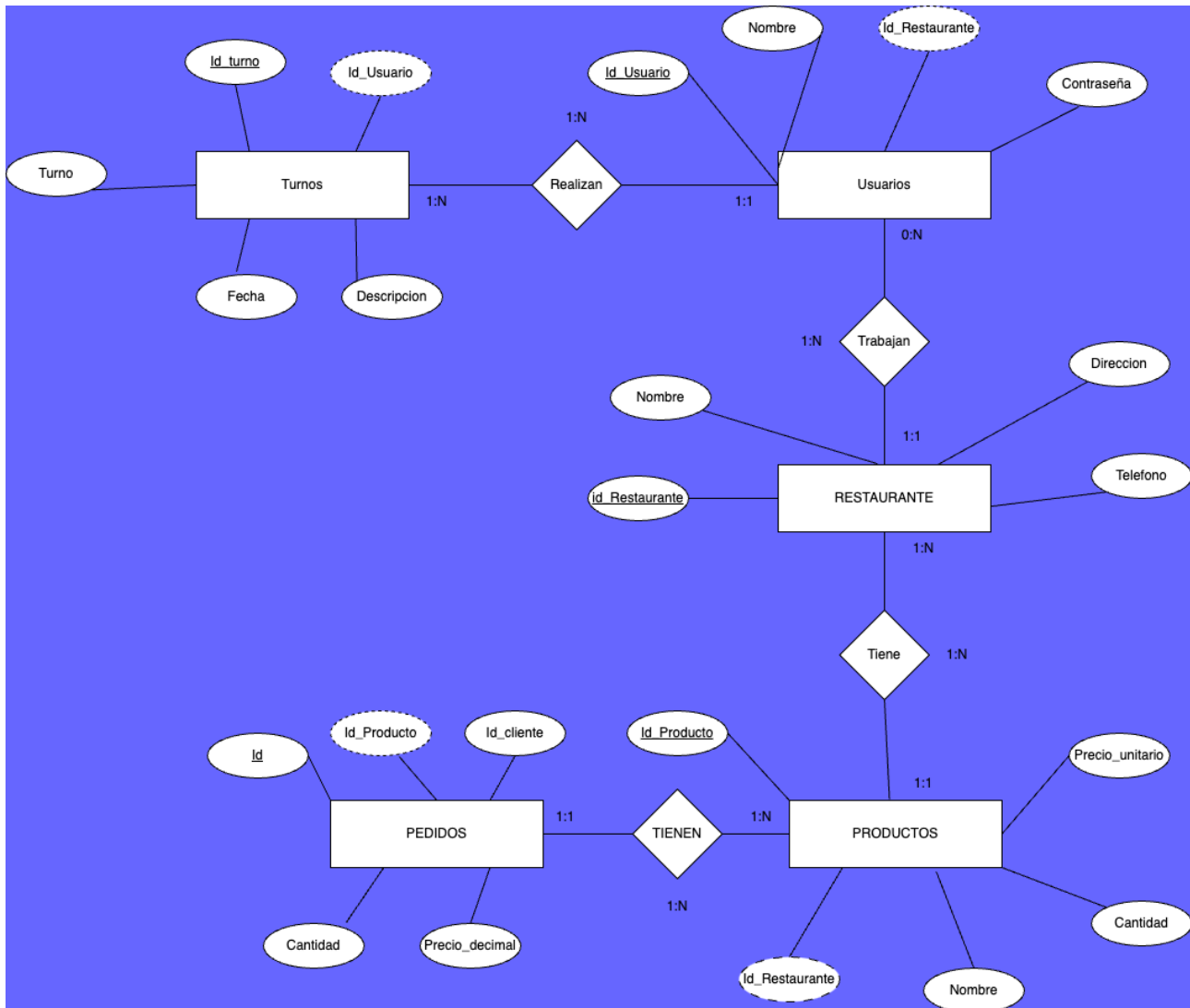
En esta sección, los usuarios podrán anotar de manera eficiente, los pedidos de los clientes, y posteriormente, imprimir la cuenta para servir en la mesa.

Recetas:

Por último, en esta sección, el usuario puede acceder a “formación” directa, viendo de manera visual, las distintas recetas y productos a la venta del local en el que está trabajando y así aprendiendo de manera rápida y eficaz la elaboración de los productos que se servirán.

Diagramas.

