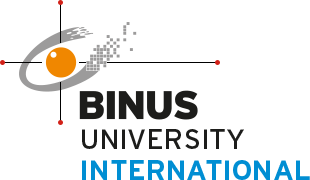
****

****

**Parking Management System**

Troy Poetra Prajoga|ID: 2702291910|Object Oriented Programming

**Table of Contents**

**Project Specification**…………………………………………...3

Project Overview…………………………………………...3

Objectives……………………………………………….….3

**Solution Design (Class Diagram)**………………………….….4

**Discussion of what was implemented and how it works**….…5

Solution Design………………………………………….…5

Main.java…………………………………………………...6

superGUI.java……………………………………………...7

TicketGUI.java……………………………………………..9

ParkingGUI.java…………………………………………..10

ParkingSlotsGUI.java……………………………………...11

LeaveGUI.java…………………………………………….12

InterfaceParkingLot.java…………………………………..14

AvailableParkingSlots.txt…………………………………..15

ParkingLot.java……………………………………………16

**Evidence of Working Program including Screenshots**……..28

**Project Specifications**

**Project Overview**

The Program created is a Parking Lot Management System. This program is designed to manage the parking operations of a parking lot, as well as simulate the parking experience. The system provides functionality for tracking and managing parking slots, calculating payment for parked vehicles, and displaying real-time information about parking occupancy.

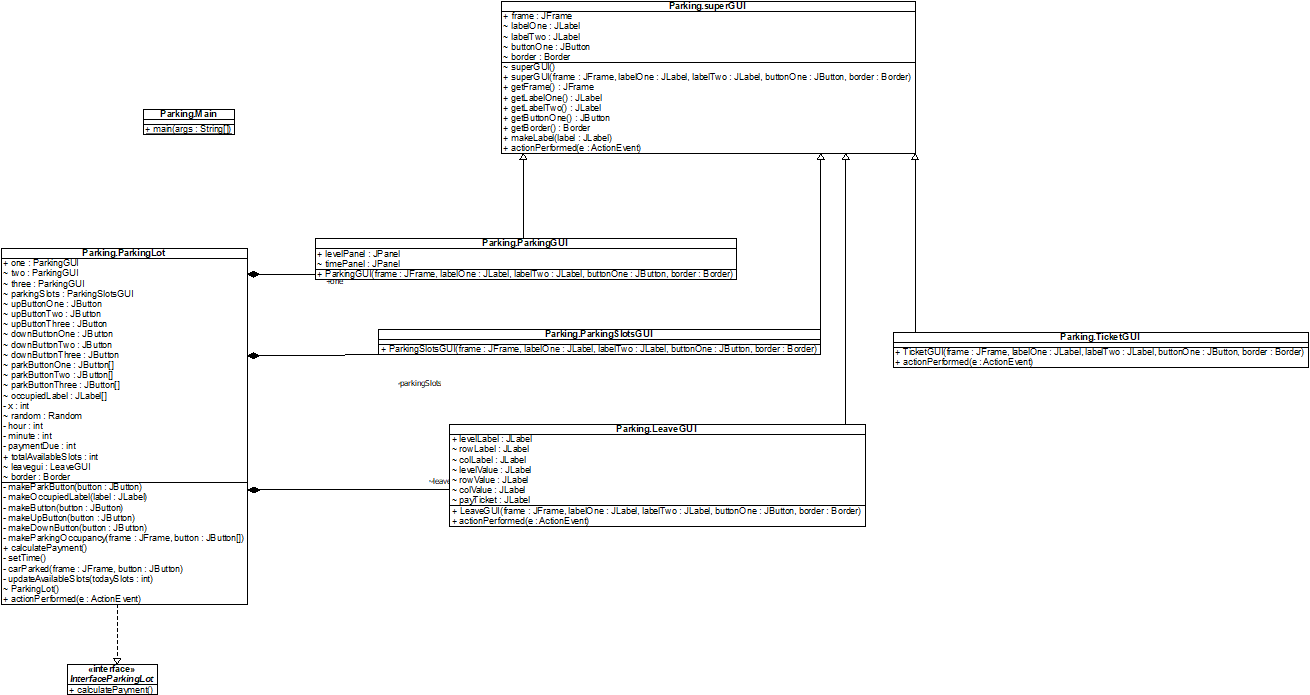
**Objectives**

The main objectives of the projects are as follows:

1. Design a user-friendly graphical user interface(GUI) for the parking system.
2. Manage the occupancy and availability of the parking slots within the parking lot.
3. Calculate the payment due for a parked vehicle based on parking duration.
4. Store and retrieve parking data from a text file for data persistence across multiple program runs.
5. Allow users to navigate through different parking levels within the parking lot.
6. Provide real-time updates on availability of parking slots.

**Solution Design**

**(Class Diagram)**

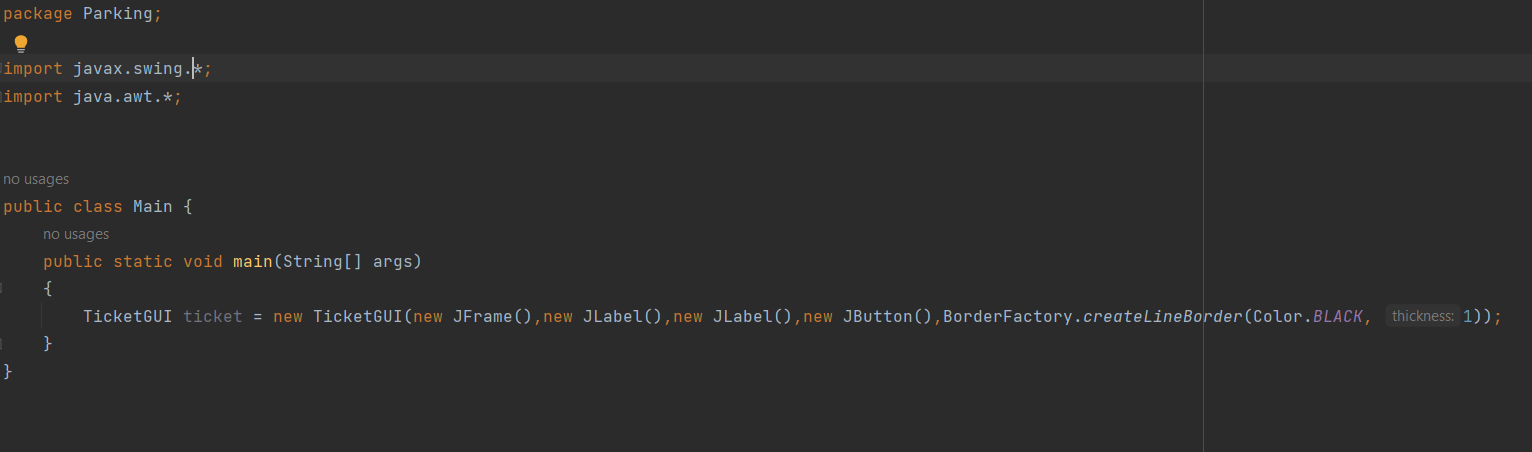


**Solution Design**

In this program, the classes that are included are Main.java, superGUI.java, ParkingSlotsGUI.java, TicketGUI.java, ParkingGUI.java, LeaveGUI.java, and ParkingLot.Java. The GUIs were created using Java Swing. The interface that is included is InterfaceParkingLot.java. The txt file that is included is AvailableParkingSlots.txt.

