

Manual Aplicative Game Time

Brayan Estiven Fernandez Jimenez

Servicio Nacional de Aprendizaje (SENA)

Análisis y desarrollo de software

Ingeniera: Enzy Zulay Angarita Bermúdez

Girón, Colombia

12 de agosto de 2025

Introducción

Propósito del manual

El presente manual tiene como propósito explicar y dar a conocer el funcionamiento del aplicativo web desarrollado para la gestión de alquiler de consolas de videojuegos. Su objetivo es proporcionar a los usuarios y administradores una guía detallada sobre el uso del sistema, describiendo sus funcionalidades, características y procesos.

Asimismo, este documento especifica las herramientas y requisitos necesarios para la correcta instalación, configuración y puesta en marcha del aplicativo, garantizando así su óptimo rendimiento. De esta manera, se facilita la comprensión del sistema y se asegura que el usuario pueda interactuar con él de forma eficiente y sin contratiempos.

Alcance del software

El aplicativo web está diseñado para gestionar el proceso de alquiler de consolas de videojuegos en un único local. Actualmente, su uso está limitado a entornos locales, ya que no se encuentra desplegado para acceso remoto desde cualquier ubicación.

El sistema contempla dos tipos de usuarios:

- Usuario: Puede realizar reservas para su propio uso, consultar el estado de estas y visualizar los productos consumidos durante el tiempo de alquiler.
- Administrador: Tiene acceso a funciones avanzadas como la creación y gestión de reservas para cualquier usuario registrado, control de inventario de productos, creación, edición e inhabilitación de consolas, administración de usuarios (crear, editar,

inhabilitar), registro y eliminación de productos, así como el cobro y cierre de reservas.

También puede consultar el historial de reservas finalizadas.

El software no incluye integración con pasarelas de pago en línea y está orientado exclusivamente a la administración de un único establecimiento. Su funcionamiento está previsto únicamente para navegadores web en equipos de escritorio.

descripción general del sistema

Objetivos

General.

Desarrollar una aplicación web que permita gestionar de forma automatizada el proceso de alquiler de consolas en un local de videojuegos, incluyendo la creación de reservas, el control del consumo de productos, la gestión de inventario y el registro de un historial de pagos, brindando acceso diferenciado según el rol del usuario.

Específicos.

- Implementar un sistema de reservas que permita a los clientes reservar consolas para sí mismos y al administrador realizar reservas para cualquier usuario registrado.
- Desarrollar un módulo de control de consumo, donde se puedan agregar productos consumidos a cada reserva y estos se descuenten automáticamente del inventario.
- Diseñar un historial de reservas cobradas que almacene el detalle de cada transacción, incluyendo el valor pagado por tiempo de uso y el valor pagado por los productos consumidos.

- Establecer roles de usuario diferenciados, donde el administrador tenga acceso completo a la gestión del sistema y los usuarios solo puedan interactuar con sus propias reservas y consultar información relevante.
- Optimizar la experiencia del usuario a través de una interfaz intuitiva que facilite la interacción tanto para clientes como para el administrador del local.

Requisitos funcionales

- Registrar nuevos clientes en el sistema para permitirles crear una cuenta personal con rol de usuario.
- Iniciar sesión con verificación de roles, de modo que el sistema identifique si el usuario es administrador o cliente y muestre las funcionalidades correspondientes.
- Realizar reservas de consolas por parte de los clientes para uso propio, y por parte del administrador para cualquier cliente registrado
- Controlar el consumo de productos durante una reserva, permitiendo agregarlos desde un listado desplegable que muestra su precio y cantidad disponible, y descontándolos automáticamente del inventario.
- Gestionar consolas y productos por parte del administrador, permitiendo registrar, editar y eliminar estos elementos, mientras que los clientes solo podrán visualizarlos sin posibilidad de modificarlos.

Requisitos no funcionales

- Garantizar una interfaz intuitiva y fácil de usar, tanto para el administrador como para los clientes.

- Permitir el acceso al sistema desde cualquier navegador moderno.
- Hay que asegurar que la interfaz sea clara y comprensible para los usuarios.
- Implementar control de acceso por roles para proteger las funcionalidades y los datos del sistema.
- Evitar conflictos de horario en las reservas y mantener actualizados los inventarios de manera automática.

Requerimientos técnicos del sistema

Requerimientos técnicos de hardware

Servidor de desarrollo.

- Procesador: Intel Core i5 o superior
- Memoria RAM: 8 GB como mínimo (recomendado 16 GB)
- Almacenamiento: 250 GB SSD mínimo.
- Conectividad: Red estable.

Equipo de usuario.

- Procesador: Intel Core i3 o superior
- Memoria RAM: 4 GB como mínimo
- Almacenamiento libre: 2 GB para archivos temporales y navegación
- Conectividad: Conexión a internet

Requerimientos técnicos de software

Equipo de desarrollo.

- Sistema Operativo: Windows 11 (64 bits)
- Servidor de Aplicaciones: Apache Tomcat 9
- Entorno de Desarrollo Integrado (IDE): NetBeans 21
- Kit de Desarrollo de Java (JDK): Versión 21
- Gestor de Base de Datos: MySQL 8
- Herramientas Adicionales: Navegador web actualizado (Google Chrome, Mozilla Firefox o Microsoft Edge)

Usuario.

- Sistema Operativo: Windows 10 o superior (64 bits)
- Navegador Web Compatible: Google Chrome, Mozilla Firefox o Microsoft Edge (última versión)
- Servidor de Aplicaciones: Apache Tomcat 9 (en el servidor donde se aloje la aplicación)
- Gestor de Base de Datos: MySQL 8 (en el servidor de base de datos)

Instalación de herramientas y configuración.**Gestor de base de datos MySQL**

1. En el navegador, busca MySQL Workbench. (Silva, 2024)
2. Seleccione Dowload Now. (Silva, 2024)
3. Por defecto, el sitio de MySQL ya identifica cuál sistema estás utilizando. Serás redirigido a una página de inicio de sesión o registro, que es opcional. Si no deseas hacer

ninguno de los dos, haz clic en 'No, thanks, just start my download'; esto iniciará la descarga automáticamente. Luego, da permiso para que se descargue completamente. (Silva, 2024)

4. Como queremos configurar todo el entorno de desarrollo, elegimos la opción Completa (Full), aquí tenemos todos los requisitos necesarios para un entorno de desarrollo completo. (Silva, 2024)

5. Haz clic en Ejecutar para que se descarguen todos los productos seleccionados. Esto puede llevar un tiempo, dependiendo de los productos seleccionados y la velocidad de tu conexión a Internet. (Silva, 2024)

6. Descarga completa. Haz clic en el botón Next para continuar. (Silva, 2024)

7. Visión general de la configuración. Haz clic en el botón Next para configurar el servidor de la base de datos MySQL. (Silva, 2024)

8. Para realizar la configuración del servidor MySQL, elige el tipo de configuración y el puerto de MySQL (3006 por defecto) y haz clic en el botón Next para continuar. (Silva, 2024)

9. Elige una contraseña para la cuenta root. Anota la contraseña y mantenla segura si estás instalando el servidor de base de datos MySQL en un servidor de producción. Si deseas agregar un usuario adicional a MySQL, puedes hacerlo en este paso. (Silva, 2024)

10. Selecciona los detalles del servicio de Windows, incluyendo el Nombre del servicio de Windows y el tipo de cuenta, luego pulsa en el botón Next para continuar (Silva, 2024)}

11. Haz clic en Execute para aplicar todas las configuraciones y finalmente en Finish. (Silva, 2024)

Laragon o XAMPP

No es necesario instalar ni configurar Laragon o XAMPP, ya que MySQL se instaló anteriormente de forma global en el sistema operativo. Por esta razón, la base de datos puede ser gestionada directamente sin el uso de estos entornos de servidor locales.

