Terminal punto de venta (TPV) para un bar con conexión a una aplicación móvil

Yufeng Yang

Resumen— Actualmente, en el sector de la hostelería, como bares y restaurantes, es común el uso de software para gestionar las comandas de los clientes, conocido como Terminal Punto de Venta (TPV). Aunque existen numerosos servicios comerciales disponibles en internet que ofrecen este tipo de software, no todos cuentan con la funcionalidad de integrar una aplicación móvil para enviar las comandas al sistema. Además, otro inconveniente frecuente es el elevado costo de estas soluciones comerciales. El objetivo de este proyecto es desarrollar un TPV con aplicación móvil que ofrezca un buen rendimiento a un precio accesible.

Parablas clave—TPV, Java, App, FrontEnd, BackEnd, BD, UI

Abstract— Currently, in the hospitality sector, such as bars and restaurants, it is common to use software to manage customer orders, known as Terminal Point of Sale (POS). Although there are numerous commercial services available on the internet that offer this type of software, not all of them have the functionality of integrating a mobile application to send orders to the system. In addition, another frequent drawback is the high cost of these commercial solutions. The objective of this project is to develop a POS with mobile application that offers good performance at an accessible price.

Index Terms—POS, Java, App, FrontEnd, BackEnd, BD, UI

1 Introducción

Actualmente, la tecnología juega un papel esencial en la gestión y funcionamiento de los establecimientos de hostelería, como bares y restaurantes. La automatización de procesos mediante los sistemas de Terminal Punto de Venta (TPV) ha mejorado significativamente la atención al cliente, la eficiencia en el flujo de trabajo y el control de ventas. Sin embargo, la creciente adopción de dispositivos móviles y la necesidad de ofrecer un servicio más ágil han generado nuevas oportunidades para agilizar el servicio.

A pesar de las ventajas que ofrecen los sistemas TPV tradicionales, muchos de ellos todavía presentan limitaciones en términos de flexibilidad y personalización. Esto dificulta que los trabajadores puedan gestionar los procesos de forma dinámica y adaptada a las necesidades específicas del servicio.

Con el objetivo de superar estas limitaciones, se propone el desarrollo de un sistema de TPV integral conectado a una aplicación móvil diseñado específicamente para la hostelería. Este sistema permitirá a los camareros gestionar pedidos de forma eficiente desde cualquier punto del establecimiento y acceso a la información de todas las mesas.

En definitiva, la integración de un TPV con una aplicación móvil supone una oportunidad para revolucionar el funcionamiento de los bares, ofreciendo una solución flexible, eficiente y personalizada que beneficiará tanto al personal como a los clientes, contribuyendo a un servicio de alta calidad y una gestión empresarial más efectiva.

2 Objetivos

El objetivo principal de este proyecto es crear un terminal de venta (TPV) que se integre con una aplicación móvil para un bar. El uso de este sistema mejorará la eficiencia operativa y la calidad del servicio al optimizar la gestión de pedidos y el control de ventas.

Se pueden identificar varios objetivos específicos a partir de este objetivo principal:

Gestión eficiente de pedidos y reducción de tiempo de espera: El software ayudará a los camareros a gestionar los pedidos de manera más rápida, enviándolos directamente a la cocina desde dispositivos móviles, lo que reduce el tiempo de espera y los errores en la transmisión.

Control de ventas: El TPV permitirá a los administradores del bar tener un control preciso de las ventas realizadas y de los productos más vendidos.

Interfaz de usuario fácil de usar e intuitiva: Se priorizará una interfaz de usuario simple y fácil de usar para camareros y administradores, con una navegación clara que permita la visualización y gestión rápida de pedidos y ventas.

Integración con dispositivos móviles: Debido a que el uso de una aplicación que permita al personal acceder al TPV desde cualquier lugar del proyecto, el sistema debe poder funcionar con dispositivos móviles sin problemas.

3 ANÁLISIS DE SOLUCIONES EXISTENTES

En el sector de la restauración, el uso de los sistemas TPV permite tener un control y gestión ordenado de las operaciones diarias de los establecimientos. Si bien, el TPV proporciona a los negocios un control más preciso de las ventas, el inventario y los pedidos, casi todos los sistemas existentes son dispositivos fijos sin integración con dispositivos móviles. Por lo tanto, la capacidad de servicio del personal en establecimientos de alta afluencia de clientes, como bars y restaurantes, tiende a verse reducida.

Hay varias opciones disponibles en el mercado que permitirían mejorar la capacidad de servicio del personal en establecimientos de alta demanda:

TPV tradicional con sistemas fijos: En muchos establecimientos, la opción predominante sigue siendo una terminal física que los empleados utilizan para procesar pedidos y pagos. Aunque es una solución robusta y efectiva para controlar las operaciones, tiene el inconveniente de ser costosa y poco eficiente, ya que obliga al personal a desplazarse continuamente hacia la terminal para realizar estas gestiones. Esto resulta en una pérdida de tiempo, especialmente en momentos de alta demanda, lo que impacta negativamente en la eficiencia del servicio.
Clover y **Square**, ofrecen terminales físicas con pantallas táctiles y lectores de tarjetas. Estos sistemas proporcionan una solución integral de gestión de ventas e inventarios, pero su implementación suele requerir una inversión considerable en hardware y software, lo que puede ser prohibitivo para algunos pequeños negocios.

TPV con integración en nube: Los TPV basados en la nube permiten a las empresas gestionar sus operaciones de manera remota, lo que mejora significativamente la eficiencia operativa. Estos sistemas ofrecen acceso instantáneo a datos de ventas, inventarios y análisis en tiempo real, desde cualquier dispositivo conectado a internet. Sin embargo, a pesar de sus beneficios, muchas de estas soluciones no se enfocan lo suficiente en la movilidad del personal y, además, suelen tener costos elevados tanto en suscripciones como en hardware compatible.
Lightspeed y **Vend**, ofrecen plataformas flexibles con opciones de gestión remota. Aunque mejoran la accesibilidad y el control de las operaciones, los costos asociados con su implementación y la necesidad de hardware específico los convierten en soluciones más apropiadas para medianas y grandes empresas.

