

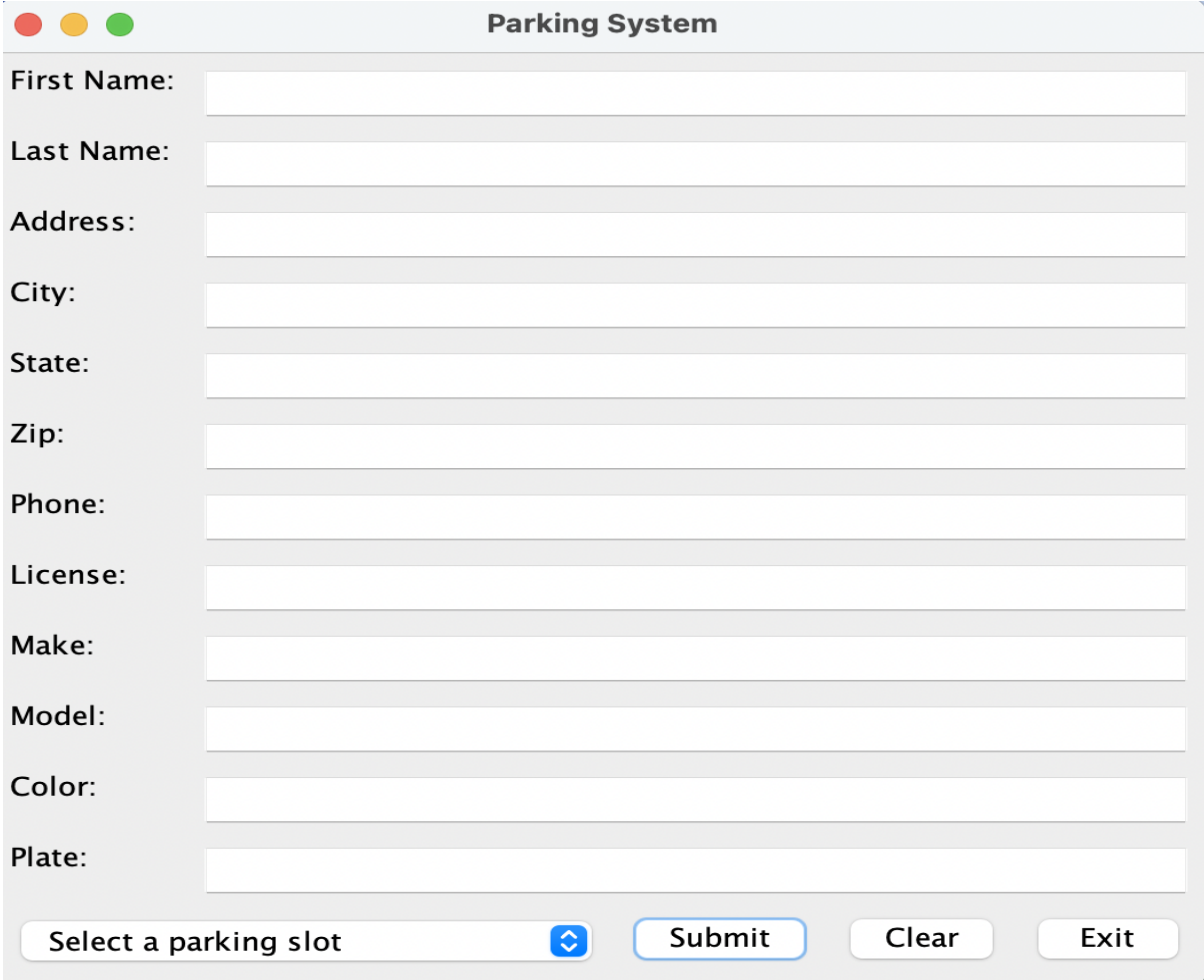
OOAD using JAVA: Assignment - 1

Demonstrate MVC architecture

Name : Abhishek Aditya BS
SRN : PES1UG19CS019
Semester : VI 'A' section

Parking System as MVC application :

We will be modelling a Parking System portal as a MVC framework. The portal will consist of a front end built using Java Swing. The front end will accept user details like First Name, Last Name, Address, Licence Number, Car Details and so on. The user will then submit their request for the parking slot by selecting the parking slot number and then by clicking the Submit button.



The screenshot shows a Java Swing window titled "Parking System". The window contains a form with the following fields and controls:

- First Name:
- Last Name:
- Address:
- City:
- State:
- Zip:
- Phone:
- License:
- Make:
- Model:
- Color:
- Plate:

At the bottom of the window, there is a row of controls:

- A dropdown menu with the text "Select a parking slot" and a downward arrow icon.
- A "Submit" button.
- A "Clear" button.
- An "Exit" button.