NetBeans

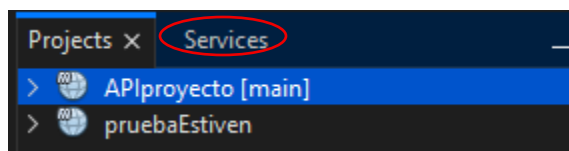
1. Visite el [sitio web oficial](#) para descargar el instalador de NetBeans. (Ultahost, 2024)
2. Una vez finalizada la descarga, localice el archivo de instalación (normalmente un .exe archivo). Haga doble clic en él para iniciar el proceso. (Ultahost, 2024)
3. Lea el acuerdo de licencia de NetBeans. Para aceptar los términos, haga clic en "Siguiente" o "Aceptar" para continuar. (Ultahost, 2024)
4. El instalador sugerirá un directorio de instalación predeterminado. Puede conservarlo o seleccionar una ubicación diferente. Se recomienda instalar NetBeans en un directorio sin espacios ni caracteres especiales para evitar posibles problemas. (Ultahost, 2024)
5. Se le presentará una lista de componentes que pueden instalarse con NetBeans. Seleccione los componentes que necesite (selecciónelos todos). (Ultahost, 2024)
6. Comenzará el proceso de instalación. Puede tardar unos minutos, dependiendo de la velocidad de su sistema y de los componentes seleccionados. Espere a que finalice la instalación. (Ultahost, 2024)
7. Una vez completada la instalación, haga clic en “Finalizar” para salir del instalador. (Ultahost, 2024).

Servidor apache Tomcat 9

1. Se deberá visitar el [sitio oficial](#) para realizar la descarga del apache tomcat 9.
2. Se deberá seleccionar el paquete de descarga que en archivo zip para Windows, ya sea de 32 bits o de 64.
3. Una vez que se haya descargado el archivo .zip se deberá descomprimir en cualquier directorio de su preferencia.
4. Una vez que el archivo se encuentre descomprimido se deberá dirigir al netbeans IDE y hacer click en la pestaña services.

Figura 1

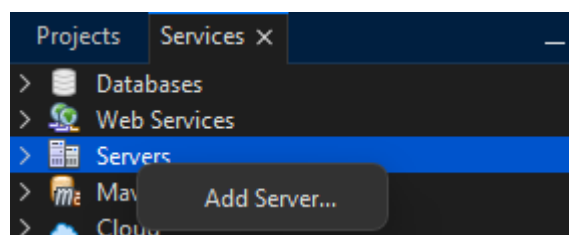
Pestaña services



5. Cuando ya se encuentre ubicado en la pestaña services deberá hacer click derecho sobre la opción servers y click en “add server”.

Figura 2

Add Server

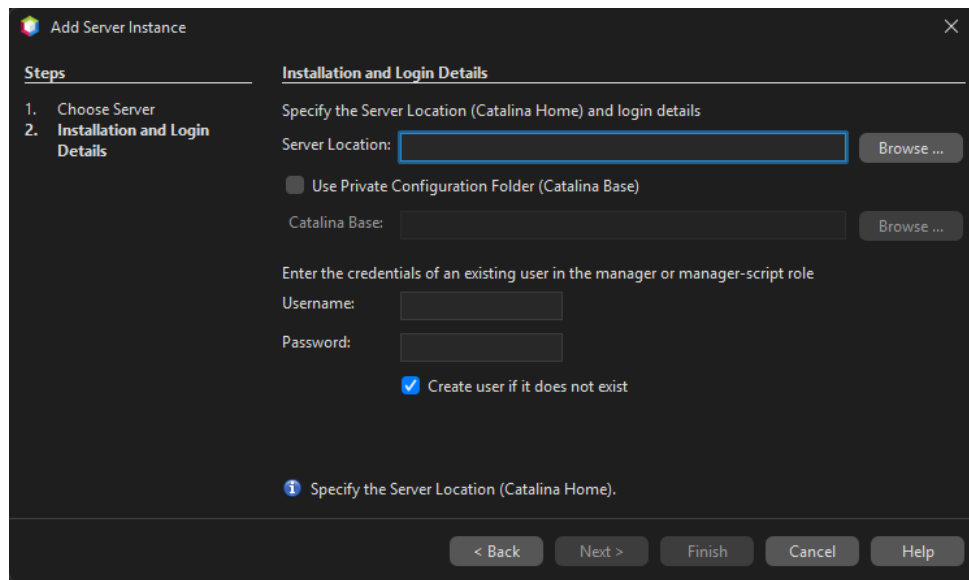


6. Cuando haga clic sobre la anterior opción se mostrara un listado de servidores los cuales se pueden agregar, deberá seleccionar la opción “Apache Tomcat or TomEE”.

7. Se abriría un menú en el cual deberá buscar la ruta en la cual descomprimió el archivo zip anteriormente, luego de esto deberá ingresar un usuario y una contraseña para acceder al servidor, una vez hecho esto se deberá hacer clic en el botón “finish”.

Figura 3

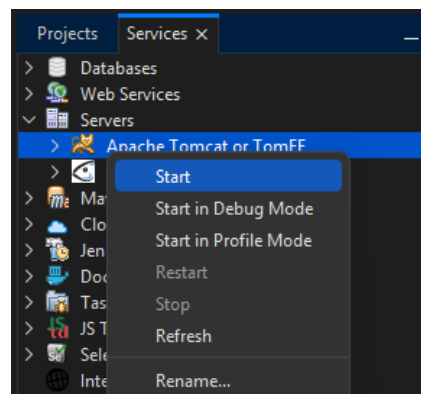
Agrega servidor



8. Por último, únicamente se deberá hacer clic sobre el servidor que se creó y start, con esto nuestro servidor quedaría funcionando correctamente.

