**PES University**
**Object Oriented Analysis and Design with Java**
**Assignment - 1: Understanding MVC Framework**

Adithya P
PES1UG19CS028

**Problem Formulation:**
This project envisions a **weather application** with a simple GUI Interface where the user can type in the name of a city as an input. As the output, the application displays the current details of the weather in that city in a dialogue box (The details are: Current Temperature, Minimum Temperature, Maximum Temperature and Weather Description. The values are in degrees celsius.). The GUI is implemented using **Java's Swing API** and the details of the weather from the city is achieved by using the **OpenWeather API,** a free weather API.
**The entire application is built using the MVC architecture.**

**Code related to Model:**

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

public class WeatherModel {

    private float temp;
    private float minTemp;
    private float maxTemp;
    private String city;
    private String APIKEY;
    private URL openweather;
    private String response;
    private String description;

    WeatherModel() {
        this.temp = 0.0f;
        this.minTemp = 0.0f;
        this.maxTemp = 0.0f;
        this.description = "Enter valid city name";
        // this.APIKEY = "secret";
        this.city = "London"; // keeping London as the default city name
        File apiFile = new File("./APIKEY.txt");
        try {
            BufferedReader br = new BufferedReader(new FileReader(apiFile));
            this.APIKEY = br.readLine();
        } catch (IOException e) {
            System.out.println(e);
        }
    }

    public void callWeatherAPI(String inputCity) {
        try {
            this.response = ""; // To contain the response got from the API call
            this.city = inputCity;
```

```
            // calling the API endpoint using a HTTPS GET request
            this.openweather = new URL(
                    "https://api.openweathermap.org/data/2.5/weather?q=" + this.city
+ "&appid=" + this.APIKEY);
            URLConnection ow = openweather.openConnection();
            BufferedReader in = new BufferedReader(new
InputStreamReader(ow.getInputStream()));
            String inputLine;
            while ((inputLine = in.readLine()) != null) {
                this.response += inputLine;
            }
            in.close();
        } catch (IOException e) {
            System.out.println(e);
             return;
        }
        if (this.response != null) {
            // Set the non-null response values to the respective private variables
            this.setTemp();
            this.setMinTemp();
            this.setMaxTemp();
            this.setDescription();
        }
    }

    // First parse the JSON string to get the temp value and set it to the temp
    // private var.
    // The functions following this follow a similar procedure.
    private void setTemp() {
        int index = this.response.indexOf("temp") + 6;
        this.temp = Float.parseFloat(response.substring(index, index + 5));
    }

    private void setMinTemp() {
        int index = this.response.indexOf("temp_min") + 6 + 4;
        this.minTemp = Float.parseFloat(response.substring(index, index + 5));
    }

    private void setMaxTemp() {
        int index = this.response.indexOf("temp") + 6;
        this.maxTemp = Float.parseFloat(response.substring(index, index + 5));
    }

    private void setDescription() {
        int index = this.response.indexOf("description") + 14;
        int end = 0;
        while (this.response.charAt(index + end) != '\"') {
            end += 1;
        }
        this.description = response.substring(index, index + end);
    }

    // Getter functions
    public float getTemp() {
```

```
        return this.temp;
    }

    public float getMinTemp() {
        return this.minTemp;
    }

    public float getMaxTemp() {
        return this.maxTemp;
    }

    public String getDescription() {
        return this.description;
    }
}
```

**Code related to View:**

```
import java.awt.event.ActionListener;
import javax.swing.*;

public class WeatherView extends JFrame {
    private JTextField inputCity = new JTextField(20);
    private JButton submitButton = new JButton("Submit");
    private String description;
    private float maxTemp;
    private float minTemp;
    private float temp;

       // Initialize JFrame Parameters in the constructor, make the GUI.
    WeatherView() {
        this.description = "Enter City name";
        this.maxTemp = 0.0f;
        this.minTemp = 0.0f;
        this.temp = 0.0f;
        JPanel weatherPanel = new JPanel();
        this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        this.setTitle("Weather App MVC");
        this.setSize(500, 400);
        weatherPanel.add(new JLabel(this.description + "   "));
        weatherPanel.add(this.inputCity);
        weatherPanel.add(this.submitButton);
        this.add(weatherPanel);
        this.setResizable(false);
        this.pack();
        this.setVisible(true);
    }

    public String getInputCity() {
        return new String(inputCity.getText());
    }

