**CLASIFICACION DE REQUISITOS – ACTIVIDAD CASO TREN**

**Danilo Andrés Montezuma Ibarra**

**Tomas Benavides Calderón**

**Ingeniero, Miguel Tovar Bastidas**

**Documentación de Requisitos Del Caso Del Tren**

**Proyecto: Sistema de Venta de Boletos en la Estación Plaza Sesamo**

Requisitos Funcionales:

1. **Registro de Usuarios:**
   * Los usuarios pueden crear cuentas proporcionando su nombre, información de la tarjeta de crédito (nombre del propietario, marca, número, fecha de vencimiento y CVV).
   * El sistema verificará la validez de los datos ingresados y evitará el registro duplicado con la misma información de tarjeta.
2. **Inicio de Sesión:**
   * Los usuarios registrados pueden iniciar sesión proporcionando su nombre.
   * El sistema autenticará al usuario y permitirá el acceso a las funciones específicas.
3. **Visualización de Información de la Cuenta:**
   * Los usuarios autenticados pueden ver su información de cuenta, incluyendo nombre, información de la tarjeta y saldo disponible.
4. **Compra de Boletos:**
   * Los usuarios autenticados pueden comprar boletos de diferentes ubicaciones disponibles.
   * El sistema mostrará los detalles de los boletos, como ubicación, precio y fecha de salida.
   * Los usuarios pueden seleccionar boletos y realizar la compra.
   * El saldo de la cuenta se actualizará y se registrará la compra en los detalles de la cuenta del usuario.
5. **Cierre de Sesión:**
   * Los usuarios pueden cerrar sesión en cualquier momento para proteger su información y salir del sistema.

Requisitos No Funcionales:

1. **Usabilidad:**
   * La interfaz de usuario debe ser intuitiva y fácil de usar para personas con diferentes niveles de experiencia.
   * Las instrucciones y mensajes de error deben ser claros y comprensibles.
2. **Seguridad:**
   * La información de la tarjeta de crédito debe almacenarse de manera segura y encriptada.
   * El acceso a las cuentas debe estar protegido por autenticación y autorización adecuadas.
3. **Rendimiento:**
   * El sistema debe ser capaz de manejar múltiples usuarios al mismo tiempo sin experimentar demoras significativas.
   * Las consultas de la base de datos deben ser eficientes para asegurar una respuesta rápida.
4. **Escalabilidad:**
   * El sistema debe diseñarse para poder manejar un número creciente de usuarios y transacciones sin degradar el rendimiento.
5. **Mantenibilidad:**
   * El código fuente debe seguir buenas prácticas de programación y estar documentado para facilitar futuras actualizaciones y correcciones.
6. **Compatibilidad:**
   * La aplicación debe ser compatible con diferentes navegadores web y dispositivos, incluyendo computadoras de escritorio y dispositivos móviles.
7. **Disponibilidad:**
   * El sistema debe estar disponible y accesible para los usuarios en todo momento, con un tiempo de inactividad planificado mínimo para actualizaciones.
8. **Documentación:**
   * Se debe proporcionar documentación detallada para la instalación, configuración y uso del sistema, así como para el mantenimiento y resolución de problemas.
9. **Interfaz de Usuario:**
   * La interfaz de usuario debe ser atractiva visualmente y tener un diseño coherente y organizado.
10. **Clase Ticket:**
    * Representa un boleto con atributos como ubicación, precio y fecha de salida.
    * Genera un ID aleatorio para cada boleto al inicializarse.
11. **Lista de Tickets:**
    * Una lista que contiene objetos de la clase **Ticket** con información sobre diferentes boletos disponibles en ubicaciones reales.
12. **Clase Client:**
    * Representa un cliente con atributos como nombre, información de la tarjeta, saldo y boletos comprados.
    * La clase incluye un método **display\_info** que muestra la información relevante del cliente.
13. **Clase Saldo:**
    * Representa el saldo disponible en la cuenta del cliente.
14. **Función save\_clients(clients):**
    * Guarda la información de los clientes en un archivo JSON.
    * Itera a través de la lista de clientes y almacena sus nombres, información de tarjeta y boletos comprados.
15. **Función load\_clients():**
    * Carga la información de los clientes desde el archivo JSON.
    * Crea objetos de la clase **Client** a partir de los datos almacenados en el archivo.
16. **Función main():**
    * La función principal que controla el flujo del programa.
    * Ofrece opciones para crear una cuenta, iniciar sesión, ver información de la cuenta, comprar boletos y cerrar sesión.
17. **Función buy\_ticket(client, clients, tickets):**
    * Permite a los clientes comprar boletos.
    * Muestra los boletos disponibles y permite al cliente seleccionar y comprar un boleto si tienen saldo suficiente.
18. **Bloque if \_\_name\_\_ == "\_\_main\_\_":**
    * Inicia la ejecución del programa si se ejecuta como un archivo independiente.
    * Llama a la función **main()** para iniciar el flujo del programa.

Estas son las partes fundamentales del código que definen las clases, funciones y flujos de control para el sistema de venta de boletos en la Estación Plaza Sesamo. Cada parte cumple un papel importante en la funcionalidad y operación del sistema, permitiendo a los usuarios crear cuentas, comprar boletos, ver información y más. Es fundamental entender cómo interactúan estas partes para comprender completamente el funcionamiento del programa.

1. **Creación de cuenta:**
   * El usuario puede crear una cuenta proporcionando su nombre, información de la tarjeta (propietario, marca, número, vencimiento y CVV).
   * La información se valida para asegurarse de que cumple con ciertos criterios.
   * Se crea un objeto **Client** con la información proporcionada y se agrega a la lista de clientes.
   * Si es uno de los primeros 10 clientes, se otorga un saldo inicial de $500.
2. **Iniciar sesión:**
   * El usuario ingresa su nombre y el sistema verifica si existe un cliente con ese nombre.
   * Si el cliente se encuentra, se le presenta un menú con opciones.
3. **Ver información de la cuenta:**
   * El cliente puede ver su información personal, incluido su nombre, información de tarjeta, boletos comprados y saldo.
4. **Comprar boletos:**
   * El cliente puede ver los boletos disponibles con su información, como ubicación, precio y fecha de salida.
   * Selecciona un boleto para comprar y se verifica si ya lo ha comprado previamente.
   * Se verifica si tiene suficiente saldo para comprar el boleto y, en caso afirmativo, se deduce el precio del saldo.
   * El boleto se agrega a la lista de boletos comprados por el cliente.
5. **Mirar horarios de salida:**
   * El cliente puede ver los horarios de salida de los boletos disponibles, mostrando la ubicación y la fecha de salida.
6. **Cerrar sesión:**
   * El cliente puede cerrar sesión y salir del programa.
   * La información actualizada se guarda en el archivo JSON.
7. **Funciones auxiliares:**
   * Las funciones **save\_clients** y **load\_clients** permiten guardar y cargar la información de los clientes en un archivo JSON.
   * Estas funciones aseguran que los datos se conserven entre sesiones.

En resumen, el programa permite a los usuarios crear cuentas, iniciar sesión, comprar boletos, ver información de cuentas y horarios de salida, y luego cerrar sesión. La estructura de clases y funciones asegura que la funcionalidad se mantenga organizada y fácil de entender, al tiempo que garantiza que los datos de los clientes y los boletos se almacenen de manera persistente en un archivo.