**INTEGRANTES:**

* **lUis alberto luna arana**
* **Joel josue ORRALA YUGLA**
* **GIOVANNI ANDRE SAMBONINO PINCAY**

**URL Repositorio: https://github.com/JoelOrrala/POO4\_PROY2P\_Luna\_Orrala\_Sambonino.git**

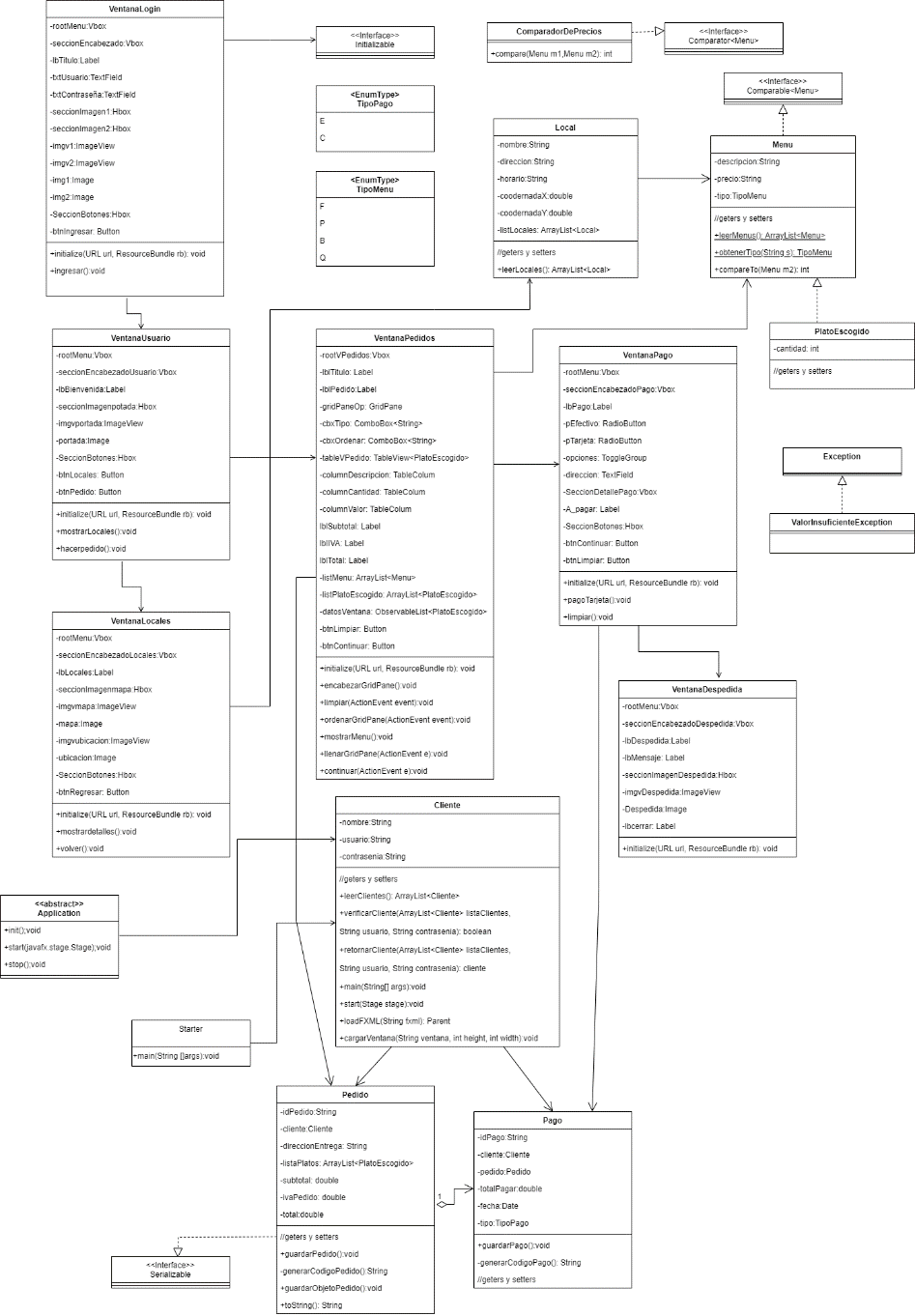
Fecha: 29/ENE/2023

Reporte – proyecto SEGUNDO parcial-par 4

PROGRAMACIÓN ORIENTADA A OBJETOS TÉRMINO II 2022-2023

# Diagrama de clases

En esta sección deberán incluir el diagrama de clases que elaboraron como primera versión de su proyecto.



# Tareas

En esta sección incluirán el detalle de las tareas que se asignó a cada estudiante.

Estudiante (Joel Josué Orrala Yugla):

1. Creación de las clases del paquete modelo
2. Creación de los métodos de las clases del paquete modelo
3. Adicionalmente ayudo a los demás compañeros con algunas dudas que tenían con los métodos de sus clases
4. Creación de archivos.

Estudiante (Giovanni Andre Sambonino Pincay):

1. Creación de la Ventanas PagoController y DespedidaController con sus respectivos métodos.
2. Creación de las los FXML de dichas ventanas.
3. Creación de los enums utilizados.
4. Adicionalmente ayudo a los demás compañeros con algunas dudas que tenian con los métodos de sus clases.

Estudiante (Luis Alberto Luna Arana):

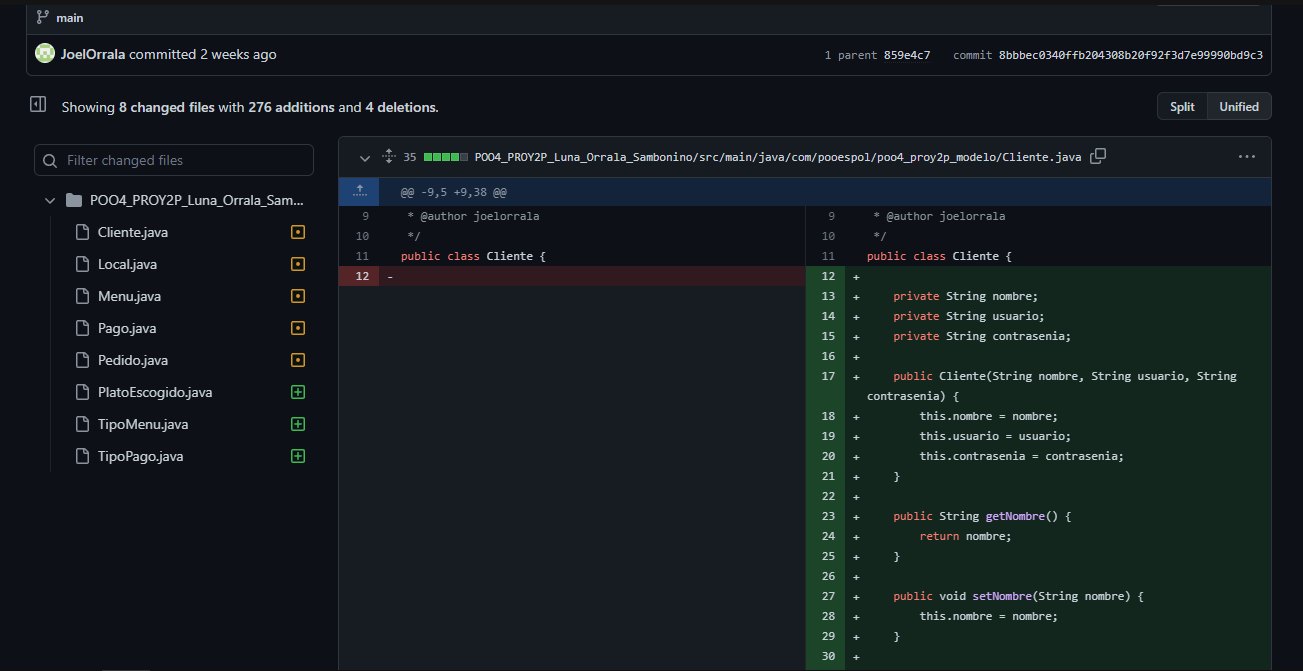
1. Creación de la Ventanas IngresoController, PedidosController y UsuarioController con sus respectivos métodos.
2. Creación de las los FXML de dichas ventanas.
3. Adicionalmente ayudo a los demás compañeros con algunas dudas que tenian con los métodos de sus clases.
4. Creación de archivos.

