Proyecto: Sistema de Reserva de Asientos de Autobuses

Integrantes: Darwin Albornoz R.

Grupo: 14

Profesor: Geoffrey Hecht

Asignatura: Programación II

Tema Proyecto: Sistema de reserva de asientos de autobús

- El sistema de reserva de asientos de autobús permite al personal de una empresa de autobús elegir y reservar asientos de forma conveniente por su cliente. Los usuarios pueden visualizar una representación gráfica de los asientos disponibles en el autobús y seleccionar los que deseen ocupar. El sistema muestra información detallada sobre cada asiento, como su ubicación, número y categoría (por ejemplo, semi cama, Salón Cama).
- Una vez que los usuarios seleccionan los asientos deseados, el sistema verifica la disponibilidad y permite confirmar la reserva mostrando el precio a pagar. En caso de que algún asiento ya esté reservado por otro pasajero, se informa al usuario para que pueda elegir otro asiento disponible. El personal confirma el pago (no gestionado por el sistema) lo que reserva los asientos.
- El sistema debe gestionar varios tipos de autobuses (por ejemplo, con diferente número de plazas, o de 1 o 2 pisos...).
- El sistema debe mostrar un menú que permita seleccionar el autobús en función de su horario y recorrido (se supone que estos datos están disponibles con los autobuses vacíos cuando se lanza el software)