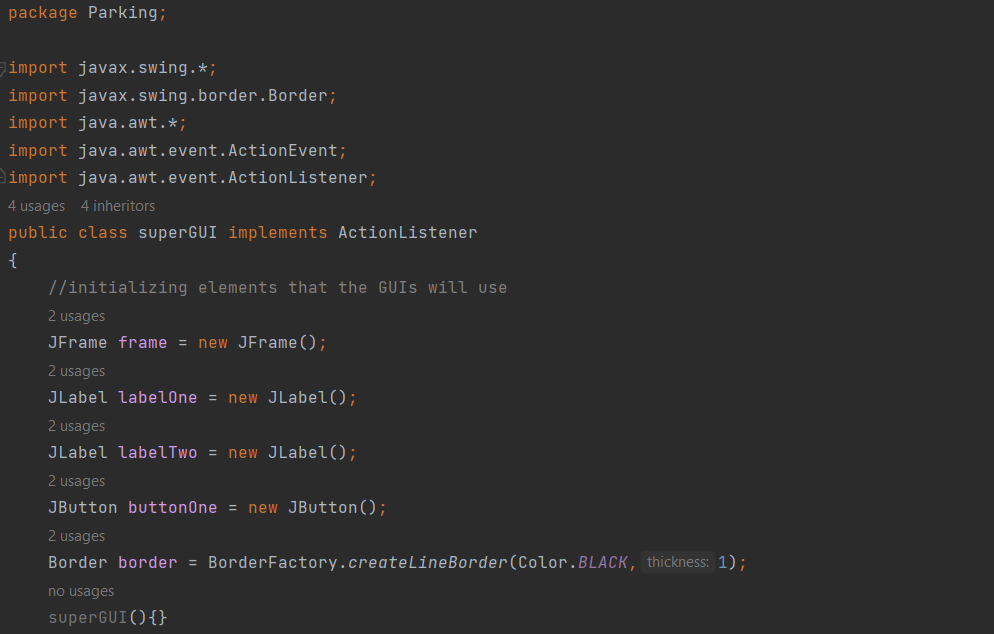
**Main.java**

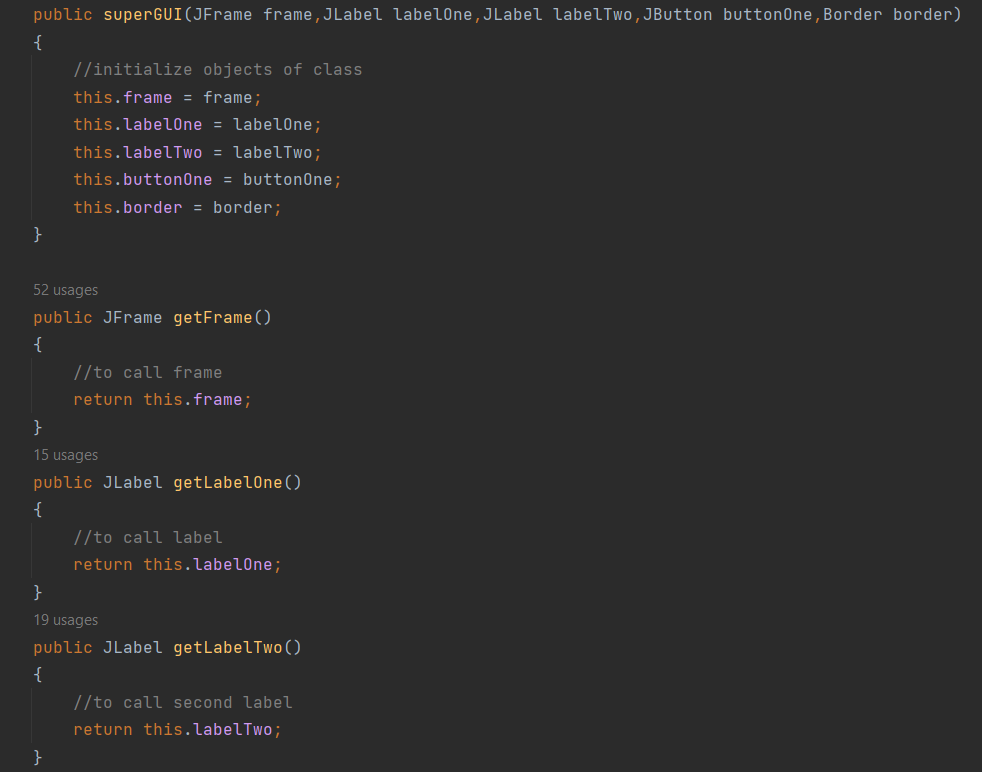
The Main.java class is the main class that runs the whole program. It starts by creating a new instance of the TicketGUI class. Since the TicketGUI uses Java Swing, the imports used in this class are javax.swing.\* and java.awt\*.



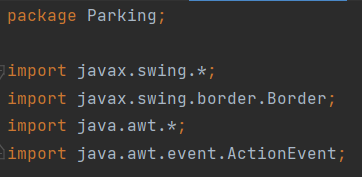
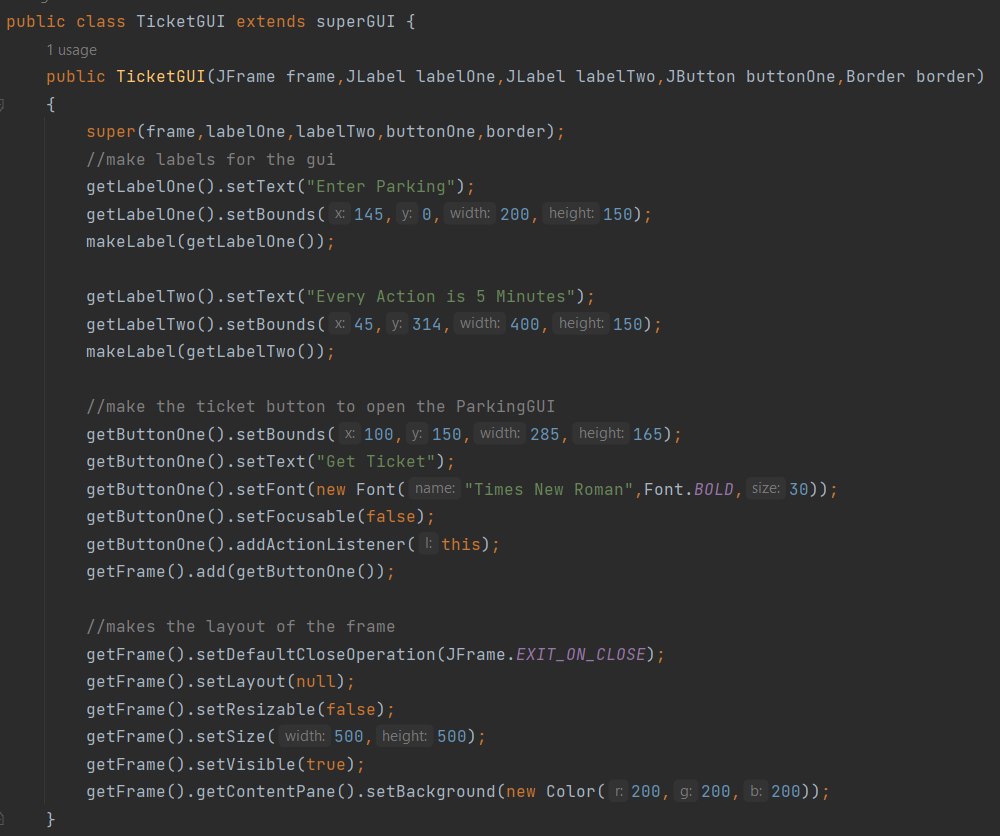
**superGUI.java**

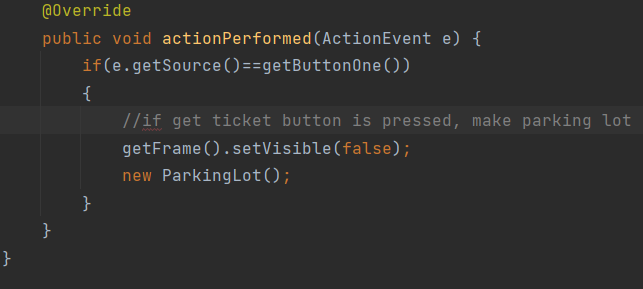
The superGUI.java class is the parent class for the GUIs. It contains the main attributes that most of the GUIs have so that the child classes can extend from this class. This class contains one JFrame, two JLabels, one JButton, and one Border.