Sistemas TPV integrados con aplicaciones móviles comerciales: Algunas soluciones en el mercado ya incluyen integración con dispositivos móviles, haciendo posible para el personal del bar hacer pedidos directamente en un teléfono o tableta. Estas aplicaciones también permiten facilitar la comunicación entre el personal de la sala y la cocina y hacer un seguimiento del proceso de preparación de los platos. Sin embargo, la principal desventaja es que muchos de los desarrollos no son totalmente personalizables y son mucho más caros en términos de licencias.
Toast POS y **Revel Systems**, permiten al personal gestionar pedidos desde dispositivos móviles, mejorando la experiencia del cliente y aumentando la eficiencia operativa. No obstante, el elevado costo de licencias y la falta de flexibilidad en algunos casos pueden ser desventajas para pequeños negocios que buscan soluciones más adaptadas a sus necesidades.

Modificar un sistema de TPV de código abierto: Esta opción permite adaptar un sistema TPV existente de código abierto para ajustarlo a las necesidades específicas del bar, como la integración con una aplicación móvil. Las ventajas incluyen un menor coste inicial y la flexibilidad para personalizar el software según los requisitos del negocio, además de un tiempo de implementación más corto al partir de una base funcional.
Odoo POS, ofrece gestión de ventas e inventarios, y puede integrarse con otros módulos como aplicaciones móviles, y **Floreant POS**, diseñado específicamente para restaurantes y bares, que permite una fácil adaptación para mejorar la gestión de comandas y pagos desde dispositivos móviles.
Las desventajas pueden incluir posibles incompatibilidades o limitaciones del sistema escogido, lo que puede requerir mayor esfuerzo en personalización y mantenimiento.

Creación de un sistema TPV personalizado con integracion con una aplicación móvil: Esta opción resulta atractiva porque permite desarrollar una solución específica para el bar, adaptada a sus necesidades únicas. En este caso, el sistema se construirá a partir de elementos clave y funcionalidades especiales que respondan a los requerimientos del negocio.
Sin embargo, la desventaja de esta alternativa es que el desarrollo requiere considerable tiempo y recursos, lo que la convierte en una opción significativamente más costosa que otras disponibles en el mercado.

A pesar de esto, se ha optado por la creación de un sistema TPV desde cero, ya que permite personalizar el software en su totalidad, asegurando que todas las especificaciones se integren de manera efectiva.
Este enfoque personalizado podría aportar ventajas únicas sobre las soluciones existentes, como una interfaz de usuario adaptada a las preferencias del personal, funciones específicas para la gestión del menú y la integración directa con sistemas de inventario o contabilidad utilizados en el bar. Además, la posibilidad de actualizar y modificar el sistema de acuerdo con la evolución del negocio ofrece una flexibilidad que muchos TPV comerciales no pueden igualar.

4 REQUISITOS

Los siguientes listados definen el conjunto de requisitos base del proyecto. También se define la prioridad de cada requisito.

4.1 Funcionals

RF-01: El sistema debe permitir a los usuarios iniciar sesión con una cuenta de administrador o camarero para acceder a las funciones del TPV. **(Prioridad: Alta)**

RF-02: El sistema debe permitir a los usuarios introducir pedidos desde el TPV o desde la aplicación móvil. **(Prioridad: Alta)**

RF-03: El sistema debe enviar los pedidos al software. **(Prioridad: Alta)**

RF-04: El sistema debe permitir consultar y gestionar los pedidos en tiempo real, desde cualquier dispositivo conectado al sistema (TPV o aplicación móvil). **(Prioridad: Alta)**

RF-05: El sistema debe permitir la modificación de pedidos existentes, añadiendo o eliminando productos antes de cerrar el pedido. **(Prioridad: Media)**

RF-06: El sistema debe proporcionar una vista general de todas las mesas del bar, con la posibilidad de asignar pedidos a cada mesa. **(Prioridad: Media)**

RF-07: El sistema debe permitir dividir el pago entre varios clientes de una misma mesa. **(Prioridad: Media)**

RF-08: El sistema debe permitir generar facturas automáticas una vez pedido por el programa. **(Prioridad: Alta)**

RF-09: El sistema debe permitir a los usuarios realizar el pago desde terminales fijas o móviles, aceptando varios métodos de pago (efectivo, tarjeta, etc.). **(Prioridad: Alta)**

RF-10: El sistema debe registrar y almacenar todas las transacciones y ventas en una base de datos para su consulta posterior. **(Prioridad: Alta)**

RF-11: El sistema debe permitir generar informes de ventas diarias, semanales y mensuales de forma automática. **(Prioridad: Media)**

RF-12: El sistema debe permitir a los administradores configurar y modificar los precios de los productos desde el TPV. **(Prioridad: Baja)º**

RF-13: El sistema debe proporcionar una interfaz de usuario intuitiva para facilitar su uso a camareros y administradoras, tanto en el TPV como en la aplicación móvil. **(Prioridad: Alta)**

RF-14: El sistema principal debe poder utilizarse sobre todo con la pantalla táctil. **(Prioridad: Baja)**

4.2 No funcionales

RNF-01: El sistema debe estar altamente disponible y asegurar una respuesta rápida en las operaciones de registro de pedidos, con un tiempo de respuesta máximo de 2 segundos por operación. **(Prioridad: Alta)**

RNF-02: El sistema debe ser escalable para permitir su uso en distintos establecimientos y soportar múltiples dispositivos conectados simultáneamente. **(Prioridad: Alta)**

RNF-03: La aplicación móvil debe ser responsive, adaptándose a diferentes tamaños de pantalla, incluyendo teléfonos y tablets. **(Prioridad: Alta)**

RNF-04: El diseño de la interfaz de usuario debe ser intuitivo y fácil de usar para camareros con distintos niveles de formación tecnológica. **(Prioridad: Alta)**

RNF-05: El sistema debe garantizar la seguridad de las datos personales y financieros. **(Prioridad: Alta)**

RNF-06: El sistema debe ser compatible con distintos sistemas operativos, tanto para terminales fijas (Windows, Linux) como para dispositivos móviles (Android). **(Prioridad: Media)**

RNF-07: La base de datos debe ser relacional (SQL) para asegurar una gestión eficiente de los datos de pedidos, inventario y ventas. **(Prioridad: Media)**

RNF-08: El sistema debe poder funcionar en modo offline, almacenando temporalmente los datos y sincronizándolos cuando la conexión a internet se restablezca. **(Prioridad: Media) (Sin hacer)**

RNF-09: Las actualizaciones del inventario y ventas deben realizarse en tiempo real para asegurar que los datos estén actualizados constantemente entre el TPV y la aplicación móvil. **(Prioridad: Media)**

