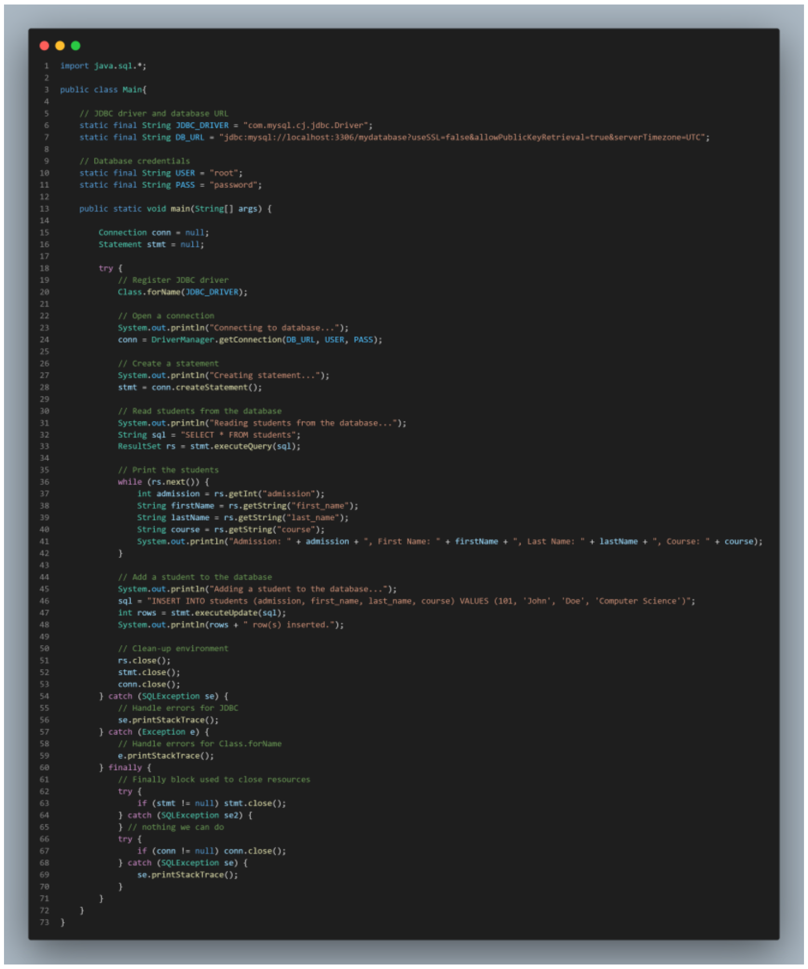
Derek Pesa

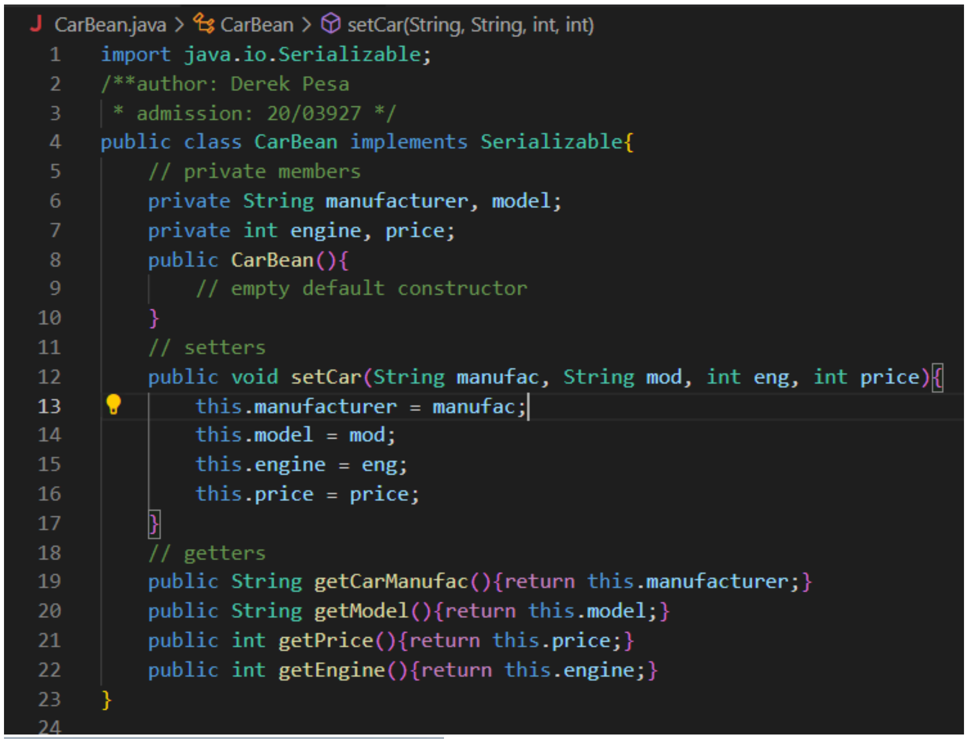
20/03927

1.



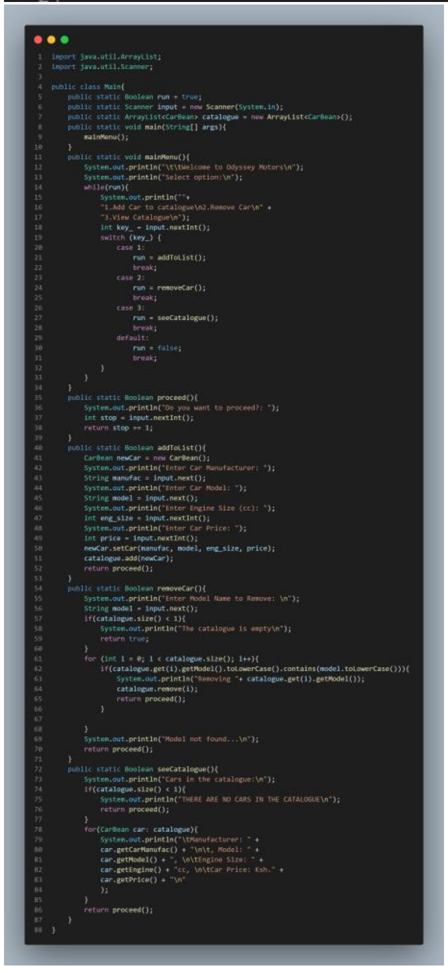
```java
import java.sql.*;

public class Main{

    // JDBC driver and database URL
    static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
    static final String DB_URL = "jdbc:mysql://localhost:3306/mydatabase?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC";

    // Database credentials
    static final String USER = "root";
    static final String PASS = "password";

    public static void main(String[] args) {

        Connection conn = null;
        Statement stmt = null;

        try {
            // Register JDBC driver
            Class.forName(JDBC_DRIVER);

            // Open a connection
            System.out.println("Connecting to database...");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);

            // Create a statement
            System.out.println("Creating statement...");
            stmt = conn.createStatement();

            // Read students from the database
            System.out.println("Reading students from the database...");
            String sql = "SELECT * FROM students";
            ResultSet rs = stmt.executeQuery(sql);

            // Print the students
            while (rs.next()) {
                int admission = rs.getInt("admission");
                String firstName = rs.getString("first_name");
                String lastName = rs.getString("last_name");
                String course = rs.getString("course");
                System.out.println("Admission: " + admission + ", First Name: " + firstName + ", Last Name: " + lastName + ", Course: " + course);
            }

            // Add a student to the database
            System.out.println("Adding a student to the database...");
            sql = "INSERT INTO students (admission, first_name, last_name, course) VALUES (101, 'John', 'Doe', 'Computer Science')";
            int rows = stmt.executeUpdate(sql);
            System.out.println(rows + " row(s) inserted.");

            // Clean-up environment
            rs.close();
            stmt.close();
            conn.close();
        } catch (SQLException se) {
            // Handle errors for JDBC
            se.printStackTrace();
        } catch (Exception e) {
            // Handle errors for Class.forName
            e.printStackTrace();
        } finally {
            // Finally block used to close resources
            try {
                if (stmt != null) stmt.close();
            } catch (SQLException se2) {
            } // nothing we can do
            try {
                if (conn != null) conn.close();
            } catch (SQLException se) {
                se.printStackTrace();
            }
        }
    }
}
```

2.