Figura 4

Iniciar Servidor



Arquitectura de software y base de datos

Figura 5

Diagrama de casos de uso administrador

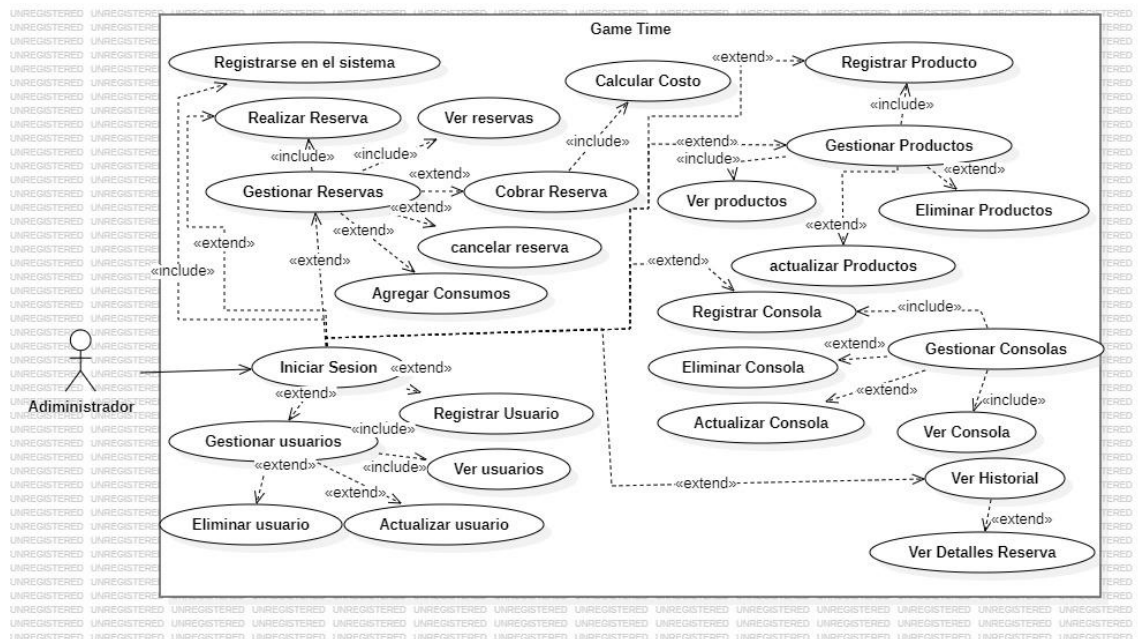


Figura 6

Diagrama de casos de uso usuario

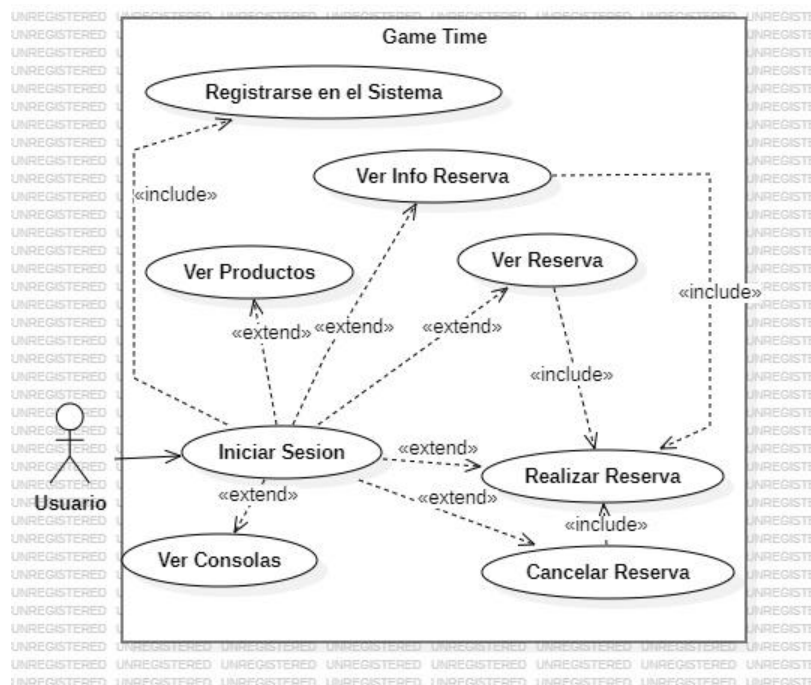
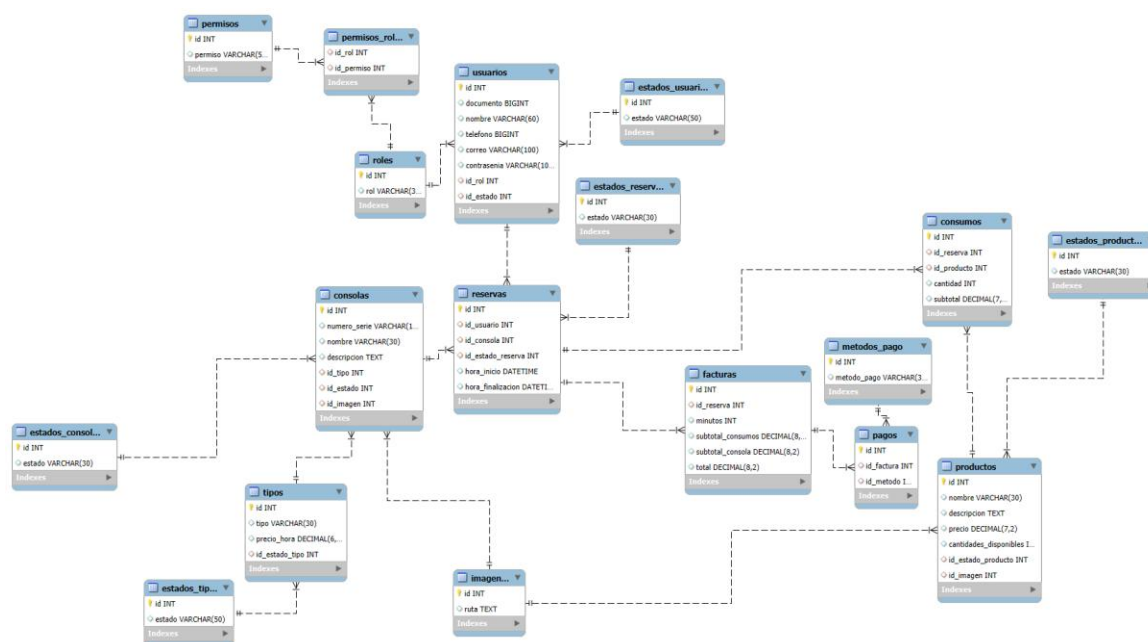


Figura 7*Estructura de tablas*

La base de datos del sistema está diseñada para gestionar de forma integral el proceso de alquiler de consolas de videojuegos, incluyendo la administración de usuarios, consolas, productos, reservas, facturación y pagos. A continuación, se describen sus principales tablas:

- **roles**: Almacena los tipos de roles disponibles en el sistema (administrador, usuario).
- **permisos**: Contiene la lista de permisos disponibles.
- **permisos_rol**: Relaciona roles con permisos específicos.
- **estados_usuarios**: Registra los posibles estados de los usuarios (activo, inactivo).
- **usuarios**: Almacena la información de cada usuario, incluyendo sus datos personales, credenciales, rol y estado.