RNF-10: El sistema debe cumplir con las normativas locales de protección de datos (como el GDPR si aplica) y normativas fiscales para la facturación. **(Prioridad: Alta)**

RNF-11: La interfaz debe ser optimizada para su uso en pantallas táctiles, asegurando que los botones y controles sean fácilmente accesibles. **(Prioridad: Media)**

RNF-12: El sistema debe soportar un volumen de hasta 500 transacciones por día, garantizando el rendimiento sin caídas ni lentitud. **(Prioridad: Alta)**

RNF-13: La aplicación móvil debe ser capaz de gestionar notificaciones push para alertar al personal de nuevos pedidos o actualizaciones importantes. **(Prioridad: Baja)**

RNF-14: El sistema debe permitir la integración con dispositivos de impresión de tickets. **(Prioridad: Media)**

4.3 Requisitos técnicos

RT-01: El sistema debe utilizar una arquitectura cliente-servidor, gestionando las conexiones entre el TPV, la aplicación móvil y la base de datos central. El servidor central manejará las solicitudes de pedidos, gestión de usuarios, y actualizaciones en tiempo real. **(Prioridad: Alta)**

RT-02: El backend será implementado en Java, mientras que el frontend del TPV y la aplicación móvil se desarrollará con React. Spring Boot será utilizado para gestionar el servidor y la comunicación con el frontend. Esto garantizará un desarrollo eficiente y una integración fluida. **(Prioridad: Alta)**

RT-03: El sistema utilizará MySQL como base de datos relacional para gestionar eficientemente los datos de usuarios, pedidos, inventarios y transacciones. **(Prioridad: Alta)**

RT-04: El sistema debe ser escalable y soportar múltiples dispositivos conectados simultáneamente, permitiendo el crecimiento del negocio. **(Prioridad: Media)**

5 METODOLOGIA

En este proyecto se hará uso de la metodología de desarrollo incremental, donde las tareas se distribuirán en diferentes etapas, permitiendo la construcción del sistema en partes funcionales y mejorando gradualmente el producto a lo largo del tiempo. Cada etapa tendrá sus objetivos específicos, que podrán ajustarse en función del feedback del cliente y las prioridades del proyecto durante el proceso de desarrollo.
Se priorizarán las tareas relacionadas con las funcionalidades más esenciales del TPV (Prioridad Alta), asegurando que se cubran las necesidades más críticas del bar de forma rápida y eficiente.

5.1 Trello

Trello es una herramienta que permite gestionar y organizar proyectos de forma visual y colaborativa. Esta plataforma facilita el seguimiento exhaustivo de las tareas y el estado actual con solo verlo.

En Trello, las tareas pueden gestionarse mediante tarjetas, que se distribuyen en diferentes listas según su estado. Para el desarrollo del proyecto, se han definido las siguientes listas para organizar y realizar seguimiento de las tareas:

- Lista de tareas: Incluye todas las tareas planificadas para el proyecto, pero que todavía no se han empezado.
- Lista de tareas semana actual: Contiene las tareas que se han priorizado para la semana en curso.
- En proceso: Aquí se colocan las tareas que ya están en fase de ejecución o desarrollo activo.
- Hecho: Esta lista incluye las tareas completadas, reflejando el progreso realizado hasta el momento.

Durante el desarrollo del proyecto, se actualizava en tiempo real el estado de cada tarea moviendo las tarjetas entre listas según su progreso. Esta metodología visual e intuitiva facilita un seguimiento preciso, mejora la gestión de los sprints.

5.2 GitHub

GitHub se ha utilizado como repositorio central para control de versiones del proyecto. Este contiene todos los archivos del back y del front.

6 PLANIFICACIÓN BASE

6.1 Fases

La planificación que se ha decidido seguir cuenta con las siguientes fases:

Fase 1 - Concepción: Se desarrollará el concepto y en la determinación de las principales características del sistema TPV. En esta fase se definen las funcionalidades básicas y cómo se conectará el TPV con la aplicación móvil, así como los requisitos generales del proyecto.

Fase 2 - Planificación: En esta fase se definirán los requisitos detallados y se aclara la visión global del proyecto. Además, se planifican las tecnologías a utilizar, como el lenguaje de programación, las bases de datos y los dispositivos necesarios para su implementación. También se realiza una estimación del alcance y las principales funcionalidades.

Fase 3 - Diseño: Se crean los diseños técnicos y de la interfaz del TPV y de la aplicación móvil. Con la ayuda de diagramas y mock up, se especifica cómo funcionarán las diferentes partes del sistema.

Fase 4 - Desarrollo: Es la fase de implementación del proyecto definido en las fases anteriores. Se utilizará la metodología Scrum para desarrollar el sistema de forma iterativa, trabajando con sprints, que se pueden ajustar en función de las necesidades durante el desarrollo.

Fase 5 - Formalización, entrega y cierre: En esta última fase, se verifica que el sistema cumpla con los requisitos y diseños definidos anteriormente. Se validan las funcionalidades desarrolladas, asegurando que son aceptadas por los stakeholders (en este caso, el personal del bar). Por último, se hace la entrega oficial del sistema TPV completo con la aplicación móvil conectada.

6.2 Planificación de desarrollo

Iteración 1 (1 semanas):

- Implementación del sistema de validación de usuarios y gestión de roles (RF-01).
- Implementación de la estructura básica del sistema, incluyendo la creación de la base de datos inicial y la conexión con el backend.

Iteración 2 (3 semanas):

- Gestión de pedidos desde el TPV (RF-02, RF-03, RF-04).

- Integración de la base de datos para almacenar y consultar los pedidos en tiempo real.

Iteración 3 (3 semanas):
- Funcionalidad para gestionar las mesas y los pedidosasociados (RF-06).
- Desarrollo de la aplicación movil
- Modificación de pedidos en curso, así como divisiónde pagos entre clientes (RF-05, RF-07).

Iteración 4 (1 semanas):
- Sistema de pagos (RF-09).
- Registro de transacciones y listado de transacciones pasadas (RF-10).
- Ajustes de precios y configuraciones administrativas (RF-12).