// To handle the button click for submitting the value to the controller
    public void addWeatherEventListener(ActionListener weatherListener) {
        submitButton.addActionListener(weatherListener);
```

```
    }

    public void displayError(String ErrorMessage) {
        JOptionPane.showMessageDialog(this, ErrorMessage);
    }

// Setter functions
    public void setDesc(String d) {
        this.description = d;
    }

    public void setTemp(float t) {
        this.temp = t;
    }

    public void setMinTemp(float t) {
        this.minTemp = t;
    }

    public void setMaxTemp(float t) {
        this.maxTemp = t;
    }

    public void setResult() {
        String res = this.getInputCity() + "<br>";
        res += "Current Temp. : " + this.KelvinToCelcius(this.temp) + " °C<br>";
        res += "Min Temp.     : " + this.KelvinToCelcius(this.minTemp) + " °C<br>";
        res += "Max Temp.     : " + this.KelvinToCelcius(this.maxTemp) + " °C<br>";
        res += this.description;
        res = "<html><br><p>" + res + "</p></html>";
        JOptionPane.showMessageDialog(this, res);
    }

    public void setError() {
        JOptionPane.showMessageDialog(this, this.description);
    }

// to convert Kelvin to Celsius values
    private float KelvinToCelcius(float k) {
        return k - 273.15f;
    }
}
```

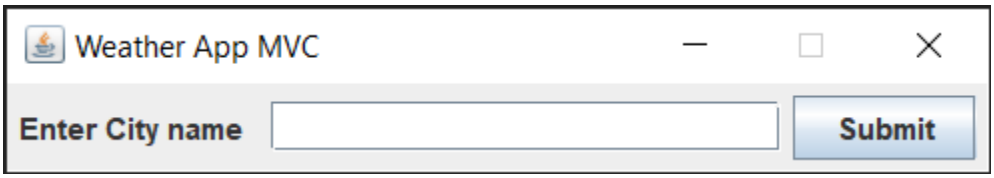
**Code related to Controller:**

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class WeatherController {
    private WeatherModel wm;
    private WeatherView wv;
```

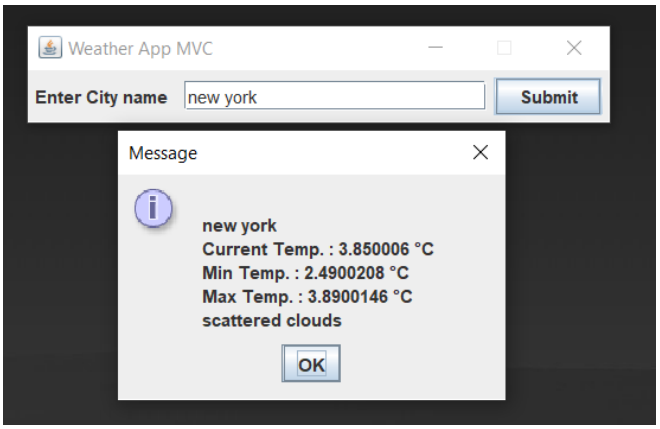
```
    public WeatherController(WeatherModel wm, WeatherView wv) {
        this.wm = wm;
        this.wv = wv;

        this.wv.addWeatherEventListener(new weatherListener());
    }
       // subclass for implementing the button click action listener
    class weatherListener implements ActionListener {

        @Override
        public void actionPerformed(ActionEvent e) {
            String inputCity = "London"; // the default one
            try {
                inputCity = wv.getInputCity(); // Get input from view instance
                wm.callWeatherAPI(inputCity); // pass it to the model for API Call
                wv.setDesc(wm.getDescription()); // Get desc. from model to view
                wv.setTemp(wm.getTemp()); // Get temp from model, set it to view's
temp value
                wv.setMaxTemp(wm.getMaxTemp()); // Get max temp from model to view
                wv.setMinTemp(wm.getMinTemp()); // Get min temp from model to view
                wv.setResult(); // Call the set result function to display the result
            } catch (Exception ex) {
                System.out.println(ex);
            }
        }
    }
}
```