# Evidencias de Tareas

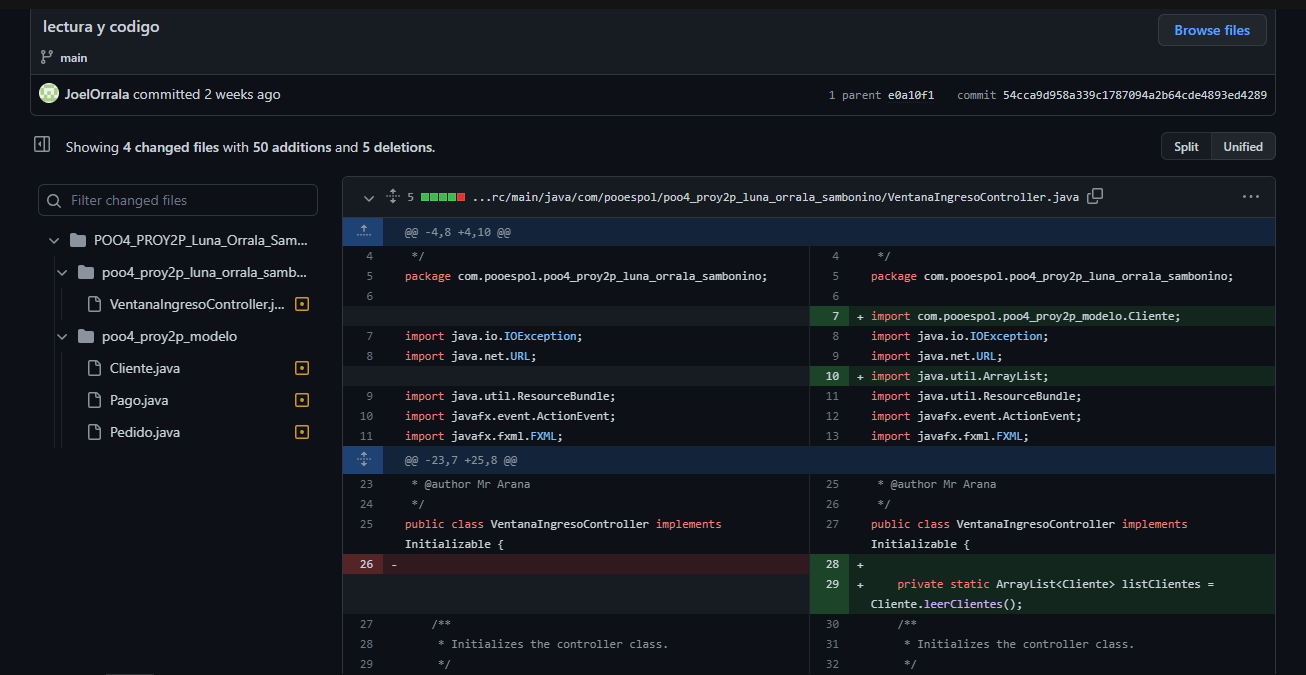
En esta sección incluirán screenshots de la página de Bitbucket donde consten los commits realizados. Además, por cada estudiante deben elegir un par de commits y mostrar el detalle. Esto debería concordar con las tareas definidas para cada uno. Ejemplo:

Estudiante (Joel Josué Orrala Yugla)):