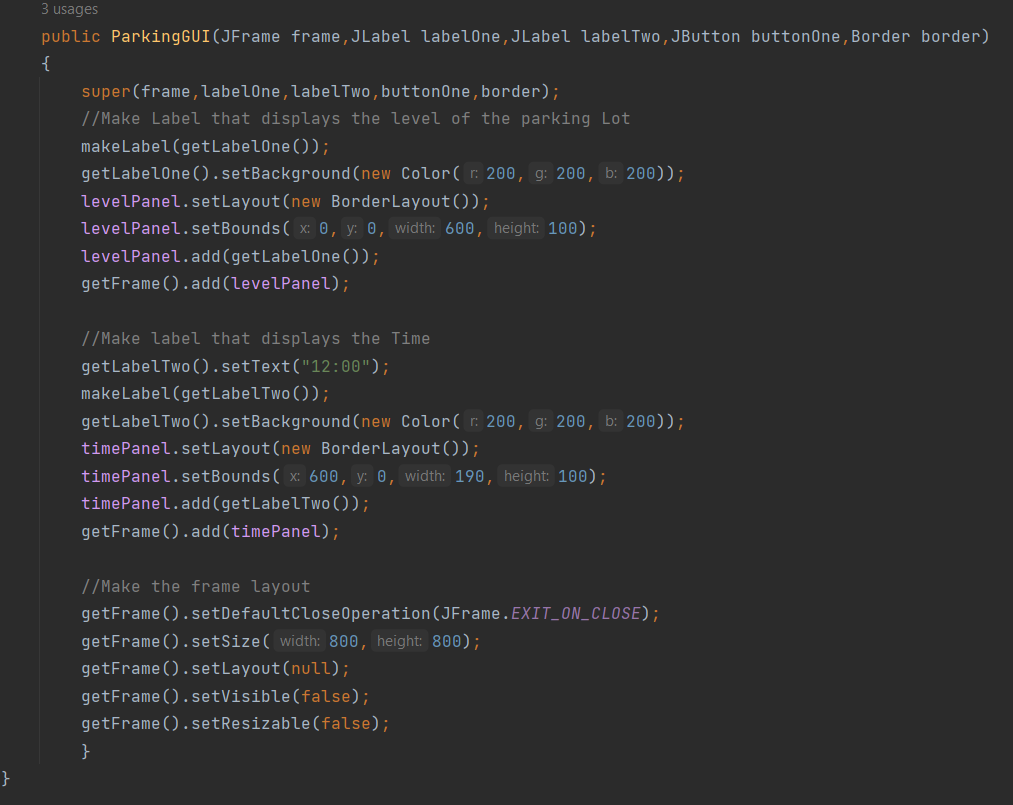
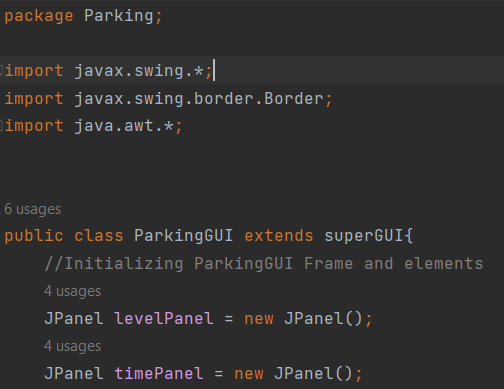
This class also includes a constructor that initializes the GUI attributes, and also getter methods such as getFrame( ), getLabelOne( ), getLabelTwo( ), getButtonOne( ), and getBorder( ). This class also includes a setter method called makeLabel(JLabel label) which creates the general label attributes. The last thing that this class has is an actionPerformed method for the child classes.

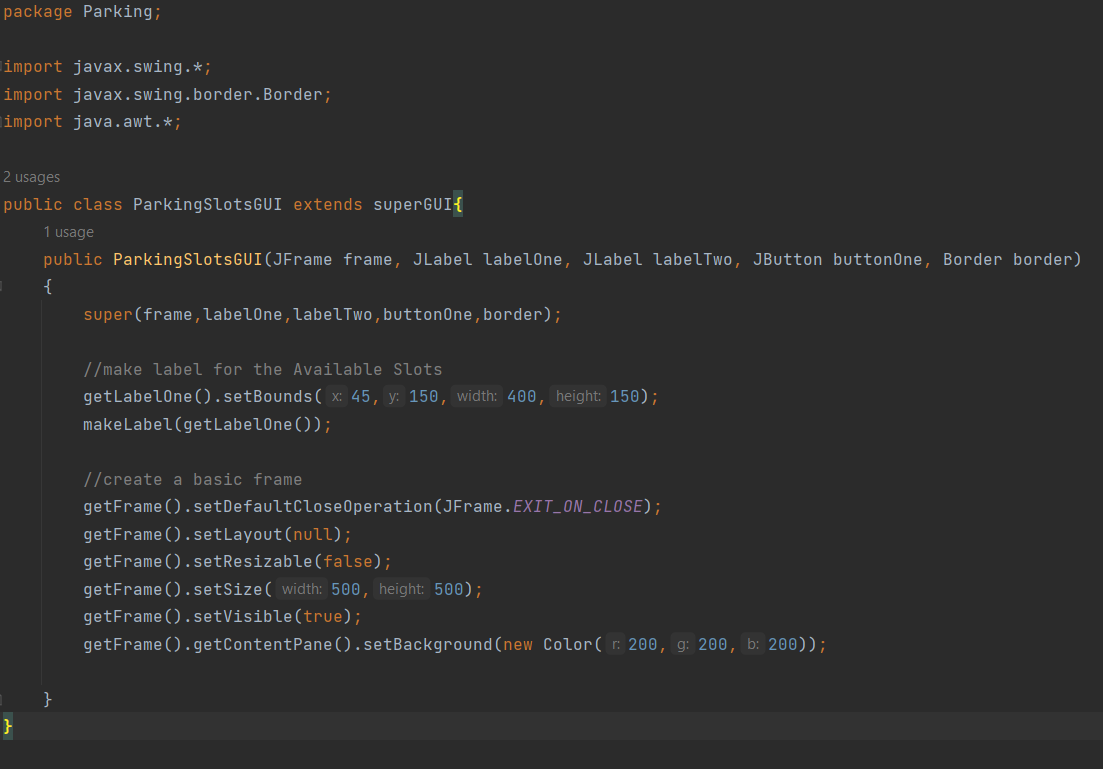
**TicketGUI.java**

The TicketGUI.java class is a child class that extends from the superGUI.java class. Its purpose is to show the initial GUI. The imports used for this class are javax.swing.\*, javax.swing.border.Border, java.awt.\*, and java.awt.event.ActionEvent. This class creates two labels, one button, and a frame using inheritance to avoid code repetition. The action listener creates a new instance of the ParkingLot.java class and makes the ticketGUI invisible.



**ParkingGUI.java**

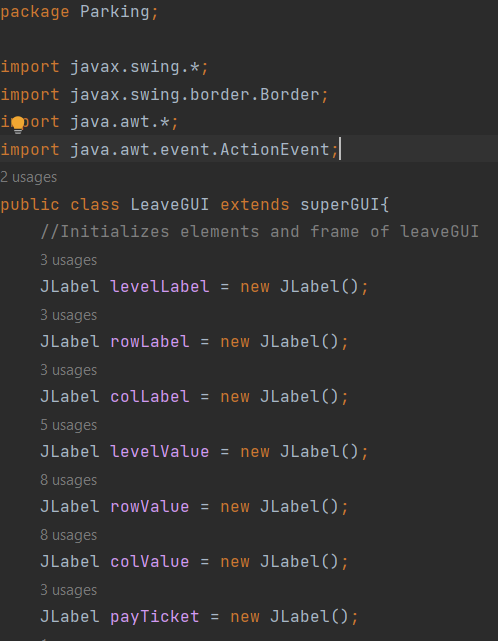
The ParkingGUI.java class creates the frame of the parking lot that includes a level JPanel and a time JPanel. The imports used in this class are javax.swing.\*, javax.swing.border.Border and java.awt.\*

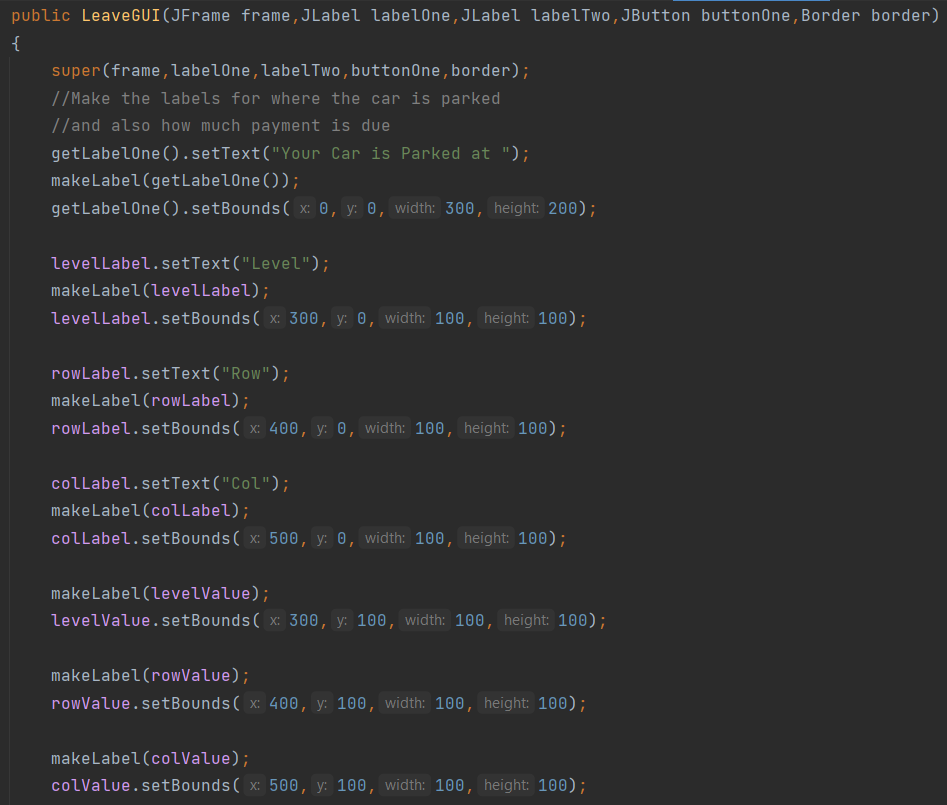
**ParkingSlotsGUI.java**

The ParkingSlotsGUI.java class creates a new GUI that displays the total available parking slots. It has one label to display the information. The imports used in this class are javax.swing.\*, javax.swing.border.Border and java.awt.\*

