MY WAY POINTS

DISTRIBUTED SYSTEMS

CSE 586

SAI SAKET REGULAPATI

UBID: saisaket

Person Number: 50286747

PROJECT CONTENTS:

INTRODUCTION

MyWayPoints is a java web application used to navigate through cities through which you will be travelling. By the use of this application you will be able to see the weather and temperature of the cities you are travelling through. This application uses Google Maps API and OpenWeatherMap API to find the cities you will be visiting while you are on your journey between two places and finds out the weather and other interesting points for you.

The application makes use of Google Maps Directions API to find the route between the Source and Destination. Then makes use of the result retrieved to find Way Points and then calls the OpenWeatherMap API to query for temperature and weather to display it to the user. Once these are retrieved, they are displayed on markers of the Way Points on the Google Maps.

SOFTWARE REQUIREMENT SPECIFICATION

Software requirements specification describes the requirements for external functioning of the application. The Software and hardware requirements are mentioned below.

Software Requirements:

OPERATING SYSTEM:	Windows 7/8/8.1/10
Platform:	Eclipse, Apache Tomcat Server
Programming:	Java, JavaScript, JSP

Hardware Requirements:

Processor:	Pentium IV (minimum)
Hard Disk:	40 GB (minimum)
RAM:	512MB (minimum)

SOFTWARE DEVELOPMENT LIFECYLCE

Analysis:

In this stage, the analysis on how the project is to be implemented is made.

Design:

A design of implementation is made at this stage to satisfy all the requirements of the users collected from the Analysis stage. A project design is made by the end of this stage.

Implementation:

The design made is implemented in this stage by the use of various platforms and programming languages.

Testing:

The product is tested for it's functionality. The project has been tested manually for it's functionality in our project.

PROGRAMMING LANGUAGES AND PLATFORMS USED

APACHE TOMCAT:

Apache Tomcat has been used as the server for the current project. The tomcat server is installed on the local ports 8005,8019 and 8080. This server is installed in the Eclipse IDE for implementation.

ECLIPSE IDE:

The photon version of the Eclipse is used as the IDE for the current project. Eclipse is used as it supports Java Web Development to the fullest. The Apache Tomcat Server is installed in the IDE where the project is implemented.

JAVA:

Java has been mainly used for the server side implementation of the web application. The main part of the java programs is to redirect between the web pages and build a co-ordination between them.

JSP:

JSP is mainly used to implement the design of the web pages on which the user will be working. The JSP implements the UI for the user. JSP is dynamic and better version of HTML.

JAVASCRIPT:

Javascript is mainly used to call and implement the APIs that are being called. The javascript call the Google Maps API and the OpenWeatherMaps API. It retrieves the data from the APIs and processes the data and represents in such a way that the user finds it comfortable. JavaScript

CSS:

CSS is mainly used to stylize the webpages design using the JSP.

APIs USED

OPENWEATHERMAP API:

The OpenWeatherMap API takes input from the user in the form of co-ordinates or city name or city ID and gives back a JSON form or XML form based on the request of the user.

The application has used the OpenWeatherMap based on the JSON data from the API. The application calls the OpenWeatherMap based on co-ordinates and the city name.

GOOGLE MAPS DIRECTION API:

The Google Maps Direction API takes two cities as input from the user and then gives the direction or route between those two cities (in the form of JSON) back to the user.

The application has made use of the steps in the JSON file to retrieve the way points. The co-ordinates of these way points are used as the parameters to get the Weather information for the user.