DIAGRAMA ENTIDAD - RELACIÓN



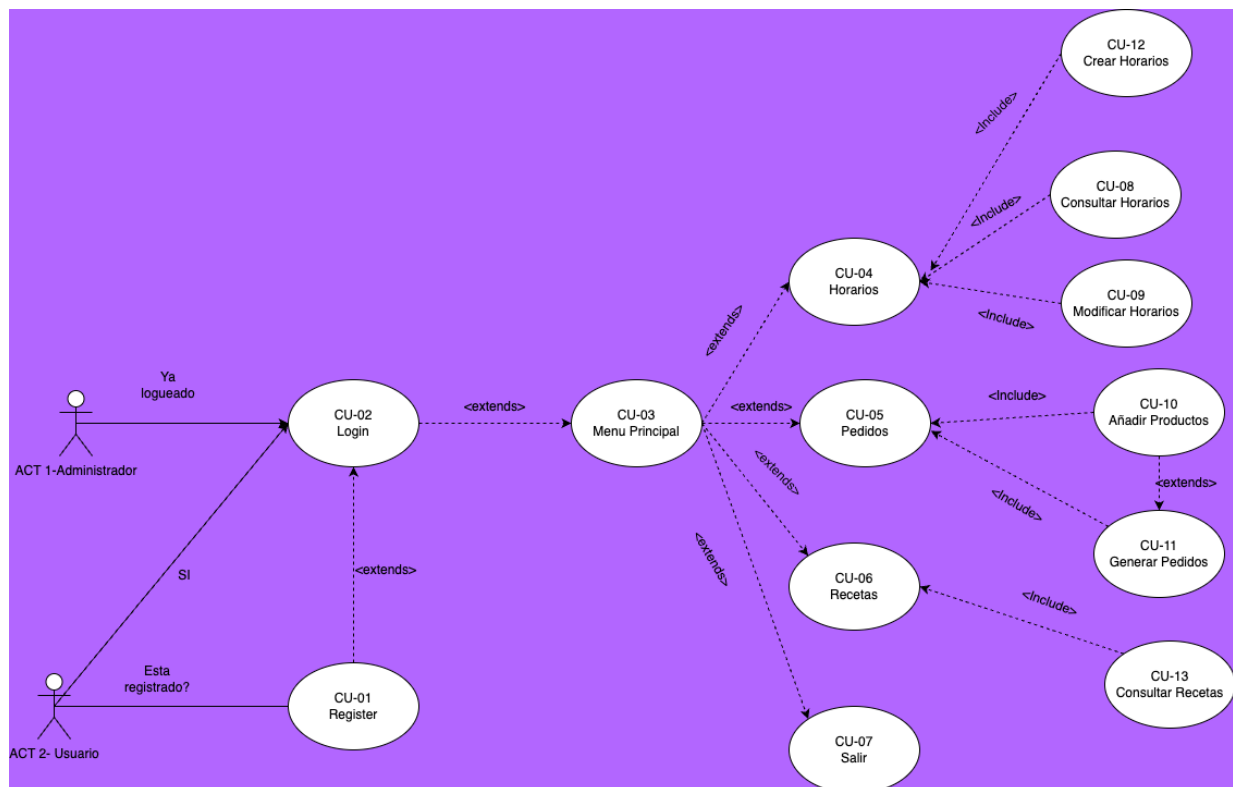
Explicación:

Cardinalidades: En las entidades Pedidos y Productos, se encontraron fallos durante el desarrollo así que se ajustaron las cardinalidades para reflejar el diseño bien.

Respecto a la relación Restaurante-Recetas, un Restaurante puede contar con varios productos para su venta, pero un producto solo está en un restaurante.

Se ha interpretado que un Usuario pueda trabajar en varios restaurantes de la franquicia, y, por último, un empleado puede realizar diferentes turnos.

DIAGRAMA DE CASOS DE USO.



Explicación:

La secuencia normal de uso de la aplicación es que un usuario se registre antes de usarla, excepto el administrador, el cual, ya estará registrado.