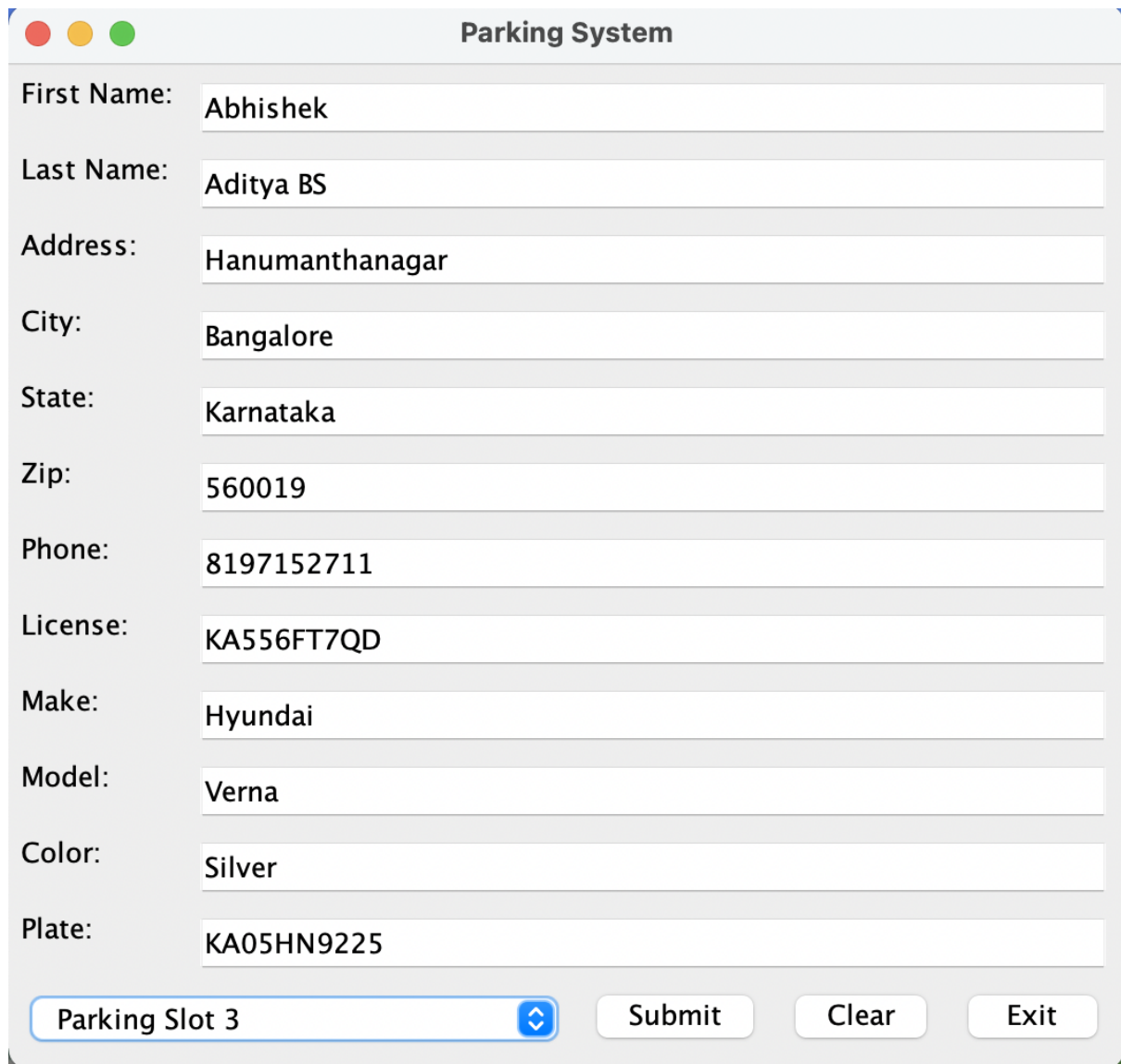
Figure : View of Parking Slot Booking Portal built using Swing UI.

Once the user clicks the submit button, a request will be sent to the model layer by the controller of the MVC framework. The model layer consists of the business logic of the application. In this assignment, the business logic is just accepting the details from the user and checking if a parking slot is available or not and then storing the details in the MongoDB database. Upon submission, the form will be reset by clearing all the fields. The user can then submit a new request.


DEMO

Submitting a new application on the portal.

The user will need to enter all the requested details in the corresponding text boxes.