Iteracion 5 (2 semanas):
- Sistema de pagos mejorada (Test de usuario)
- Generación de facturas automáticas (RF-08)
- Generación de informes básicos (RF-11)

Iteración 6 (2 semanas):
- Mejoras en general

6.4 Costes

Recursos Humanos
- **Tarifa Promedio por Hora**: Supongamos una tarifa de **20 € por hora**.
- **Horas Estimadas por Fase**:
  - **Fase 1 - Concepción**: 10 horas
  - **Fase 2 - Planificación**: 10 horas
  - **Fase 3 - Diseño**: 20 horas
  - **Fase 4 - Desarrollo**: 200 horas
  - **Fase 5 - Formalización, entrega y cierre**: 60 horas
- **Total de Horas Estimadas**: 300 horas
- **Costo Total de Recursos Humanos**: 300 horas * 20 €/hora = 6000 €

Licencias
- No hay costes de licencia, ya que se usará software de libre uso.

Hardware
- **Ordenador de Sobremesa - Personal**
- **Portátil - Personal**
- **Teléfono móvil - Personal**
- **Impresora de Tickets – 20€**

Coste estimado total
- Recursos humanos 6000€
- Licencias Gratis
- Hardware 20€

Total 6020€

6.5 Riesgos

Mala priorización de tareas
Efecto: El desarrollo del sistema TPV no se centra en las funcionalidades esenciales desde el principio, provocando la carencia de tiempo para completar las funciones críticas.
Probabilidad: Baja
Nivel de impacto: Crítico
Solución: Revisar la priorización de los requisitos con frecuencia, teniendo en cuenta las necesidades del bar y de la aplicación móvil para asegurar que se desarrollen las funcionalidades más importantes primero.

Cantidad de funcionalidades poco realista
Efecto: La lista de funcionalidades a implementar podría ser demasiado grande o pequeña, desajustándose con el tiempo disponible.
Probabilidad: Baja
Nivel de impacto: Crítico
Solución: Realizar estimaciones realistas del tiempo necesario para cada tarea, añadiendo un margen de seguridad, y ajustar las expectativas con el cliente de forma continua.

Conocimientos insuficientes
Efecto: Las carencias en el conocimiento técnico podrían frenar el desarrollo del TPV o la aplicación móvil, causando retrasos en el resto de las tareas.
Probabilidad: Media
Nivel de impacto: Crítico
Solución: Establecer márgenes en el plan de trabajo que permitan tiempo para el aprendizaje y la investigación de tecnologías desconocidas.

Resultados del proyecto insatisfactorios
Efecto: Un resultado final que no cumpla con las expectativas del cliente podría suponer una pérdida de tiempo y recursos.
Probabilidad: Baja
Nivel de impacto: Crítico
Solución: Cerciorarse de mantener una comunicación constante con el cliente durante todo el proceso para ajustar las funcionalidades según su feedback.

Incompatibilidad entre componentes del sistema
Efecto: Problemas de integración entre el TPV, la aplicación móvil y otros sistemas podrían interrumpir su desarrollo.
Probabilidad: Baja
Nivel de impacto: Crítico
Solución: Realizar pruebas de integración desde las primeras fases del desarrollo y tener preparadas alternativas tecnológicas para los componentes que presenten problemas de compatibilidad.

Datos insuficientes o inaccesibles
Efecto: La falta de datos o su inaccesibilidad podría impedir que el TPV funcione correctamente o que la aplicación móvil no muestre la información correcta.
Probabilidad: Baja
Nivel de impacto: Crítico
Solución: Establecer sistemas de backup y asegurarse de que los datos esenciales están siempre disponibles y que pueden introducirse manualmente en caso de emergencia.

Corrupción o inaccesibilidad al código
Efecto: La pérdida o corrupción del código podría interrumpir el desarrollo del proyecto de forma crítica.
Probabilidad: Baja
Nivel de impacto: Crítico
Solución: Utilizar un sistema de control de versiones

(como Git) y realizar copias de seguridad regularmente en un servicio cloud para asegurar la continuidad del proyecto en caso de problemas.

7 HERRAMIENTAS

Se han evaluado diferentes herramientas para implementar el sistema TPV y la aplicación móvil. Los marcos que facilitan la construcción de las diferentes partes del sistema optimizan el desarrollo y garantizan una mejor experiencia de usuario son convenientes para este tipo de proyecto.

El principal criterio para seleccionar las tecnologías ha sido su popularidad, madurez en el desarrollo de aplicaciones y compatibilidad. Se consideran frameworks y librerías modernas que son fáciles de integrar para proyectos de TPV y aplicaciones móviles.

7.1 BackEnd

Java: El backend del sistema TPV será desarrollado utilizando Java, un lenguaje de programación orientado a objetos ampliamente utilizado en sistemas empresariales y aplicaciones a gran escala. Java destaca por su portabilidad, rendimiento y la amplia comunidad de soporte, lo que facilita el desarrollo de aplicaciones robustas y escalables.

Spring Boot: Framework de Java diseñado para simplificar la creación de aplicaciones basadas en Spring. Proporciona una configuración automatizada para desarrollar APIs y servicios web de forma rápida y eficiente.

Docker: Docker es una plataforma de contenedores que permite empaquetar, distribuir y ejecutar aplicaciones de manera aislada y consistente en cualquier entorno. Docker facilita la gestión de aplicaciones, mejora la portabilidad y optimiza el uso de recursos en comparación con máquinas virtuales tradicionales.

7.2 FrontEnd

React: Framework de Javascript centrado en la construcción de interfaces de usuario dinámicas mediante componentes reutilizables. Se trata de una opción consolidada y de código abierto.

7.3 Base de datos

Para este proyecto se ha decidido utilizar una base de datos relacional debido a la familiaridad del cliente con este tipo de sistemas, lo que facilitará el mantenimiento futuro.

MySQL: Sistema de gestión de bases de datos relacional de código abierto, ampliamente utilizado por su rendimiento, robustez y escalabilidad.

Para la base de datos, se diseñó un esquema que se encuentra en el apéndice A.4. Las tablas necesarias se detallan a continuación:

- **Usuario**: Tabla que almacena los datos de los usuarios registrados, incluyendo el nombre de usuario, contraseña y su rol dentro del sistema. El campo número_usuario tiene una restricción ≪Unique≫ para evitar duplicidades. También se registra la fecha de creación de cada usuario.

- **Mesa**: Tabla que identifica las tablas del bar con un número único y su estado actual (ocupada o libre).

- **Categoría**: Tabla que almacena las diferentes categorías de los productos disponibles en el sistema, con un campo número que identifica cada categoría.