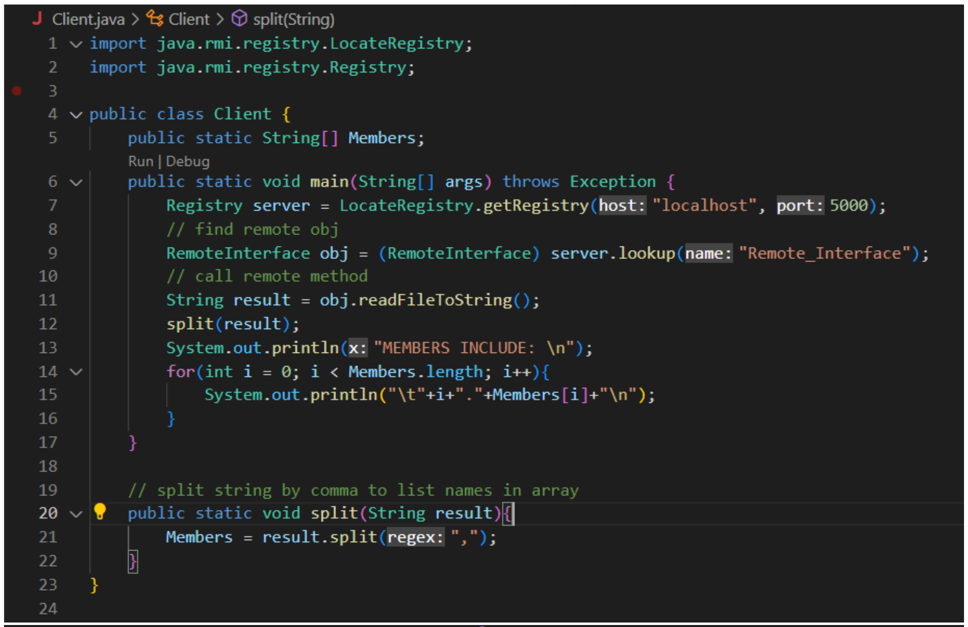
CarBean.java > CarBean > setCar(String, String, int, int)

```java
import java.io.Serializable;
/**author: Derek Pesa
 * admission: 20/03927 */
public class CarBean implements Serializable{
    // private members
    private String manufacturer, model;
    private int engine, price;
    public CarBean(){
        // empty default constructor
    }
    // setters
    public void setCar(String manufac, String mod, int eng, int price){
        this.manufacturer = manufac;
        this.model = mod;
        this.engine = eng;
        this.price = price;
    }
    // getters
    public String getCarManufac(){return this.manufacturer;}
    public String getModel(){return this.model;}
    public int getPrice(){return this.price;}
    public int getEngine(){return this.engine;}
}
```