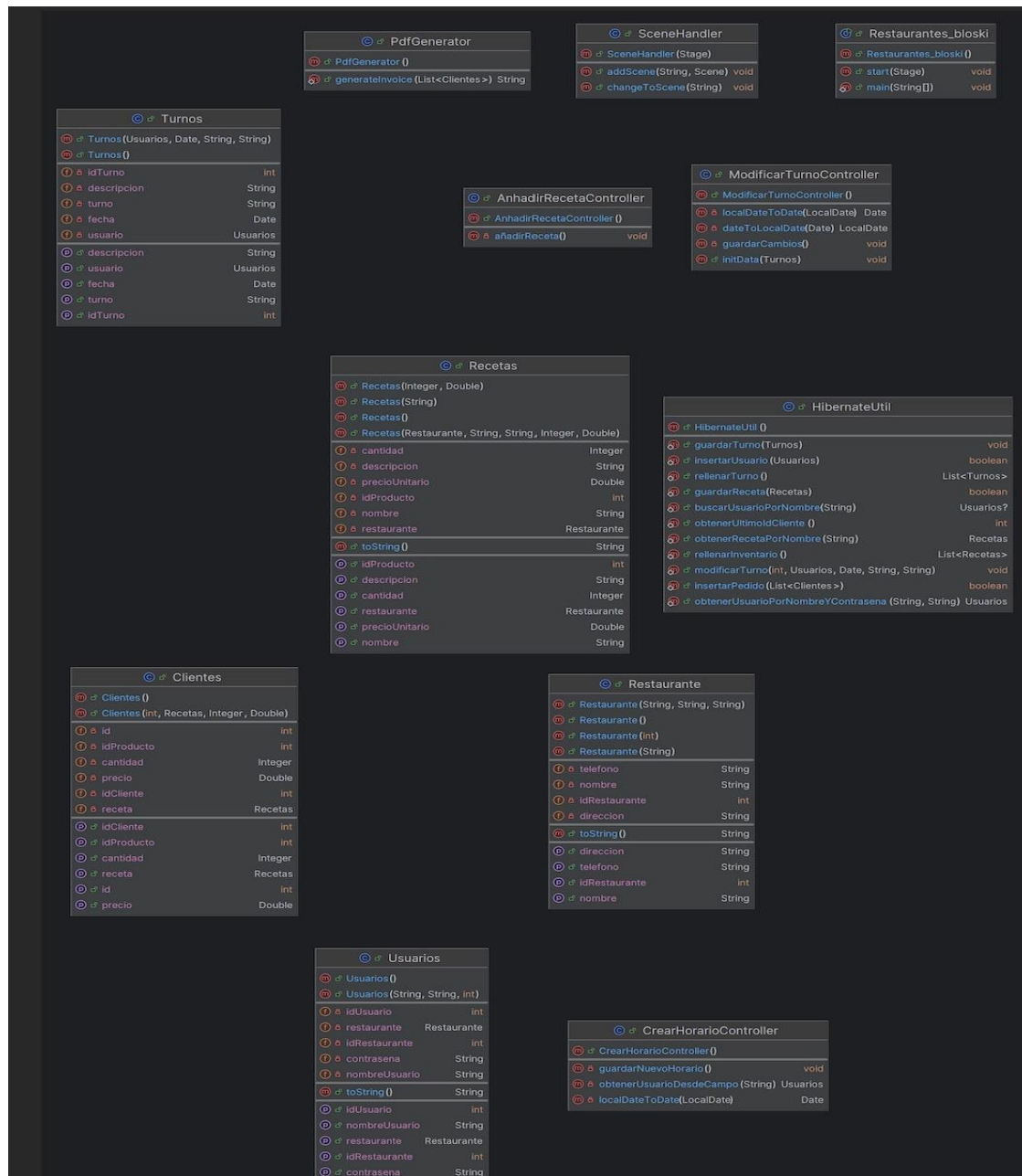
En las relaciones de extensión indica la secuencia natural de los casos de uso, mientras que las de inclusión simbolizan que los casos de uso están incluidos en otros casos de uso.

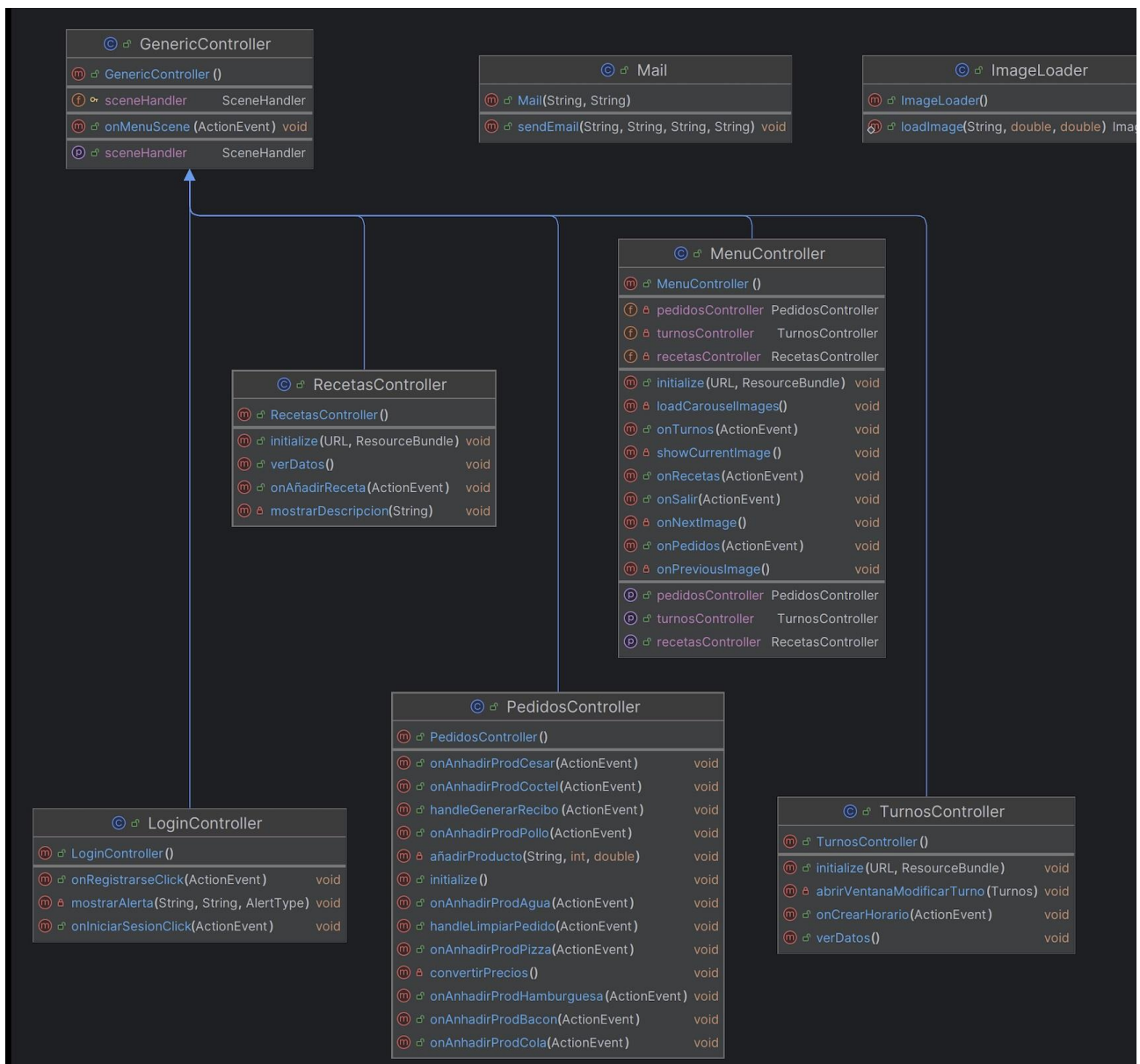
Casos de Uso:

- ❖ CU-01 Register: El usuario se registra.
- ❖ CU-02 Login: El usuario se logea.
- ❖ CU-03 Menú Principal: El usuario accede al menú principal.
- ❖ CU-04 Horarios: El usuario accede a la sección de horarios.
- ❖ CU-05 Pedidos: El usuario accede a la sección de pedidos.
- ❖ CU-06 Recetas: El usuario accede a la sección de Recetas.
- ❖ CU-07 Salir: El usuario sale de la aplicación.
- ❖ CU-08 Consultar Horarios: El usuario consulta sus horarios.
- ❖ CU-09 Modificar Horarios: El usuario modifica sus horarios.
- ❖ CU-10 Añadir Productos: El usuario añade productos al pedido.
- ❖ CU-11 Generar Pedido: El usuario envía el pedido por e-mail. [OBJ]
- ❖ CU-12 Crear Horarios: El usuario crea un horario.

❖ CU-13 Consultar Recetas: El usuario consulta una Receta.

UML:





EXPLICACIÓN:

En el UML solo observamos relaciones de herencia, en los controllers ,ya que todos extienden del generic Controller.

La foto fué dividida en dos partes por que había demasiadas clases.

Desarrollo del proyecto

El desarrollo de este proyecto se divide en las siguientes etapas:

- **Análisis de requisitos:** Previamente a programar o tocar el código, se establecieron una serie de necesidades que podría tener el usuario que emplee la aplicación, se estudió además el impacto revolucionario de la idea y si implementarla mejoraría las aplicaciones del mercado o no aportaría innovación.
- **Diseño de la base de datos:** Una vez establecidos los objetivos, se sabía que la base de datos sería necesario, por lo que se empezó a modelar un boceto no definitivo para visualizar el funcionamiento de la aplicación cuando se implemente la lógica.
- **Diseño de la Interfaz gráfica:** Como en el apartado anterior, durante esta etapa, se diseñaron distintos bocetos de para las diferentes secciones de la aplicación donde más tarde se implementaría la lógica.
- **Implementación:** Durante esta fase, se procedió a ensamblar todo mediante la lógica de la programación creando clases y métodos para unificar todo, así como usando dependencias para algunos apartados.
- **Pruebas y mejoras:** Por último, se realizaron pruebas, para solucionar los errores existentes, así como explorar nuevos apartados de mejora. Esta última fase, fue de las más importantes, ya que, se hicieron grandes cambios tanto en el diseño de la base de datos como en la programación, así como en el desarrollo de la interfaz gráfica.

Manual Administrador

Requisitos para el funcionamiento:

Para el correcto funcionamiento de la aplicación, es indispensable contar con el siguiente software:

- ❖ Entorno de desarrollo (IDE): IntelliJ Idea.
- ❖ Java JDK: openJDK-20 de oracle.
- ❖ Servidor de Base de Datos local: (MAMP, XAMP, WAMP)
- ❖ Gestor de dependencias: Maven versión 3.8.5
- ❖ Java: versión de Java: 21.0.2

Ensamblamiento del programa:

Lo primero será clonar el proyecto usando la siguiente URL:

- ❖ SSH: <git@github.com:pablorodri001/TFCPRR.git>
- ❖ HTTPS: <https://github.com/pablorodri001/TFCPRR.git>

Después, iniciar el servidor local de base de datos, procederemos a abrir en la consola, el phpMyAdmin, o el workbench de MySQL. Ejecutaremos el script .SQL que se encuentra en la carpeta principal, llamado Restaurantes_Bloski's.sql .