Diagrama de clases UML

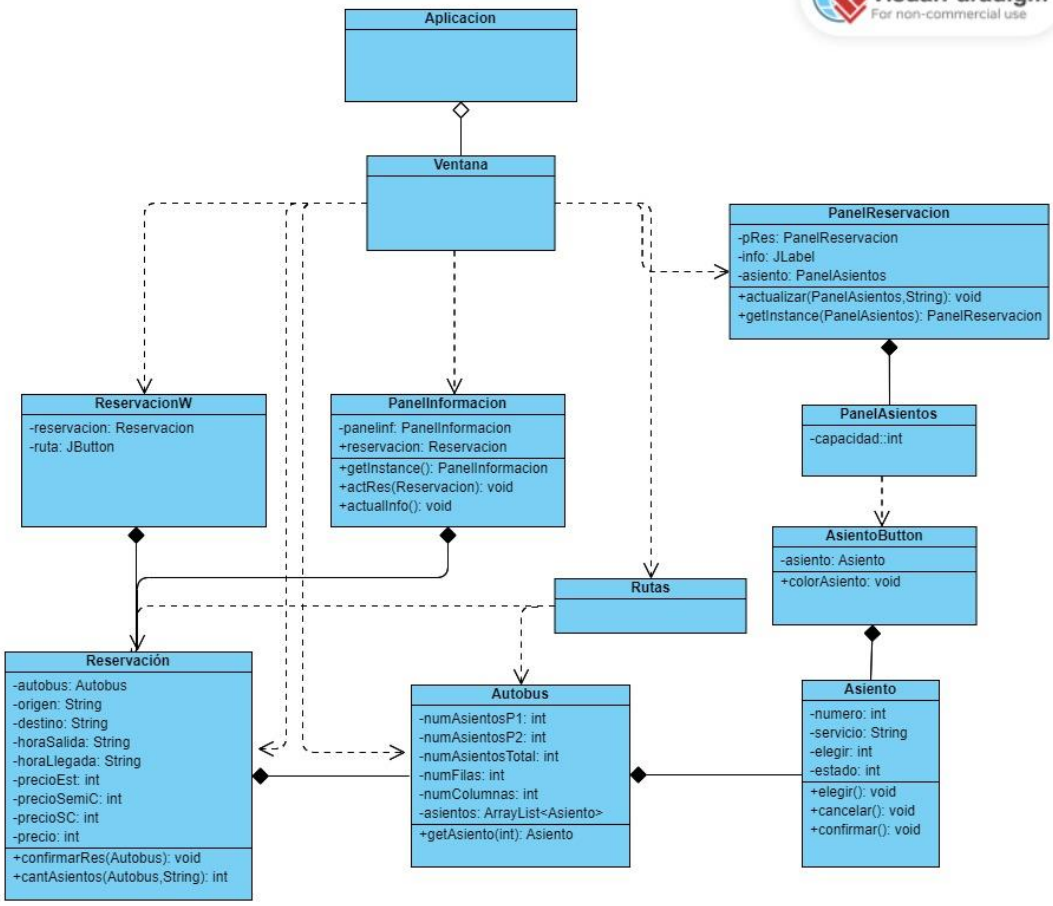
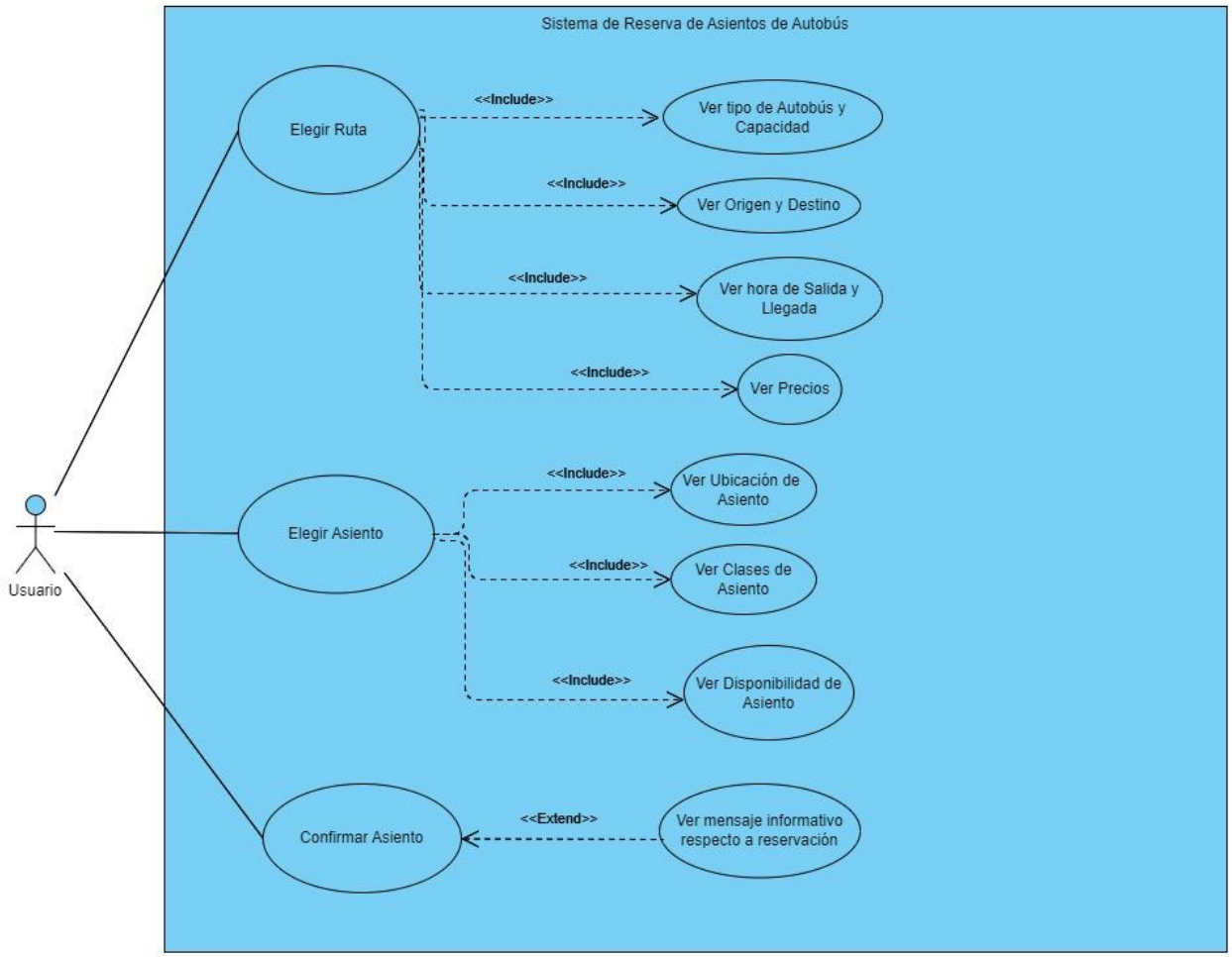


Diagrama de casos de uso



Patrones utilizados

Para el desarrollo del proyecto se consideró el patrón Singleton. La principal característica de este patrón es que permite que una clase solo posea una única instancia, donde no se permite crear otras instancian que perjudiquen la ejecución de un programa. Además, tiene la particularidad de poder tener acceso global a esta instancia.