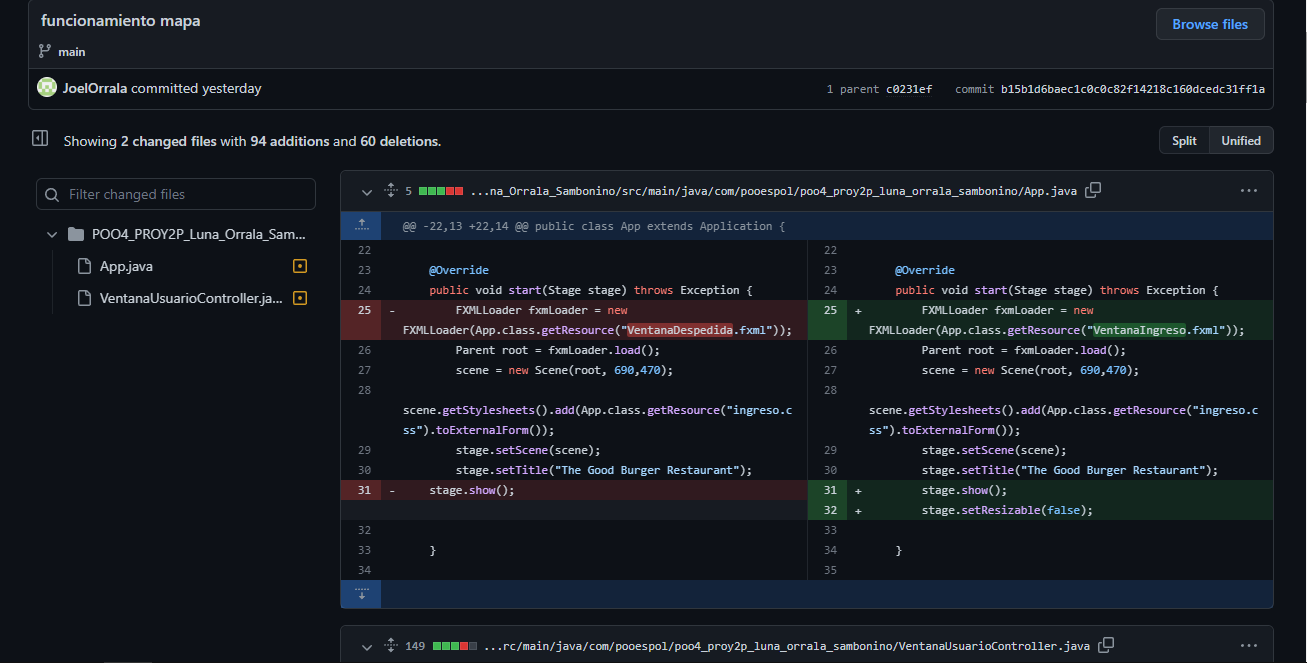
Commit 1



Commit 2

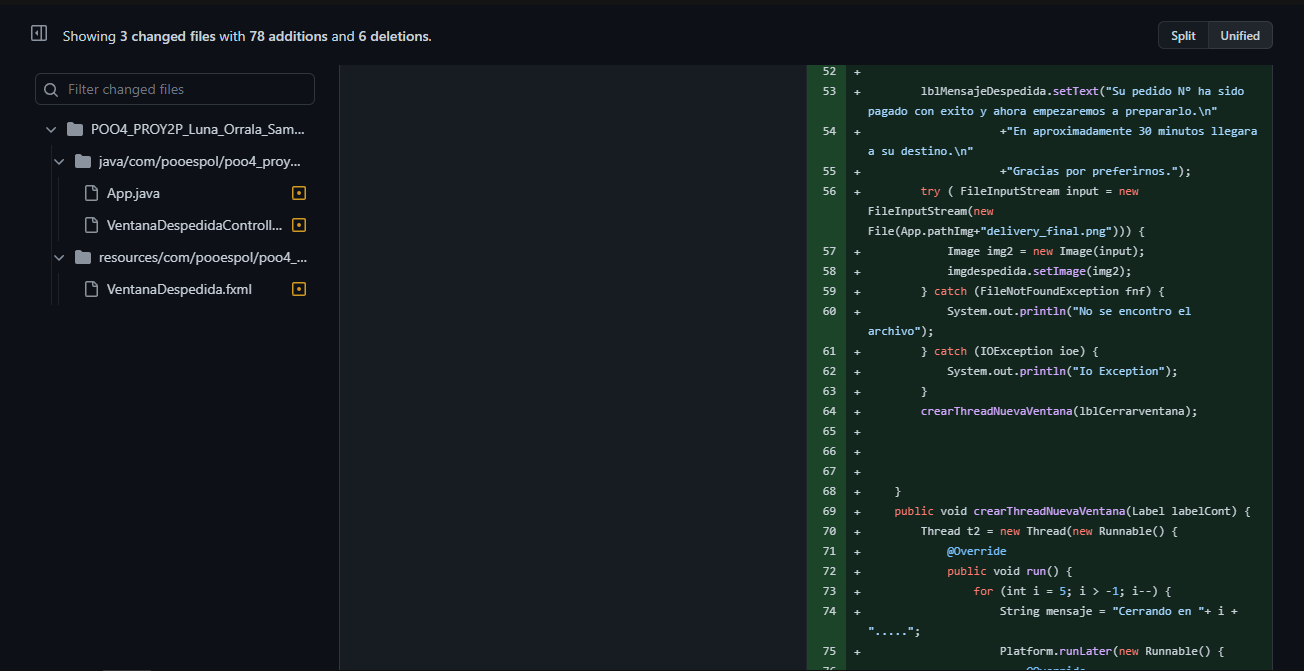


Commit 3

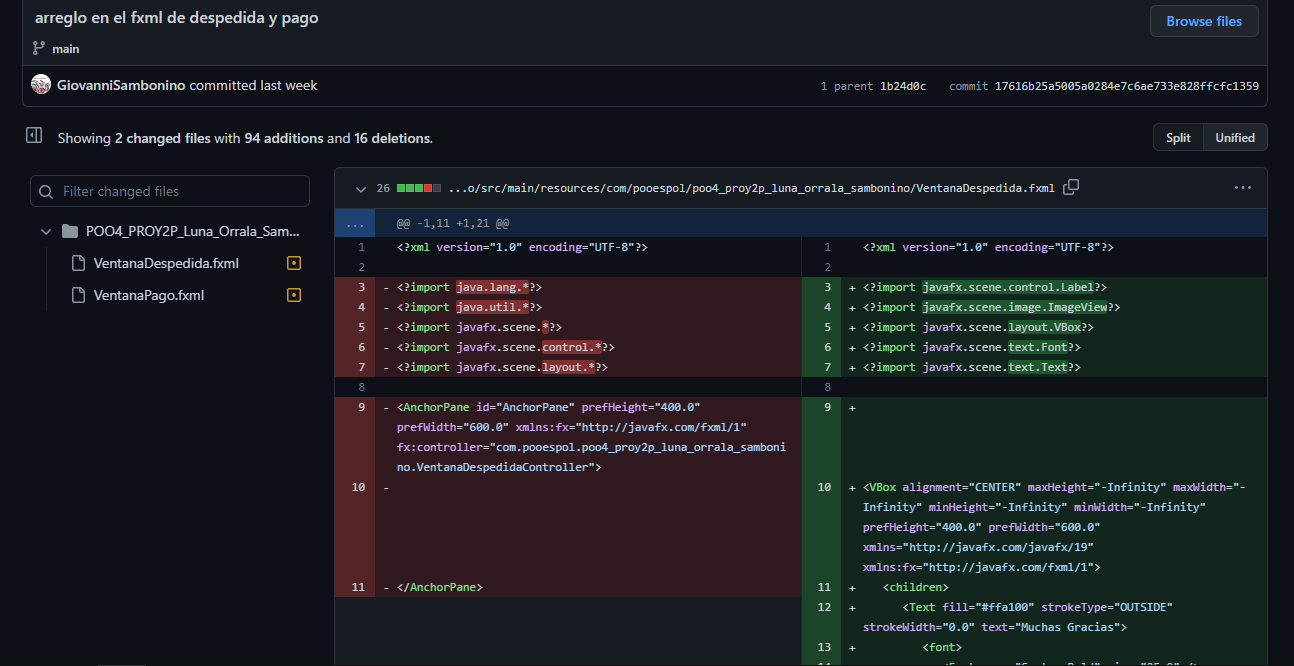


Estudiante (Giovanni Andre Sambonino Pincay):

