Grupo 2 : Proyecto final

“Sistema de reserva de asientos autobús”

Integrantes: Cristóbal Henríquez

Jocabed Aileen López Flores

Ignacio Soto

Profesor: Geoffrey Hecht

Referente: Angie Ramírez

Fecha: 16 de diciembre, 2024

1 Enunciado

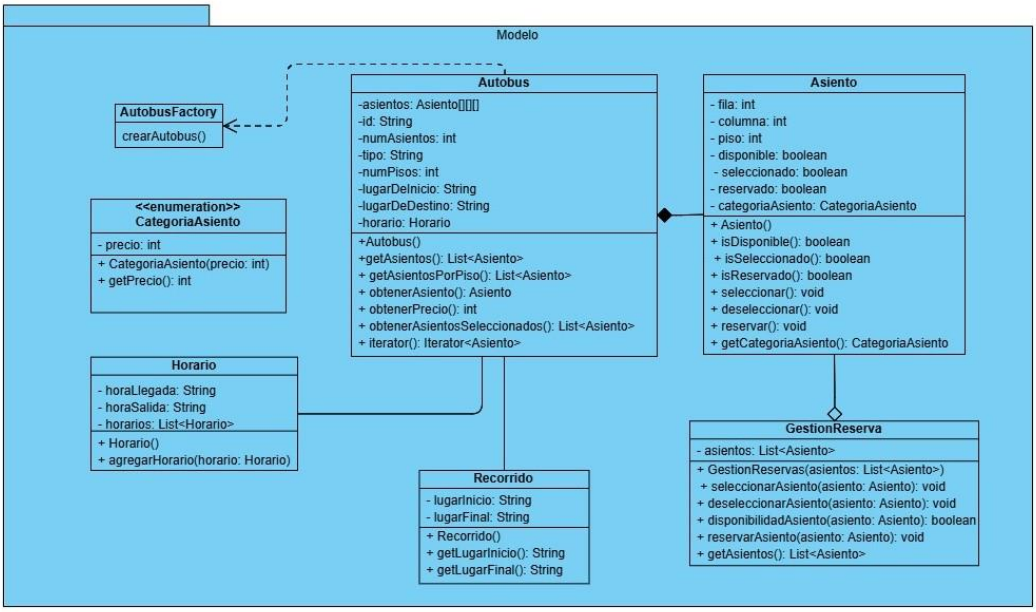
La interfaz que hemos desarrollado es parte de un sistema de reserva de asientos de autobús. Este sistema permite gestionar los recorridos de autobuses que viajan desde Concepción hacia algunas ciudades de Chile, específicamente a Santiago, Chillán y Los Ángeles. El sistema está diseñado para ser flexible, de modo que, con el tiempo, se pueden agregar nuevos destinos y rutas sin necesidad de realizar modificaciones extensivas en la infraestructura del sistema.

El sistema permite a los usuarios seleccionar su asiento en función de varios factores, como el tipo de autobús (económico o premium), el tipo de asiento (semi-cama o salón-cama) y el piso del autobús. Además, los usuarios pueden acceder a la interfaz para visualizar los recorridos disponibles, elegir sus asientos y realizar reservas.