Una vez hecho esto, abriremos el proyecto llamado TFCPRR en el IDE IntelliJ Idea.

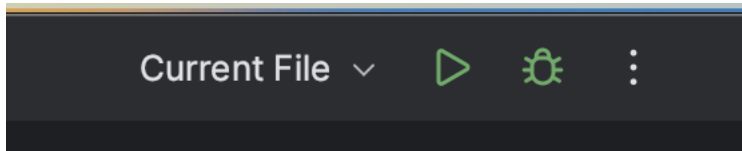
En el fichero de configuración, deberemos modificar el puerto localhost:8889, en caso de que nuestro servidor local, use un puerto diferente, y modificaremos el campo de password con nuestras credenciales de inicio de sesión en el servidor MySQL.

```
<hibernate-configuration>
  <session-factory>
    <property name="connection.driver_class">com.mysql.cj.jdbc.Driver</property>
    <property name="connection.password">root</property>
    <property name="connection.url">jdbc:mysql://localhost:8889/Restaurante_bloski?serverTimezone=UTC</property>
    <property name="connection.username">root</property>
    <property name="dialect">org.hibernate.dialect.MySQL5InnoDBDialect</property>
    <property name="show_sql">true</property>
    <property name="format_sql">true</property>
    <property name="current_session_context_class">thread</property>
    <property name="hbm2ddl.auto">update</property>

    <mapping class="Entidades.Usuarios"/>
    <mapping class="Entidades.Recetas"/>
    <mapping class="Entidades.Restaurante"/>
    <mapping class="Entidades.Turnos"/>
```

Después nos dirigiremos a la clase principal, dentro del paquete de `src/main/java/com.example.MainPackage`, la cual se llama `Restaurantes_bloski`.

Encontraremos el siguiente botón:



Lo pulsaremos, y ya se nos desplegará la interfaz principal.

Para la funcionalidad del apartado de `GenerarPedido` dentro de la sección de pedidos, cambiar el correo de destino, por el de interés, en la clase `Pedidos controller` en la línea 223:

```
String toEmail = "pablorodriodri2001@gmail.com";  
String subject = "Factura de pedido";  
String body = "Adjuntamos la factura de su pedido.";
```

PRECAUCIONES:

Es imperativo asegurarse de tener el puerto del fichero `hibernate.cfg.xml` bien configurado con los apartados anteriores, ya que, de no ser así, la aplicación no se lanzará.