- **imagenes:** Guarda las rutas de las imágenes asociadas a consolas y productos.
- **estados_tipos:** Define el estado de los tipos de consola (activo, inactivo).
- **tipos:** Contiene los tipos de consolas y sus precios por hora.
- **estados_consolas:** Define los estados de las consolas (disponible, inactivo).
- **consolas:** Registra cada consola disponible para alquiler, con sus características, tipo, estado e imagen.
- **estados_productos:** Almacena los estados posibles de los productos (disponible, agotado, inactivo).
- **productos:** Contiene los datos de los productos disponibles para consumo, incluyendo precio, cantidad en inventario, estado e imagen.
- **estados_reservas:** Define el estado de una reserva (pendiente, en proceso, terminada, cobrada).
- **reservas:** Registra las reservas realizadas por los usuarios, vinculando consola, usuario, estado y tiempos de uso.
- **facturas:** Almacena la información de facturación de cada reserva, incluyendo subtotales, total y minutos de uso.
- **consumos:** Registra los productos consumidos en el transcurso de una reserva, con sus cantidades y subtotales.
- **metodos_pago:** Lista los métodos de pago aceptados (transferencia, Nequi, efectivo).
- **pagos:** Registra los pagos realizados, vinculando factura y método de pago.

La estructura de la base de datos está normalizada para optimizar la integridad referencial y minimizar la redundancia de datos. Cada entidad clave está relacionada mediante llaves foráneas, lo que garantiza coherencia entre registros y facilita la trazabilidad de la información.

Script de creación de la base de datos

```
CREATE DATABASE bd_proyecto_brayan;
```

```
USE bd_proyecto_brayan;
```

```
-- ROLES Y PERMISOS
```

```
CREATE TABLE roles (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    rol VARCHAR(30)
```

```
);
```

```
INSERT INTO roles (rol) VALUES ('administrador');
```

```
INSERT INTO roles (rol) VALUES ('usuario');
```

```
CREATE TABLE permisos (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
permiso VARCHAR(50)
```

```
);
```

```
INSERT INTO permisos(permiso) VALUES ('ver');
```

```
CREATE TABLE permisos_roles (
```

```
    id_rol INT,
```

```
    id_permiso INT,
```

```
    FOREIGN KEY (id_rol) REFERENCES roles(id) ON DELETE SET NULL,
```

```
    FOREIGN KEY (id_permiso) REFERENCES permisos(id) ON DELETE SET NULL
```

```
);
```

```
INSERT INTO permisos_roles (id_rol, id_permiso) VALUES (2, 1);
```

```
CREATE TABLE estados_usuarios(
```

```
    id int auto_increment primary key,
```

```
    estado varchar(50)
```

```
);
```

```
INSERT INTO estados_usuarios(estado) VALUES ('Activo');
```

```
INSERT INTO estados_usuarios(estado) VALUES ('Inactivo');
```

```
-- USUARIOS
```

```
CREATE TABLE usuarios (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    documento BIGINT UNIQUE,
```

```
    nombre VARCHAR(60),
```

```
    telefono BIGINT,
```

```
    correo VARCHAR(100) UNIQUE,
```

```
    contrasenia VARCHAR(100),
```

```
    id_rol INT,
```

```
    id_estado INT,
```

```
    FOREIGN KEY (id_rol) REFERENCES roles(id) ON DELETE SET NULL,
```

```
    FOREIGN KEY (id_estado) REFERENCES estados_usuarios(id) ON DELETE SET NULL
```

```
);
```


-- IMÁGENES

CREATE TABLE imagenes (

id INT AUTO_INCREMENT PRIMARY KEY,

ruta TEXT

);

CREATE TABLE estados_tipos(

id INT AUTO_INCREMENT PRIMARY KEY,

estado VARCHAR(50)

);

INSERT INTO estados_tipos(estado) VALUES ('Activo');

INSERT INTO estados_tipos(estado) VALUES ('Inactivo');

-- TIPOS DE CONSOLA

CREATE TABLE tipos (

id INT AUTO_INCREMENT NOT NULL PRIMARY KEY,

tipo VARCHAR(30),

```
precio_hora DECIMAL(6,2),  
  
id_estado_tipo INT,  
  
FOREIGN KEY (id_estado_tipo) REFERENCES estados_tipos(id)  
  
);
```

-- ESTADOS DE CONSOLAS

```
CREATE TABLE estados_consolas (  
  
    id INT AUTO_INCREMENT PRIMARY KEY,  
  
    estado VARCHAR(30)  
  
);
```

```
INSERT INTO estados_consolas(estado) VALUES ('disponible');
```

```
INSERT INTO estados_consolas(estado) VALUES ('Inactivo');
```