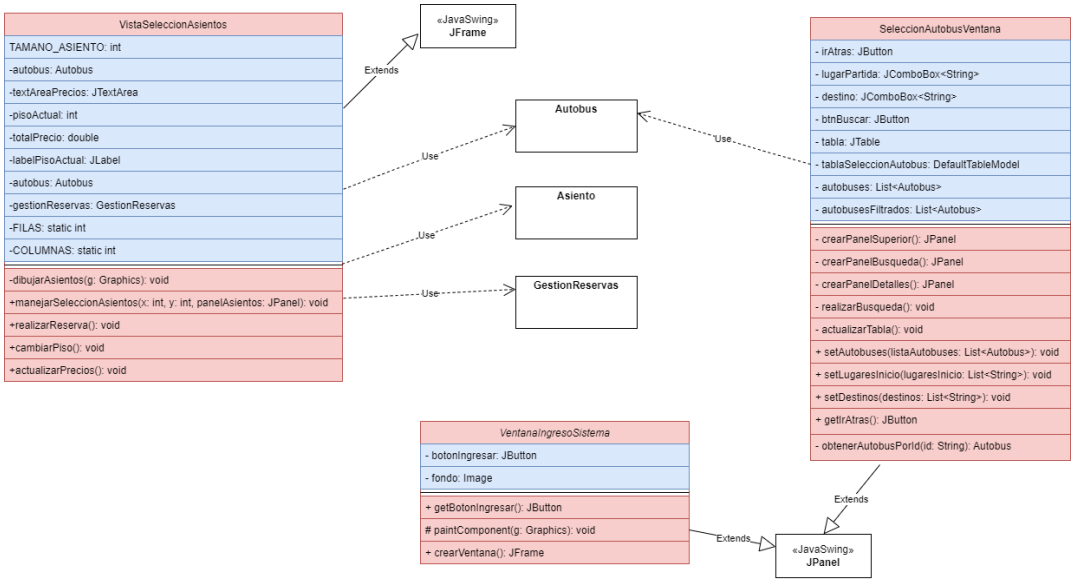
Una vez se efectúe la reserva, el sistema debería mostrar los asientos que están ocupados para que los clientes puedan escoger entre los asientos que están disponibles y no existan problemas al momento de realizar el viaje con reserva de asientos repetidos.