First Name:	Abhishek
Last Name:	Aditya BS
Address:	Hanumanthanagar
City:	Bangalore
State:	Karnataka
Zip:	560019
Phone:	8197152711
License:	KA556FT7QD
Make:	Hyundai
Model:	Verna
Color:	Silver
Plate:	KA05HN9225

Parking Slot 3 

Submit Clear Exit

When the user clicks submit a database entry will be made with the entered details.

```
test> use parkingSystem
switched to db parkingSystem
parkingSystem> db.users.find()
[
  {
    _id: ObjectId("623f76b63daaa915b53035eb"),
    firstname: 'Abhishek ',
    lastname: 'Aditya BS',
    address: 'Hanumanthanagar',
    city: 'Bangalore',
    state: 'Karnataka',
    zip: '560019',
    phone: '8197152711',
    license: 'KA556FT7QD',
    make: 'Hyundai',
    model: 'Verna',
    color: 'Silver',
    plate: 'KA05HN9225',
    parkingslot: 'Parking Slot 3'
  }
]
parkingSystem> 
```

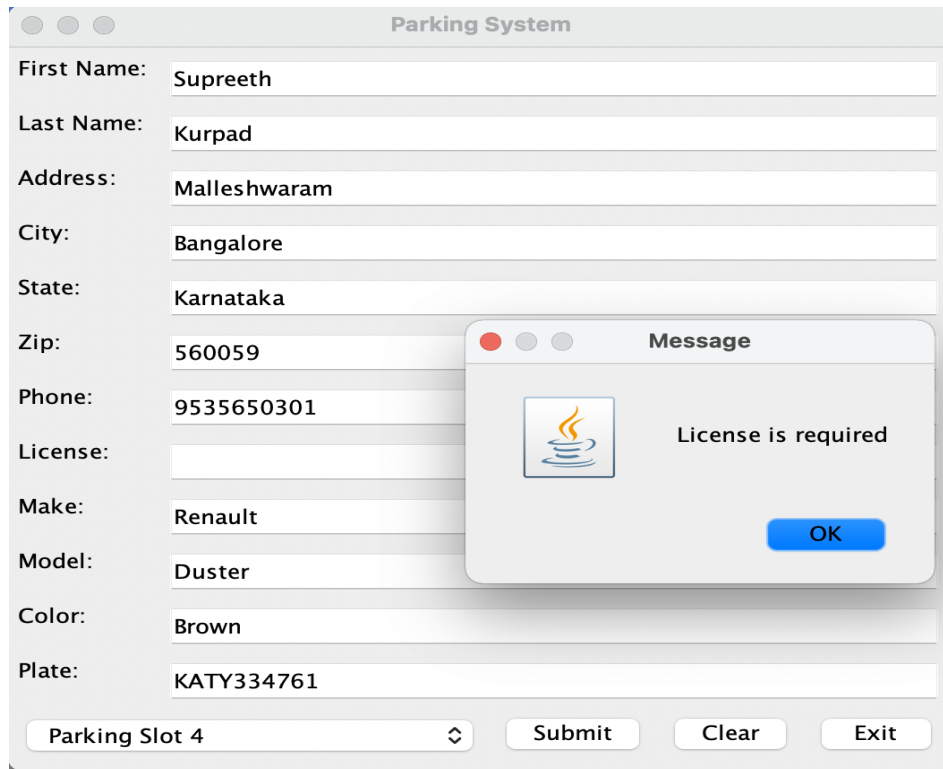
The details in the database are stored as MongoDB document. This will make information retrieval easier from the database by querying certain key-value pairs.

Inserting a few more users

```
{
  _id: ObjectId("623f76b63daaa915b53035eb"),
  firstname: 'Abhishek ',
  lastname: 'Aditya BS',
  address: 'Hanumanthanagar',
  city: 'Bangalore',
  state: 'Karnataka',
  zip: '560019',
  phone: '8197152711',
  license: 'KA556FT7QD',
  make: 'Hyundai',
  model: 'Verna',
  color: 'Silver',
  plate: 'KA05HN9225',
  parkingslot: 'Parking Slot 3'
},
{
  _id: ObjectId("624020454f0d091989038486"),
  firstname: 'T Vijay',
  lastname: 'Prashant',
  address: 'Kengari',
  city: 'Bangalore',
  state: 'Karnataka',
  zip: '560098',
  phone: '7712655986',
  license: 'K228761H7',
  make: 'Audi',
  model: 'Q7',
  color: 'Red',
  plate: 'KA0177231',
  parkingslot: 'Parking Slot 1'
},
{
  _id: ObjectId("624025654f0d091989038489"),
  firstname: 'Vishal ',
  lastname: 'R',
  address: 'Kathriguppe',
  city: 'Bangalore',
  state: 'Karnataka',
  zip: '560085',
  phone: '9976253781',
  license: 'KA772363HY',
  make: 'Hyundai',
  model: 'Creta',
  color: 'White',
  plate: 'KA02YT7761',
  parkingslot: 'Parking Slot 7'
}
```