**Output Snapshots:**

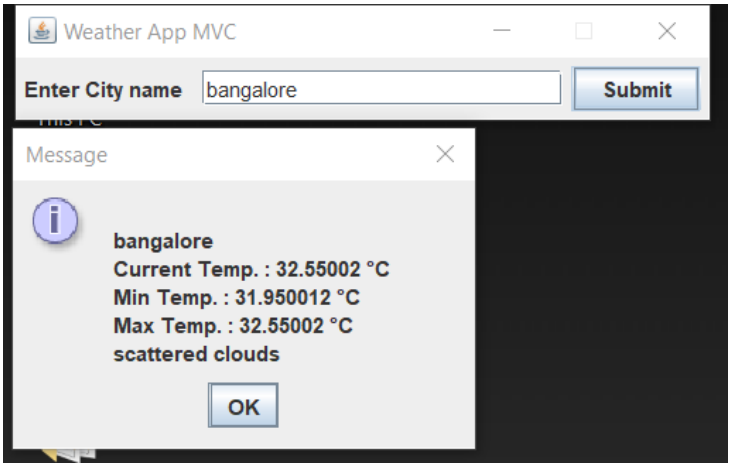
The initial GUI for user Input:



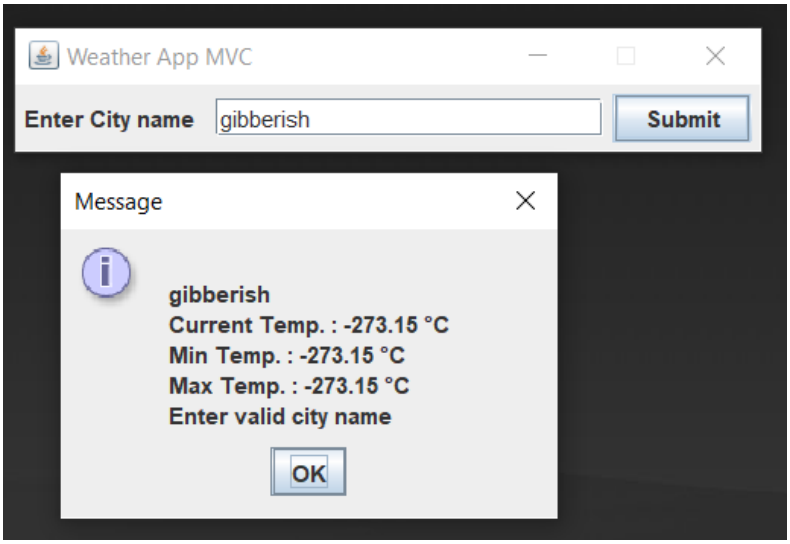

On Submitting a city name with a space:



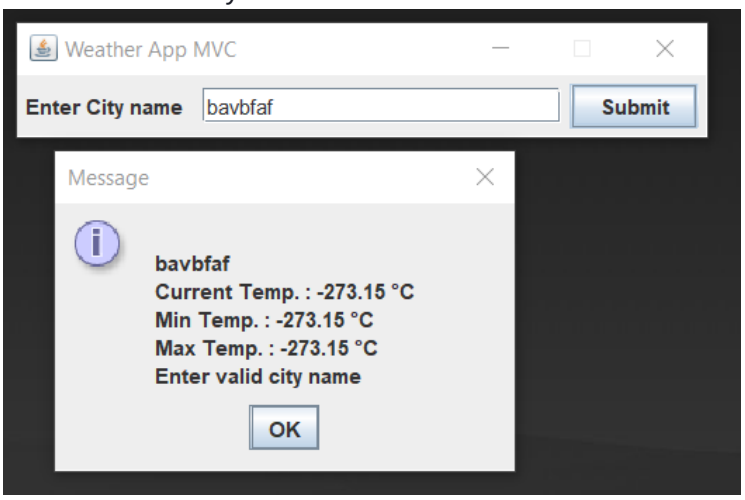
On submitting a valid city name:




On submitting an invalid city name:




Another Invalid city name case:



The main function:

```
public class App {
    public static void main(String[] args) {
        WeatherView wv = new WeatherView();
        WeatherModel wm = new WeatherModel();
        WeatherController wc = new WeatherController(wm, wv);
    }
}
```