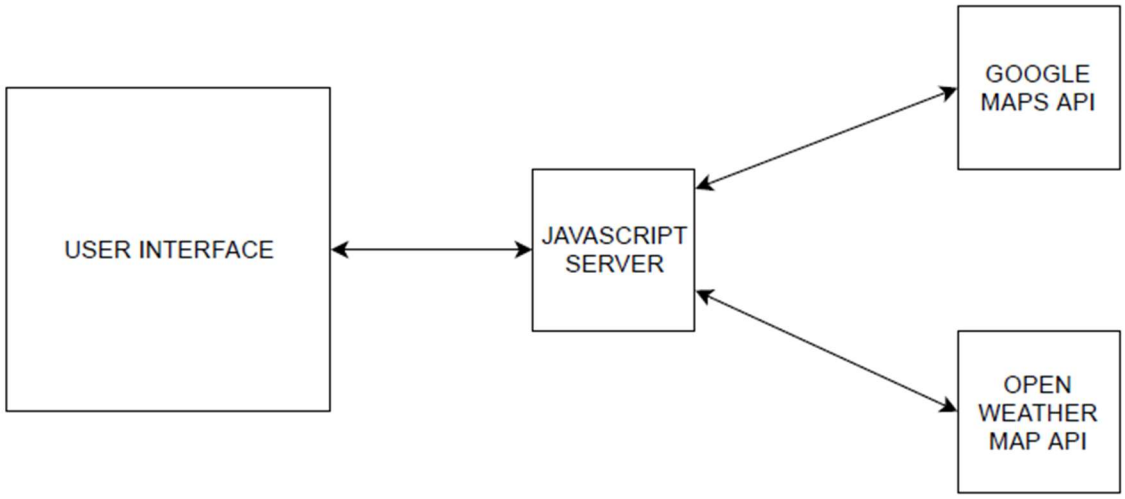
APPLICATION FLOW

The user inputs data through the select boxes on the window of the web page. Once the user inputs the request, the Javascript calls the Maps API to find the route from source to the destination. We also put the way points on the Map route for further use. Once the maps gives the markers along the route, we call the OpenWeatherMap API to get the weather information of the way points. OpenWeatherMap is given the co-ordinates of the way point and the data is retrieved and shown to the user on clicking the marker.

The user can change the source or destination anytime and will be immediately able to see the changes on the map.

ARCHITECTURE

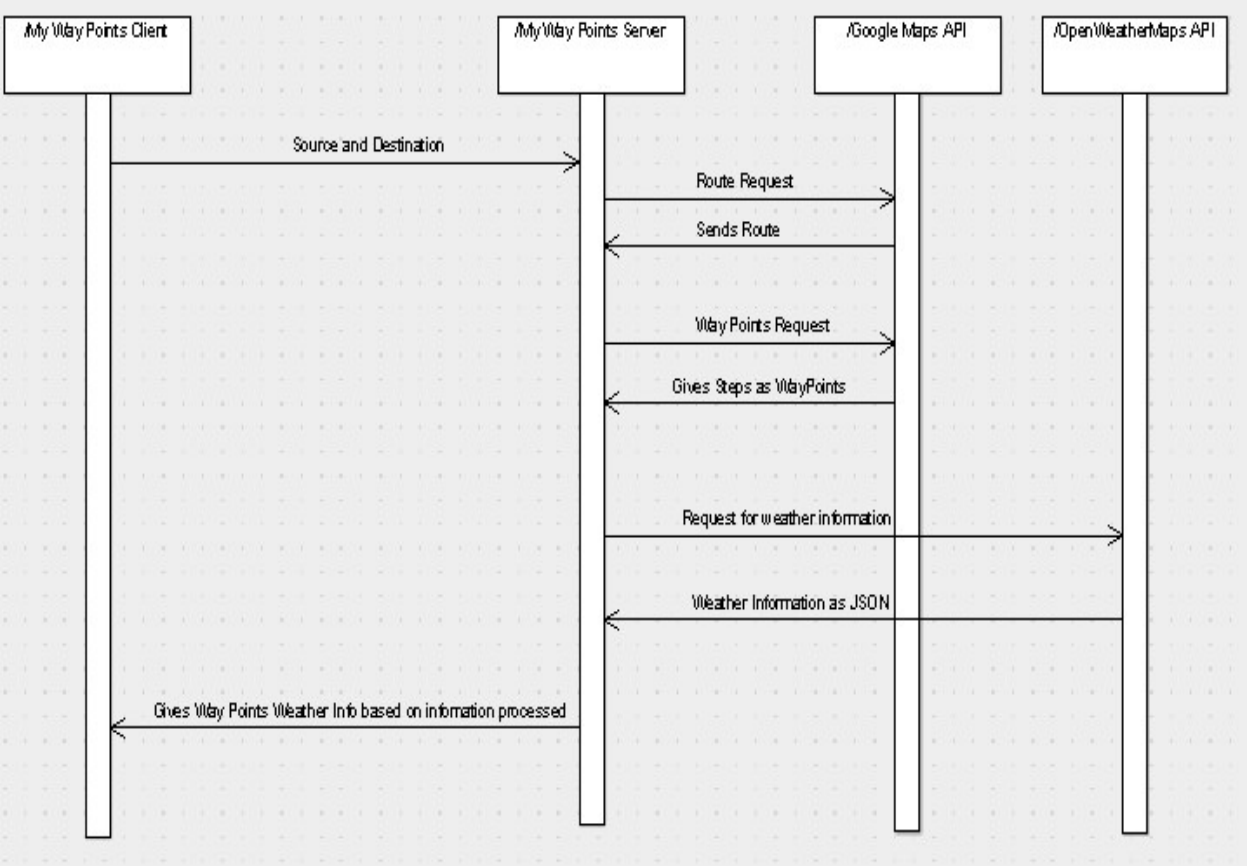
The MYWAYPOINTS application interacts with the user, takes the request, finds the GOOGLE maps API, finds the route and then tracks the Way Points through the steps of the google map result, finds the location of the steps, and then processes the temperature at the location through the use of OPENWEATHERMAPS API and gives back the temperature of these locations on the google map.