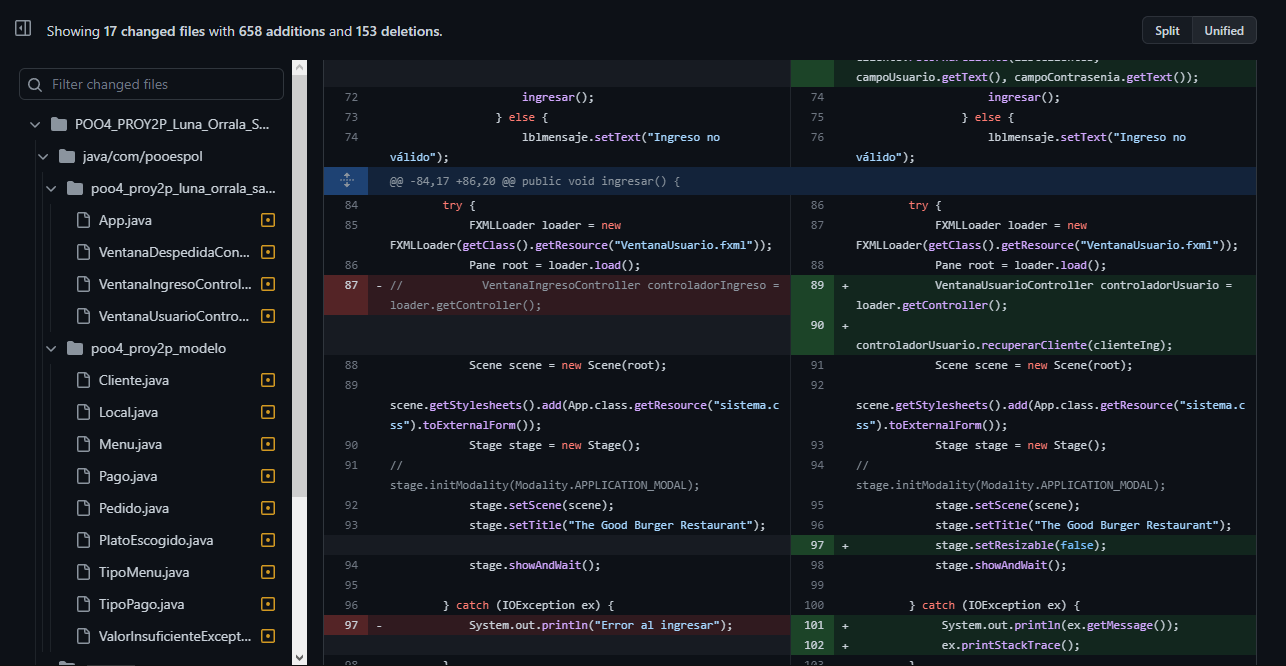
Commit 1



Commit 2

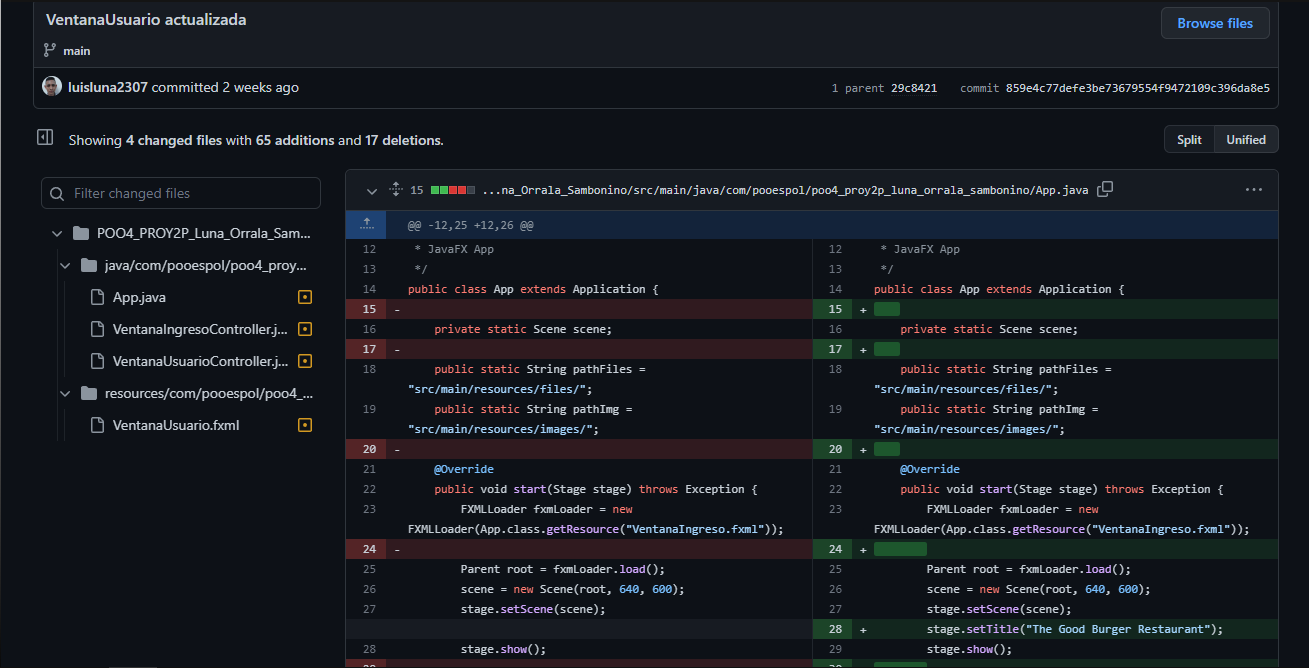


Commit 3

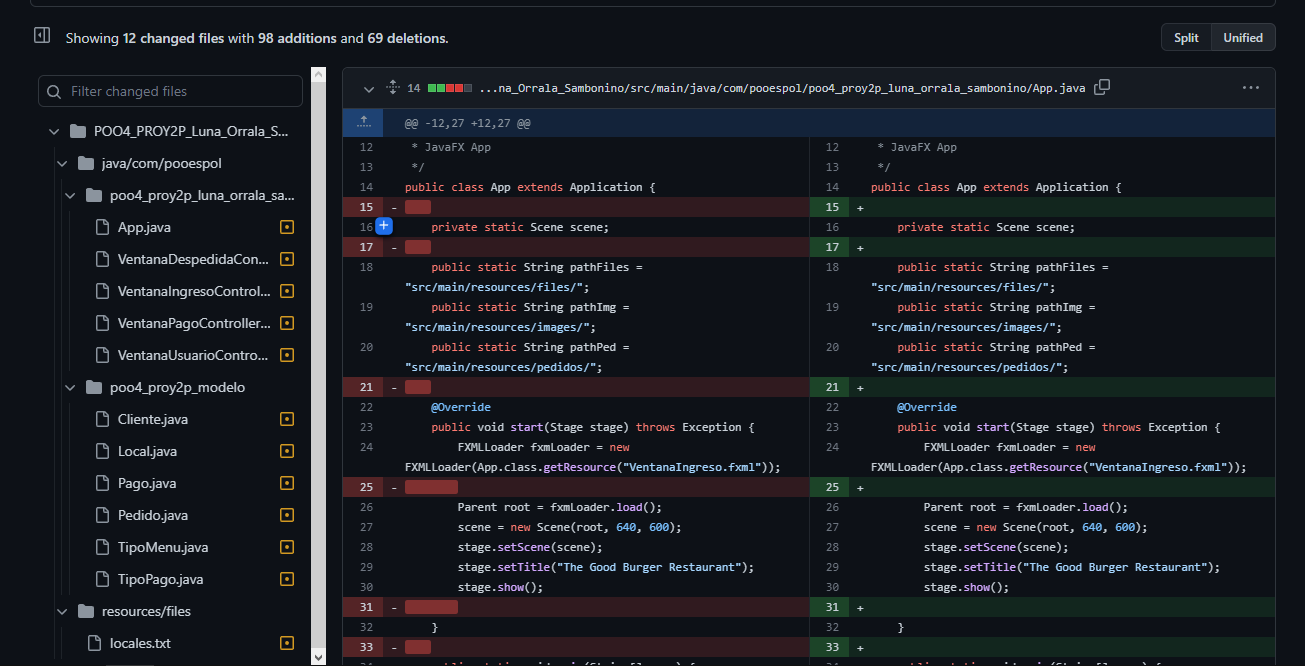


Estudiante (Luis Alberto Luna Arana)):