-- CONSOLAS

```
CREATE TABLE consolas (  
  
    id INT AUTO_INCREMENT PRIMARY KEY,  
  
    numero_serie VARCHAR(10) UNIQUE,
```

```
nombre VARCHAR(30),

descripcion TEXT,

id_tipo INT,

id_estado INT,

id_imagen INT,

FOREIGN KEY (id_tipo) REFERENCES tipos(id) ON DELETE SET NULL,

FOREIGN KEY (id_estado) REFERENCES estados_consolas(id) ON DELETE SET NULL,

FOREIGN KEY (id_imagen) REFERENCES imagenes(id) ON DELETE SET NULL

);


-- ESTADOS DE PRODUCTOS

CREATE TABLE estados_productos (

    id INT AUTO_INCREMENT PRIMARY KEY,

    estado VARCHAR(30)

);


INSERT INTO estados_productos(estado) VALUES ('disponible');

INSERT INTO estados_productos(estado) VALUES ('agotado');
```

```
INSERT INTO estados_productos(estado) VALUES ('inactivo');
```

```
-- PRODUCTOS
```

```
CREATE TABLE productos (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    nombre VARCHAR(30),
```

```
    descripcion TEXT,
```

```
    precio DECIMAL(7,2),
```

```
    cantidades_disponibles INT,
```

```
    id_estado_producto INT,
```

```
    id_imagen INT,
```

```
    FOREIGN KEY (id_imagen) REFERENCES imagenes(id) ON DELETE SET NULL,
```

```
    FOREIGN KEY (id_estado_producto) REFERENCES estados_productos(id) ON DELETE  
SET NULL
```

```
);
```

```
-- ESTADOS DE RESERVAS
```

```
CREATE TABLE estados_reservas (
```

```
id INT AUTO_INCREMENT PRIMARY KEY,  
  
estado VARCHAR(30)  
  
);
```

```
INSERT INTO estados_reservas (estado) VALUES  
  
('pendiente'), ('en proceso'), ('terminada'), ('cobrada');
```

```
-- RESERVAS
```

```
CREATE TABLE reservas (  
  
id INT AUTO_INCREMENT PRIMARY KEY,  
  
id_usuario INT,  
  
id_consola INT,  
  
id_estado_reserva INT,  
  
hora_inicio DATETIME,  
  
hora_finalizacion DATETIME,  
  
FOREIGN KEY (id_usuario) REFERENCES usuarios(id) ON DELETE SET NULL,  
  
FOREIGN KEY (id_consola) REFERENCES consolas(id) ON DELETE SET NULL,
```

```
FOREIGN KEY (id_estado_reserva) REFERENCES estados_reservas(id) ON DELETE SET  
NULL  
  
);
```

```
-- FACTURAS (con subtotales en DECIMAL(8,2))
```

```
CREATE TABLE facturas (  
  
    id INT AUTO_INCREMENT PRIMARY KEY,  
  
    id_reserva INT UNIQUE,  
  
    minutos INT,  
  
    subtotal_consumos DECIMAL(8,2) DEFAULT 0,  
  
    subtotal_consola DECIMAL(8,2) DEFAULT 0,  
  
    total DECIMAL(8,2) DEFAULT 0,  
  
    FOREIGN KEY (id_reserva) REFERENCES reservas(id) ON DELETE SET NULL  
  
);
```

```
-- CONSUMOS TEMPORALES (NO CAMBIAR)
```

```
CREATE TABLE consumos (  
  
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
id_reserva INT,

id_producto INT,

cantidad INT,

subtotal DECIMAL(7,2),

FOREIGN KEY (id_reserva) REFERENCES reservas(id) ON DELETE SET NULL,

FOREIGN KEY (id_producto) REFERENCES productos(id) ON DELETE SET NULL

);


-- MÉTODOS DE PAGO

CREATE TABLE metodos_pago (

    id INT AUTO_INCREMENT PRIMARY KEY,

    metodo_pago VARCHAR(30)

);


INSERT INTO metodos_pago(metodo_pago) VALUES ('Tranferencia'), ('Nequi'), ('Efectivo');


-- PAGOS

CREATE TABLE pagos (
```

```

id INT AUTO_INCREMENT PRIMARY KEY,

id_factura INT,

id_metodo INT,

FOREIGN KEY (id_factura) REFERENCES facturas(id) ON DELETE SET NULL,

FOREIGN KEY (id_metodo) REFERENCES metodos_pago(id) ON DELETE SET NULL

);

-- USUARIO DE PRUEBA

INSERT INTO usuarios(documento,nombre,telefono,correo,contrasenia,id_rol,id_estado)

VALUES

(1096512824,'Brayan

Fernandez',3112114081,'brayan@gmail.com','$2a$10$XWDD07M527ov2C1.R/wYnedQuxhK2f

5ACmUTysVXAE0Az752TKQqq',1,1); -- Brayan123.

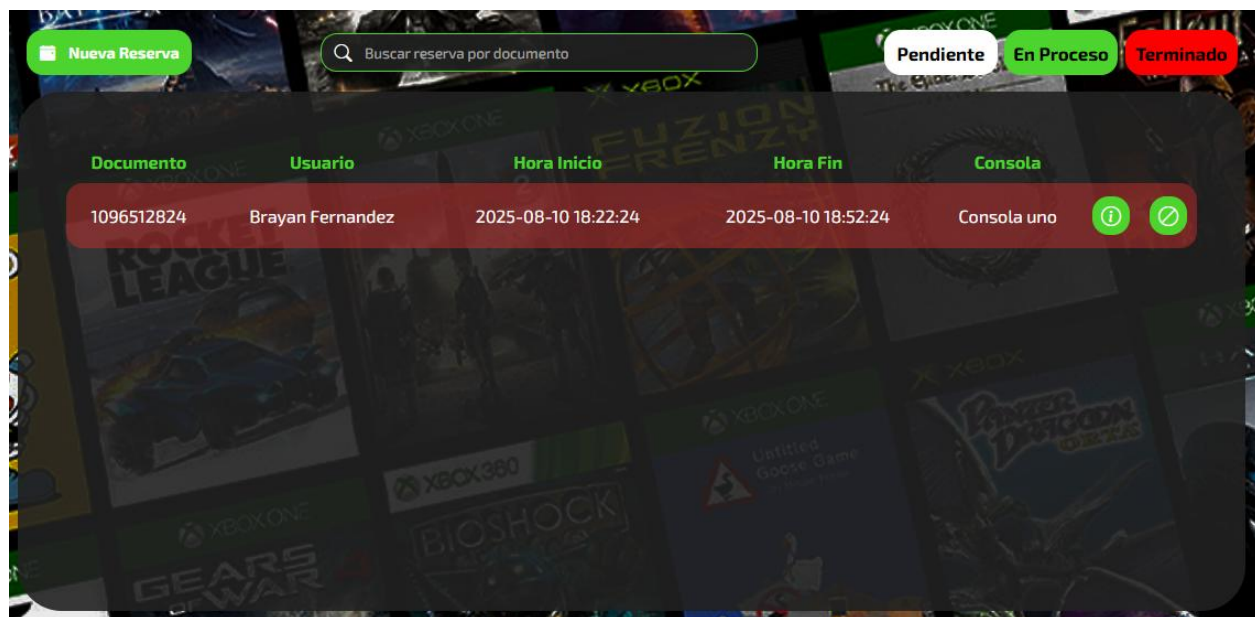
```


Diseño del Software

Descripción de los módulos del aplicativo

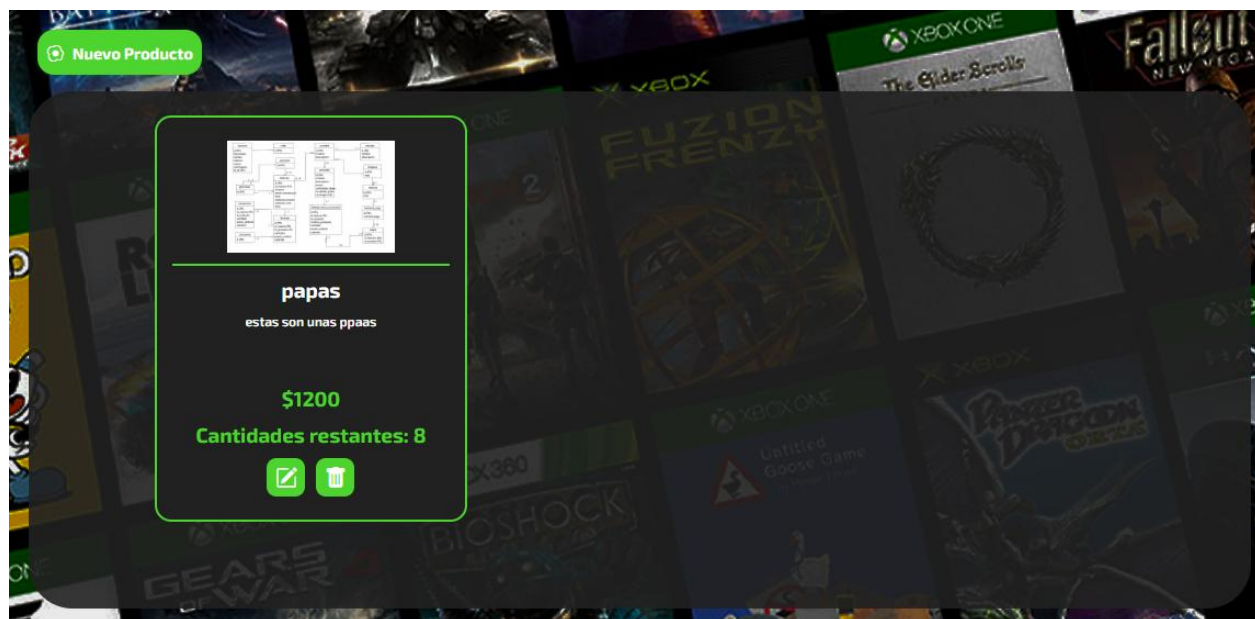
Figura 8

Modulo reservas



En este módulo, el administrador podrá visualizar todas las reservas registradas en el sistema, organizadas desde la más próxima a iniciar hasta la más lejana. El usuario, por su parte, únicamente podrá ver las reservas que le correspondan.