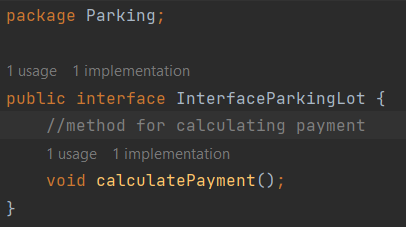
**LeaveGUI.java**

The LeaveGUI.java class creates the last GUI that allows the user to exit the program. This class makes 7 labels. The first 6 labels in the picture below, are labels to display what level, row and column the user’s car is parked on. The last label called payTicket will display “Payment due:”.The imports used are javax.swing.\*, javax.swing.border.Border, java.awt.\* and java.awt.event.ActionEvent.



The LeaveGUI uses inheritance for the frame, two labels, and the button. The first label inherited is used to display “Your Car is Parked at”, and the second label is used to display the ticket price. The button inherited is used to exit the program.

**InterfaceParkingLot.java**

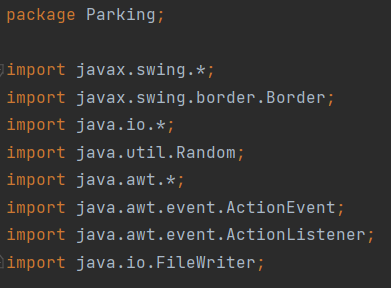
The InterfaceParkingLot.java class is an interface that has a calculatePayment( ) method. 

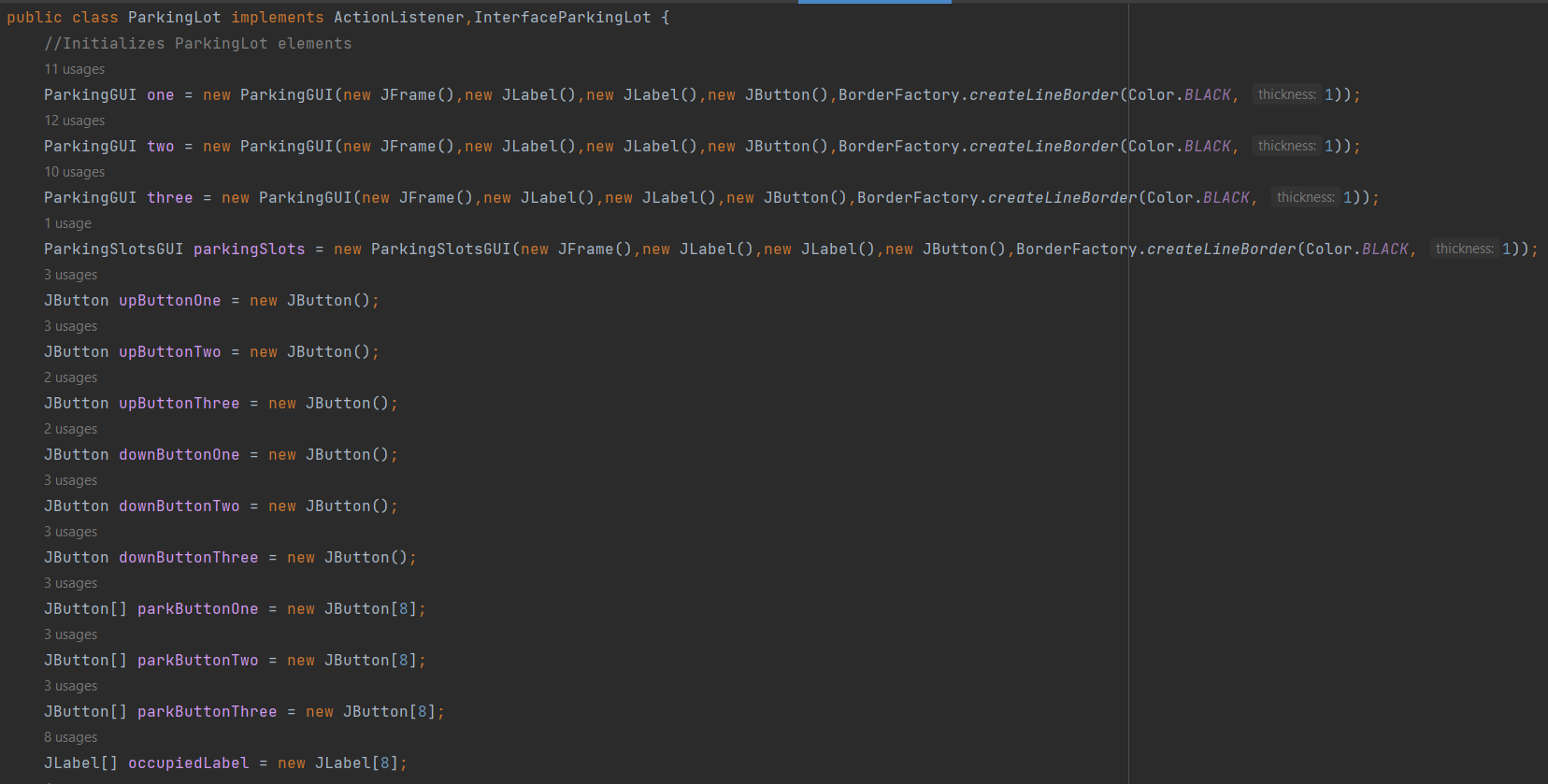
**AvailableParkingSlots.txt**

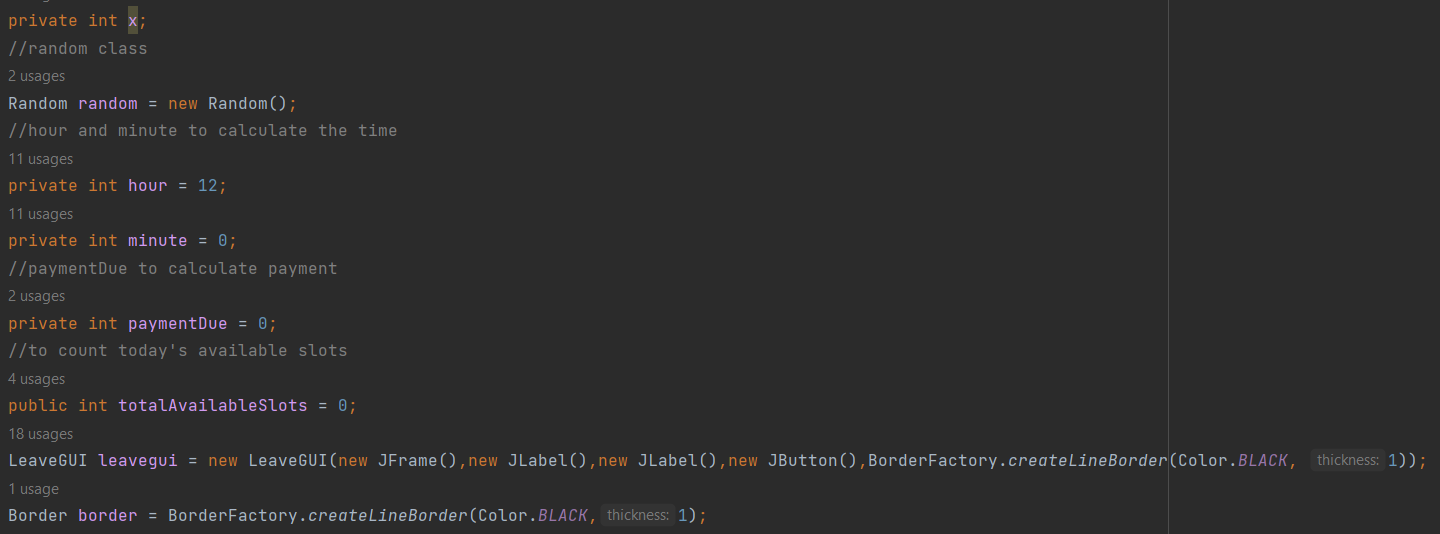
The AvailableParkingSlots.txt is a text file that keeps the information on how many available slots there are in the parking lot today. This information is updated every time the program runs.