2 Diagrama UML de clases

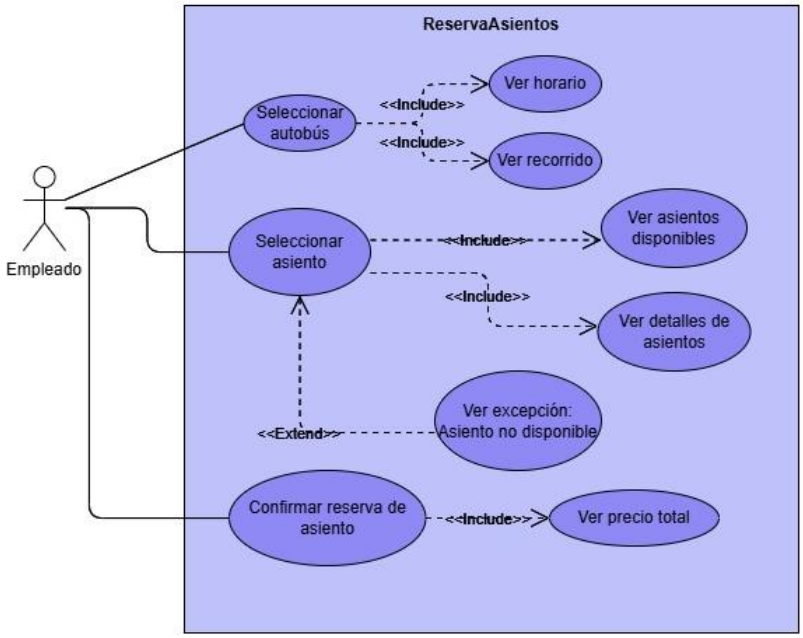
- Package Modelo



- Package Vista



3 Diagrama UML de casos de uso

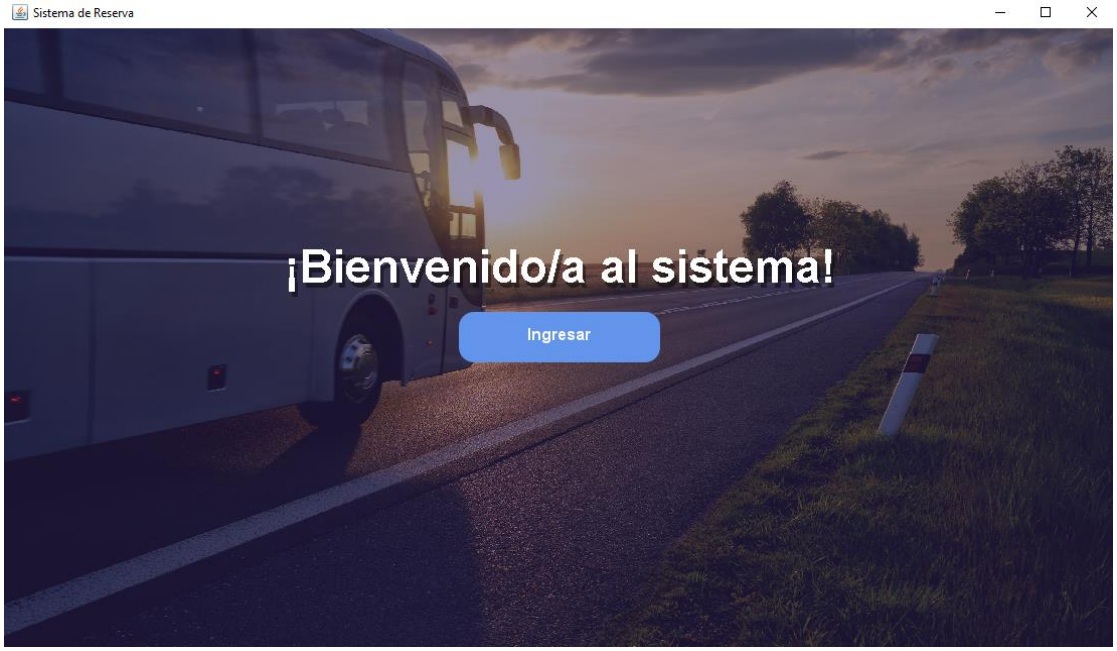


4 Patrones de diseño utilizados

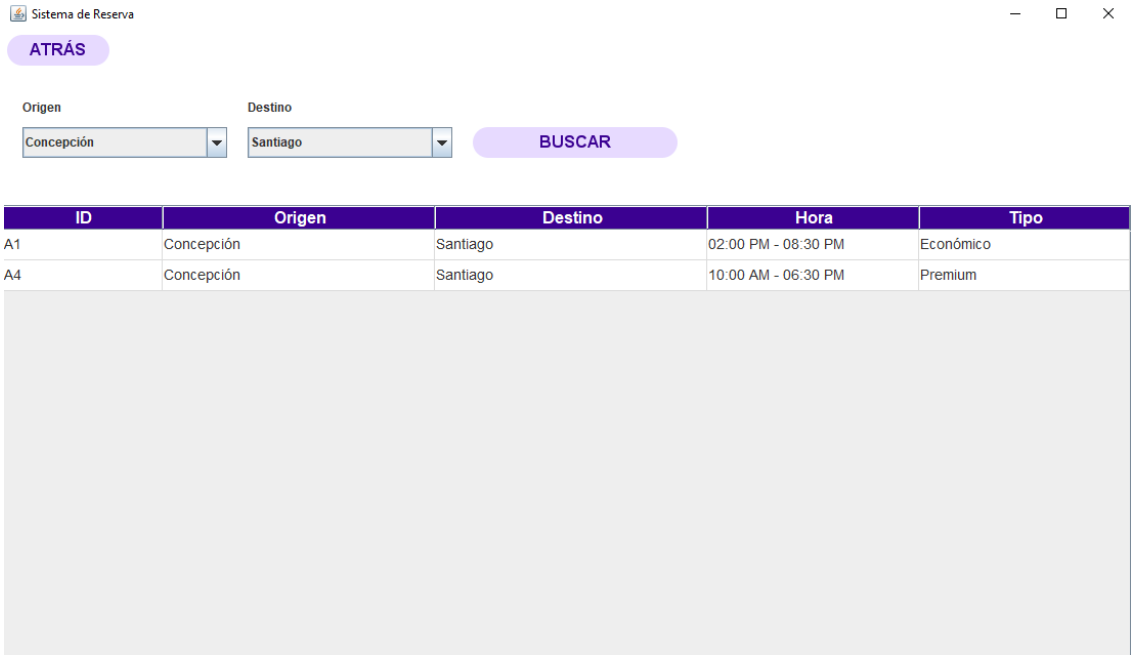
- Factory: Se utilizó este patrón de diseño dentro de la clase “Autobus” para encapsular la lógica de creación de instancias de autobuses. Este patrón permite crear objetos específicos según el tipo de autobús (por ejemplo, autobuses económicos o premium) sin exponer la lógica de creación al resto del sistema. Así, se puede instanciar un autobús adecuado para cada tipo, asegurando que sus características (como el tipo de asientos, los pisos disponibles y otras configuraciones) se ajusten a los requisitos establecidos para cada clase de autobús.
- Iterator: Este patrón de diseño se utiliza en la clase “Autobus” para proporcionar una forma ordenada de recorrer los asientos del autobús. La implementación del patrón Iterator permite acceder a los asientos del autobús de manera secuencial, sin exponer los detalles internos de su almacenamiento