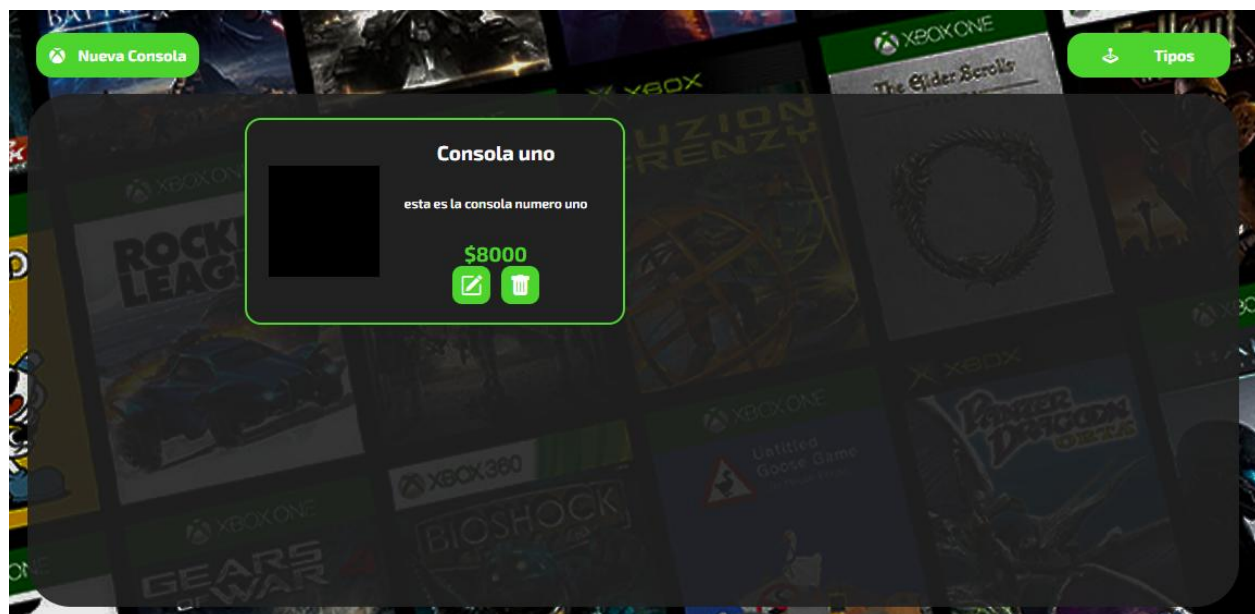
Desde esta sección, el usuario tendrá la opción de acceder al apartado para generar una nueva reserva, consultar la información detallada de una reserva existente o cancelarla, siempre que falte al menos una hora para su inicio.

Figura 9*Modulo productos*

En este módulo, el administrador podrá visualizar todos los productos disponibles junto con información relevante como su nombre, descripción, cantidad en inventario y precio.

Además, tendrá acceso al formulario para crear nuevos productos, así como para editar o eliminar los existentes. Los productos que se encuentren en estado agotado se resaltarán con un borde rojo para una identificación rápida.

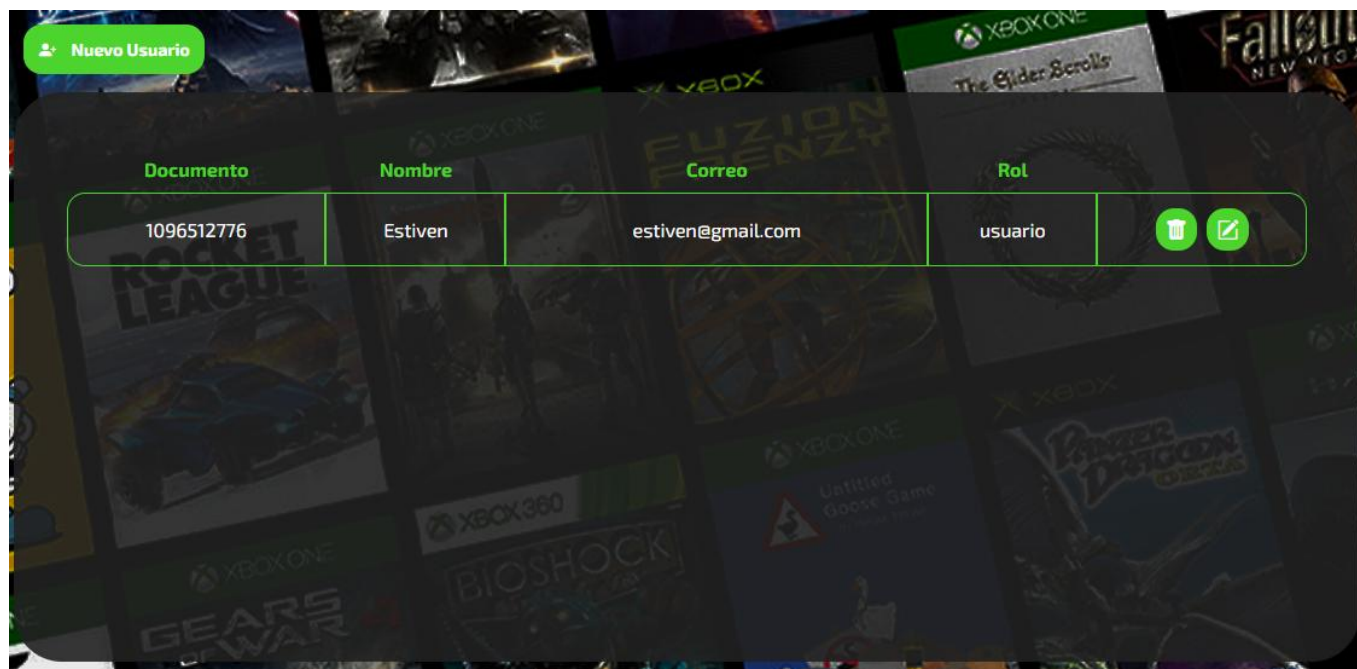
El usuario, por su parte, solo podrá consultar la información de los productos, sin posibilidad de realizar modificaciones.

Figura 10*Modulo consolas*

En este módulo, el administrador del local podrá visualizar todas las consolas registradas. Tendrá la opción de editar su información o desactivarlas (cambio de estado en lugar de eliminación definitiva). Las consolas que se encuentren desactivadas se mostrarán con un borde rojo para su fácil identificación y podrán ser reactivadas en cualquier momento.

Desde esta sección, el administrador también podrá acceder al formulario para crear una nueva consola o registrar un nuevo tipo de consola.