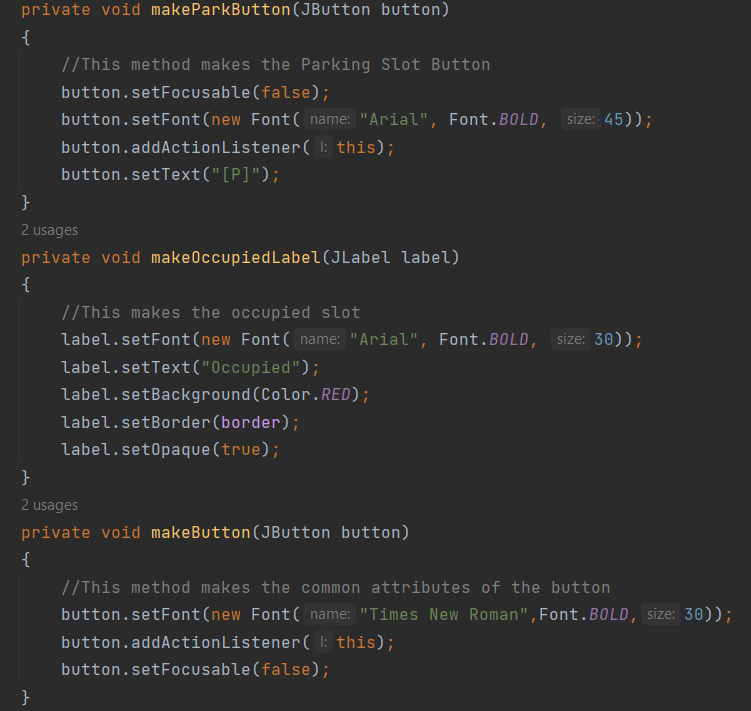
**ParkingLot.java**

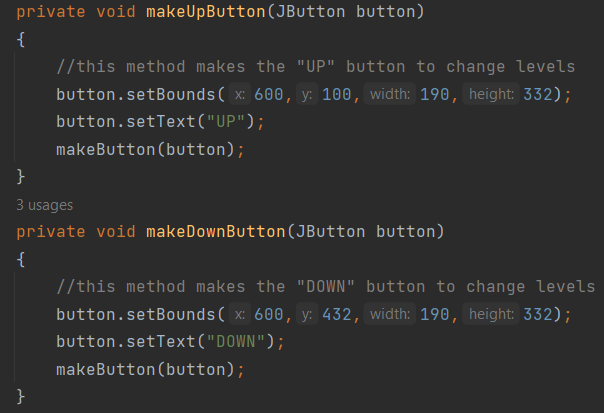
The ParkingLot.java class is where all the classes come together to create the Parking Management System. The imports used in this class are javax.swing.\*, javax.swing.border.Border, java.io.\*, java.util.Random, java.awt.\*, java.awt.event.ActionEvent, java.awt.event.ActionListener, and java.io.FileWriter.

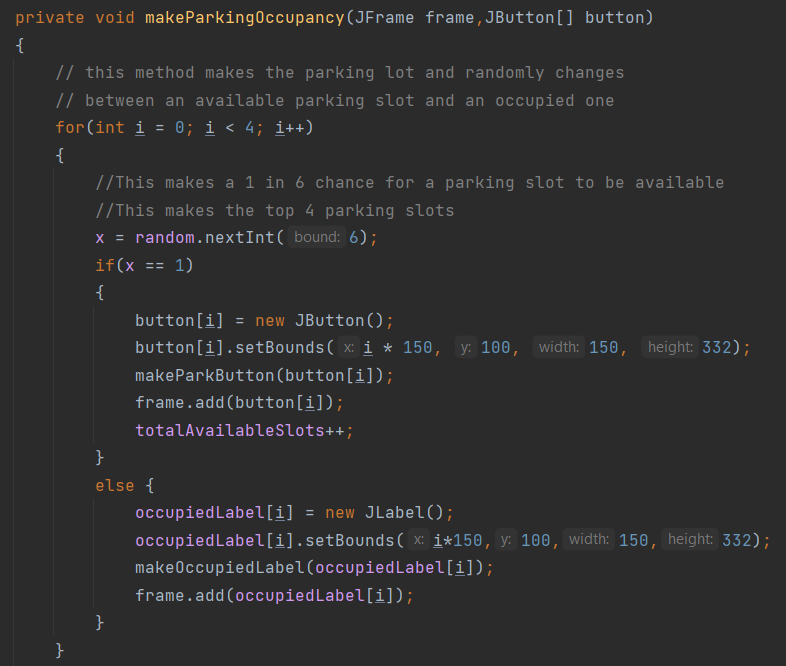
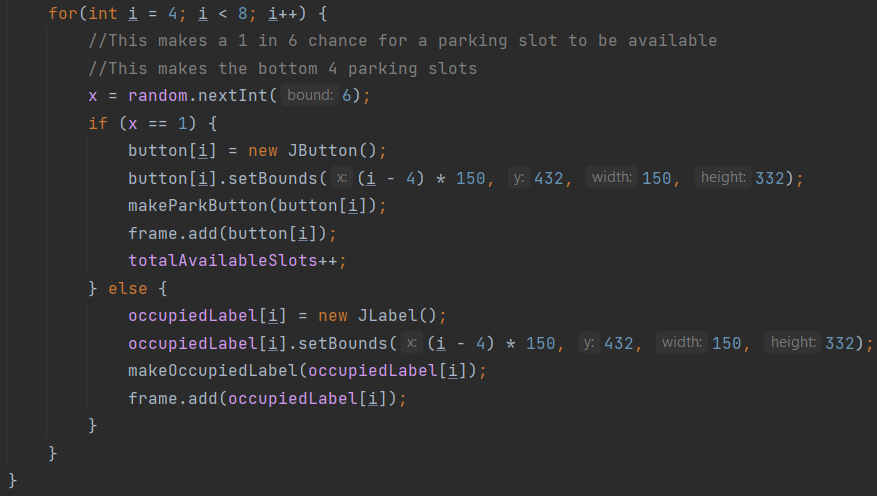


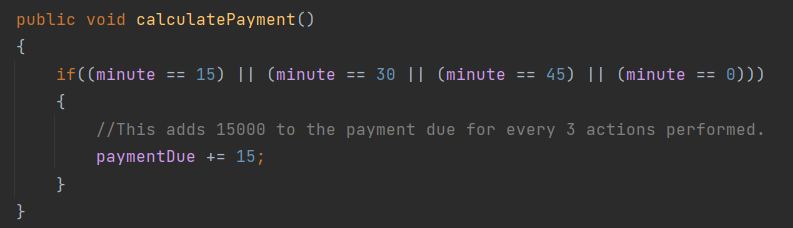
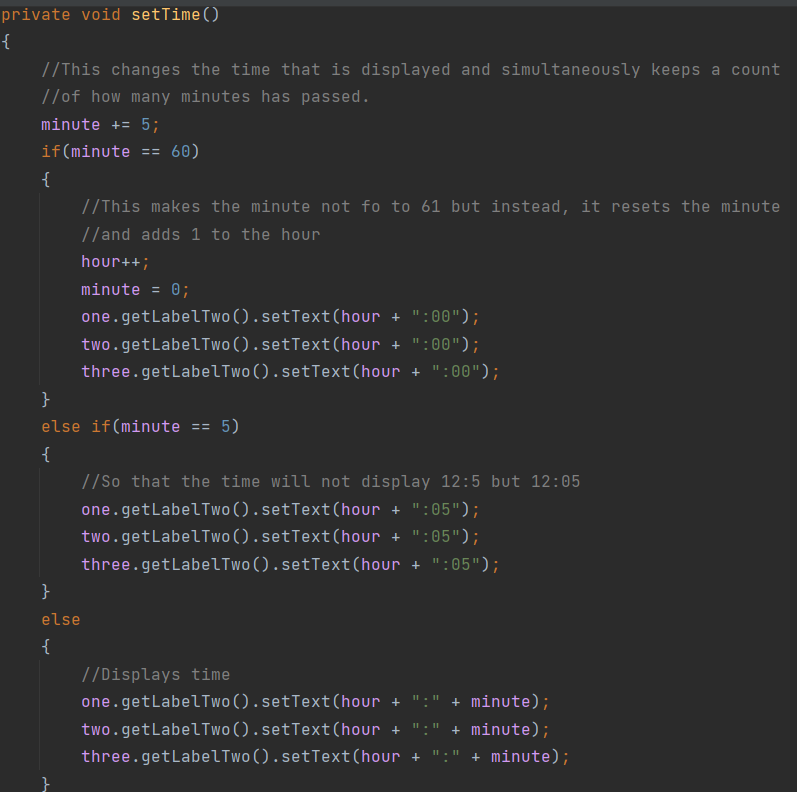
In this class, three instances of the ParkingGUI class are created in order to create three levels of the parking system. One instance of the ParkingSlotsGUI class is created to display the total available parking slots. Six buttons were also made for the user to be able to navigate through different parking levels. The Array data structure was used to make the program more compact. In this class there are three JButton arrays that will display the ability to park in a parking slot and one JLabel array that will display the occupied slots.