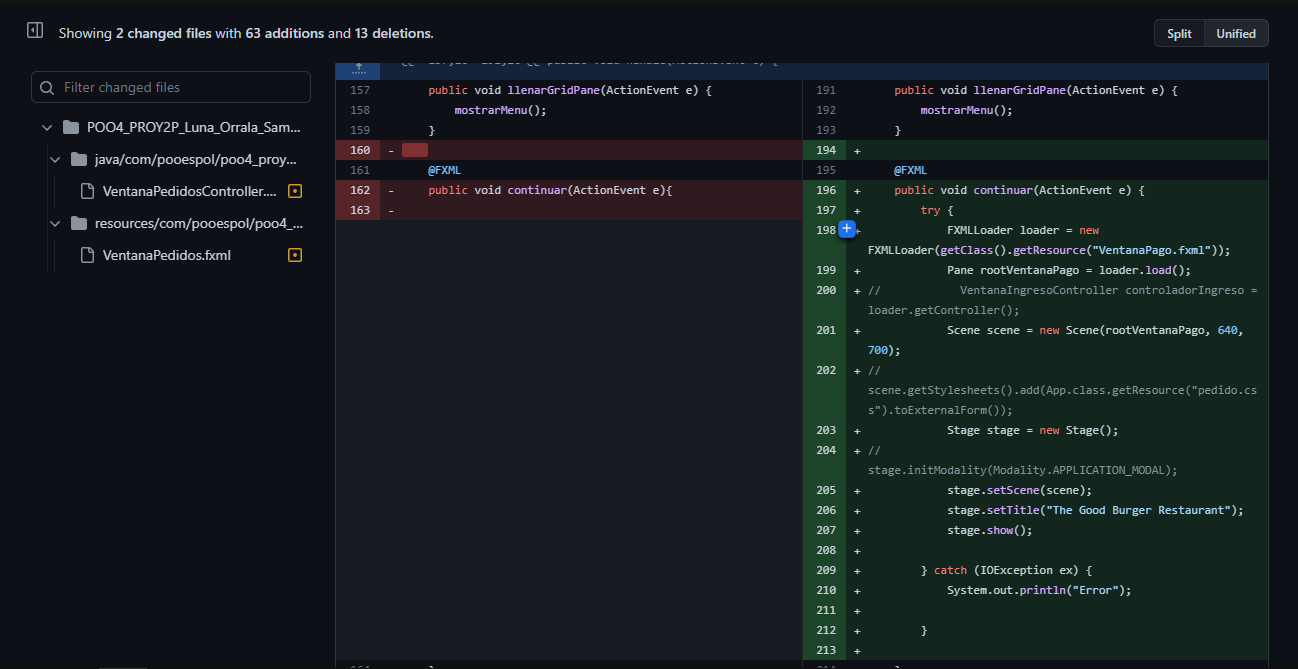
Commit 1



Commit 2



Commit 3



# Identificación de teoría aplicada en programación orientada a objetos.

En esta sección deberán incluir el detalle de los bloques de código de su proyecto donde identifican cada pilar de este paradigma: Abstracción, Encapsulamiento, Herencia y Polimorfismo. Además, deberán mostrar donde usan la interface Comparable y una descripción de porqué se ha decidido aplicar el usar de cada uno de estos conceptos en sus proyectos. Si no incluyen la explicación adecuada, este punto no será válido. Ejemplo:

## **Threads**

Tabla

Descripción generada automáticamente con confianza baja

Utilizamos los hilos para hacer que las ventanas se cierren pasados x segundos

Imagen que contiene Escala de tiempo

Descripción generada automáticamente

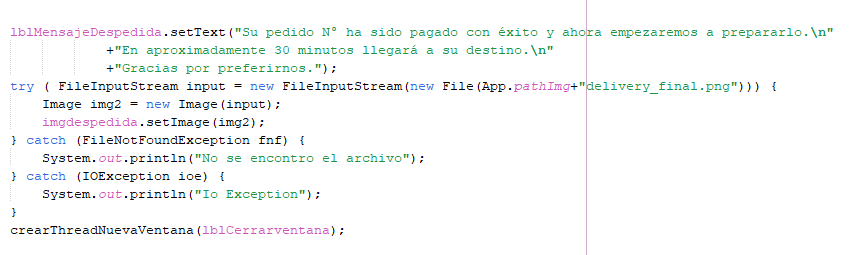
También se utilizó el hilo para hacer aparecer las flechas de información de los restaurantes aleatoriamente entre un lapso de 1 a 10 segundos

## **Manejo de Excepciones**

Texto

Descripción generada automáticamente

Utilizamos excepciones en varios momentos del programa por ejemplo aquí la utilizamos para que capture una IOException en caso de que no se encuentre el archivo de los usuarios.

Este es otro caso en el que utilizamos excepciones para que atrape las excepciones en caso de que no se encuentre el archivo.



Aquí utilizamos excepciones para verificar si al momento de presionar el botón agregar, tenemos un valor valido en la el texfield de cantidad.

## **Manejo de Excepciones propias**

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Creamos una clase para la excepción personalizada que se nos pedía

## **Manejo de Archivos**

Texto

Descripción generada automáticamente con confianza media

El método guardarPago escribirá el pago en el archivo pagos.txt

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

El método leerClientes nos retorna un arraylist de clientes a partir del archivo usuarios.txt

Interfaz de usuario gráfica, Texto, Aplicación

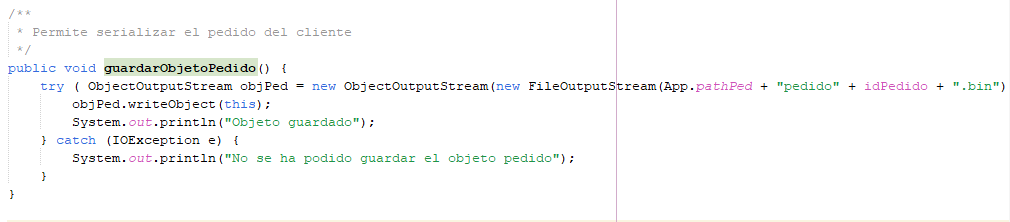
Descripción generada automáticamenteAsí mismo el método leerLocales retornara un ArrayList de locales a partir del archivo locales.txt

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Así mismo el método leerMenus retornara un ArrayList de menú a partir del archivo menu.txt

**Serialización**



El método guardarObjetoPedido serializa el pedido guardado que se realizó.

**Interfaz Comparable**

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

En este método estamos comparando los objetos menú por descripción

Texto, Carta

Descripción generada automáticamente

En este método estamos comparando los objetos menú por el precio de cada producto del menú.

Interfaz de usuario gráfica, Texto, Aplicación

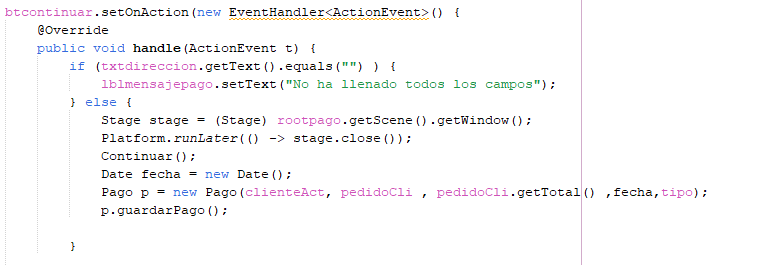
Descripción generada automáticamente

Y aquí los estamos llamando para que se ordenen en función del nombre del menú o en función del precio.

## **Controladores de eventos**

Imagen que contiene Texto

Descripción generada automáticamenteEste controlador de evento se utiliza para que cuando se oprima el botón ingresar de la ventana ingreso verifique si el cliente ingresado coincide con el que está en el archivo.



Este controlador de evento se utiliza para que cuando se oprima el botón continuar en la ventana de pedido verifique si los campos están llenos y si o están que pase a la siguiente ventana.

## **Programación dinámica de GUI**

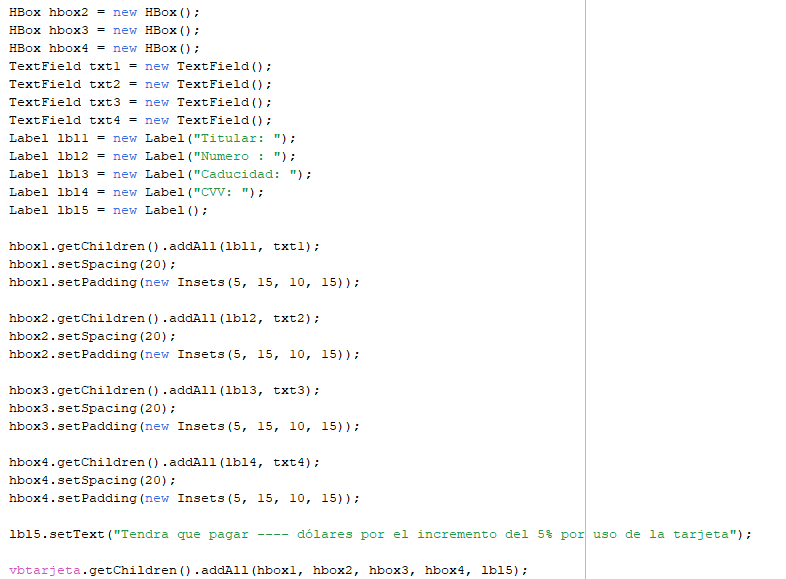


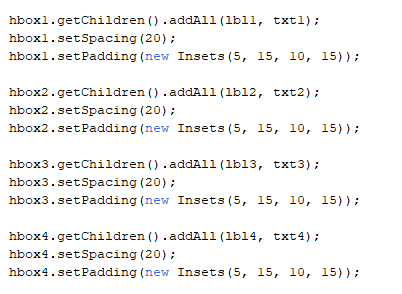
Imagen que contiene Texto

Descripción generada automáticamenteEn estas capturas se crean elementos que luego se implementan en la ventana de manera dinámica