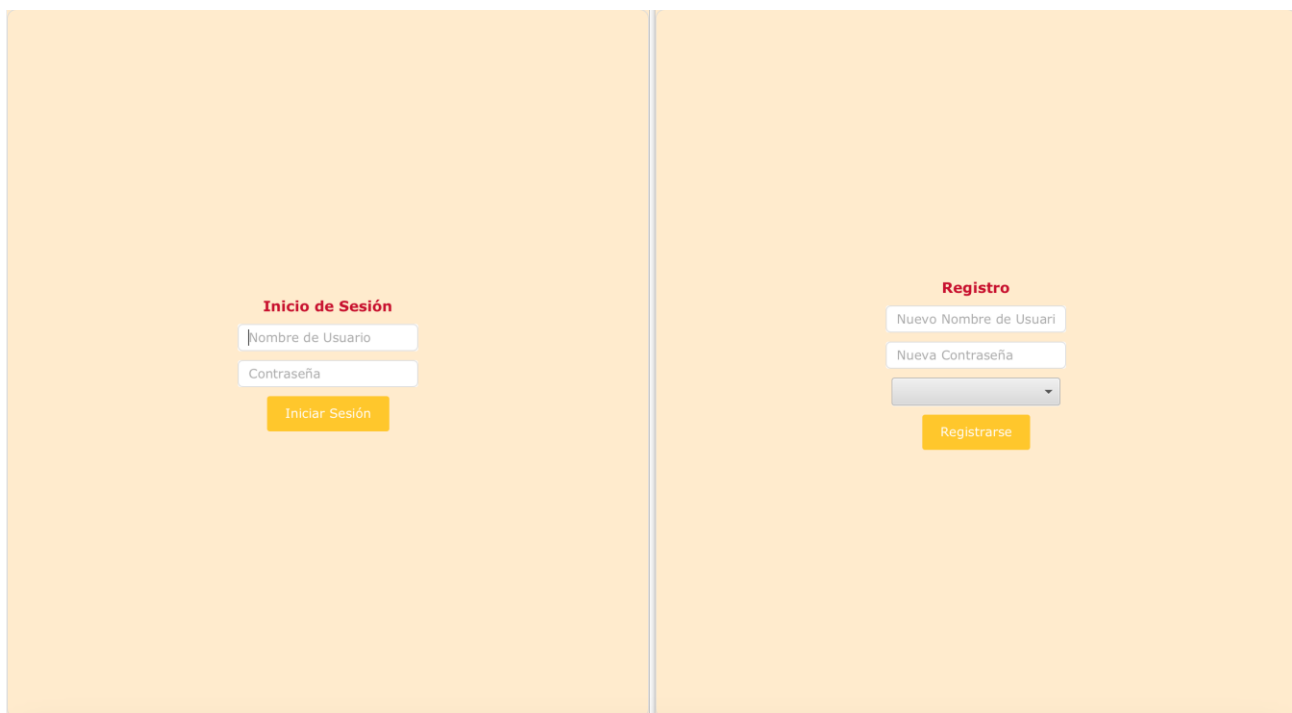
Manual Usuario

SECCION DE LOGIN/REGISTER:

Aquí encontraremos dos secciones divididas por un separator , donde podremos iniciar sesión o registrarnos si no tenemos cuenta (En el script hay inserts de prueba para el login directo).

En el apartado de Registro, tendremos que establecer un nombre de usuario y una contraseña, seleccionando también en que restaurante se va a trabajar.

Para el Inicio de sesión, simplemente se introducirán las credenciales, y el sistema procederá a mostrar la siguiente pantalla.



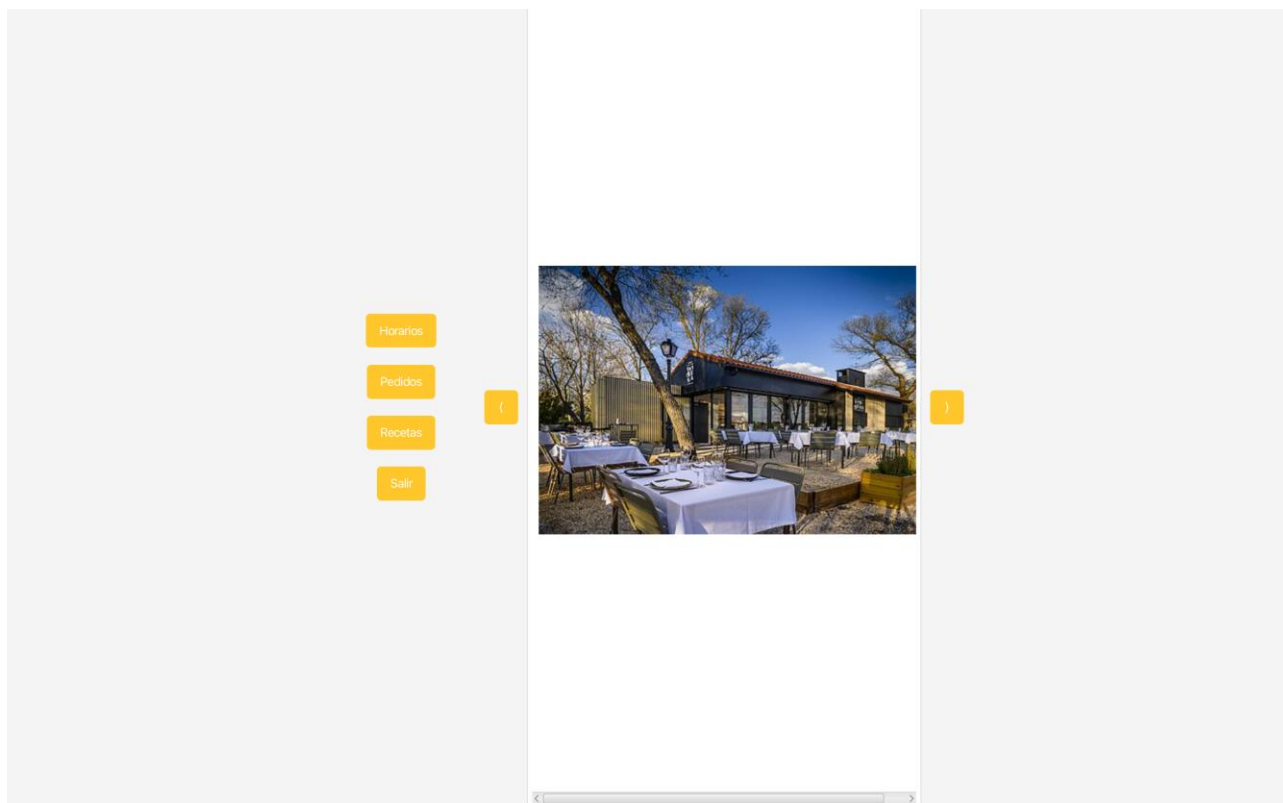
The image shows a user interface with two panels side-by-side on a light orange background. The left panel is titled 'Inicio de Sesión' in red. It contains two input fields: 'Nombre de Usuario' and 'Contraseña', both with a small blue cursor icon. Below these fields is a yellow button labeled 'Iniciar Sesión'. The right panel is titled 'Registro' in red. It contains three input fields: 'Nuevo Nombre de Usuario', 'Nueva Contraseña', and a dropdown menu. Below these fields is a yellow button labeled 'Registrarse'.