The above diagram represents the architecture of the MYWAYPOINTS web application.

SEQUENCE DIAGRAM:

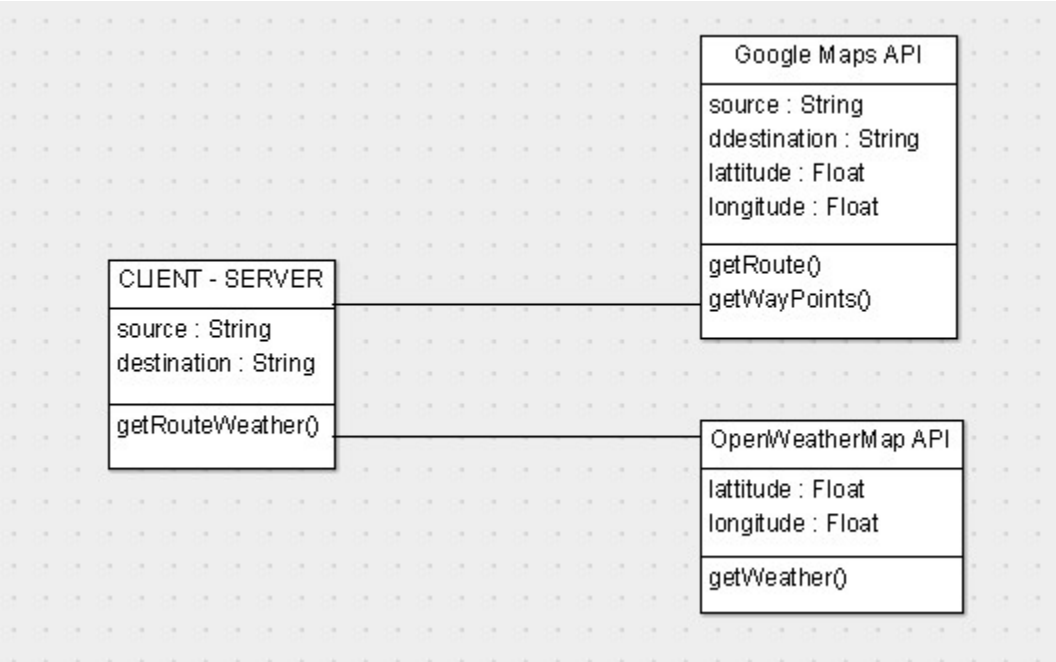
The sequence diagram gives the systematic flow of the web application. The sequence diagrams represents the object interactions arranged in time space.



The above diagram represents the Sequence view of the MyWayPoints web application.

CLASS DIAGRAM:

Class Diagram gives the static view of the model which helps in visualizing and documenting different aspects of the application.



The above diagram represents the class diagram of the model.