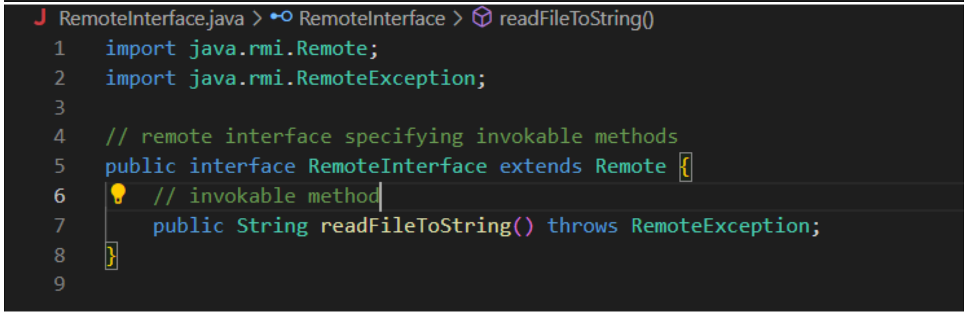
```java
import java.util.ArrayList;
import java.util.Scanner;

public class Main{
    public static Boolean run = true;
    public static Scanner input = new Scanner(System.in);
    public static ArrayList<CarBean> catalogue = new ArrayList<CarBean>();
    public static void main(String[] args){
        mainMenu();
    }
    public static void mainMenu(){
        System.out.println("\t\tWelcome to Odyssey Motors\n");
        System.out.println("Select option:\n");
        while(run){
            System.out.println(""+
            "1.Add Car to catalogue\n2.Remove Car\n" +
            "3.View Catalogue\n");
            int key_ = input.nextInt();
            switch (key_) {
                case 1:
                    run = addToList();
                    break;
                case 2:
                    run = removeCar();
                    break;
                case 3:
                    run = seeCatalogue();
                    break;
                default:
                    run = false;
                    break;
            }
        }
    }
    public static Boolean proceed(){
        System.out.println("Do you want to proceed?: ");
        int stop = input.nextInt();
        return stop == 1;
    }
    public static Boolean addToList(){
        CarBean newCar = new CarBean();
        System.out.println("Enter Car Manufacturer: ");
        String manufac = input.next();
        System.out.println("Enter Car Model: ");
        String model = input.next();
        System.out.println("Enter Engine Size (cc): ");
        int eng_size = input.nextInt();
        System.out.println("Enter Car Price: ");
        int price = input.nextInt();
        newCar.setCar(manufac, model, eng_size, price);
        catalogue.add(newCar);
        return proceed();
    }
    public static Boolean removeCar(){
        System.out.println("Enter Model Name to Remove: \n");
        String model = input.next();
        if(catalogue.size() < 1){
            System.out.println("The catalogue is empty\n");
            return true;
        }
        for (int i = 0; i < catalogue.size(); i++){
            if(catalogue.get(i).getModel().toLowerCase().contains(model.toLowerCase())){
                System.out.println("Removing "+ catalogue.get(i).getModel());
                catalogue.remove(i);
                return proceed();
            }

        }
        System.out.println("Model not found...\n");
        return proceed();
    }
    public static Boolean seeCatalogue(){
        System.out.println("Cars in the catalogue:\n");
        if(catalogue.size() < 1){
            System.out.println("THERE ARE NO CARS IN THE CATALOGUE\n");
            return proceed();
        }
        for(CarBean car: catalogue){
            System.out.println("\tManufacturer: " +
            car.getCarManufac() + "\n\t, Model: " +
            car.getModel() + ", \n\tEngine Size: " +
            car.getEngine() + "cc, \n\tCar Price: Ksh." +
            car.getPrice() + "\n"
            );
        }
        return proceed();
    }
}
```

3.



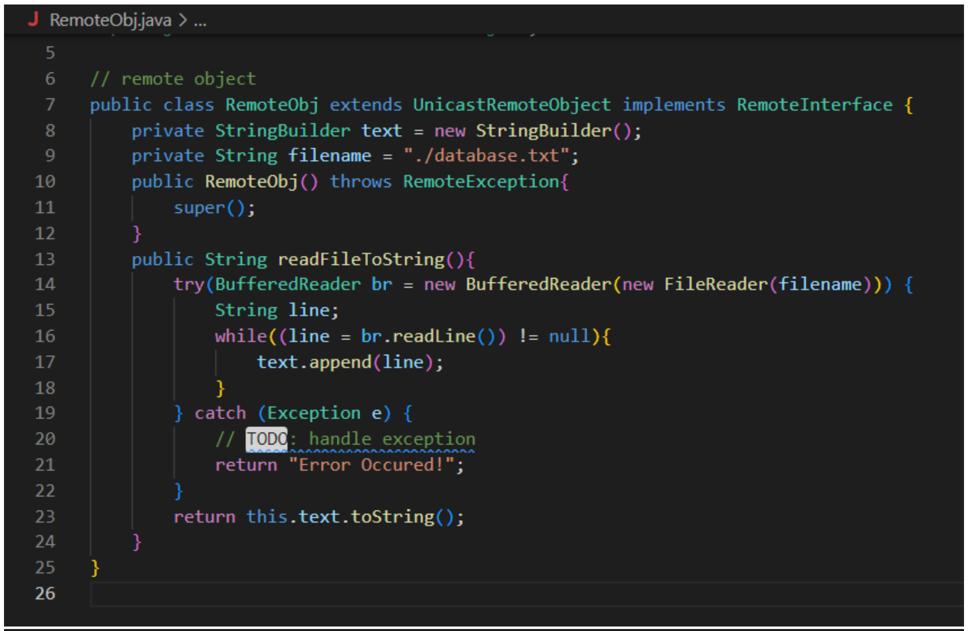
Client.java > Client > split(String)

```java
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Client {
    public static String[] Members;
    Run | Debug
    public static void main(String[] args) throws Exception {
        Registry server = LocateRegistry.getRegistry(host: "localhost", port: 5000);
        // find remote obj
        RemoteInterface obj = (RemoteInterface) server.lookup(name: "Remote_Interface");
        // call remote method
        String result = obj.readFileToString();
        split(result);
        System.out.println(x: "MEMBERS INCLUDE: \n");
        for(int i = 0; i < Members.length; i++){
            System.out.println("\t"+i+"."+Members[i]+"\n");
        }
    }

    // split string by comma to list names in array
    public static void split(String result){
        Members = result.split(regex: ",");
    }
}
```