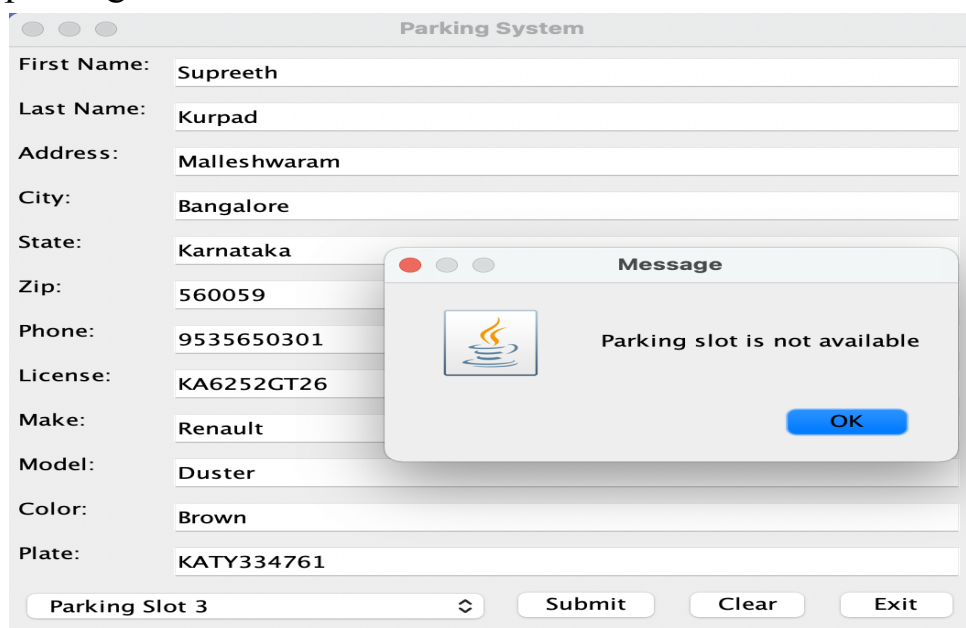
Error Handling

An error message will be shown if the user forgets to enter any required details. The error message will be displayed as a dialogue box indicating which field is missing.



The screenshot shows a web application titled "Parking System". It contains a form with the following fields: First Name (Supreeth), Last Name (Kurpad), Address (Malleshwaram), City (Bangalore), State (Karnataka), Zip (560059), Phone (9535650301), License (empty), Make (Renault), Model (Duster), Color (Brown), and Plate (KATY334761). At the bottom, there is a dropdown menu for "Parking Slot" currently set to "4", and three buttons: "Submit", "Clear", and "Exit". A modal dialog box titled "Message" is overlaid on the form, displaying a warning icon (a flame over a stack of plates) and the text "License is required". An "OK" button is at the bottom right of the dialog.

An error message will be shown if the user selects an already occupied parking slot.



The screenshot shows the same "Parking System" form as above, but the "Parking Slot" dropdown is now set to "3". The "License" field is now filled with "KA6252GT26". A modal dialog box titled "Message" is overlaid on the form, displaying the same warning icon and the text "Parking slot is not available". An "OK" button is at the bottom right of the dialog.

CODE :