CODE SNIPPETS:

Javascript code to find the route between two cities:

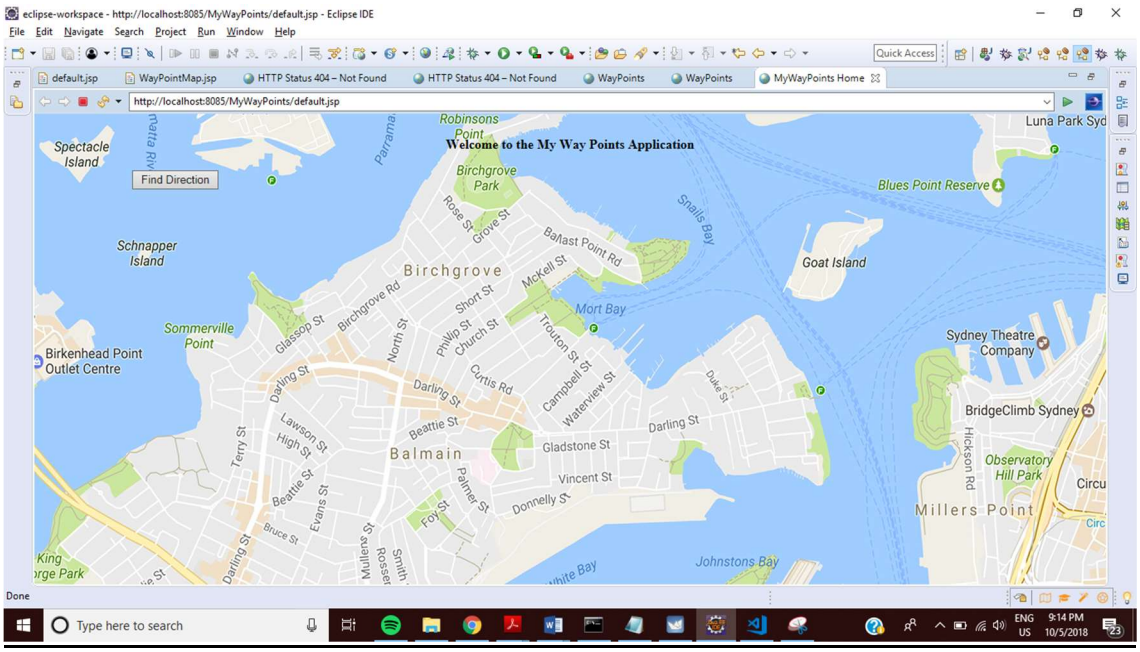
```
function calculateAndDisplayRoute(directionsDisplay, directionsService,
       markerArray, stepDisplay, map) {
       // First, remove any existing markers from the map.
       if(document.getElementById('start').value!=""&&document.getElementById('end').
       value!=""){
              for (var i = 0; i < markerArray.length; i++) {
                     markerArray[i].setMap(null);
              }

        // Retrieve the start and end locations and create a DirectionsRequest using
        // DRIVING directions.
       directionsService.route({
          origin: document.getElementById('start').value,
          destination: document.getElementById('end').value,
          travelMode: 'DRIVING'
       }, function(response, status) {
          // Route the directions and pass the response to a function to create
markers for each step.
              if (status === 'OK') {
                     document.getElementById('warnings-panel').innerHTML =
                     '<b>' + response.routes[0].warnings + '</b>';
              directionsDisplay.setDirections(response);
              showSteps(response, markerArray, stepDisplay, map);
              }
              else {
                     //error-failed
                     window.alert('Directions request failed due to ' + status);
              }
          });
        }
      }
```

Javascript code to implement the use of OpenWeatherMap:

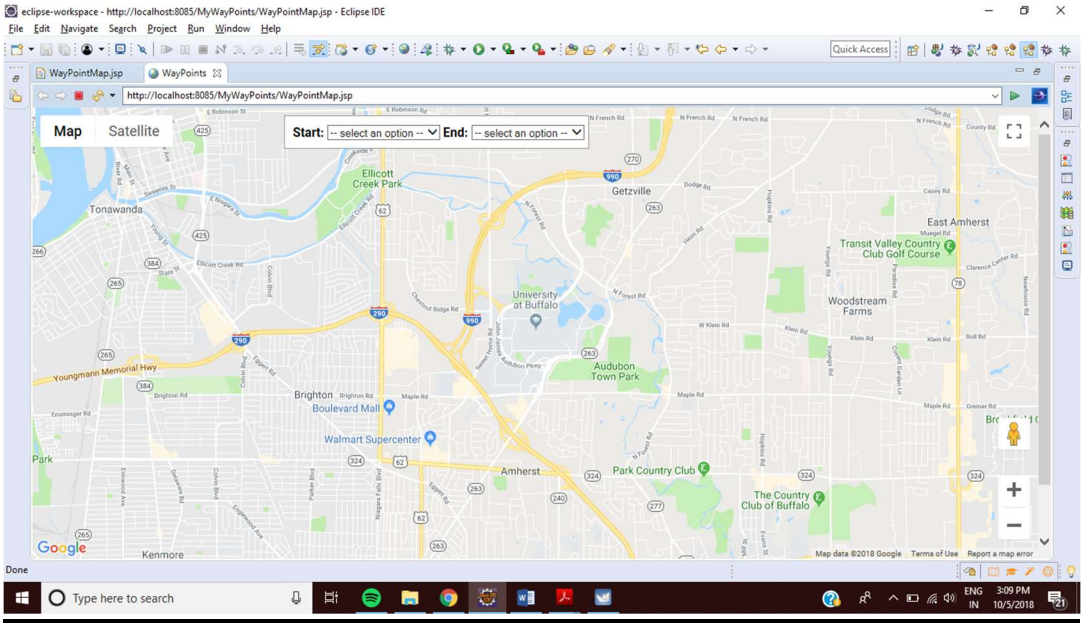
```
function showSteps(directionResult, markerArray, stepDisplay, map) {
        // At each step, place a marker, and add the text to the marker's infowindow.
        // Also attach the marker to an array so we can keep track of it and remove
it for calculating new routes.
        var myRoute = directionResult.routes[0].legs[0];
        for (var i = 0; i < myRoute.steps.length; i++) {
          var marker = markerArray[i] = markerArray[i] || new google.maps.Marker;
          marker.setMap(map);
          marker.setPosition(myRoute.steps[i].start_location);
          var arr=(myRoute.steps[i].start_location.toString()).split(",");
          var lat=arr[0].substr(1).toString();
          var lng=arr[1].slice(0,-1).toString();

          try{

             dataurl="https://api.openweathermap.org/data/2.5/weather?lat="+lat+"&lon
             ="+lng+"&appid=b3445eb438bb411b0bb3d991a0076a6f";
             var xhr =new XMLHttpRequest();
             xhr.open("GET", dataurl, false);
             xhr.send();
             var jsontext=JSON.parse(xhr.responseText);
             var name=jsontext.name;
             var temp=jsontext.main.temp;
          }
          catch(Exception){
               console.log(Exception);
          }
          attachInstructionText(stepDisplay, marker, name, map);
        }
}
```

WORKING SCREENSHOTS:

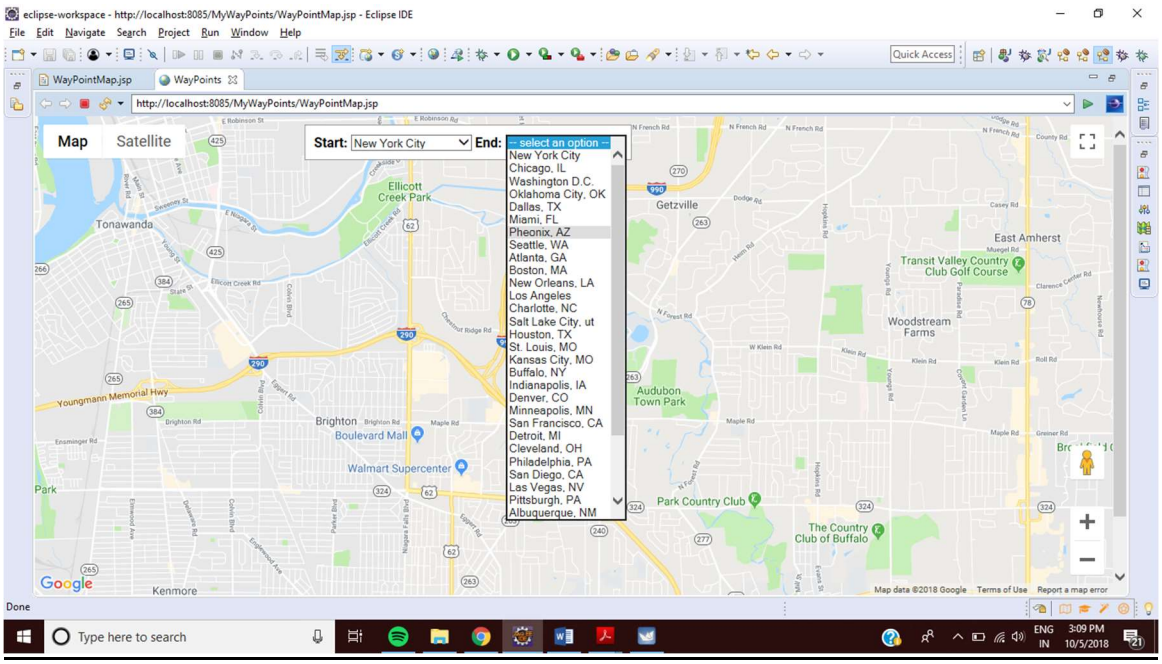
Screenshot 1 – Welcome Web Page



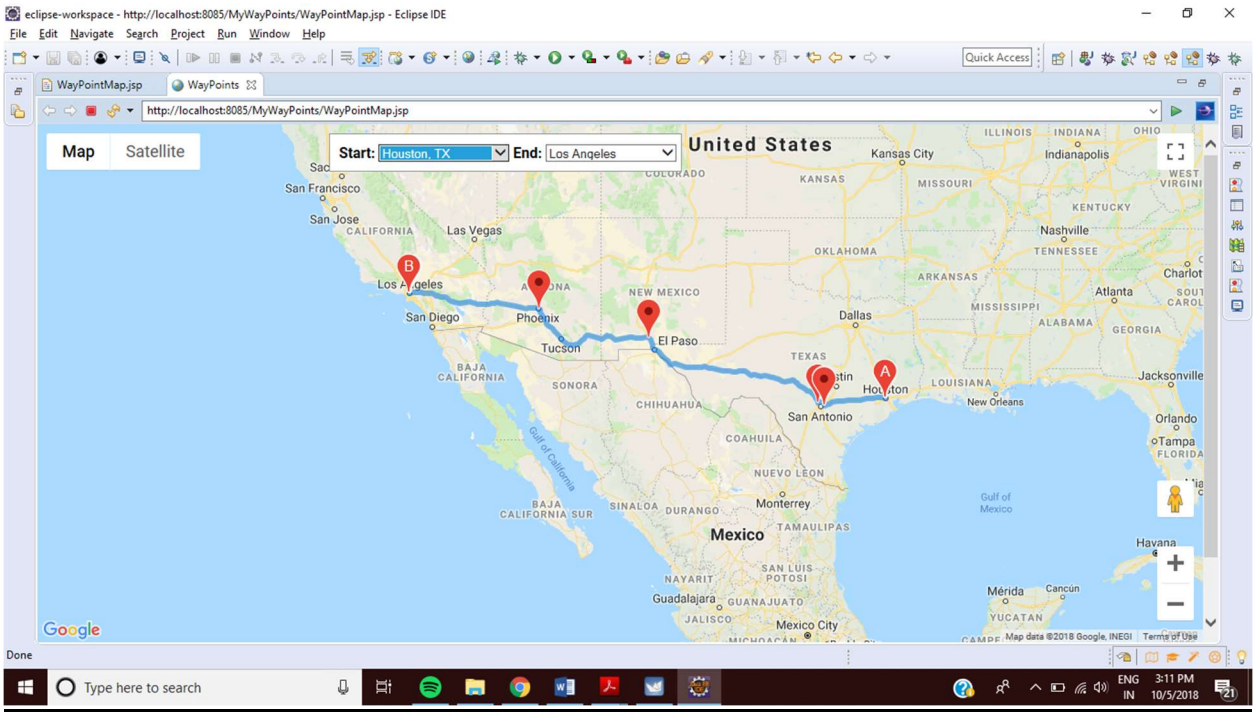
Screenshot 2 - User page to select source and destination