Para efectos del proyecto, el patrón singleton se utiliza para instancias de las clases PanelInformacion y PanelReservacion, esto por que ofrece la posibilidad de acceder a una de estas dos instancias de manera global, es decir, desde cualquier clase se puede acceder y darle el uso que necesite según la situación. El programa requiere que exista una constante actualización de información y las instancias de las clases mencionadas anteriormente son indispensables en el funcionamiento del sistema. Particularmente, la instancia de PanelInformacion, llamada *panelinf*, se necesita en 4 clases diferentes:

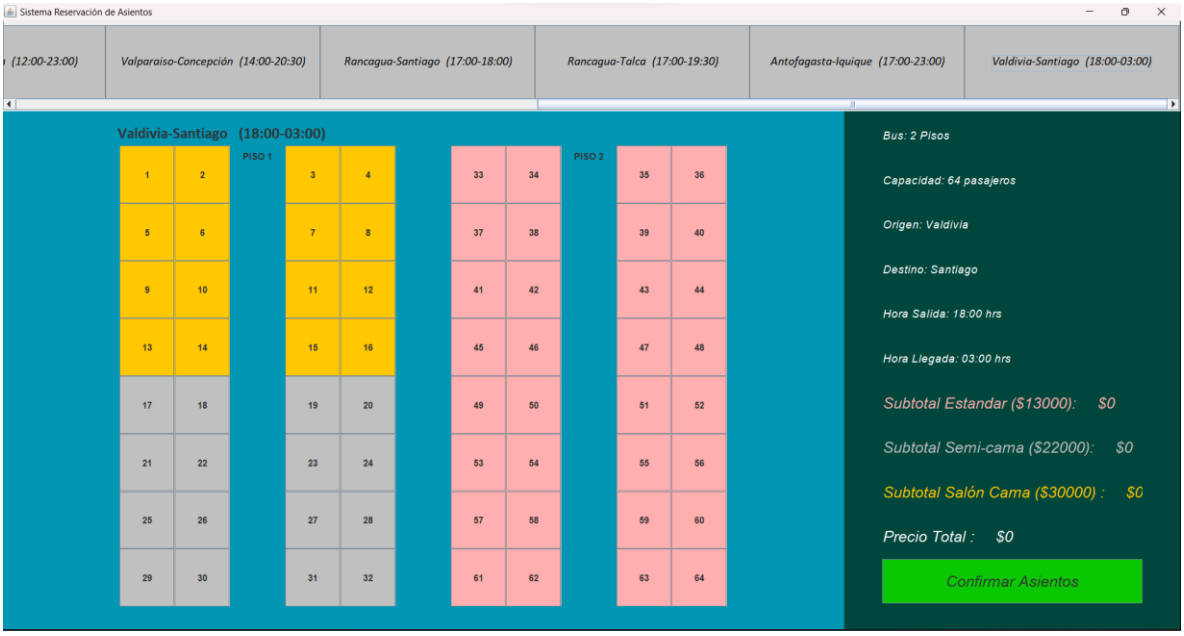
- Clase Ventana, donde comienza a ejecutarse el código permitiendo el funcionamiento de GUI.

- Clase Panel Reservación, donde la información debe ser actualizada de acuerdo con los cambios que realice el usuario al interactuar con la interfaz.

- Clase AsientoButton, en la cual la instancia es fundamental para actualizar datos relacionados a la elección de un asiento determinado.

- Clase ReservaciónW, aquí tiene la función de renovar la información según sea la ruta elegida por el usuario.

Captura de pantalla de interfaz



Sección de decisiones

Las primeras decisiones se relacionaban con la estética que debía tener la interfaz, la distribución que debía tener cada elemento para mejorar la visualización y el manejo que un usuario le podría dar. Por temas de sentido común, se decide dejar un listado de rutas en la zona superior, donde el usuario puede desplazarse horizontalmente para escoger la ruta que requiera en ese momento. En la zona inferior izquierda se mostraría un panel con asientos vinculados a la ruta seleccionada anteriormente y luego en la zona inferior derecha estarían distribuidos datos acerca del autobús, itinerario, precios junto con un botón que permitiera confirmar la selección de asientos.

Luego, habría que definir la construcción de clases y el desarrollo del código necesario para el correcto funcionamiento del programa. Se puede destacar el uso de una relación de composición entre las diferentes clases, donde cada clase contribuía al funcionamiento de otra, y esta a su vez a la de una siguiente clase. La decisión del patrón se basó en la necesidad de acceso a ciertas instancias, desde otras clases, sin perjudicar la ejecución del programa al crear nuevas instancias. De esta forma, se pudo conseguir la elaboración de un sistema de reserva de asientos que sea lo más accesible y amigable con el usuario.

Sección de problemas y autocrítica

Los problemas estaban íntimamente relacionados con el desarrollo de código, donde cada pequeño fallo afectaba gravemente el funcionamiento del programa. La escaza experiencia en los temas de GUI complicaban la búsqueda de una pronta solución. Había momentos donde costaba encontrar el error, y se procedía a revisar clase por clase, método a método cual podría ser la razón. Es por eso que considero, como autocrítica, indispensable mejorar la distribución de cada parte del código, hacer comentarios constantemente, para saber que utilidad tiene cada variable, cada método, cada condicional, de esta forma se facilita enormemente la tarea de solucionar ciertos imprevistos.