- **Producto**: Tabla que contiene los productos disponibles en el bar, incluyendo el nombre, el precio, la categoría asociada y la cantidad de stock disponible.

- **Pedido**: Tabla que registra los pedidos realizados, con referencias a los usuarios y tablas asociadas, el precio total del pedido y la fecha en que se creó.

- **Detalle de Pedido**: Tabla intermedia que detalla los productos de un pedido, especificando la cantidad, precio unitario y el número de unidades ya pagadas.

- **Transacción**: Tabla que almacena los pagos asociados a cada pedido. Incluye el método de pago, la fecha de la transacción y el importe pagado.

7.4 FrontEnd App

React Native: React Native es un framework de desarrollo de aplicaciones móviles creado por Facebook, que permite construir aplicaciones para iOS y Android utilizando JavaScript y React. A diferencia de otras soluciones, React Native permite escribir código nativo, lo que significa que las aplicaciones pueden aprovechar características específicas de cada plataforma.

8 IMPLEMENTACION

8.1 Iteracion 1 – Estructura base

En la primera iteración del proyecto se desarrolló la infraestructura básica tanto del backend como del frontend. Este trabajo incluyó la configuración inicial de los frameworks, la estructuración de la arquitectura del sistema y la definición de los componentes esenciales para garantizar una sólida base para futuras funcionalidades.

Además, se diseñó e implementó el primer prototipo de la base de datos, que incluye las tablas. Este prototipo sirve como punto de partida para validar la integración entre el backend y la base de datos, así como asegurar una gestión eficiente de la información.

Para la base de datos, se diseñó un esquema inicial que define las tablas y datos requeridos por cada una de ellas.

Además, se creó un archivo de inicialización de la base de datos que genera todas las tablas, atributos y relaciones de clave necesarias. También se incluye un usuario de administrador (root) para probar la aplicación. Esta

configuración permite realizar pruebas funcionales de la base de datos con datos consistentes.

En el backend se desarrolló la primera petición para gestionar el inicio de sesión en el sistema TPV. Esta funcionalidad permitía autenticar a los usuarios y diferenciar entre usuarios con rol de administrador (root) y usuarios normales. Esta diferenciación era fundamental para habilitar funciones específicas para cada tipo de usuario, garantizando un acceso seguro y controlado a las funcionalidades del sistema según su rol asignado.

8.2 Iteración 2 – Almacenamiento en Tiempo Real

En la segunda iteración del proyecto, se enfocó en establecer toda la infraestructura necesaria en el backend para interactuar con las tablas de la base de datos, garantizando un manejo eficiente y seguro de la información. Este trabajo incluyó la creación de controladores, servicios y repositorios que permitieron implementar una arquitectura modular y escalable, facilitando futuras ampliaciones del sistema.

Se desarrollaron las primeras peticiones API REST para gestionar pedidos, lo que incluyó funcionalidades clave como:

- Crear nuevos pedidos asociados a una mesa y un usuario.
- Consultar pedidos en tiempo real, incluyendo sus detalles y estado.

Además, se realizaron pruebas exhaustivas con Postman para validar el correcto funcionamiento de las APIs. Esto incluyó verificar respuestas esperadas, manejo de errores y el comportamiento frente a casos límite. Estas pruebas garantizaron la fiabilidad y robustez del sistema al gestionar pedidos.

En frontend se desarrollo la pagina de iniciar sesión y la pagina de selección de mesa.

Finalmente, la integración con la base de datos se validó en tiempo real, confirmando que las operaciones CRUD (Crear, Leer, Actualizar, Eliminar) funcionaban correctamente con las tablas de pedidos y detalles de pedido.

8.3 Iteración 3 – Gestión de Mesas y Desarrollo de la Aplicación Móvil

En la tercera iteración del proyecto, se implementaron nuevas funcionalidades clave para mejorar la experiencia de usuario en el sistema TPV, tanto en el backend como en la aplicación móvil.

Se desarrolló la funcionalidad para gestionar las mesas, permitiendo a los usuarios visualizar las mesas activas y gestionar los pedidos correspondientes a cada una de ellas. Este avance incluyó la integración con la base de datos para almacenar los estados de las mesas y actualizar los pedidos en tiempo real.

Se implemento en frontend la pagina de la mesa donde se puede gestionar el pedido de la mesa.

Además, se inició el desarrollo de la aplicación móvil, que permitió a los usuarios interactuar con el sistema TPV desde sus dispositivos móviles. Esta parte del proyecto incluyó la implementación de las pantallas de inicio de sesión, selección de mesa y visualización de pedidos, facilitando la interacción con el sistema en un formato más accesible y móvil.

Por otro lado, se implementó la funcionalidad de modificación de pedidos en curso, lo que permitió a los usuarios actualizar los detalles de un pedido existente, como cambiar productos, cantidades o eliminar artículos. También, se desarrolló la funcionalidad para dividir pagos entre clientes, permitiendo a los usuarios gestionar el pago de un pedido de manera flexible, según las necesidades de cada cliente.

8.4 Iteración 4 – Pagos y ajustes

En esta iteración, se implementó el sistema de pagos, permitiendo guardar los pagos hechos en la base de datos con el front de la iteración 3, facilitando la consulta de pagos anteriores. Además, se añadieron los ajustes de precios y configuraciones administrativas, permitiendo a los administradores modificar precios y gestionar configuraciones del sistema. Estas mejoras optimizan la gestión de pagos y el control administrativo, asegurando la funcionalidad y flexibilidad del TPV.

En frontend también se creo una pagina para gestionar los productos, categorías y usuarios.

En esta iteración también se hizo algunas pruebas en el sistema.

Test de API

- **Pruebas de Funcionalidad**: Se realizaron pruebas para cada uno de los endpoints de la API, verificando que devolviesen las respuestas esperadas para todas las solicitudes (GET, POST, PUT, DELETE). Se comprobó que la API gestionara correctamente las operaciones relacionadas con pedidos, usuarios y gestión de inventario.
- **Validación de Respuestas**: Se verificó que las respuestas de la API incluyeran los códigos de estado HTTP correctos y los datos estructurados en el formato JSON adecuado. Esto incluye el manejo de errores, asegurando que se devuelvan mensajes claros y precisos en caso de fallos.