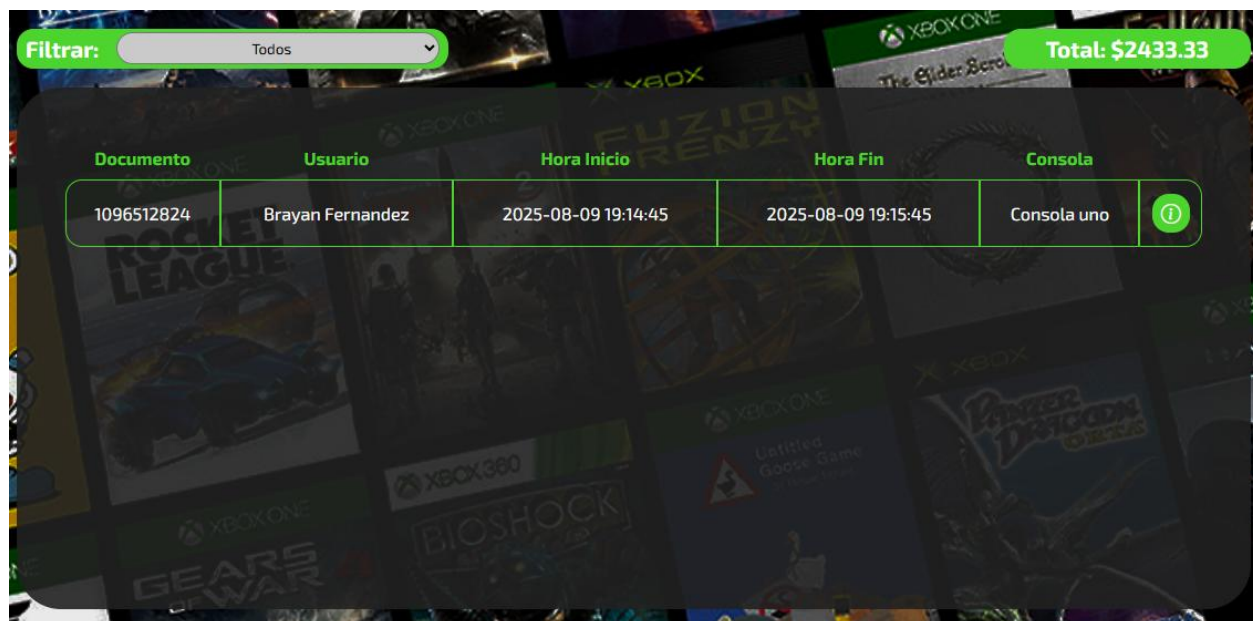
El usuario únicamente podrá visualizar las consolas disponibles, sin posibilidad de realizar modificaciones.

Figura 11*Módulo de usuarios*

Este módulo será de uso exclusivo para el administrador. Desde aquí podrá visualizar todos los usuarios registrados en el sistema, así como editar su información o eliminarlos cuando sea necesario.

Además, contará con acceso a un formulario que le permitirá registrar nuevos usuarios en la plataforma.

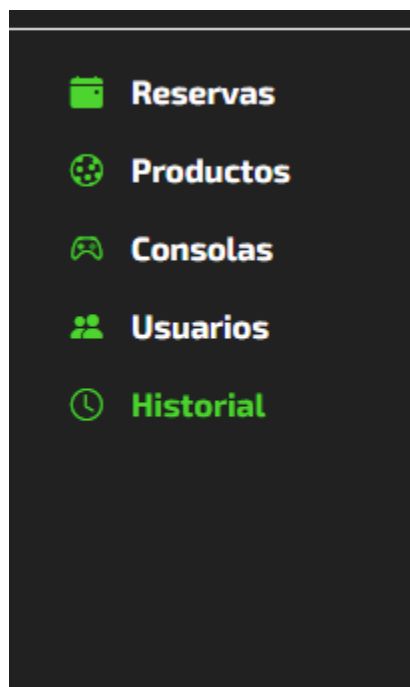
Figura 12
Modulo historial



Documento	Usuario	Hora Inicio	Hora Fin	Consola
1096512824	Brayan Fernandez	2025-08-09 19:14:45	2025-08-09 19:15:45	Consola uno

Este módulo será visible únicamente para el administrador. En él se mostrarán todas las reservas cuyo estado sea “cobrado”. El administrador podrá filtrar las reservas por método de pago y visualizar el total recaudado por cada uno.

Además, desde esta sección también podrá acceder a la información detallada de cobro de cada reserva.

Figura 13*Sidebar*

El sidebar permitirá al usuario navegar entre los diferentes módulos disponibles en el sistema. En caso de que el usuario no tenga acceso a un módulo específico, este no será mostrado en el menú.

descripción de los botones que se utilizaron

- botón “Nueva Reserva”: Este botón redirige al usuario al módulo donde se encuentran las consolas disponibles, permitiéndole realizar una nueva reserva.
- botón “Reservas”: Este botón abre un calendario en el que se muestran los horarios disponibles para reservar la consola seleccionada.
- Buscador: Esta herramienta permite al administrador ingresar el número de documento de un usuario para visualizar todas las reservas asociadas a dicho usuario.

- botón con icono de “Cancelar”: Este botón permite al usuario cancelar la reserva a la que está asociado. La cancelación solo podrá realizarse si falta, como mínimo, una hora para el inicio de la reserva.
- botón con icono de “información”: Este botón permite al usuario visualizar los detalles de la reserva asociada, incluyendo los productos consumidos y la información básica de la misma.
- botón “Nuevo Producto”: Al hacer clic en este botón, el usuario será redirigido a un formulario donde deberá ingresar los datos básicos para registrar un nuevo producto.
- botón con icono de “Editar”: Este botón permite modificar la información del elemento al que está asociado. Al hacer clic, el usuario será redirigido a un formulario que contiene los datos previamente registrados, donde podrá realizar los cambios necesarios.
- botón con icono de “Eliminar”: Este botón permite eliminar el elemento al que está asociado.
- botón “Nueva Consola”: Este botón redirige al usuario a un formulario donde deberá ingresar todos los datos necesarios para registrar una nueva consola.
- botón “Tipos”: Este botón redirige al usuario a un módulo donde se encuentran todos los tipos registrados. Desde allí podrá crear un nuevo tipo, editar uno existente o eliminarlo si así lo desea
- botón “Nuevo Usuario”: Este botón redirige al usuario a un formulario donde deberá ingresar todos los datos personales necesarios para realizar el registro del nuevo usuario.
- Select “Filtrar”: Esta opción permite al usuario seleccionar el método de pago por el cual desea filtrar todas las reservas que ya han sido pagadas. Una vez elegido, se mostrarán

únicamente las reservas pagadas mediante ese método y el total de dinero recibido por dicho medio de pago.

- botón “Agregar producto”: Este botón abre un formulario en el que el usuario puede seleccionar el producto que desea añadir a la reserva y especificar la cantidad a agregar.
- botón “Cobrar”: Este botón muestra la factura correspondiente a la reserva, junto con un selector que permite elegir el método de pago para realizar el cobro. Una vez registrado el pago, la reserva dejará de aparecer en el módulo de reservas y pasará al módulo de historial.

