MID TERM EXAM

-PARAMPREET SINGH

-200535733

CODES-

CARS.JAVA-

|  |
| --- |
| package com.example.midtermexam;    import javafx.application.Application; import javafx.fxml.FXMLLoader; import javafx.scene.Scene; import javafx.scene.image.Image; import javafx.scene.layout.Pane; import javafx.stage.Stage;    public class CarApp extends Application {  @Override  public void start(Stage primaryStage) throws Exception { FXMLLoader loader = new FXMLLoader();  loader.setLocation(getClass().getResource("/fxml/carview.fxml"));  Pane root = loader.load();  Scene scene = new Scene(root, 1000, 800);    primaryStage.setTitle("Car Lot");  primaryStage.getIcons().add(new Image("/images/car.png")); primaryStage.setScene(scene); primaryStage.show();  } public static void main(String[] args) {  Application.*launch*(args);  }  } |

CARDBMANAGER.JAVA-

|  |
| --- |
| package com.example.midtermexam;    import java.sql.\*; import java.time.LocalDate; import java.util.ArrayList; import java.util.List;    public class CarDBManager { |

|  |  |  |  |
| --- | --- | --- | --- |
| private Connection connection;    public CarDBManager(String url, String username, String password) throws SQLException {  this.connection = DriverManager.*getConnection*(url, username, password);  }  public List<Car> fetchAllCars() throws SQLException {  List<Car> cars = new ArrayList<>();   |  | | --- | | SELECT \* FROM carSales |   String sqlQuery = " ";  try (Statement stmt = this.connection.createStatement()) {  ResultSet rs = stmt.executeQuery(sqlQuery); while (rs.next()) {  Car car = extractCarFromResultSet(rs); cars.add(car);  }  rs.close();  }  return cars;  }  public List<Car> getCarsByYear(int year) throws SQLException {  List<Car> cars = new ArrayList<>();   |  | | --- | | SELECT \* FROM carSales WHERE modelYear = ? |   String sqlQuery = " "; try (PreparedStatement stmt = this.connection.prepareStatement(sqlQuery)) { stmt.setInt(1, year);  ResultSet rs = stmt.executeQuery();  while (rs.next()) {  Car car = extractCarFromResultSet(rs); cars.add(car);  }  rs.close();  }  return cars;  }  public List<Integer> getAllYears() throws SQLException {  List<Integer> years = new ArrayList<>();   |  | | --- | | SELECT DISTINCT modelYear FROM carSales |   String sqlQuery = " ";  try (Statement stmt = this.connection.createStatement()) {  ResultSet rs = stmt.executeQuery(sqlQuery); while (rs.next()) {  int year = rs.getInt("modelYear"); years.add(year);  }  rs.close();  }  return years;  }  private Car extractCarFromResultSet(ResultSet rs) throws SQLException { int carID = rs.getInt("carID");  int modelYear = rs.getInt("modelYear"); String make = rs.getString("make"); String model = rs.getString("model"); int price = rs.getInt("price");  LocalDate dateSold = rs.getDate("dateSold").toLocalDate(); return new Car(carID, modelYear, make, model, price, dateSold); |

}

}

CARSAPP.JAVA-

|  |
| --- |
| package com.example.midtermexam;    import javafx.application.Application; import javafx.fxml.FXMLLoader; import javafx.scene.Scene; import javafx.scene.layout.Pane; import javafx.stage.Stage;  import org.springframework.boot.autoconfigure.SpringBootApplication; import org.springframework.boot.builder.SpringApplicationBuilder; import org.springframework.context.ConfigurableApplicationContext;  @SpringBootApplication  public class CarsApp extends Application {  private ConfigurableApplicationContext springContext; private FXMLLoader fxmlLoader;    public static void main(String[] args) {  Application.*launch*(args);  }    @Override  public void init() {  springContext = new SpringApplicationBuilder(CarsApp.class).run(); fxmlLoader = new FXMLLoader();  fxmlLoader.setControllerFactory(springContext::getBean); }    @Override  public void start(Stage primaryStage) throws Exception {    fxmlLoader.setLocation(getClass().getResource("/fxml/carview.fxml"));  Pane root = fxmlLoader.load();  Scene scene = new Scene(root, 1000, 800);    primaryStage.setTitle("Car Lot"); primaryStage.setScene(scene); primaryStage.show();  }    @Override  public void stop() { springContext.stop();  }  } |