Model.java

```
package com.parkingmvc;

public class Model {

    private String firstname;
    private String lastname;
    private String address;
    private String city;
    private String state;
    private String zip;
    private String phone;
    private String license;
    private String make;
    private String model;
    private String color;
    private String plate;
    private String parkingslot;

    public Model(String firstname, String lastname, String address, String city,
String state, String zip, String phone, String license, String make, String model,
String color, String plate, String parkingslot) {
        this.firstname = firstname;
        this.lastname = lastname;
        this.address = address;
        this.city = city;
        this.state = state;
        this.zip = zip;
        this.phone = phone;
        this.license = license;
        this.make = make;
        this.model = model;
        this.color = color;
        this.plate = plate;
        this.parkingslot = parkingslot;
    }

    public String getFirstname() {
        return firstname;
    }

    public void setFirstname(String firstname) {
        this.firstname = firstname;
    }
}
```

```
public String getLastname() {  
    return lastname;  
}  
  
public void setLastname(String lastname) {  
    this.lastname = lastname;  
}  
  
public String getAddress() {  
    return address;  
}  
  
public void setAddress(String address) {  
    this.address = address;  
}  
  
public String getCity() {  
    return city;  
}  
  
public void setCity(String city) {  
    this.city = city;  
}  
  
public String getState() {  
    return state;  
}  
  
public void setState(String state) {  
    this.state = state;  
}  
  
public String getZip() {  
    return zip;  
}  
  
public void setZip(String zip) {  
    this.zip = zip;  
}  
  
public String getPhone() {  
    return phone;  
}  
  
public void setPhone(String phone) {  
    this.phone = phone;  
}  
  
public String getLicense() {  
    return license;  
}  
  
public void setLicense(String license) {  
    this.license = license;  
}  
  
public String getMake() {  
    return make;  
}  
}
```

```

    public void setMake(String make) {
        this.make = make;
    }
    public String getModel() {
        return model;
    }
    public void setModel(String model) {
        this.model = model;
    }
    public String getColor() {
        return color;
    }
    public void setColor(String color) {
        this.color = color;
    }
    public String getPlate() {
        return plate;
    }
    public void setPlate(String plate) {
        this.plate = plate;
    }

    public String getParkingslot() {
        return parkingslot;
    }

    public void setParkingslot(String parkingslot) {
        this.parkingslot = parkingslot;
    }
}

```

Controller.java

```

package com.parkingmvc;

import javax.swing.JOptionPane;
import org.bson.Document;

public class Controller {

    private Model model;
    private View view;
}

```

```
private Database database;

public Controller(Model model, View view, Database database) {
    this.model = model;
    this.view = view;
    this.database = database;
}

public void initController() {
    view.getSubmitButton().addActionListener(e -> {
        String firstname = view.getFirstnameTextField().getText();
        if (firstname.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Firstname is required");
            return;
        }
        String lastname = view.getLastnameTextField().getText();
        if (lastname.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Lastname is required");
            return;
        }
        String address = view.getAddressTextField().getText();
        if (address.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Address is required");
            return;
        }
        String city = view.getCityTextField().getText();
        if (city.isEmpty()) {
            JOptionPane.showMessageDialog(null, "City is required");
            return;
        }
        String state = view.getStateTextField().getText();
        if (state.isEmpty()) {
            JOptionPane.showMessageDialog(null, "State is required");
            return;
        }
        String zip = view.getZipTextField().getText();
        if (zip.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Zip is required");
            return;
        }
        String phone = view.getPhoneTextField().getText();
        if (phone.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Phone is required");
            return;
        }
    })
}
```



```

String license = view.getLicenseTextField().getText();
if (license.isEmpty()) {
    JOptionPane.showMessageDialog(null, "License is required");
    return;
}

String make = view.getMakeTextField().getText();
if (make.isEmpty()) {
    JOptionPane.showMessageDialog(null, "Make is required");
    return;
}

String car_model = view.getModelTextField().getText();
if (car_model.isEmpty()) {
    JOptionPane.showMessageDialog(null, "Model is required");
    return;
}

String color = view.getColorTextField().getText();
if (color.isEmpty()) {
    JOptionPane.showMessageDialog(null, "Color is required");
    return;
}

String plate = view.getPlateTextField().getText();
if (plate.isEmpty()) {
    JOptionPane.showMessageDialog(null, "Plate is required");
    return;
}

String parkingslot = view.getComboBox().getSelectedItem().toString();
if (parkingslot == "Select a parking slot") {
    JOptionPane.showMessageDialog(null, "Parking slot is required");
    return;
}

if (database.isParkingSlotAvailable(parkingslot) == false) {
    JOptionPane.showMessageDialog(null, "Parking slot is not
available");
    return;
}

model.setFirstname(firstname);
model.setLastname(lastname);
model.setAddress(address);
model.setCity(city);
model.setState(state);
model.setZip(zip);
model.setPhone(phone);
model.setLicense(license);
model.setMake(make);
model.setModel(car_model);