5 Interfaz

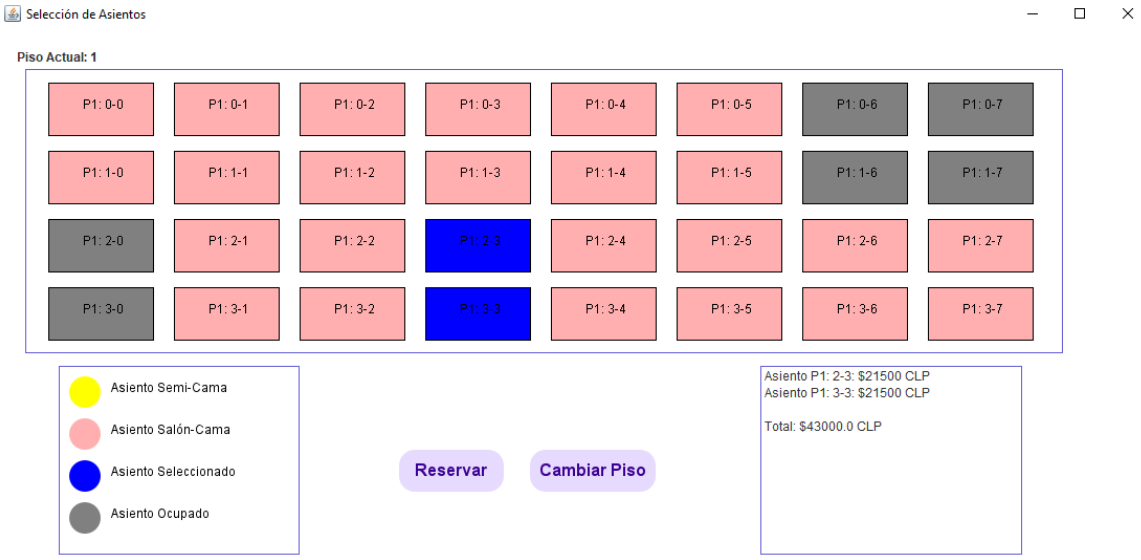
- Vista de ingreso



- Vista de selección de autobús



- Vista de selección de asientos



6 Decisiones tomadas

Las decisiones tomadas para realizar el proyecto se basaron principalmente en los objetivos iniciales de crear un prototipo funcional. Este enfoque nos permitió generar ideas gradualmente y definir la estructura del diseño conforme avanzábamos en la implementación. Sin embargo, a medida que el código fue desarrollándose, surgieron ajustes necesarios. Un ejemplo de esto fue la decisión de simplificar el sistema de asignación de asientos. Inicialmente, se había planeado que un solo piso pudiera tener asientos de distintas categorías, como salón-cama y semi-cama, al mismo tiempo, pero se optó por modificar este enfoque para que cada piso tuviera un solo tipo de asiento. Esta modificación se hizo con el objetivo de reducir la complejidad y mejorar la claridad del sistema.

7 Problemas encontrados

Uno de los principales problemas surgió al intentar implementar un filtro para mostrar los autobuses según su recorrido. A pesar de implementar la lógica necesaria para filtrar por destino, tuvimos dificultades para hacer que los resultados se actualizaran correctamente en la interfaz al presionar el botón “BUSCAR”.

Para resolver este problema, implementamos una `JTable` que sirve como contenedor para mostrar los autobuses filtrados en una tabla. Esto nos permitió representar los datos de forma más clara y dinámica, además de poder gestionar la visualización y la interacción con los datos de manera eficiente.

Otro desafío importante que enfrentamos fue la dificultad para mantener un ritmo constante en la ejecución de las tareas y el cumplimiento de los plazos establecidos. Esto afectó la planificación general del proyecto, ya que algunas funciones y modificaciones tuvieron que adaptarse a medida que el tiempo avanzaba. Este retraso en la colaboración dentro del equipo complicó la finalización de ciertos aspectos y, en algunos casos, afectó la calidad del código implementado.