Interacción entre componentes

En el aplicativo web, los componentes interactúan entre sí para facilitar la gestión de reservas, ventas y control de consolas.

- Botones de acción: Al hacer clic en botones como “Reservar”, “Agregar producto” el sistema procesa la acción solicitada, actualiza la base de datos y refleja los cambios en la interfaz de forma inmediata.
- Sidebar: Permite moverse entre las secciones principales cargando dinámicamente la información correspondiente.
- Formularios de registro y edición: Reciben información del usuario (por ejemplo, datos de una nueva reserva o producto), validan los campos y envían la información al servidor para su almacenamiento.

- Tablas dinámicas: Muestran la información actualizada de reservas, ventas o inventario.
- Modales emergentes: Se activan desde botones o enlaces, permitiendo realizar acciones rápidas como agregar consumos a una reserva.

Interfaces del sistema

API

El sistema implementa una API de tipo REST desarrollada en JAX-RS. Esta API permite la comunicación entre el cliente (frontend) y el servidor (backend), ofreciendo una interfaz para realizar todas las operaciones necesarias, como gestión de consolas, reservas, consumos y facturación.

La API utiliza el protocolo HTTP para el intercambio de información, empleando métodos como GET, POST, PUT y DELETE, y maneja los datos en formato JSON para entrada y salida. Su diseño sigue principios de modularidad, agrupando los endpoints por recursos (por ejemplo: /consolas, /reservas, /facturas) para facilitar su uso y mantenimiento.

Formatos de entrada y salida

La comunicación entre cliente y servidor se realiza exclusivamente en formato JSON.

- Entrada: los datos enviados al servidor deben cumplir con la estructura definida para cada recurso, incluyendo todos los campos obligatorios según la operación (por ejemplo, para crear una reserva se envían campos como idConsola, horaInicio, etc.).

- Salida: las respuestas de la API también se devuelven en formato JSON, conteniendo la información solicitada o los resultados de la operación realizada.
- Manejo de errores: en caso de error, el servidor responde con un código de estado HTTP adecuado (por ejemplo, 400 Bad Request, 404 Not Found, 500 Internal Server Error) y un objeto JSON que incluye un mensaje descriptivo del error.

Autenticación y autorización

La API utiliza el mecanismo de JSON Web Tokens (JWT) para autenticar a los usuarios y verificar que tengan una sesión activa.

- Autenticación: el usuario debe iniciar sesión enviando sus credenciales al endpoint correspondiente. Si las credenciales son válidas, el servidor genera y devuelve un token JWT que el cliente deberá incluir en el encabezado Authorization de cada solicitud.
- Autorización: actualmente, el token emitido permite el acceso a los recursos y operaciones disponibles para un usuario autenticado.

Mensajes de ayuda y error

Mensajes de ayuda

se utilizan principalmente en los formularios de la aplicación para guiar al usuario en la correcta introducción de datos.

Por ejemplo, en el campo de correo electrónico, a medida que el usuario escribe se valida el formato y, si es incorrecto, aparece un mensaje debajo del campo indicando que el correo no es

válido. Una vez que el dato cumple con el formato requerido, el mensaje desaparece automáticamente. Este mismo enfoque se aplica en otros campos de entrada, adaptando el mensaje de ayuda según el tipo de dato solicitado.

Mensajes de error

cuando ocurre un error en el backend o no se cumplen las validaciones establecidas, se muestra un mensaje al usuario utilizando la librería SweetAlert. Estos mensajes indican de forma clara el tipo de error sucedido, como errores de validación, problemas de conexión o fallos en la ejecución de la operación solicitada.

Conclusión

El desarrollo del aplicativo web para la gestión del alquiler de consolas en un local de videojuegos representa una solución eficiente y práctica para optimizar los procesos operativos del negocio. A través de la implementación de módulos que permiten la creación y control de reservas, la gestión del inventario de productos y consolas, y el registro detallado de pagos y consumos, se logra automatizar tareas que anteriormente se realizaban de forma manual.

El sistema ofrece un acceso diferenciado basado en roles, lo que garantiza seguridad y un adecuado control de permisos entre usuarios y administradores. Además, la integración de tecnologías modernas como JAX-RS para la construcción de la API REST y JWT para la autenticación, asegura una comunicación robusta y segura entre el frontend y backend.

La interfaz amigable y los mensajes de ayuda en tiempo real facilitan la interacción del usuario, mejorando la experiencia y reduciendo errores durante el uso del sistema. Por último, la

estructura modular y escalable del aplicativo permite futuras ampliaciones y mantenimiento, garantizando la adaptabilidad del sistema a las necesidades cambiantes del negocio.

En resumen, este proyecto contribuye significativamente a la digitalización y mejora de la administración en el local de videojuegos, ofreciendo una herramienta confiable y eficiente para sus usuarios y administradores.

Referencias

ChatGPT. (12 de agosto de 2025). *CharGPT*. Obtenido de <https://chatgpt.com/share/689be990-e9e8-8004-9edc-388f48cb3fde>

ChatGPT. (12 de agosto de 2025). *ChatGpt*. Obtenido de <https://chatgpt.com/share/689be88f-9324-8004-91b9-a8e55e0a5a1f>

Silva, I. (01 de 07 de 2024). *alura LATAM*. Obtenido de <https://www.aluracursos.com/blog/descargar-mysql-serve>

Ultahost. (31 de julio de 2024). *Ultahost*. Obtenido de <https://ultahost.com/knowledge-base/install-netbeans-windows/#:~:text=Instalaci%C3%B3n%20de%20NetBeans%20en%20Windows,especial es%20para%20evitar%20posibles%20problemas.>

Glosario

- API (Interfaz de Programación de Aplicaciones): Conjunto de reglas y protocolos que permiten la comunicación entre diferentes componentes de software, como el frontend y backend de una aplicación.
- Autenticación: Proceso de verificar la identidad de un usuario para permitirle acceso al sistema.

- **JWT (JSON Web Token):** Método estándar para transmitir información segura entre partes como un token firmado digitalmente, usado para autenticar usuarios.
- **Backend:** Parte del sistema que se encarga de la lógica de negocio, procesamiento de datos y comunicación con la base de datos, generalmente ejecutándose en un servidor.
- **Frontend:** Interfaz gráfica de usuario del sistema, la parte con la que interactúan los usuarios desde su navegador.
- **Reserva:** Acción de apartar una consola para uso en un horario determinado dentro del local.
- **Roles de Usuario:** Categorías que definen los permisos y nivel de acceso que tiene cada usuario dentro del sistema (por ejemplo, administrador o cliente).
- **SweetAlert:** Librería de JavaScript utilizada para mostrar mensajes emergentes estilizados en la interfaz, usada para notificaciones y alertas.

Contacto del Equipo

- **Encargado:** Brayan Estiven Fernandez Jimenez
- **Teléfono:** 3112114081
- **Correo:** brayanfernandezjimenez18@gmail.com