```

```

        model.setColor(color);
        model.setPlate(plate);
        model.setParkingslot(parkingslot);

        Document document = new Document("firstname", model.getFirstname())
            .append("lastname", model.getLastname())
            .append("address", model.getAddress())
            .append("city", model.getCity())
            .append("state", model.getState())
            .append("zip", model.getZip())
            .append("phone", model.getPhone())
            .append("license", model.getLicense())
            .append("make", model.getMake())
            .append("model", model.getModel())
            .append("color", model.getColor())
            .append("plate", model.getPlate())
            .append("parkingslot", model.getParkingslot());

        database.insertDocument(document);

        view.getFirstnameTextField().setText("");
        view.getLastnameTextField().setText("");
        view.getAddressTextField().setText("");
        view.getCityTextField().setText("");
        view.getStateTextField().setText("");
        view.getZipTextField().setText("");
        view.getPhoneTextField().setText("");
        view.getLicenseTextField().setText("");
        view.getMakeTextField().setText("");
        view.getModelTextField().setText("");
        view.getColorTextField().setText("");
        view.getPlateTextField().setText("");
        view.getComboBox().setSelectedItem("Select a parking slot");
        JOptionPane.showMessageDialog(null, "Thank you for your using our
parking system");
    });

    view.getClearButton().addActionListener(e -> {
        view.getFirstnameTextField().setText("");
        view.getLastnameTextField().setText("");
        view.getAddressTextField().setText("");
        view.getCityTextField().setText("");
        view.getStateTextField().setText("");
        view.getZipTextField().setText("");
        view.getPhoneTextField().setText("");
    });

```

```

        view.getLicenseTextField().setText("");
        view.getMakeTextField().setText("");
        view.getModelTextField().setText("");
        view.getColorTextField().setText("");
        view.getPlateTextField().setText("");
        view.getComboBox().setSelectedItem("Select a parking slot");
    });

    view.getExitButton().addActionListener(e -> {
        database.closeConnection();
        System.exit(0);
    });
}
}

```

View.java

```

package com.parkingmvc;

import java.awt.BorderLayout;
import javax.swing.GroupLayout;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JTextField;
// import javax.swing.SwingConstants;

public class View {

    private JFrame frame;
    private JLabel firstnameLabel;
    private JLabel lastnameLabel;
    private JLabel addressLabel;
    private JLabel cityLabel;
    private JLabel stateLabel;
    private JLabel zipLabel;
    private JLabel phoneLabel;
    private JLabel licenseLabel;
    private JLabel makeLabel;
    private JLabel modelLabel;
    private JLabel colorLabel;
    private JLabel plateLabel;
    private JTextField firstnameTextField;

```

```

private JTextField lastnameTextField;
private JTextField addressTextField;
private JTextField cityTextField;
private JTextField stateTextField;
private JTextField zipTextField;
private JTextField phoneTextField;
private JTextField licenseTextField;
private JTextField makeTextField;
private JTextField modelTextField;
private JTextField colorTextField;
private JTextField plateTextField;
String p[] = {"Select a parking slot",
              "Parking Slot 1", "Parking Slot 2",
              "Parking Slot 3", "Parking Slot 4",
              "Parking Slot 5", "Parking Slot 6",
              "Parking Slot 7", "Parking Slot 8",
              "Parking Slot 9", "Parking Slot 10"};

private JComboBox<String> comboBox;
private JButton submitButton;
private JButton clearButton;
private JButton exitButton;
// Get a drop down list of options

public View(String title) {
    frame = new JFrame(title);
    frame.setLayout(new BorderLayout());
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setSize(400, 400);
    frame.setLocationRelativeTo(null);
    frame.setVisible(true);
    frame.setResizable(true);

    firstnameLabel = new JLabel("First Name: ");
    lastnameLabel = new JLabel("Last Name: ");
    addressLabel = new JLabel("Address: ");
    cityLabel = new JLabel("City: ");
    stateLabel = new JLabel("State: ");
    zipLabel = new JLabel("Zip: ");
    phoneLabel = new JLabel("Phone: ");
    licenseLabel = new JLabel("License: ");
    makeLabel = new JLabel("Make: ");
    modelLabel = new JLabel("Model: ");
    colorLabel = new JLabel("Color: ");
    plateLabel = new JLabel("Plate: ");