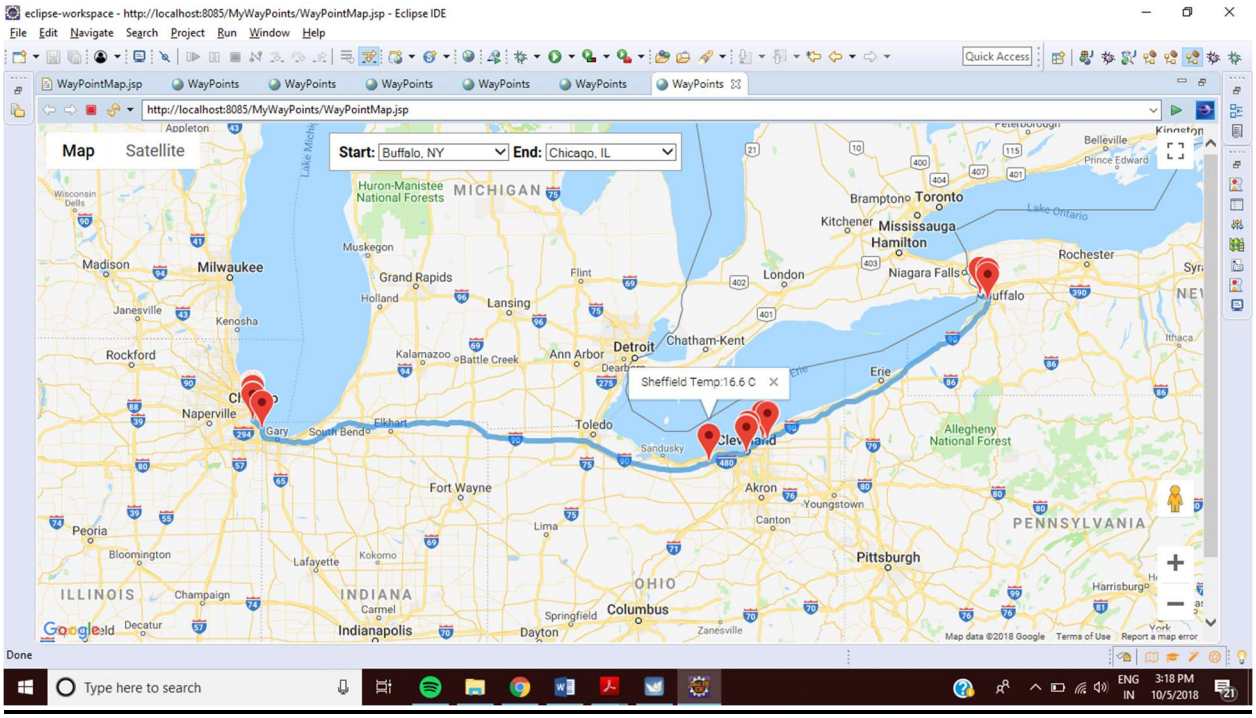
Screenshot 3 - User Selecting his destination



Screenshot 4 – Route Map showing the directions from Source to Destination:



Screenshot 5 – Route Map showing Place name and Temperature at the WayPoint



FUTURE ENHANCEMENTS:

To optimize the performance of the MyWayPoints application use of databases can be helpful. The query results obtained from the Google Maps Direction API can be can be stored in the database. Whenever the same query is requested, the user can simply establish a connection with the database. This reduces the time and the cost function of the Web Application.

CONCLUSION

The MyWayPoints web application is designed in four stages analysis, design, implementation and testing. The application makes use of the Google Maps API and the OpenWeatherMap API to retrieve the information from the user.

The web application can be enhanced by the use of databases. The queries can be stored in the databases. Whenever there is a popular request queried, instead of calling both the APIs, simply a query to the database suffices.

REFERENCES

[1]. https://developers.google.com/maps/documentation/directions/start

[2]. https://openweathermap.org/current

[3]. https://stackoverflow.com/

[4]. https://www.geeksforgeeks.org/

[5]. https://www.javatpoint.com/

[6]. https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest

[7]. https://www.eclipse.org/eclipse/

[8]. https://www.w3schools.com/js/