## **Posicionamiento absoluto**

Texto, Carta

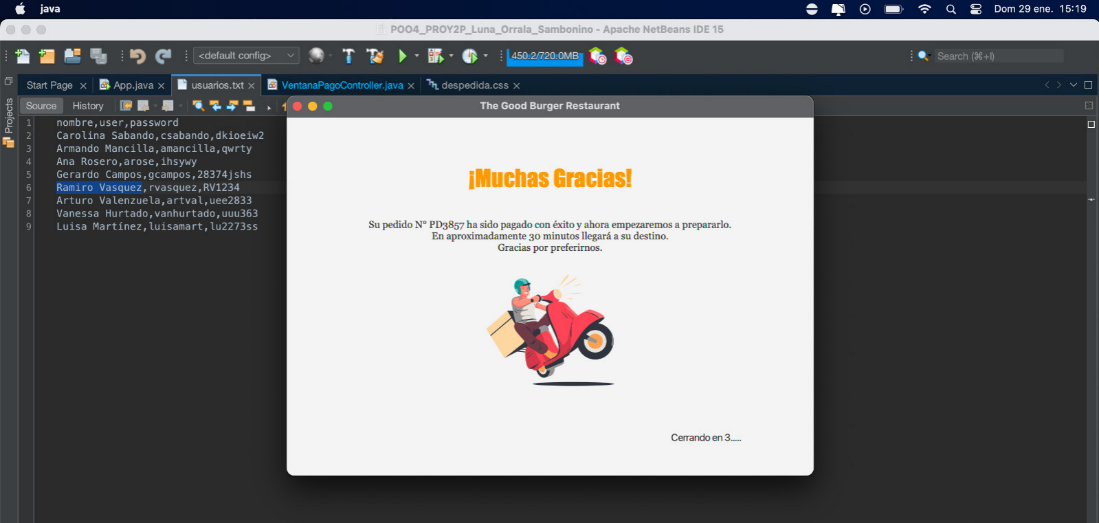
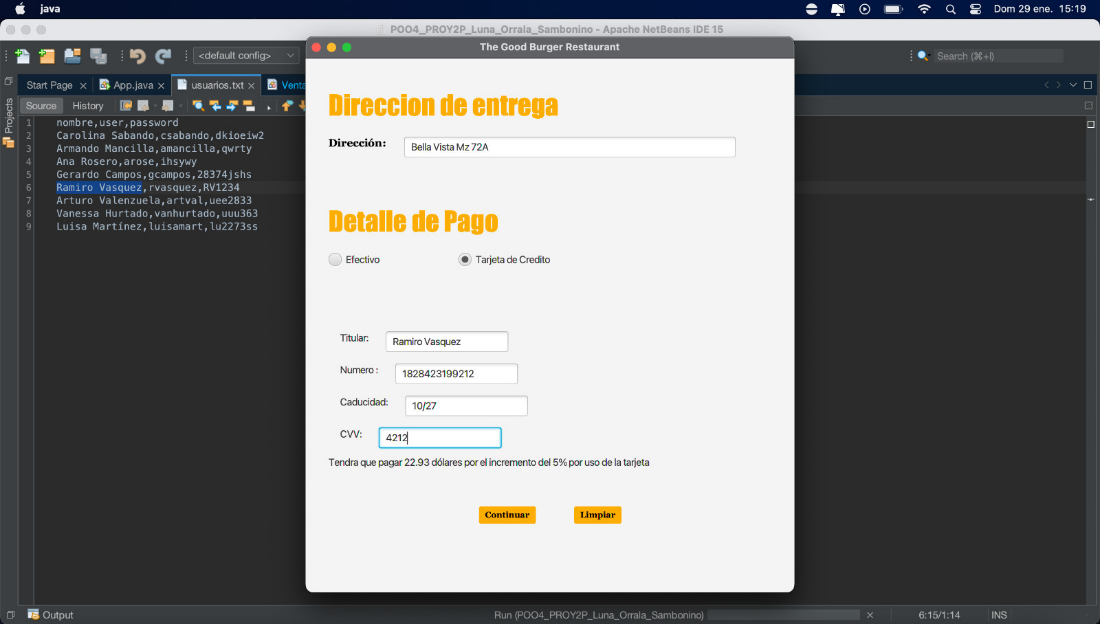
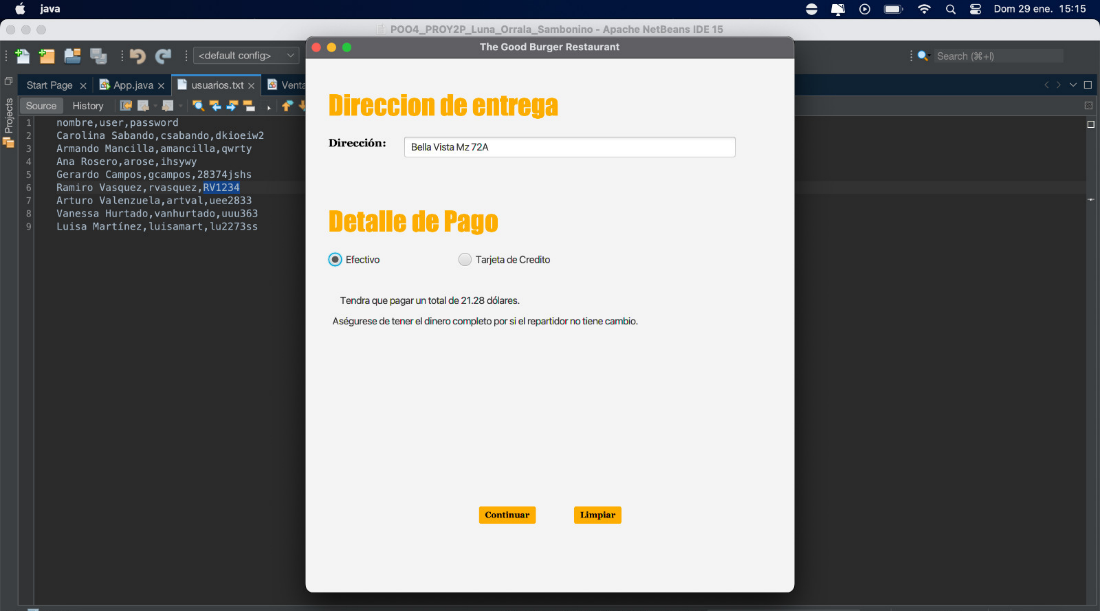
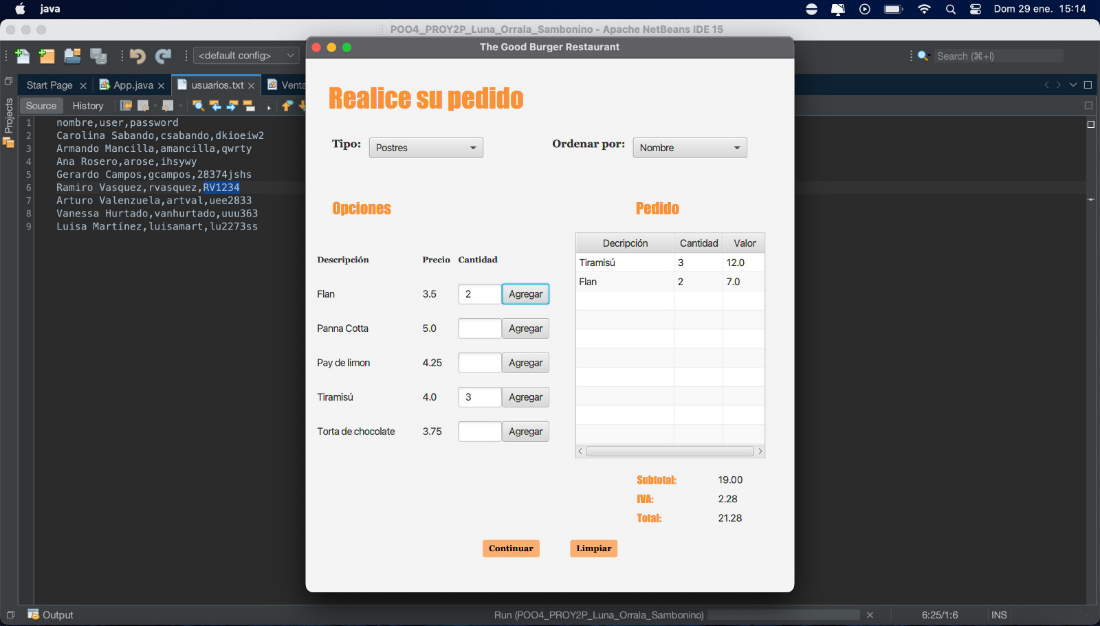
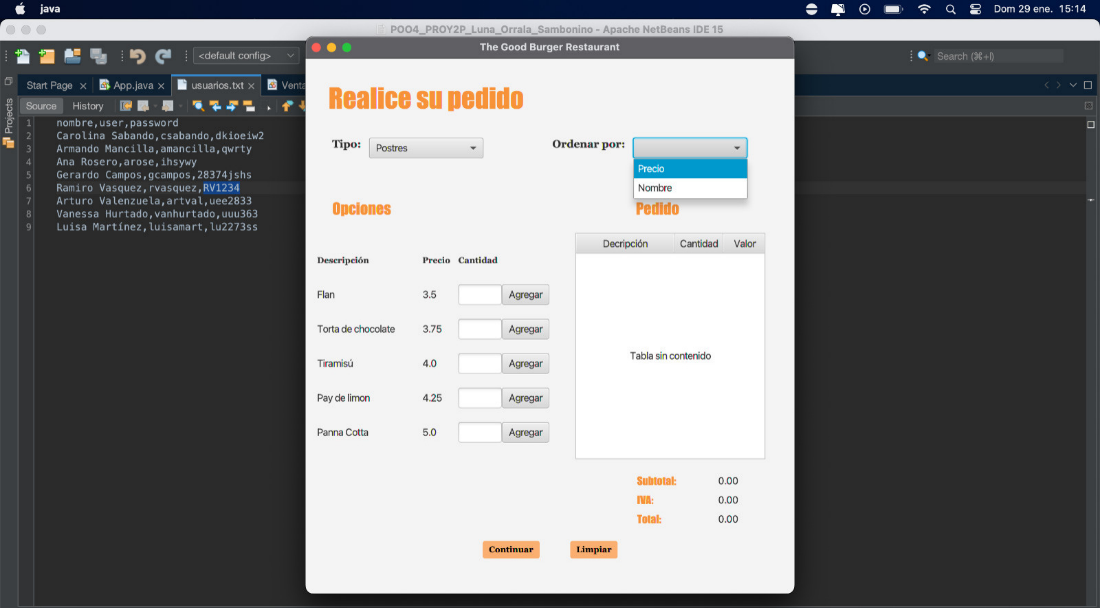
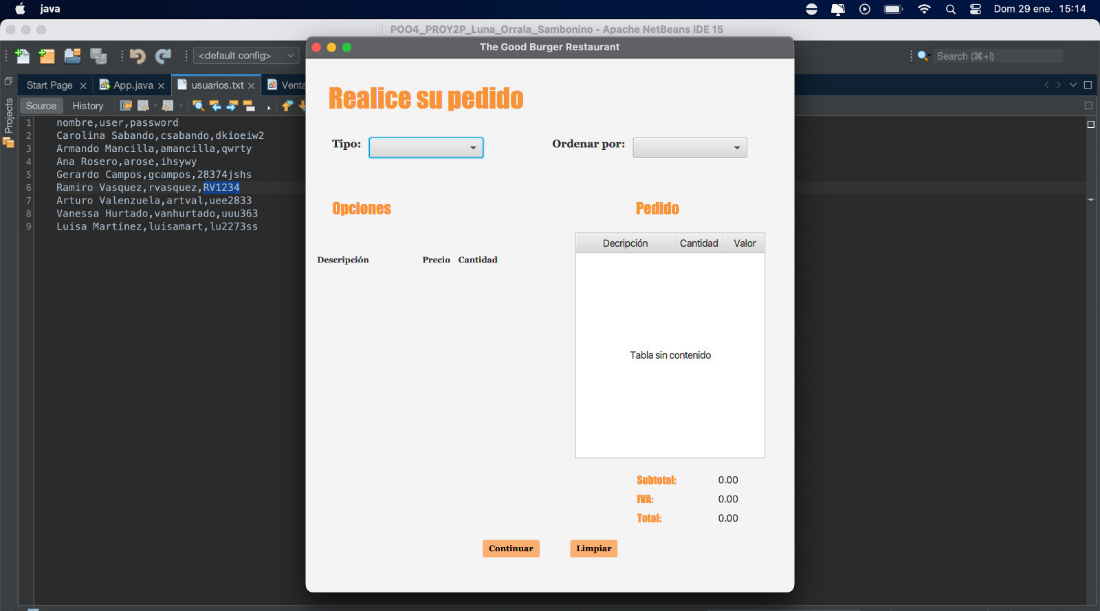