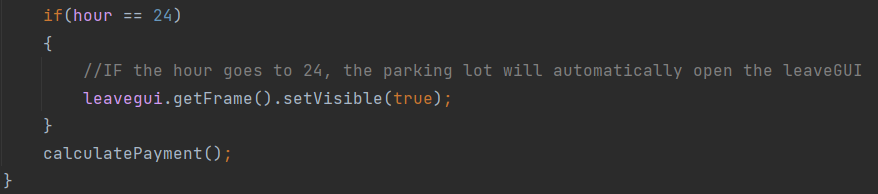
In this class, the private int x that is declared and the random variable is used to randomize the occupancy of the parking slots. The hour and minute integers are used to display the time. The paymentDue integer is used to calculate the payment ticket price depending on how long the user stays in the parking system. The totalAvailableSlots integer will display the number of available slots throughout the three levels of the parking system. An instance of the LeaveGUI class is created to prepare for the user to exit the system.

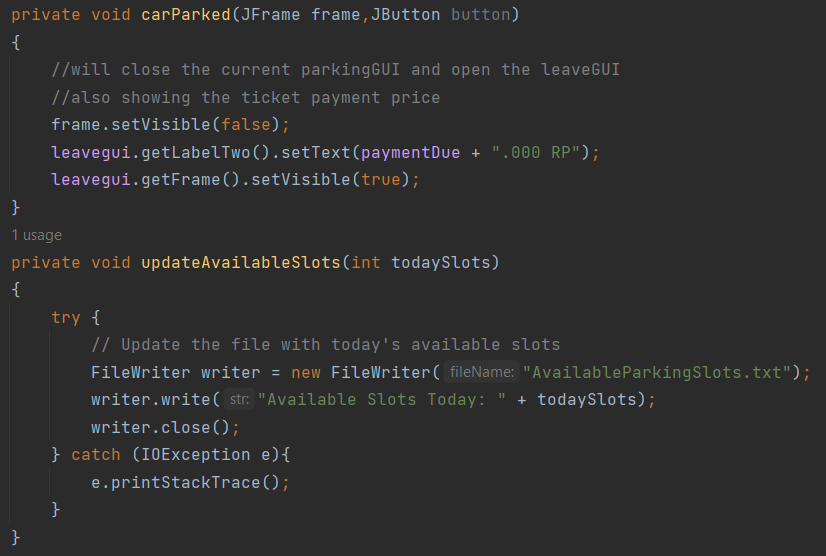
The first few methods of this class are makeParkButton(JButton button), makeOccupiedLabel(JLabel label) and makeButton(JButton button). The first method creates the attributes of all the parking buttons. The second method creates the attributes of all the occupied labels. The third method creates the attributes for the up and down buttons.

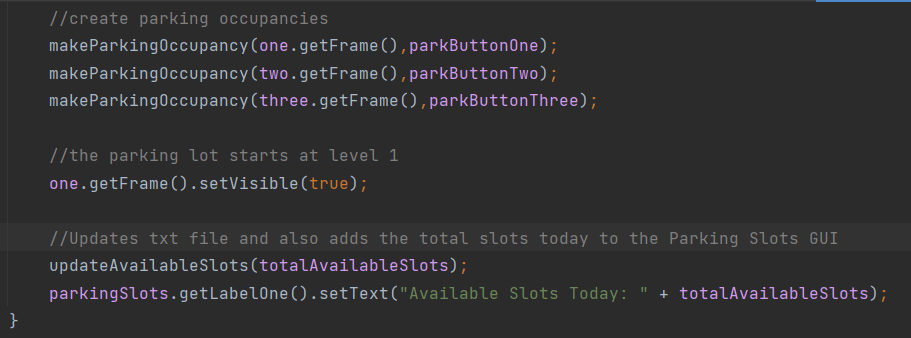
These two methods are makeUpButton(Jbutton button) and makeDownButton(JButton button). These methods are separate so that one button created will say “UP” and the other button will say “DOWN”. In these methods, the makeButton method from previously is used so that both the up and down buttons can obtain the common attributes.

This method is called makeParkingOccupancy(JFrame frame, JButton[] button). It takes two inputs because this method needs to make the parking occupancy of three floors (frame input) and 8 parking slots (button input). This method creates a 2 x 4 matrix of parking slots. This method is split into two for loops, the first for loop creates the top row of methods and creates an 1 in 6 chance of creating an available parking slot. The second for loop does the same thing, but for the bottom row. Everytime an available slot is created, it is added to the total available slots counter.

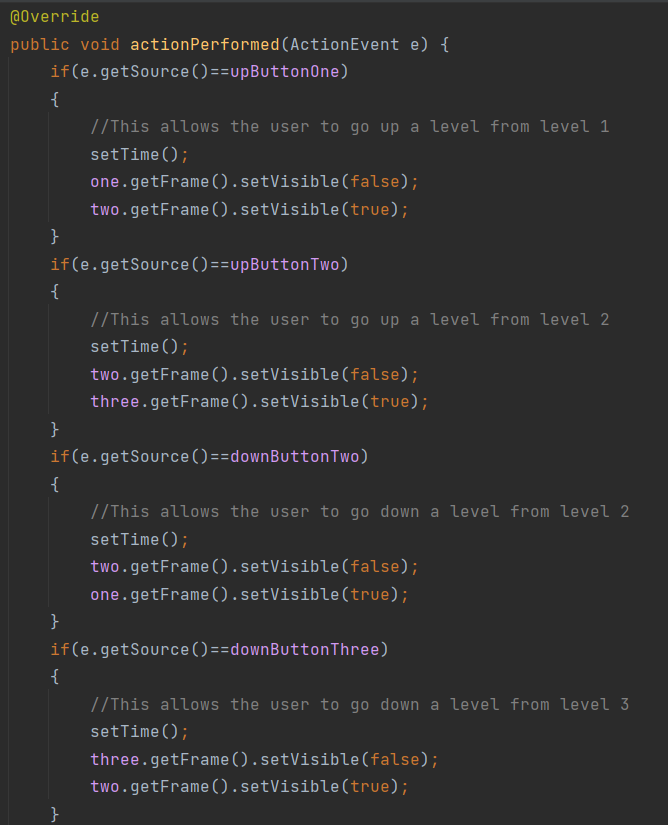
These are two methods called calculatePayment( ) and setTime( ). The calculatePayment( ) simply adds 15 to the paymentDue counter for every 15 minutes that passes by. The payment due will be displayed in thousand of rupiah.

The setTime( ) adds five minutes to the minute counter and everytime the minute count gets to 60, the minute count resets to zero and the hour count gets added by one. This method also displays the time and if the hour gets to 24, the program will immediately close the screen and open the LeaveGUI. At the end of this method, the calculatePayment ( ) is called to calculate the payment.