CARS.JAVA-

|  |
| --- |
| package com.example.midtermexam;    import javafx.beans.property.\*;    import java.time.LocalDate;    public class Cars {  private IntegerProperty carID; private IntegerProperty modelYear; private StringProperty make; private StringProperty model; private IntegerProperty price;  private ObjectProperty<LocalDate> dateSold;    public Cars(int carID, int modelYear, String make, String model, int price, LocalDate dateSold) { validateCarID(carID); validateMake(make); validateModel(model); validatePrice(price); validateDateSold(dateSold);    this.carID = new SimpleIntegerProperty(carID); this.modelYear = new SimpleIntegerProperty(modelYear); this.make = new SimpleStringProperty(make); this.model = new SimpleStringProperty(model); this.price = new SimpleIntegerProperty(price); this.dateSold = new SimpleObjectProperty<>(dateSold); }  private void validateCarID(int carID) { if (carID <= 0) {  throw new IllegalArgumentException("Car ID should be greater than 0.");  }  }  private void validateMake(String make) {  String[] validMakes = {"Acura", "Ford", "Honda", "Nissan",  "Tesla"};  boolean isValid = false;  for (String validMake : validMakes) { if (validMake.equals(make)) { isValid = true; break;  } }  if (!isValid) {  throw new IllegalArgumentException("Invalid make. Valid makes are: Acura, Ford, Honda, Nissan, Tesla."); }  } private void validateModel(String model) { if (model.length() < 2) {  throw new IllegalArgumentException("Model should be at least 2 characters long.");  }  }  private void validatePrice(int price) { |

|  |
| --- |
| if (price <= 0) {  throw new IllegalArgumentException("Price should be greater than 0.");  }  }  private void validateDateSold(LocalDate dateSold) { LocalDate currentDate = LocalDate.*now*(); if (dateSold.isAfter(currentDate)) {  throw new IllegalArgumentException("Invalid date sold. Date cannot be in the future.");  }  }    // Getters and property access methods...  public int getCarID() { return carID.get();  }  public IntegerProperty carIDProperty() { return carID;  } public int getModelYear() { return modelYear.get();  }  public IntegerProperty modelYearProperty() { return modelYear;  } public String getMake() { return make.get();  }  public StringProperty makeProperty() { return make;  } public String getModel() { return model.get();  } public StringProperty modelProperty() { return model;  } public int getPrice() { return price.get();  } public IntegerProperty priceProperty() { return price;  } public LocalDate getDateSold() { return dateSold.get();  }  public ObjectProperty<LocalDate> dateSoldProperty() { return dateSold; |

} }

carsview.fxml-

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>    <?import javafx.scene.chart.\*?>  <?import javafx.scene.control.\*?>  <?import javafx.scene.layout.\*?>    <GridPane xmlns="http://javafx.com/javafx/17.0.2-ea" xmlns:fx="http://javafx.com/fxml/1"  fx:controller="com.example.midtermexam.CarsController">  <children>  <TableView fx:id="carTable" GridPane.rowIndex="0"  GridPane.columnSpan="2">  <columns>  <TableColumn fx:id="carIDColumn" text="Car ID" />  <TableColumn fx:id="modelYearColumn" text="Year" />  <TableColumn fx:id="makeColumn" text="Make" />  <TableColumn fx:id="modelColumn" text="Model" />  <TableColumn fx:id="priceColumn" text="Price" />  <TableColumn fx:id="dateSoldColumn" text="Date Sold" />  </columns>  </TableView>    <HBox alignment="CENTER" spacing="10" GridPane.rowIndex="1">  <Label text="Select Year: " />  <ComboBox fx:id="yearComboBox" />  </HBox>  <HBox alignment="CENTER" spacing="10" GridPane.rowIndex="2">  <Label text="Total Cars Sold: " /> <Label fx:id="totalCarsLabel" />  </HBox>  <HBox alignment="CENTER" spacing="10" GridPane.rowIndex="3">  <Label text="Total Sales: " />  <Label fx:id="totalSalesLabel" />  </HBox>  <BarChart fx:id="barChart" prefHeight="400.0" prefWidth="454.0"  GridPane.rowIndex="4" GridPane.columnSpan="2">  <xAxis>  <CategoryAxis />  </xAxis>  <yAxis>  <NumberAxis />  </yAxis>  </BarChart>  </children>  </GridPane> |