Test por parte de usuarios

- **Test sin explicar nada a usuarios:** Se ha presentado la aplicación a conocidos para que prueben sin explicar nada de como funciona.
  Se les ha dado una lista de tareas para que hagan.
  1. Crear una mesa
  2. Añadir productos
  3. Modificar cantidad

4. Modificar precio del producto
5. Guardar pedido/mesa
6. Facturar una mesa
7. Cobrar una mesa
8. Crear un producto y categoría
9. Crear otro usuario

El resultado ha sido bastante positivo, menos la parte de modificar precio del producto, la otra parte del programa ha sido bastante entendible.

- **Test de usuarios clave:** Se ha puesto delante del programa personas experimentadas en el sector. No se les ha dado ninguna pauta, solo que prueben el programa.
  Del resultado se ha decidido cambiar el apartado de pago y agrandar los botones.

8.5 Iteración 5 – Informes

En esta iteración, se **mejoró el sistema de pagos** mediante pruebas de usuario para garantizar su eficacia y usabilidad. Además, se implementaron las funcionalidades de **generación de facturas automáticas**, permitiendo la creación de facturas de manera rápida y precisa, y se desarrolló la capacidad de **generación de informes básicos**, facilitando la consulta de datos clave para el control de ventas e inventario. Estas mejoras optimizan la experiencia del usuario y proporcionan herramientas administrativas para el análisis de datos.

8.6 Iteración 6 – Mejoras

A partir de los test hechos en la iteración 4, se descarto la funcionalidad de tener la gestión de inventario. Ya que el usuario final para que se está desarrollando el proyecto tiene una aplicación dedicada a la gestión de inventario dada por su proveedor.
Así que se ha decidido que en esta iteración se dedicara en mejorar las funcionalidades ya implementadas.
También se ha creado el primer release del proyecto, utilizando Docker para desplegar el servidor y la pagina frontend.

10 Conclusión

Gracias a las decisiones tomadas, la planificación, las tecnologías utilizadas, los conocimientos previos y los conocimientos aprendidos, el proyecto se ha llevado a cabo de forma satisfactoria. Cumpliendo los requisitos especificados e incluso incorporando algunas mejoras respecto a la idea inicial, como el nuevo método de pago por produto.

He podido aprender bastante sobre las tecnologías utilizadas como docker para implementar el desplagamiento del proyecto.

Para mejoras futuras, se podría subir el servidor en alguna maquina virtual en la nube para poder acceder desde cualquier lugar deseado al TPV. Con el proyecto dockerizado no sería difícil la migración.

Finalmente, quería agradecer a mis padres y mis amigos por ayudarme a testear el proyecto y haberme dado ánimos. También al soporte por parte de mi tutor Marc Talló, por su gestión y soporte durante el proyecto.

BIBLIOGRAFÍA

[1] **JavaScript**. Documentación en MDN Web Docs. Un recurso exhaustivo para aprender sobre JavaScript, incluyendo su sintaxis, funciones, objetos y un amplio conjunto de ejemplos. Recuperado de MDN Web Docs.

[2] **Node.js**. Sitio oficial de Node.js. Plataforma de JavaScript del lado del servidor que permite ejecutar código JavaScript fuera de un navegador, ideal para construir aplicaciones web escalables y eficientes. Recuperado de Node.js.

[3] **Java**. Sitio oficial de Java. Lenguaje de programación orientado a objetos ampliamente utilizado en el desarrollo de aplicaciones de escritorio, móviles y web. Proporciona documentación, recursos de aprendizaje y descargas. Recuperado de Java.

[4] **React**. Documentación de React: Una biblioteca de JavaScript para construir interfaces de usuario. Proporciona información sobre cómo desarrollar componentes reutilizables y gestionar el estado en aplicaciones de una sola página. Recuperado de React.

[5] **Recharts**. Biblioteca de gráficos para React. Facilita la creación de gráficos interactivos y visualizaciones de datos utilizando componentes de React, con una API intuitiva y fácil de usar. Recuperado de Recharts.

[6] **MUI**. La biblioteca de componentes React que siempre has querido. Ofrece una colección de componentes pre-diseñados y personalizables basados en Material Design, que ayudan a crear aplicaciones modernas y atractivas. Recuperado de MUI.

[7] **Material Design**. Guía de diseño de Material Design. Conjunto de principios de diseño y directrices para crear interfaces de usuario coherentes y atractivas en aplicaciones web y móviles, promoviendo la usabilidad y la estética. Recuperado de Material Design.

[8] **MySQL**. Sitio oficial de MySQL. Sistema de gestión de bases de datos relacional de código abierto, ampliamente utilizado para almacenar y gestionar datos en aplicaciones web. Proporciona documentación, recursos de aprendizaje y descargas. Recuperado de MySQL.

[9] **Spring Boot**. Sitio oficial de Spring Boot. Framework para construir aplicaciones Java robustas y escalables, simplificando la configuración y el despliegue de aplicaciones basadas en Spring. Proporciona documentación y herramientas para acelerar el desarrollo de aplicaciones backend. Recuperado de Spring Boot.

[10] **Certidevs**. Tutorial de Spring Boot con ReactJS: Cómo integrar Spring Boot y ReactJS para crear aplicaciones web completas y modernas. Recuperado de Certidevs.

[11] **Postman**. Sitio oficial de Postman. Herramienta de colaboración para la API que permite probar, documentar y monitorear APIs de manera eficiente. Ideal para desarrolladores que buscan optimizar la comunicación entre frontend y backend. Recuperado de Postman.

[12] **Docker**. Sitio oficial de Docker. Plataforma de contenedores que permite empaquetar, distribuir y ejecutar aplicaciones de manera aislada y consistente en cualquier entorno. Recuperado de Docker.

[13] **React Native**. Sitio oficial de React Native. Framework de desarrollo de aplicaciones móviles que permite construir aplicaciones nativas para iOS y Android usando JavaScript y React. Recuperado de React Native.