MENU PRINCIPAL:

En esta sección, observamos 4 botones que nos ayudarán a navegar entre las distintas secciones de la aplicación: Horarios, Pedidos, Recetas y, por último, un botón de salir.

Proyecto Desarrollo de Aplicaciones Multiplataforma

También observamos un carrusel de imágenes de restaurantes.




HORARIOS:

En esta sección, disponemos de un TableView el cual nos describe la tabla horarios de la base de datos, y nos permite al clicar encima de un horario, nos saldrá una ventana con los campos disponibles para cambiarlos y hacer un alter table en la base de datos. También dispondremos de un botón para volver al menú principal. Se implementó al final un botón más para crear horarios nuevos.

Modificar Turno

Usuario:

Fecha:




Turno:

Descripción:

Guardar Cambios

En esta sección, de la aplicación, puedes seleccionar un producto, y te mostrara un pop up como el de la segunda foto, con la receta del producto, esto puede ser útil para los empleados para evitarse formación tradicional.

Descripción del Producto

 Freímos las papas hasta que estén doradas y crujientes. Las cubrimos con queso cheddar rallado y bacon crujiente troceado. Gratinar hasta que el queso se derrita. Servir con salsa ranch.

[Aceptar](#)

Después de apuntar todos los pedidos, si seleccionan el botón de “Generar Recibo”, insertaran el pedido en la base de datos, a la vez que se genera una factura que se envía por correo electrónico al correo deseado de la empresa.

Viabilidad tecno-económica.

1. Costes de Software

JavaFX:

- Es una plataforma de código abierto y gratuito para el desarrollo de interfaces de usuario en Java, por lo que no genera costes adicionales.

IntelliJ IDEA:

- La Community Edition de IntelliJ IDEA es gratuita y adecuada para proyectos educativos y pequeños negocios. La Ultimate Edition cuesta 149€ al año para estudiantes, aunque muchos centros educativos ofrecen licencias gratuitas o con descuento.

Otras Herramientas:

- **Git** para control de versiones, Maven para la gestión de dependencias, y el JDK (Java Development Kit) son gratuitas y esenciales para el desarrollo del proyecto.

2. Costes de Hardware

Ordenadores:

- Se recomienda un ordenador de gama media para el desarrollo fluido de aplicaciones JavaFX. El coste aproximado de un ordenador de gama media (con procesador i5/i7, 8-16 GB de RAM y un SSD) es de 600€ a 1.000€.

Periféricos:

- Incluyendo monitor, teclado y ratón, el coste adicional es de aproximadamente 150€ a 300€.

Actualizaciones y Mantenimiento:

- Costes adicionales para el mantenimiento y actualización de los hardware estimados en 100€ a 200€ anuales.

3. Modelo de Negocio

Venta de Licencias:

Los restaurantes pagarán una licencia anual para usar la aplicación, con un precio estimado de 500€.

Soporte y Mantenimiento:

Ingresos adicionales por ofrecer servicios de soporte y mantenimiento a un coste de 100€/mes por cliente.

4. Estimación de ingresos

Clientes Iniciales:

- Se espera vender 20 licencias en el primer año.

Ingresos Estimados:

- 20 licencias vendidas a 500€ cada una: 10.000€ anuales.
- Servicios de soporte y mantenimiento: 100€/mes por cliente, sumando 24.000€ anuales (asumiendo soporte a todos los clientes).

5. Costes Adicionales

Marketing y Publicidad:

- Coste: 2,000€ - 5,000€ anuales.
- Descripción: Gastos para la promoción de la aplicación en el mercado hostelero.

Servidor y Hosting:

- Coste: 500€ - 1,000€ anuales.
- Descripción: Costes para mantener el servidor y el hosting necesario para la aplicación.

Formación y Capacitación:

Coste: 500€ - 1,000€ anuales.

Descripción: Gastos para la formación del personal del cliente en el uso de la aplicación.

Actualizaciones y Nuevas Funcionalidades:

Coste: 2,000€ - 5,000€ anuales.

Descripción: Costes asociados con el desarrollo de nuevas funcionalidades y actualizaciones de la aplicación.

6. Análisis de Rentabilidad

Costes Totales Anuales:

Proyecto Desarrollo de Aplicaciones Multiplataforma

- Desarrollo de Software: 149€ (si se usa la Ultimate Edition de IntelliJ IDEA).
- Hardware: 1,000€ + 300€ (periféricos) + 200€ (mantenimiento) = 1,500€.
- Marketing y Publicidad: 3,500€ (promedio).
- Servidor y Hosting: 750€ (promedio).
- Formación y Capacitación: 750€ (promedio).
- Actualizaciones y Nuevas Funcionalidades: 3,500€ (promedio).
- Total, Costes Anuales: 10,149€.