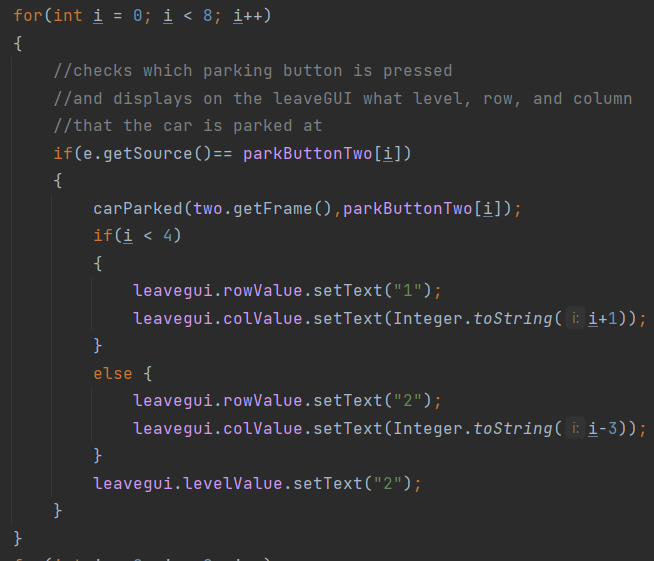
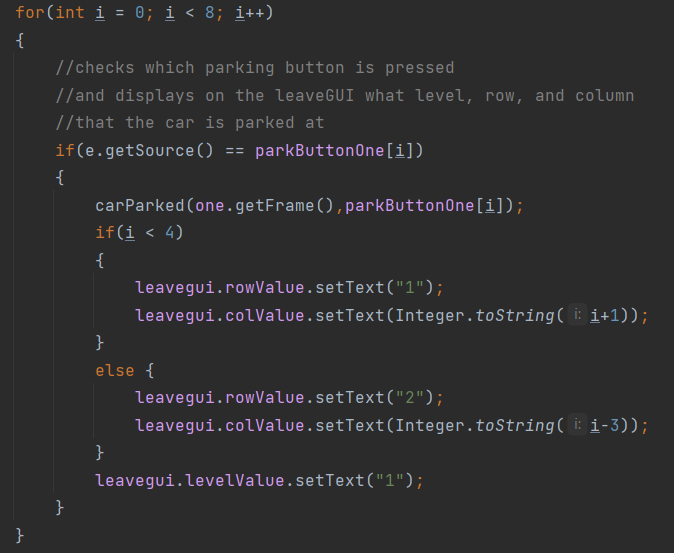
These two methods are called carParked(JFrame frame, JButton button) and updateAvailableSlots(int todaySlots). The first method is used to display the LeaveGUI, the payment due on the LeaveGUI, and close the current GUI. The second method is to update the txt file to show how many available slots there are.

The ParkingLot() combines all the previous elements and methods into one. First, it creates the up and down buttons using the makeUpButton( ) and makeDownButton( ) methods. Then, it adds these buttons to their respective parking levels (frames). Afterwards, each parking level gets different parking labels to differentiate the levels. Then, the makeParkingOccupancy( ) method is applied to each parking level and the first level of the parking system is displayed as the first GUI.

Once all the available parking slots are created, the totalAvailableSlots and both the txt files and ParkingSlotsGUI are updated.