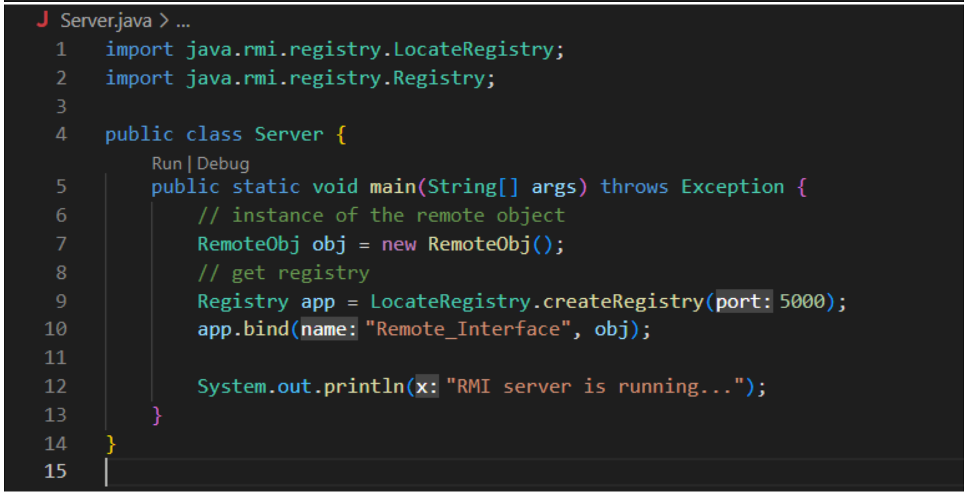


RemoteInterface.java > RemoteInterface > readFileToString()

```java
import java.rmi.Remote;
import java.rmi.RemoteException;

// remote interface specifying invokable methods
public interface RemoteInterface extends Remote {
    // invokable method
    public String readFileToString() throws RemoteException;
}
```