```

```
firstnameTextField = new JTextField();
lastnameTextField = new JTextField();
addressTextField = new JTextField();
cityTextField = new JTextField();
stateTextField = new JTextField();
zipTextField = new JTextField();
phoneTextField = new JTextField();
licenseTextField = new JTextField();
makeTextField = new JTextField();
modelTextField = new JTextField();
colorTextField = new JTextField();
plateTextField = new JTextField();
comboBox = new JComboBox(p);
submitButton = new JButton("Submit");
clearButton = new JButton("Clear");
exitButton = new JButton("Exit");

GroupLayout layout = new GroupLayout(frame.getContentPane());
frame.getContentPane().setLayout(layout);
layout.setAutoCreateGaps(true);
layout.setAutoCreateContainerGaps(true);
layout.setHorizontalGroup(layout.createParallelGroup()
    .addGroup(layout.createSequentialGroup()
        .addGroup(layout.createParallelGroup()
            .addComponent(firstnameLabel)
            .addComponent(lastnameLabel)
            .addComponent(addressLabel)
            .addComponent(cityLabel)
            .addComponent(stateLabel)
            .addComponent(zipLabel)
            .addComponent(phoneLabel)
            .addComponent(licenseLabel)
            .addComponent(makeLabel)
            .addComponent(modelLabel)
            .addComponent(colorLabel)
            .addComponent(plateLabel))
        .addGroup(layout.createParallelGroup()
            .addComponent(firstnameTextField)
            .addComponent(lastnameTextField)
            .addComponent(addressTextField)
            .addComponent(cityTextField)
            .addComponent(stateTextField)
            .addComponent(zipTextField)
            .addComponent(phoneTextField)
```

```

        .addComponent (licenseTextField)
        .addComponent (makeTextField)
        .addComponent (modelTextField)
        .addComponent (colorTextField)
        .addComponent (plateTextField)))
    .addGroup (layout.createSequentialGroup ()
        .addComponent (comboBox)
        .addComponent (submitButton)
        .addComponent (clearButton)
        .addComponent (exitButton))) ;

layout.setVerticalGroup (layout.createSequentialGroup ()
    .addGroup (layout.createParallelGroup ()
        .addComponent (firstnameLabel)
        .addComponent (firstnameTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (lastnameLabel)
        .addComponent (lastnameTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (addressLabel)
        .addComponent (addressTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (cityLabel)
        .addComponent (cityTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (stateLabel)
        .addComponent (stateTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (zipLabel)
        .addComponent (zipTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (phoneLabel)
        .addComponent (phoneTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (licenseLabel)
        .addComponent (licenseTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (makeLabel)
        .addComponent (makeTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (modelLabel)
        .addComponent (modelTextField))
    .addGroup (layout.createParallelGroup ()
        .addComponent (colorLabel)
        .addComponent (colorTextField))

```

```

        .addGroup(layout.createParallelGroup()
            .addComponent(plateLabel)
            .addComponent(plateTextField))
        .addGroup(layout.createParallelGroup()
            .addComponent(comboBox)
            .addComponent(submitButton)
            .addComponent(clearButton)
            .addComponent(exitButton));
    }

    public JFrame getFrame() {
        return frame;
    }

    public JLabel getFirstnameLabel() {
        return firstnameLabel;
    }
    public JLabel getLastnameLabel() {
        return lastnameLabel;
    }
    public JLabel getAddressLabel() {
        return addressLabel;
    }
    public JLabel getCityLabel() {
        return cityLabel;
    }
    public JLabel getStateLabel() {
        return stateLabel;
    }
    public JLabel getZipLabel() {
        return zipLabel;
    }
    public JLabel getPhoneLabel() {
        return phoneLabel;
    }
    public JLabel getLicenseLabel() {
        return licenseLabel;
    }
    public JLabel getMakeLabel() {
        return makeLabel;
    }
    public JLabel getModelLabel() {
        return modelLabel;
    }
}

