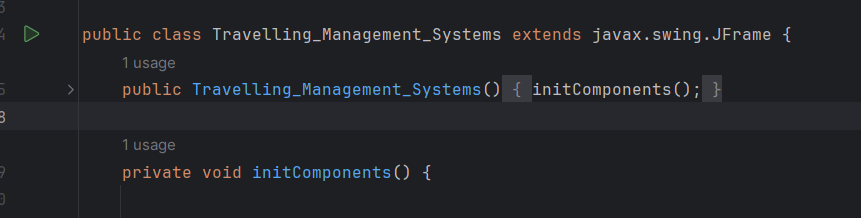
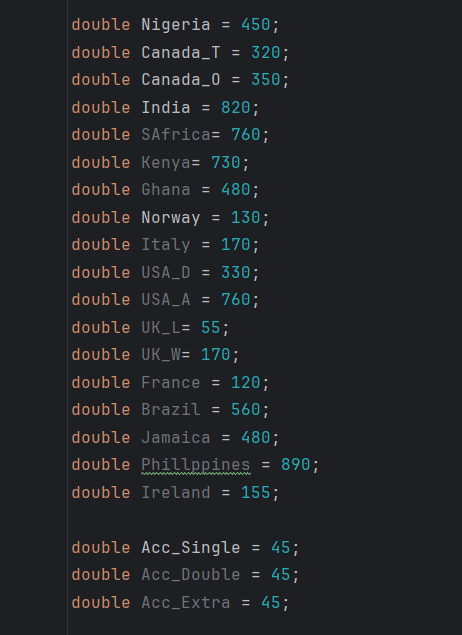
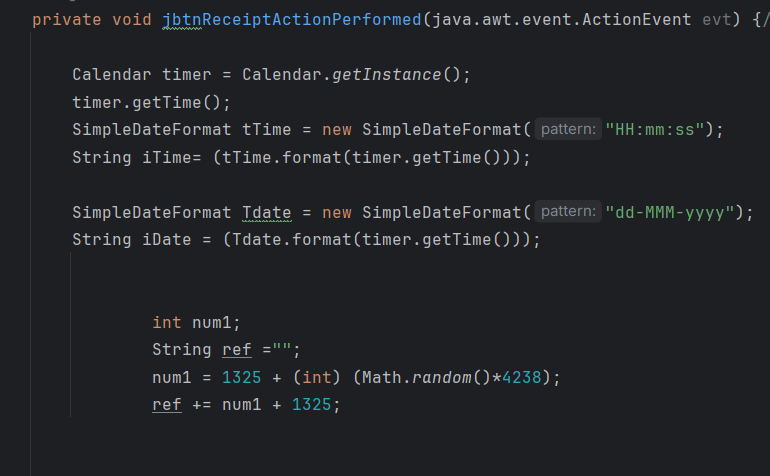
Model-View-Controller(MVC) in our project:

Model: This code represents blocks of conditions for calculating the cost of travel depending on the parameters selected by the user.









view layer:

In the code, you see many Swing components that represent the application's GUI. Each of these components serves to display information to the user or to input data from the outside:

JFrame (frame): The main window of your application. It contains all the other components.

JComboBox (jAccommodation, jDeparture, jDestination): Dropdown lists that allow the user to select options from a predefined list.

JLabel (jLabel1, jLabel11, ..., jlblTotal): Labels used to display text or titles above other components. For example, "Travel Management System", "Flight", "Ticket", etc.

JTextField (jtxtAddress, jtxtEmail, ..., jtxtSurname): Text fields for entering text information such as address, email, first name, last name and others.



Controller:

In this code, the controller is not explicitly allocated to a separate class. In MVC (Model-View-Controller) architecture, the controller is usually a separate class responsible for handling user events and managing the business logic of the application.

In our case, event-handling methods such as jbtnTotalMouseClicked, jbtnTotalMouseReleased, jbtnTotalActionPerformed, jbtnResetActionPerformed, jbtnReceiptMouseClicked and jbtnReceiptActionPerformed act as a controller. They process events from graphical components and trigger appropriate actions.