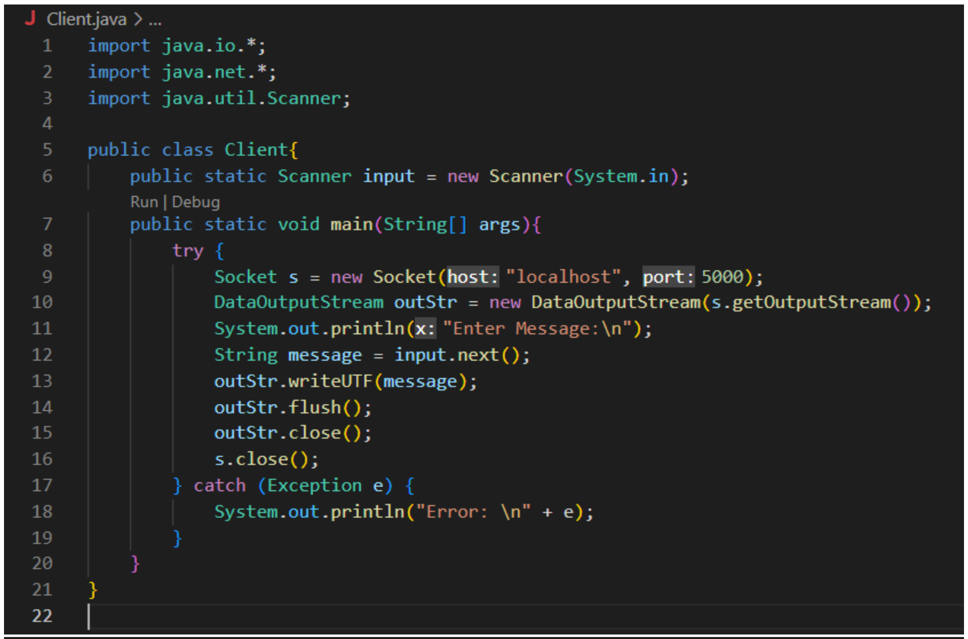

RemoteObj.java > ...

```java

// remote object
public class RemoteObj extends UnicastRemoteObject implements RemoteInterface {
    private StringBuilder text = new StringBuilder();
    private String filename = "./database.txt";
    public RemoteObj() throws RemoteException{
        super();
    }
    public String readFileToString(){
        try(BufferedReader br = new BufferedReader(new FileReader(filename))) {
            String line;
            while((line = br.readLine()) != null){
                text.append(line);
            }
        } catch (Exception e) {
            // TODO: handle exception
            return "Error Occured!";
        }
        return this.text.toString();
    }
}

```

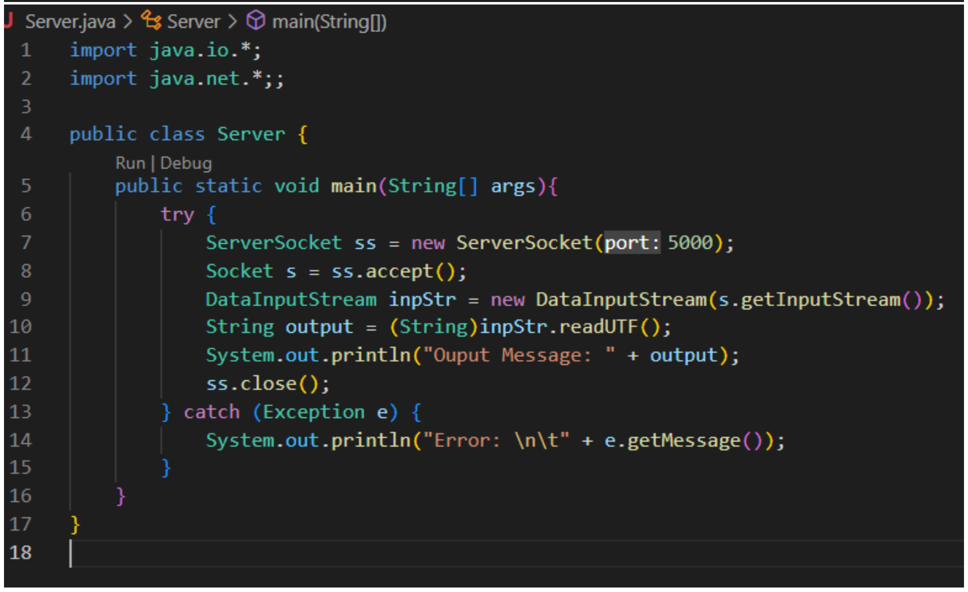

Server.java > ...

```java
import java.rmi.registry.LocateRegistry;
import java.rmi.registry.Registry;

public class Server {
    Run | Debug
    public static void main(String[] args) throws Exception {
        // instance of the remote object
        RemoteObj obj = new RemoteObj();
        // get registry
        Registry app = LocateRegistry.createRegistry(port: 5000);
        app.bind(name: "Remote_Interface", obj);

        System.out.println(x: "RMI server is running...");
    }
}

```

4.