APÈNDIX

A1. Lista de Tareas Preliminar

1. Definición y diseño de inicio de sesión (RF-01)
    a. Diseñar la interfaz de inicio de sesión para usuarios administradores y camareros.
    b. Implementar la lógica de autenticación.
2. Diseño y desarrollo del módulo de pedidos (RF-02, RF-03)
    a. Crear la funcionalidad para introducir pedidos desde el TPV y la aplicación móvil.
    b. Desarrollar el sistema para enviar pedidos al software.
3. Implementación del módulo de gestión de pedidos (RF-04, RF-05)
    a. Desarrollar la capacidad de consultar y gestionar pedidos en tiempo real.
    b. Implementar la funcionalidad para modificar pedidos existentes.
4. Desarrollo de la gestión de mesa (RF-06)
    a. Crear la vista general de las mesas del bar.
    b. Implementar la asignación de pedidos a mesa específicas.
5. Implementación de funcionalidades de pago (RF-07, RF-08, RF-09)
    a. Desarrollar la opción para dividir el pago entre varios clientes.
    b. Crear la funcionalidad para generar facturas automáticas.
    c. Implementar métodos de pago desde terminales fijas y móviles.
6. Diseño y desarrollo de la base de datos (RF-10, RNF-07)
    a. Diseñar la estructura de la base de datos relacional.
    b. Implementar el registro y almacenamiento de transacciones y ventas.
7. Generación de informes (RF-11)
    a. Desarrollar la funcionalidad para generar informes de ventas diarias, semanales y mensuales.
8. Gestión del inventario (RF-12, RF-13)
    a. Implementar la gestión de inventario, actualizando automáticamente las existencias.
    b. Crear alertas para niveles de inventario bajos.
9. Configuración de precios (RF-14)
    a. Desarrollar la interfaz para que los administradores configuren y modifiquen precios.
10. Diseño de la interfaz de usuario (RF-15, RNF-04, RNF-11)
    a. Crear un diseño intuitivo para el uso por parte de camareros y administradoras.
    b. Optimizar la interfaz para pantallas táctiles.
11. Implementación de opciones de disponibilidad y rendimiento (RNF-01, RNF-12)
    a. Garantizar que el sistema esté disponible con tiempos de respuesta máximos de 2 segundos.
    b. Asegurar que el sistema soporte hasta 500 transacciones por día.
12. Desarrollo de la funcionalidad de funcionamiento offline (RNF-08)
    a. Implementar el modo offline y la sincronización de datos cuando se restablezca la conexión.
13. Compatibilidad con dispositivos (RNF-06)
    a. Asegurar que el sistema sea compatible con distintos sistemas operativos (Windows, Linux, Android, iOS).
14. Implementación de seguridad de datos (RNF-05, RNF-10)
    a. Implementar encriptación de datos durante la transmisión y almacenamiento.
    b. Asegurar el cumplimiento de normativas de protección de datos y fiscales.
15. Gestión de notificaciones (RNF-13)
    a. Implementar la funcionalidad de notificaciones push para alertar al personal de nuevos pedidos.
16. Integración con dispositivos externos (RNF-14)
    a. Desarrollar la integración con dispositivos de pago externos e impresoras de tickets.

A2. Planificacion y costes

Fase 1: Concepción

Actividades:

- Definición del concepto y características del sistema TPV.
- Revisión de requisitos generales.
- Horas Estimadas: 10 horas
- Costo Estimado: 200€ (a 20€/hora)

Fase 2: Planificación

Actividades:

- Definición de requisitos detallados.
- Planificación de tecnologías.
- Estimación del alcance y funcionalidades.
- Horas Estimadas: 10 horas
- Costo Estimado: 200€

Fase 3: Diseño

Actividades:

- Creación de diseños técnicos y de interfaz.
- Diagramas y mockups del sistema.
- Horas Estimadas: 20 horas
- Costo Estimado: 400€

Fase 4: Desarrollo

Actividades:

- Implementación del sistema en iteraciones utilizando Scrum.
- Desarrollo de funcionalidades y pruebas.
- Horas Estimadas: 200 horas
- Costo Estimado: 4000€

Fase 5: Formalización, Entrega y Cierre

Actividades:

- Verificación de requisitos y funcionalidades.
- Entrega oficial del sistema.
- Horas Estimadas: 60 horas
- Costo Estimado: 1200€

Total de Horas y Costos

- Horas Totales: 300 horas
- Costo Total: 6000€

Costos Adicionales

Licencias de Software:

- React: Gratuito
- Node.js: Gratuito
- MySQL: Gratuito
- MUI: Gratuito

Hardware:

- Dispositivos de pago: 20€ (estimado)

Total Estimado del Proyecto

- Costo Total: 6200€

A3. Casos de uso

1. Iniciar Sesión:

- **Actor**: Administrador/Camarero
- **Descripción**: El usuario ingresa sus credenciales para acceder al sistema.

2. Registrar Pedido:

- **Actor**: Camarero
- **Descripción**: El camarero ingresa los detalles del pedido desde el TPV o la aplicación móvil.

3. Modificar Pedido:

- **Actor**: Camarero
- **Descripción**: El camarero puede agregar o eliminar productos antes de cerrar el pedido.

4. Generar Factura:

- **Actor**: Camarero
- **Descripción**: Una vez cerrado el pedido, el sistema genera automáticamente la factura.

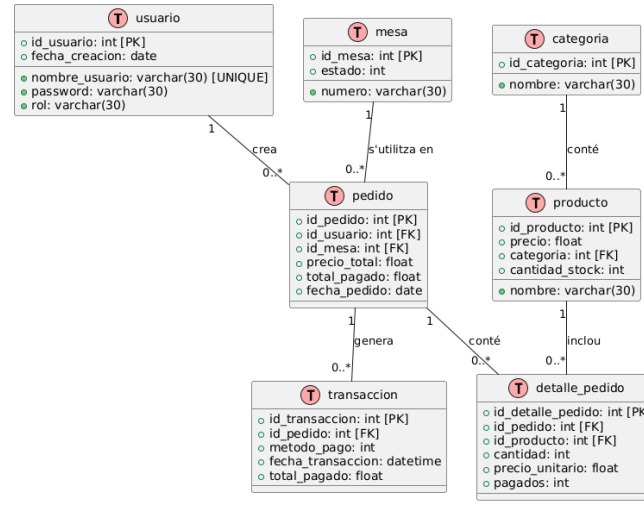
5. Consultar Inventario:

- **Actor**: Administrador
- **Descripción**: El administrador consulta el estado actual del inventario y realiza ajustes si es necesario.

6. Gestionar Pagos:

- **Actor**: Camarero
- **Descripción**: El camarero procesa el pago desde terminales fijas o móviles.

