Samuel Ramirez

Julián Ramos

ISIS 1226

Secuencias

En este documento vamos a mostrar las que a nuestra consideración son las secuencias mas importantes dentro de nuestra solución al enunciado del proyecto 1.

1. Compra de Tiquete (Online o en Taquilla)

Participantes involucrados: Cliente, VentaOnline / VentaTaquilla, GeneradorId, Tiquete

Secuencia del proceso:

- El Cliente solicita la compra de un tiquete a través de una modalidad: VentaOnline o VentaTaquilla.
- La clase de venta recibe la solicitud y genera un identificador único usando GeneradorId.
- Con el ID generado, se crea una instancia del tipo de tiquete solicitado:
    - TiqueteRegular, TiqueteTemporada, TiqueteIndividual o FastPass.
- El tiquete se almacena con su estado inicial como “activo”.
- La instancia del cliente recibe el tiquete mediante el método agregarTiquete() o similar.
- La venta también guarda los datos del cliente y el medio de pago.
- Se confirma la compra y el tiquete queda disponible para su uso.

Nota: Todos los tiquetes son instancias de clases concretas que heredan de la clase abstracta Tiquete.

2. Validación de Acceso a una Atracción

Participantes involucrados: Cliente, Tiquete, Atracción, OperadorAtracciones

Secuencia del proceso:

- El Cliente presenta un tiquete al llegar a una atracción.
- Se invoca el método estaActivo() sobre el tiquete.
- Si está activo, se llama a permiteAcceso(cliente):

    - Verifica las restricciones de la atracción (altura, peso, edad, temporada, enfermedades).
    - Verifica el nivel de exclusividad exigido por la atracción frente al del tiquete.
- Si el acceso está permitido, se llama al método usar(), que marca el tiquete como "usado".
- La atracción verifica si tiene asignado al menos un operador capacitado (nivelRiesgoCapacitado) si es una atracción mecánica.
- Si todo es correcto, el cliente accede a la atracción.

3. Asignación de Turnos a Empleados

Participantes involucrados: Administrador, Empleado (subclase de Persona), Turno

Secuencia del proceso:

- El Administrador crea una instancia de la clase Turno, definiendo:
    - Lugar del turno
    - Hora de apertura y cierre
    - Tarea asignada
- Selecciona un empleado (por ejemplo: OperadorAtracciones, Cocinero o ServicioGeneral).
- El sistema verifica si el tipo de empleado es compatible con la tarea del turno.
    - Por ejemplo, solo operadores pueden ser asignados a atracciones mecánicas.
- Si hay compatibilidad, se asigna el turno al empleado mediante un método como asignarTurno().
- El empleado recibe el turno en su lista interna.
- El sistema confirma al Administrador que el turno fue asignado correctamente.

4. Registro de Nueva Atracción

Participantes involucrados: Administrador, Atraccion (Mecánica o Cultural), Temporada, OperadorAtracciones

Secuencia del proceso:

- El Administrador registra una nueva atracción, instanciando:

    - AtraccionMecanica (si tiene restricciones físicas y nivel de riesgo)
    - o AtraccionCultural (si solo aplica restricción de edad mínima).
- Se definen los atributos de la atracción: nombre, ubicación, cupo máximo, exclusividad, restricciones, etc.
- Se asigna una instancia de Temporada, indicando la fecha de inicio y fin de vigencia.
- El Administrador asigna uno o más operadores a la atracción mediante un método como asignarOperador().
- El sistema valida si los operadores asignados están capacitados (verificando nivelRiesgoCapacitado en caso de atracciones mecánicas).
- Si todo está correcto, la atracción queda registrada y lista para operar.

5. Persistencia de Datos (Guardar y Cargar)

Participantes involucrados: Administrador, ArchivoPlano, Clases del sistema (Cliente, Empleado, Tiquete, Atracción, Turno)

Guardado de datos:

- El Administrador solicita guardar el estado del sistema.
- Se invoca la clase ArchivoPlano, que recorre todas las estructuras internas:
    - Lista de clientes
    - Lista de empleados
    - Lista de tiquetes
    - Lista de atracciones
    - Lista de turnos
- Se formatea la información como texto separado por comas (CSV).
- Los datos se escriben en archivos externos .csv por tipo de entidad.
- El sistema confirma que los datos han sido guardados correctamente.

Carga de datos:

- Al iniciar el sistema, el Administrador solicita cargar los datos persistidos.
- ArchivoPlano lee línea por línea cada archivo .csv.
- Por cada registro, se instancia el objeto correspondiente (Cliente, Tiquete, etc.).
- Las instancias reconstruidas se agregan a las listas del sistema en memoria.

- El sistema confirma que la restauración fue exitosa y el estado anterior ha sido recuperado.