Ingresos Totales Anuales:

- Total, Ingresos: 34,000€.

Beneficio Neto Anual:

- Ingresos Totales - Costes Totales: 34,000€ - 10,149€ = 23,851€.

Conclusión

El análisis de viabilidad económica muestra que el proyecto Restaurante Bloski's tiene un gran potencial de rentabilidad, con un beneficio neto anual estimado de 23,851€. La inversión inicial en hardware y marketing es significativa, pero los ingresos generados por la venta de licencias y los servicios de soporte y mantenimiento superan con creces los costes, haciendo que el proyecto sea económicamente viable.

Trabajo futuro.

En el desarrollo de la aplicación, se han cumplido casi todos los objetivos funcionales, sin embargo, queda aún mucho trabajo por delante para que la aplicación sea competente en el mercado.

Apartado Gráfico:

En este punto, sería conveniente modificar la aplicación para dar una mejor experiencia de usuario al cliente final.

Se podría construir un diseño en Figma , y reestructurar un poco los campos ya disponibles en nuestra aplicación, además de que sería conveniente contar con la ayuda de un experto diseñador.

Funcionalidad:

Hay que reconocer que en la aplicación existen algunos fallos, los cuales deberían mejorarse en futuras versiones.

Algunos de ellos son: la posibilidad de imprimir pedidos utilizando la funcionalidad de añadir recetas, es decir, poder dar de alta un producto y generar ya una funcionalidad para generar la factura de este en los pedidos. Esto sería algo bueno ya que, si no, habría que especificar con anterioridad los productos que se van a vender, en un restaurante e ir cambiando el software con el paso del tiempo.

Un elemento clave que nos hubiera gustado implementar, sería un apartado de roles, el cual se nos dificultó por un problema de paquetes, con ello cambiaría un poco la funcionalidad de la aplicación, y se investigará para hacerlo en un futuro.

Seguridad y otros apartados técnicos:

Respecto a la seguridad, debería de implementarse en un futuro, un sistema de encriptación para la contraseña, así como un sistema de protección contra inserción de SQL.

Conclusiones.

Durante el proyecto se encontraron diversos problemas, que se fueron solucionando conforme se desarrollaban.

Uno de los mayores retos de este proyecto fue trabajar con dependencias desconocidas.

En este punto nos gustaría hacer hincapié en dos de ellas:

- <iText.pdf>: Cuando se trabajaba con esta dependencia, fue todo un logro aprender a utilizarla, ya que se tuvo que hacer uso de documentación externa disponible en la página web de iTextPDF así como del repositorio Maven para utilizar la versión correcta ya que daba fallos las antiguas: <https://mvnrepository.com/artifact/com.itextpdf/itextpdf>
- <JavaSunMail>: Esta dependencia, también fue todo un reto por diversas razones: La primera fue que era una librería desconocida, y como la anterior mencionada, se tuvo que utilizar la versión correcta para su funcionamiento. Otro problema fue a la hora de usar el dominio @gmail.com, el cual, por seguridad de Google, no dejaba usar el protocolo SMTP de manera sencilla, y había que configurar las opciones de privacidad.

Otro reto fue el apartado de interfaz gráfica, ya que, aunque en la aplicación no se apreciaba un diseño del todo “bonito”, se pretendió hacer una interfaz fácil de usar y se estudió el diseño de otras aplicaciones del mercado.

Conclusión

Se han cumplido los objetivos principales, reflejando el esfuerzo puesto en el desarrollo de la aplicación.

Un último punto para destacar es la dificultad de desarrollo por la escasez de tiempo durante las prácticas, esto es importante ya que a pesar de haber pensado secciones del proyecto como el análisis de requisitos y el diseño de la base de datos mucho antes del comienzo del desarrollo, al final se dedicó mucho tiempo a la fase de pruebas y mejoras, lo cual conforme se acercaba la fecha de entrega fue un “*hándicap*” para tener en cuenta.

Biblioteca de recursos web y referencias.

1. JavaFX, I. (2024): JavaFX API Documentation, Oracle Corporation, disponible en:
<https://openjfx.io/javadoc/17/>
2. MVN Repository, I. (2024): MVN Repository, MVN Repository, disponible en:
<https://mvnrepository.com/>
3. JavaMail, I. (2024): JavaMail, Oracle Corporation, disponible en:
<https://javaee.github.io/javamail/>
4. iTextPDF, I. (2024): iTextPDF Documentation, iText Software, disponible en:
<https://itextpdf.com/en/resources/books/itext-7-jump-start-tutorial>
5. Java Oracle, I. (2024): Java Oracle Documentation, Oracle Corporation, disponible en :
<https://docs.oracle.com/javase/8/docs/>
6. Hibernate, I. (2024): Hibernate Documentation, Hibernate ORM, disponible en:
<https://hibernate.org/orm/documentation/>

Anexos.