A4. Base de Datos



A5. Imágenes

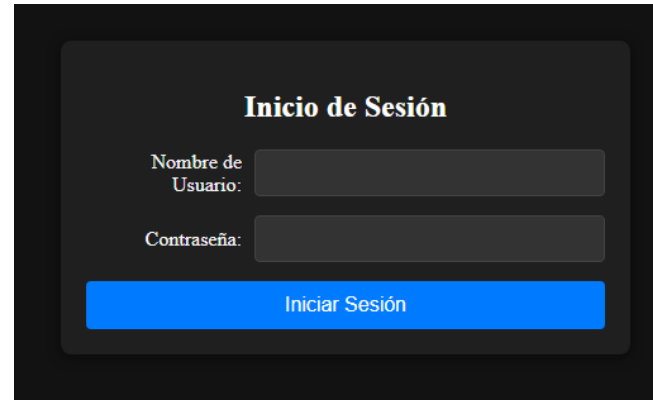


Figura 1 Inicio de sesión

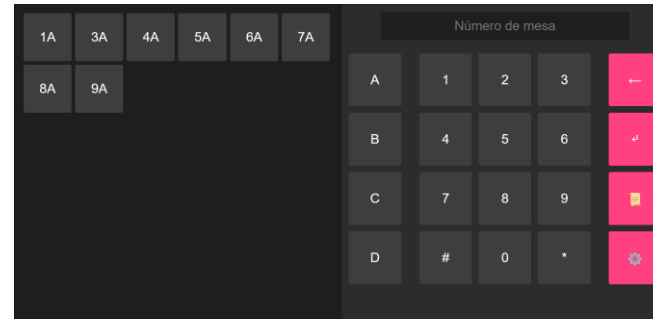


Figura 2 Seleccion de mesa

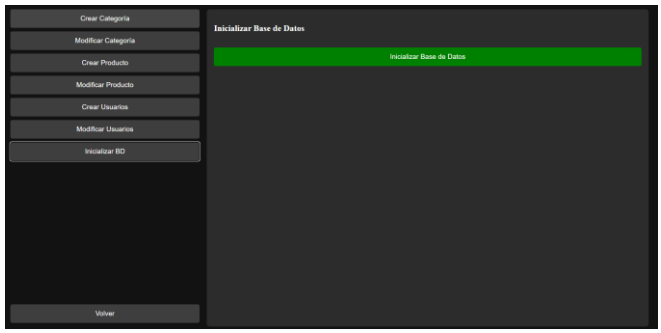


Figura 3 Pagina de Opciones

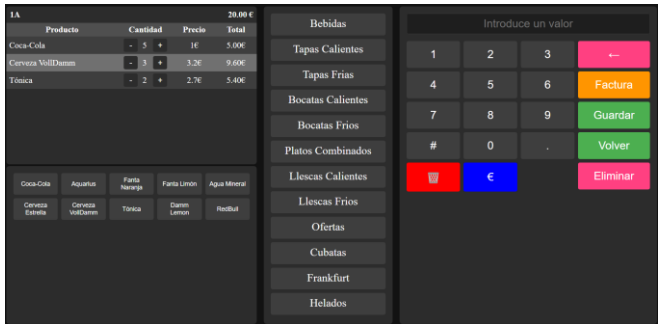


Figura 4 Pagina de mesa EN CURSO

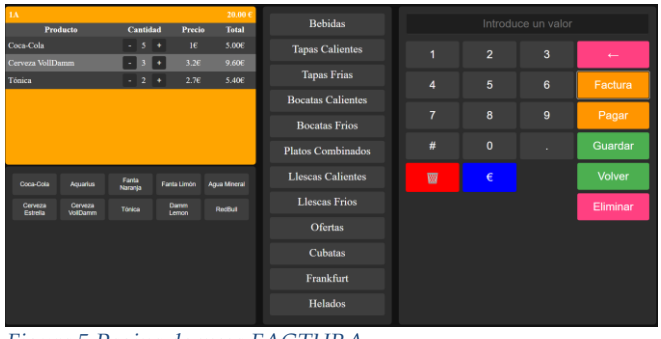


Figura 5 Pagina de mesa FACTURA

Figura 6 Pagina de mesa PARA PAGAR

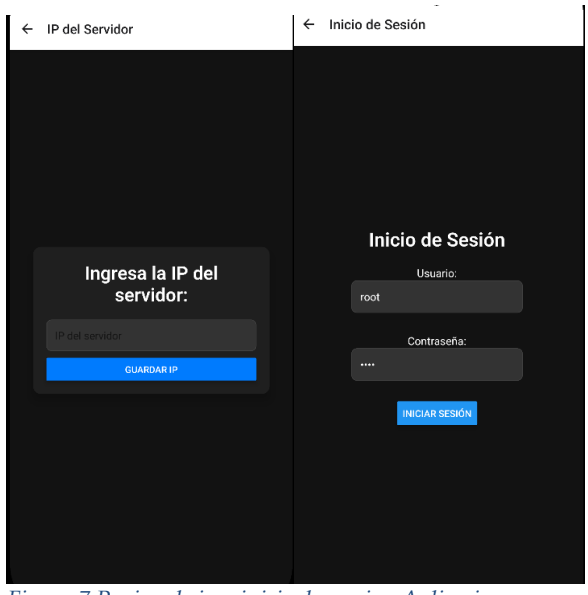


Figura 7 Pagina de ip y inicio de session Aplicacion

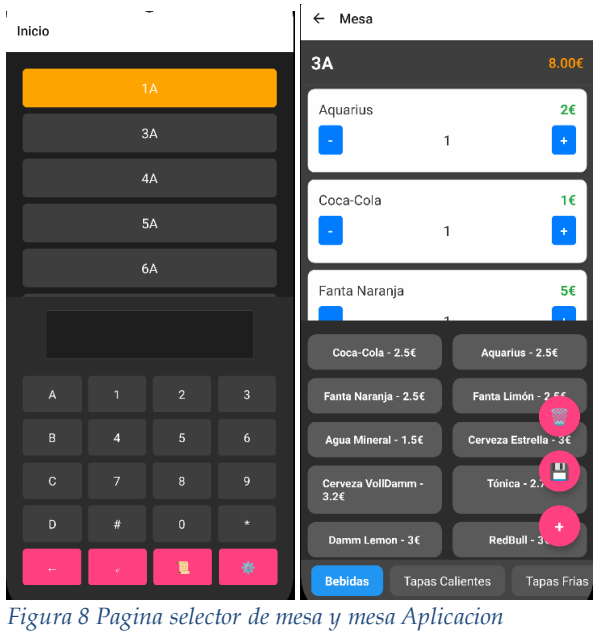


Figura 8 Pagina selector de mesa y mesa Aplicacion