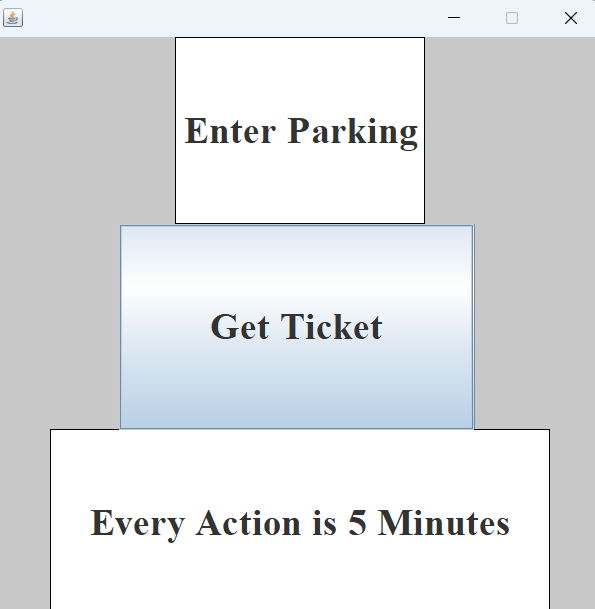


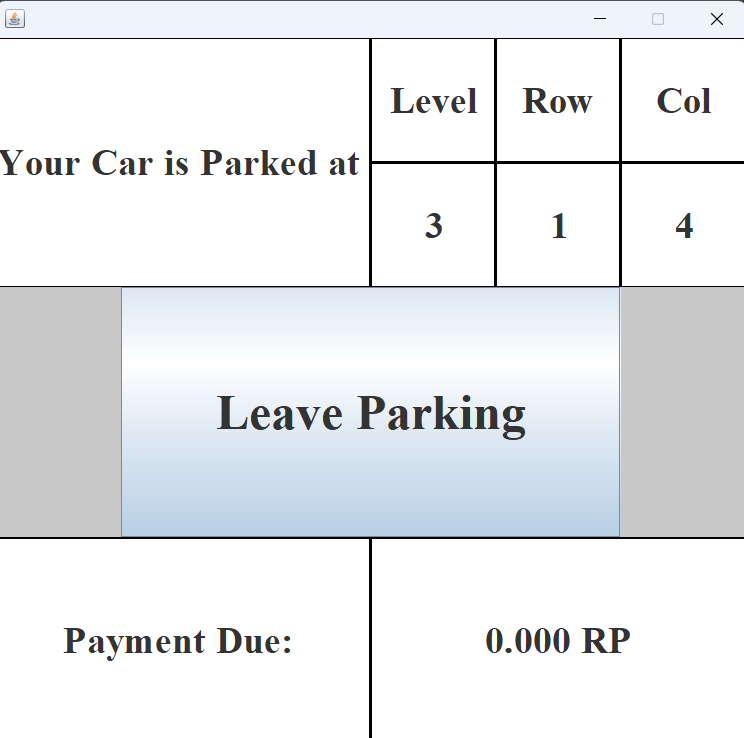
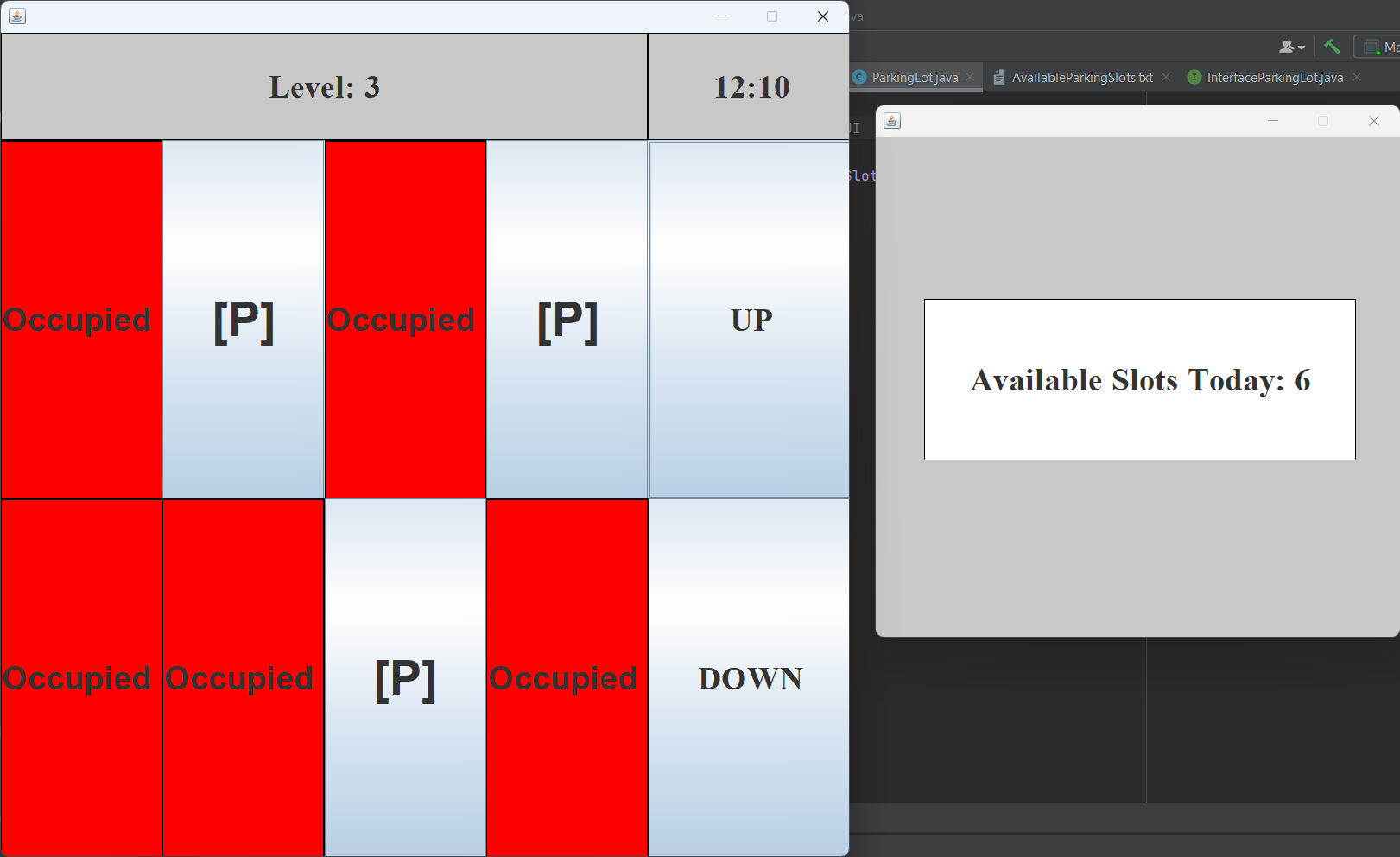
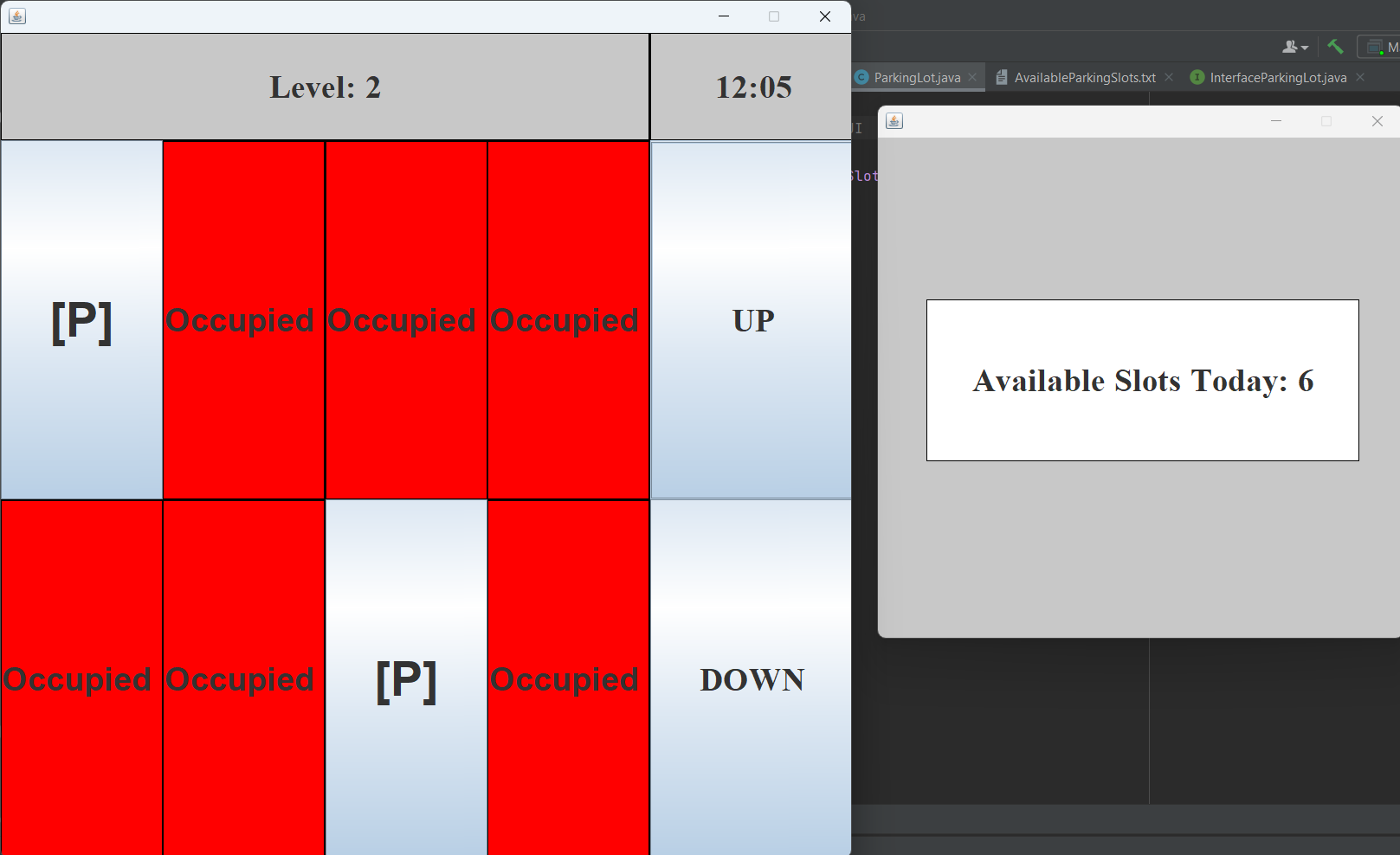
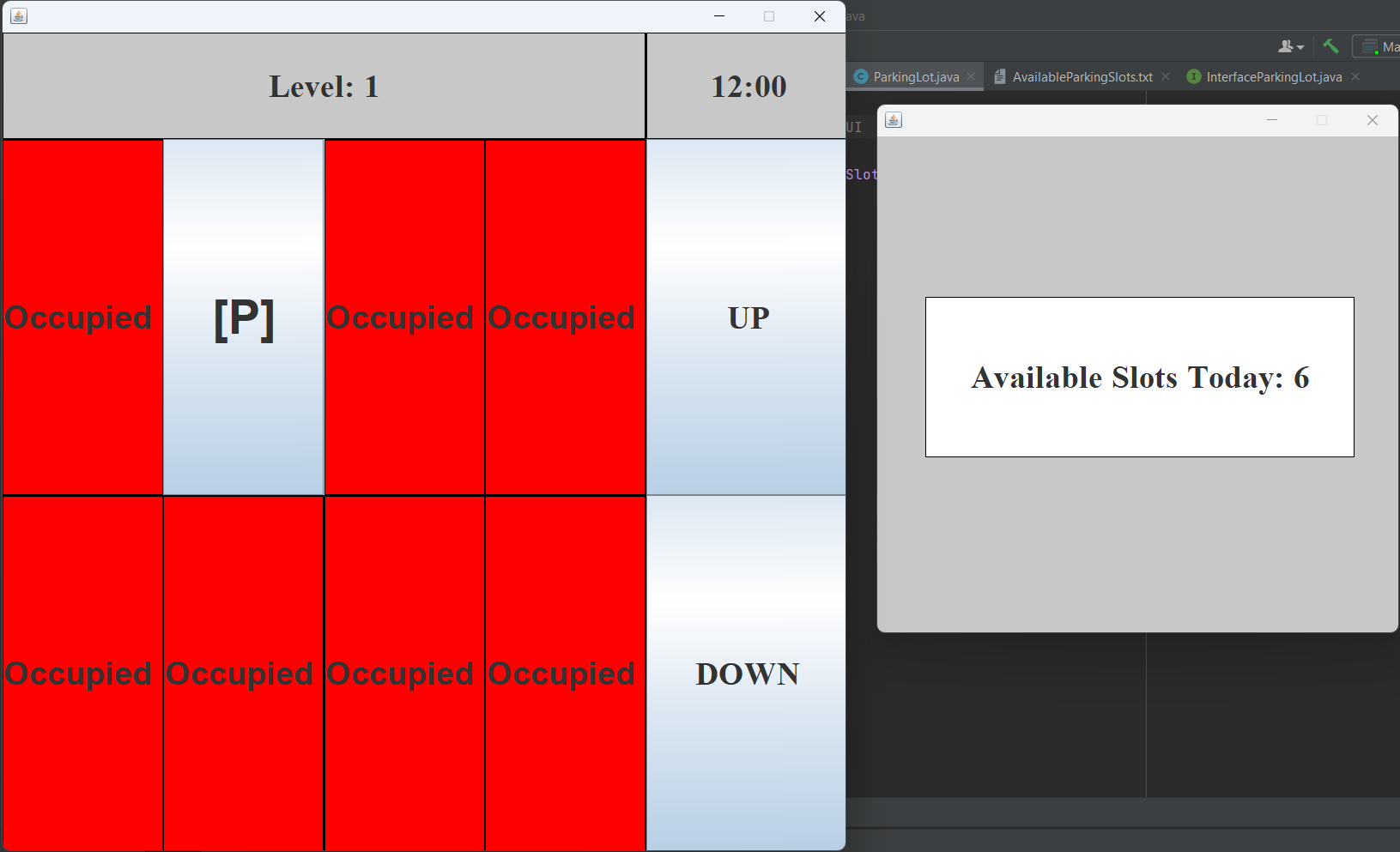
This is where all the button actions occur. These first four create actions for the up and down buttons in the different parking levels, so that users are able to navigate through them easily.

There are three for loops that are used to determine and display what level, row, and column that the user parked on. The first for loop is for when the user parks on the first level, the second for loop is for when the user parks on the second level, and the third for loop is for when the user parks on the third level.



**Evidence of Working Program**

****

****