6. Creación de Empleados por el Administrador

Participantes involucrados: Administrador, Subclase de Persona (Empleado), ArchivoPlano

Secuencia del proceso:

- El Administrador inicia el proceso de creación de un nuevo empleado.
- Selecciona el tipo de empleado (Cajero, Cocinero, Operador de Atracciones, etc.).
- Se ingresan los datos: nombre, login, contraseña, fecha de nacimiento, altura, peso, enfermedades o discapacidades.
- El sistema crea una instancia del tipo de empleado correspondiente, todos derivados de la clase Persona.
- El nuevo empleado se añade a la estructura de datos en memoria.
- El Administrador llama al módulo ArchivoPlano para guardar los datos en el archivo .csv correspondiente.
- El sistema confirma que el empleado fue creado exitosamente.

7. Uso de un FastPass para Ingreso Prioritario

Participantes involucrados: Cliente, FastPass (subclase de Tiquete), Atracción, OperadorAtracciones

Secuencia del proceso (versión precisa):

- El Cliente presenta su FastPass para una atracción específica en una fecha determinada.
- Se invoca el método estaActivo() desde el objeto FastPass (heredado de Tiquete), para verificar que la fecha actual coincide con la establecida.
- Luego, se llama a permiteAcceso(cliente) (también heredado), que verifica:
    - Si el cliente cumple con las restricciones físicas y médicas.
    - Si la atracción está disponible en esa temporada.
    - Si el nivel de exclusividad del tiquete permite el acceso.
- Si todo es válido, se invoca usar(), marcando el FastPass como usado.
- El sistema permite el ingreso del cliente con prioridad.
- El operador asignado verifica su capacitación para dicha atracción.

- Se registra el acceso exitoso del cliente.

8. Verificación de Restricciones Físicas y Médicas (desde permiteAcceso())

Participantes involucrados: Cliente, Tiquete (de cualquier tipo), Atracción (Mecánica o Cultural)

Secuencia del proceso:

- El Cliente presenta un tiquete (por ejemplo, Regular o Individual).
- El método permiteAcceso(cliente) es invocado sobre el tiquete.
- Este método accede a la lista de restricciones definidas en la atracción (altura, peso, edad mínima, enfermedades/discapacidades).
- Se verifica cada condición:
    - Altura del cliente ≥ altura mínima requerida (si aplica).
    - Peso del cliente ≤ peso máximo permitido (si aplica).
    - Edad del cliente ≥ edad mínima (en atracciones culturales).
    - El cliente no presenta ninguna enfermedad o discapacidad listada como prohibitiva.
- Si alguna condición falla, el método retorna false y se deniega el acceso.
- Si todas se cumplen, se retorna true y luego se puede invocar usar() para marcar el tiquete como usado.

9. Asignación de operadores a atracciones mecánicas según nivel de riesgo

Participantes involucrados: Administrador, AtraccionMecanica, OperadorAtracciones

Justificación:
Este proceso es esencial para garantizar la seguridad operativa de las atracciones mecánicas. Solo operadores capacitados pueden ser asignados a atracciones de cierto nivel de riesgo (bajo, medio o alto).

Secuencia del proceso:

- El Administrador selecciona una AtraccionMecanica registrada en el sistema.
- El sistema identifica el nivel de riesgo de la atracción: bajo, medio o alto.
- El Administrador elige un operador disponible (instancia de OperadorAtracciones).
- Se accede al atributo nivelRiesgoCapacitado del operador.

- Se verifica si el operador está capacitado para ese nivel de riesgo (el valor debe ser igual o superior al de la atracción).
- Si la capacitación es suficiente, el operador se asigna mediante el método asignarOperador(operador).
- Si no es suficiente, el sistema impide la asignación y emite una advertencia.
- La atracción mantiene una lista actualizada de operadores asignados, necesaria para su funcionamiento.

10. Validación de acceso considerando temporada y nivel de exclusividad del tiquete

Participantes involucrados: Cliente, Tiquete, Atraccion, Temporada

Justificación:
Este proceso es crítico para regular el uso de tiquetes según la fecha y privilegios del cliente, evitando accesos indebidos a atracciones exclusivas o fuera de temporada.

Secuencia del proceso:

- El Cliente intenta acceder a una atracción.
- Se invoca el método permiteAcceso(cliente) sobre el tiquete (cualquiera de sus subtipos).
- Dentro del método:
    - Se consulta si la atracción está activa en la fecha actual a través del método Temporada.estaVigente().
    - Se compara el nivelExclusividad requerido por la atracción con el del tiquete:
        - Familiar, Oro o Diamante.
        - Si el nivel del tiquete es inferior al de la atracción, el acceso es denegado.
- Solo si la temporada está vigente y el nivel de exclusividad del tiquete es suficiente, se devuelve true.
- Luego se llama a usar() sobre el tiquete para marcarlo como usado.
- Si alguna condición falla, el sistema impide el ingreso y notifica al cliente.