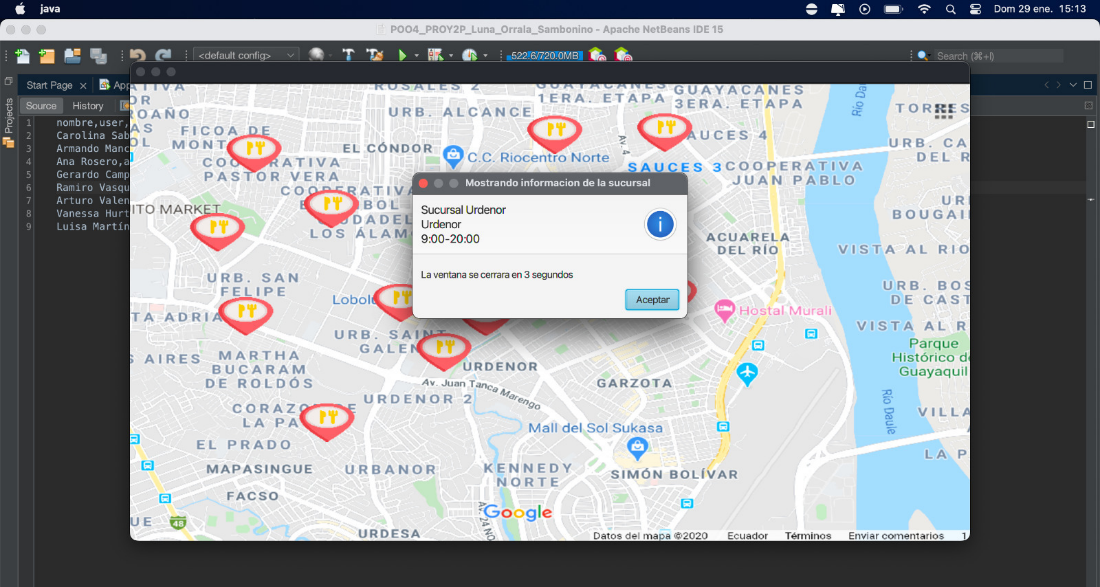
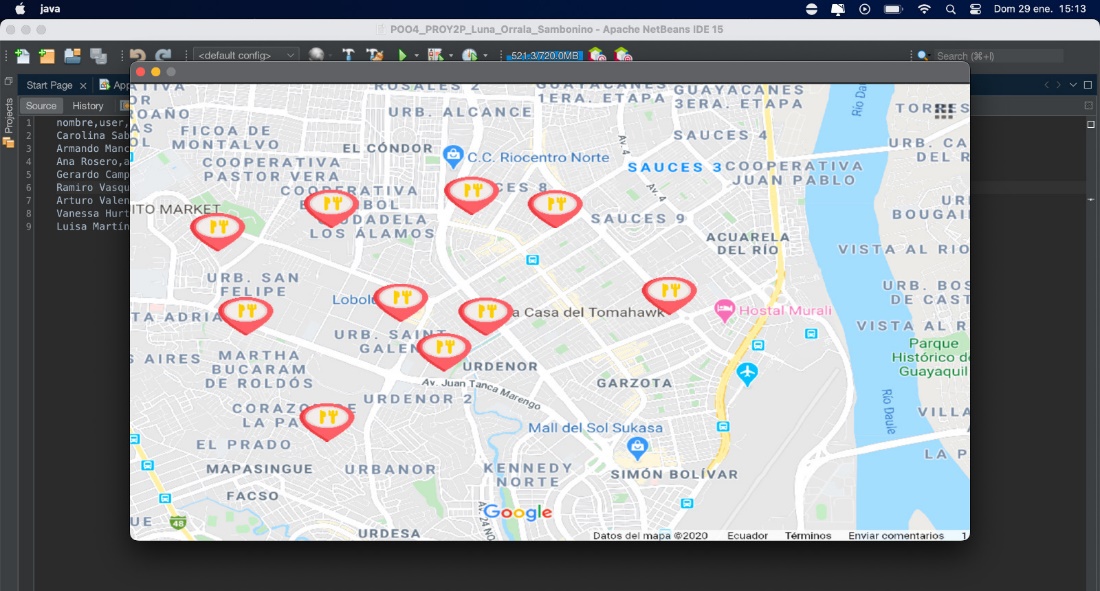
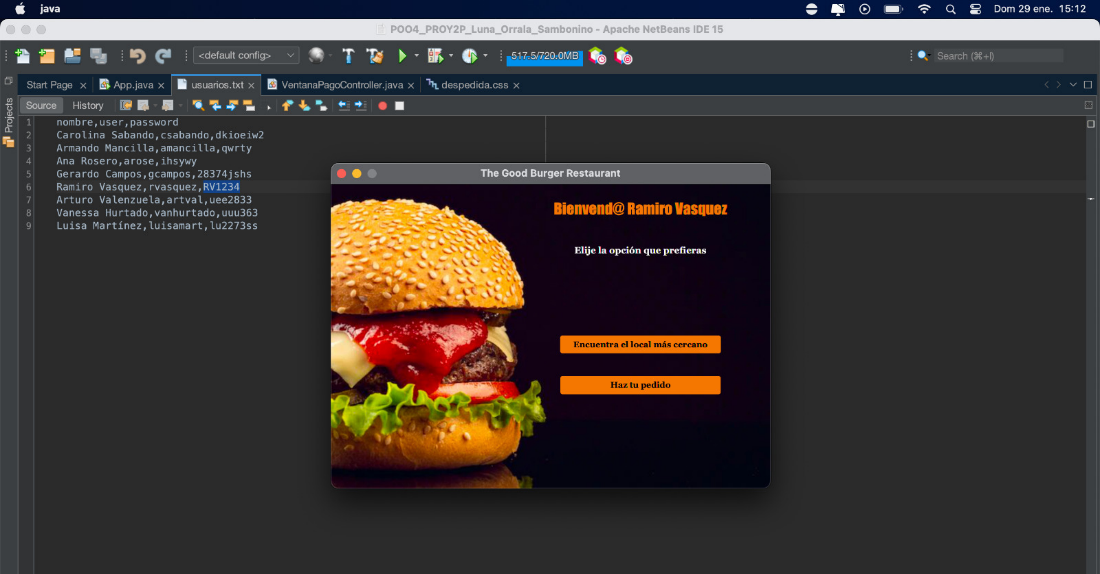
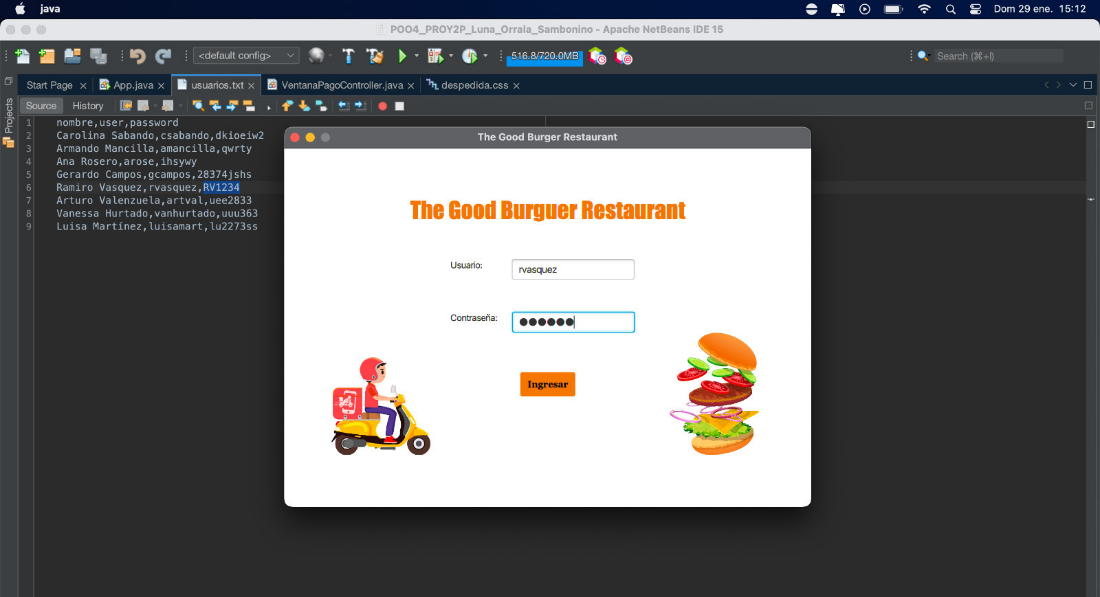
Descripción generada automáticamente



En esta captura se puede ver como se ingresan los valores creados dinámicamente a la ventana del javafx.

# Programa en ejecución

En esta sección deberán incluir screenshots de su proyecto en ejecución, con el objetivo de que se visualice la ejecución de este.



# JAVADOC

**Agregar la documentación JAVADOC como una carpeta adicional a este reporte**. Los métodos deben estar siempre comentados con el formato explicado en clase.