```

```
public JLabel getColorLabel() {
    return colorLabel;
}

public JLabel getPlateLabel() {
    return plateLabel;
}

public JComboBox getComboBox() {
    return comboBox;
}

public JTextField getFirstnameTextField() {
    return firstnameTextField;
}

public JTextField getLastNameTextField() {
    return lastnameTextField;
}

public JTextField getAddressTextField() {
    return addressTextField;
}

public JTextField getCityTextField() {
    return cityTextField;
}

public JTextField getStateTextField() {
    return stateTextField;
}

public JTextField getZipTextField() {
    return zipTextField;
}

public JTextField getPhoneTextField() {
    return phoneTextField;
}

public JTextField getLicenseTextField() {
    return licenseTextField;
}

public JTextField getMakeTextField() {
    return makeTextField;
}

public JTextField getModelTextField() {
    return modelTextField;
}

public JTextField getColorTextField() {
    return colorTextField;
}

public JTextField getPlateTextField() {
    return plateTextField;
}
```



```
public JButton getSubmitButton() {
    return submitButton;
}

public JButton getClearButton() {
    return clearButton;
}

public JButton getExitButton() {
    return exitButton;
}

public void setFirstnameLabel(JLabel firstnameLabel) {
    this.firstnameLabel = firstnameLabel;
}

public void setLastnameLabel(JLabel lastnameLabel) {
    this.lastnameLabel = lastnameLabel;
}

public void setAddressLabel(JLabel addressLabel) {
    this.addressLabel = addressLabel;
}

public void setCityLabel(JLabel cityLabel) {
    this.cityLabel = cityLabel;
}

public void setStateLabel(JLabel stateLabel) {
    this.stateLabel = stateLabel;
}

public void setZipLabel(JLabel zipLabel) {
    this.zipLabel = zipLabel;
}

public void setPhoneLabel(JLabel phoneLabel) {
    this.phoneLabel = phoneLabel;
}

public void setLicenseLabel(JLabel licenseLabel) {
    this.licenseLabel = licenseLabel;
}

public void setMakeLabel(JLabel makeLabel) {
    this.makeLabel = makeLabel;
}

public void setModelLabel(JLabel modelLabel) {
    this.modelLabel = modelLabel;
}

public void setColorLabel(JLabel colorLabel) {
    this.colorLabel = colorLabel;
}

public void setPlateLabel(JLabel plateLabel) {
    this.plateLabel = plateLabel;
}
```

```
}

public void setFirstnameTextField(JTextField firstnameTextField) {
    this.firstnameTextField = firstnameTextField;
}

public void setLastnameTextField(JTextField lastnameTextField) {
    this.lastnameTextField = lastnameTextField;
}

public void setAddressTextField(JTextField addressTextField) {
    this.addressTextField = addressTextField;
}

public void setCityTextField(JTextField cityTextField) {
    this.cityTextField = cityTextField;
}

public void setStateTextField(JTextField stateTextField) {
    this.stateTextField = stateTextField;
}

public void setZipTextField(JTextField zipTextField) {
    this.zipTextField = zipTextField;
}

public void setPhoneTextField(JTextField phoneTextField) {
    this.phoneTextField = phoneTextField;
}

public void setLicenseTextField(JTextField licenseTextField) {
    this.licenseTextField = licenseTextField;
}

public void setMakeTextField(JTextField makeTextField) {
    this.makeTextField = makeTextField;
}

public void setModelTextField(JTextField modelTextField) {
    this.modelTextField = modelTextField;
}

public void setColorTextField(JTextField colorTextField) {
    this.colorTextField = colorTextField;
}

public void setPlateTextField(JTextField plateTextField) {
    this.plateTextField = plateTextField;
}

public void setJComboBox(JComboBox comboBox) {
    this.comboBox = comboBox;
}

public void setSubmitButton(JButton submitButton) {
    this.submitButton = submitButton;
}

public void setClearButton(JButton clearButton) {
    this.clearButton = clearButton;
}
```

```

    }

    public void setExitButton(JButton exitButton) {
        this.exitButton = exitButton;
    }
}

```

Database.Java

```

package com.parkingmvc;

import org.bson.Document;
import com.mongodb.MongoClient;
import com.mongodb.MongoClientURI;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;

public class Database {

    private String connectionString; // "mongodb://localhost:27017"
    private String databaseName;
    private String collectionName;

    public Database(String connectionString, String databaseName, String
collectionName) {
        this.connectionString = connectionString;
        this.databaseName = databaseName;
        this.collectionName = collectionName;
    }

    public String getConnectionString() {
        return connectionString;
    }

    public void setConnectionString(String connectionString) {
        this.connectionString = connectionString;
    }

    public String getDatabaseName() {
        return databaseName;
    }

    public void setDatabaseName(String databaseName) {
        this.databaseName = databaseName;
    }

    public String getCollectionName() {
        return collectionName;
    }

    public void setCollectionName(String collectionName) {

```

```

        this.collectionName = collectionName;
    }

    public void insertDocument(Document document) {
        MongoClientURI uri = new MongoClientURI(connectionString);
        MongoClient mongoClient = new MongoClient(uri);
        MongoDBDatabase database = mongoClient.getDatabase(databaseName);
        MongoClientCollection<Document> collection =
database.getCollection(collectionName);
        collection.insertOne(document);
    }

    public void closeConnection() {
        MongoClientURI uri = new MongoClientURI(connectionString);
        MongoClient mongoClient = new MongoClient(uri);
        mongoClient.close();
    }

    public boolean isParkingSlotAvailable(String parkingslot) {
        MongoClientURI uri = new MongoClientURI(connectionString);
        MongoClient mongoClient = new MongoClient(uri);
        MongoDBDatabase database = mongoClient.getDatabase(databaseName);
        MongoClientCollection<Document>collection =database.getCollection(collectionName);
        Document document = collection.find(new Document("parkingslot",
parkingslot)).first();
        if (document == null) {
            return true;
        }
        return false;
    }

    public void deleteParkingSlot(String parkingslot) {
        MongoClientURI uri = new MongoClientURI(connectionString);
        MongoClient mongoClient = new MongoClient(uri);
        MongoDBDatabase database = mongoClient.getDatabase(databaseName);
        MongoClientCollection<Document> collection =
database.getCollection(collectionName);
        collection.deleteOne(new Document("parkingslot", parkingslot));
    }
}

```