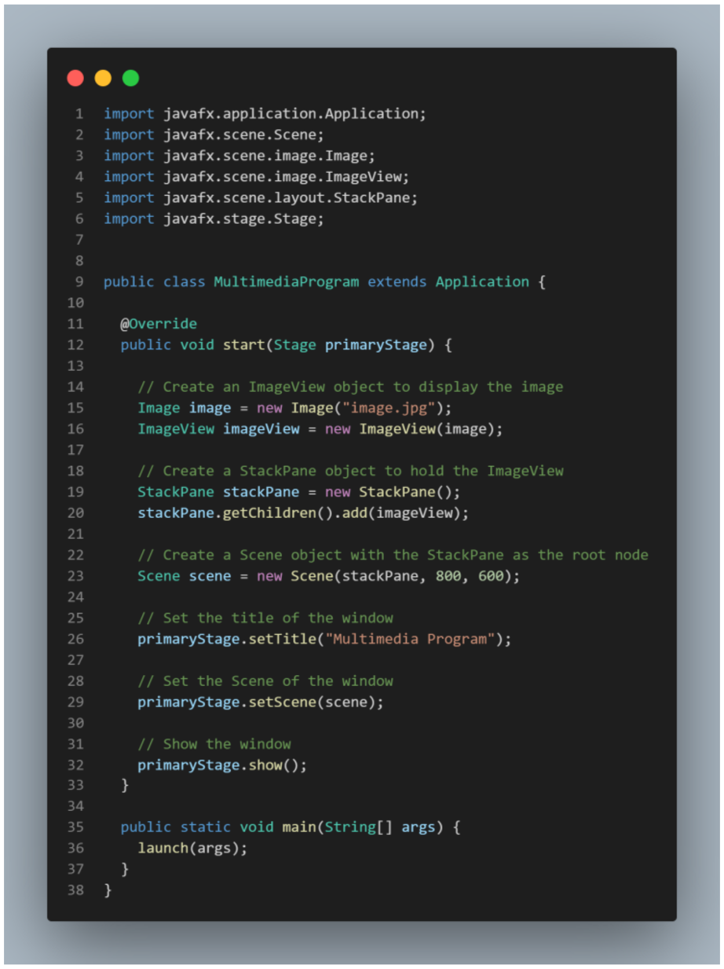

Client.java > ...

```java
import java.io.*;
import java.net.*;
import java.util.Scanner;

public class Client{
    public static Scanner input = new Scanner(System.in);
    Run | Debug
    public static void main(String[] args){
        try {
            Socket s = new Socket(host: "localhost", port: 5000);
            DataOutputStream outStr = new DataOutputStream(s.getOutputStream());
            System.out.println(x: "Enter Message:\n");
            String message = input.next();
            outStr.writeUTF(message);
            outStr.flush();
            outStr.close();
            s.close();
        } catch (Exception e) {
            System.out.println("Error: \n" + e);
        }
    }
}
```


Server.java > Server > main(String[])

```java
import java.io.*;
import java.net.*;;

public class Server {
    Run | Debug
    public static void main(String[] args){
        try {
            ServerSocket ss = new ServerSocket(port: 5000);
            Socket s = ss.accept();
            DataInputStream inpStr = new DataInputStream(s.getInputStream());
            String output = (String)inpStr.readUTF();
            System.out.println("Ouput Message: " + output);
            ss.close();
        } catch (Exception e) {
            System.out.println("Error: \n\t" + e.getMessage());
        }
    }
}
```

5.



```java
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;


public class MultimediaProgram extends Application {

  @Override
  public void start(Stage primaryStage) {

    // Create an ImageView object to display the image
    Image image = new Image("image.jpg");
    ImageView imageView = new ImageView(image);

    // Create a StackPane object to hold the ImageView
    StackPane stackPane = new StackPane();
    stackPane.getChildren().add(imageView);

    // Create a Scene object with the StackPane as the root node
    Scene scene = new Scene(stackPane, 800, 600);

    // Set the title of the window
    primaryStage.setTitle("Multimedia Program");

    // Set the Scene of the window
    primaryStage.setScene(scene);

    // Show the window
    primaryStage.show();
  }

  public static void main(String[] args) {
    launch(args);
  }
}
```