CHAPTER V

IMPLEMENTATION AND TESTING

5.1 Implementation

This is head of HTML. In the head of HTML page, there are title, CSS, and javascript. It is mostly applied to all HTML page.

```
<head>
  <title>Point of Interests Semarang - Home</title>
  <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false&language=en"></script>
  <script type="text/javascript" src="jquery/jquery-latest.min.js"></script>
      <link rel="stylesheet" type="text/css" href="style.css">
      <style type="text/css">
      .content {
      position: relative;
      left: 8%;
      top: 50px;
      margin-bottom: 100px;
      width: 80%;
      padding: 10px;
      border:2px solid;
      background: #FACC2E;
      }

      p {
      width: 60%;
      color: #6B6108;
      }

      #image1 {
      position: relative;
      float: right;
      top: 70px;
      }

      #image2 {
      margin-left: 30px;
      }

      </style>
</head>
```

Figure 5.1 Head of HTML page

In this website, there is feature to change language. For this feature, this website using CSS div switch stylesheet.. First, make two different div with same layout but different language. Each div is put attribute lang="(language)", for example: lang="en". In HTML body, put class="(language)" to set it to default. Then, make button to change the language.

```
<body class="id" onload="initialize()">

    <div align="right">
<button onclick="document.body.className='en'">English</button>
<button onclick="document.body.className='id'">Indonesian</button>
    </div>

    <h1 align="center">Point of interests Semarang</h1>

    <div lang="en" align="center" id="menu"> <!--ENGLISH-->
    <ul>
    <li><a href="index.php">Home</a></li>
    <li><a href="category.php">Category</a></li>
    <li><a href="direction.php">Direction</a></li>
    <li><a href="input.php">Input Data</a></li>
    </ul>
    </div>

    <div lang="id" align="center" id="menu"> <!--INDONESIAN-->
    <ul>
    <li><a href="index.php">Beranda</a></li>
    <li><a href="category.php">Kategori</a></li>
    <li><a href="direction.php">Petunjuk Arah</a></li>
    <li><a href="input.php">Tambah Data</a></li>
    </ul>
    </div>
```

Figure 5.2 HTML Code to change language

Second, make code in CSS to make div switches when the button clicked. In example, code body.id > [lang=en] { display:none; } means body.id is selected, body.en will not be displayed. Otherwise, body.en does so.

```
body.id > [lang=en] {
        display: none;
}
body.en > [lang=id] {
        display: none;
}
```

Figure 5.3 CSS Code to switch div

In category menu, there is map that shows marker datas which are divided by category. Data in the map is stored in txt format file and use get JSON to display it on the map. To get data from txt file, use code $.getJSON("data.txt", function(json). Then code for (var i = 0; i < json.length; i++) is used to loop all datas that are stored in txt file.

Code var marker = createMarker(point,name,html,category); is used to make marker and show category as it selected.

```
    // Read the data
  $.getJSON("data.txt", function(json) {
      var firstcat = null;
      var bounds = new google.maps.LatLngBounds();
      for (var i = 0; i < json.length; i++) {
        // obtain the attributes of each marker
        var lat = parseFloat(json[i].lat);
        var lng = parseFloat(json[i].lng);
        var point = new google.maps.LatLng(lat,lng);
        bounds.extend(point);
        var name = json[i].title;
        var category = json[i].category;
        var image = json[i].image;
        var html = "<b>"+name+"</b><br>"+json[i].description+"</p>Category:"+category+"<br><br><a href="+image+" target='_blank'><img src="+image+"
width='80' height='50' ></a>";
        // create the marker
        var marker = createMarker(point,name,html,category);
        if (category in markers == false) {
          if (firstcat == null) firstcat = category;
          markers[category] = [];
        }
        markers[category].push(marker);
      }

      // == show or hide the categories initially ==
      for (category in markers) {
        createCategoryDropdown(category);
      }
      select(firstcat);
      // == create the initial sidebar ==
      makeSidebar();
//    map.fitBounds(bounds);
    });
  }
```

Figure 5.4 Code to get data from JSON and make marker

Direction menu is used to show direction from a place to another place. This feature uses Google API code. Variable that will be used in function has to be declared first. In this code, var directionDisplay; and var directionsService = new google.maps.DirectionsService(); are declared at the beginning. Var directionDisplay is used in function initialize() to display direction and also used in function calcRoute() if selected right then showing the direction. Var directionService is used in function calcRoute to make route when request is right. Var start and end are declared to get ID for select.

```
var directionsDisplay;
var directionsService = new google.maps.DirectionsService();

var map

function initialize() {

directionsDisplay = new google.maps.DirectionsRenderer();

// menampilkan center map
var semarang = new google.maps.LatLng(-6.9667, 110.41677);
var mapOptions = {
        zoom: 13,
        center: semarang
  };
map = new google.maps.Map(document.getElementById('map'), mapOptions);
directionsDisplay.setMap(map);
directionsDisplay.setPanel(document.getElementById('directions-panel'));

  var control = document.getElementById('control');
  control.style.display = 'block';
  map.controls[google.maps.ControlPosition.TOP_CENTER].push(control);

}

function calcRoute() {
  var start = document.getElementById('start').value;
  var end = document.getElementById('end').value;
  var request = {
    origin:start,
    destination:end,
    travelMode: google.maps.TravelMode.DRIVING
    };
  directionsService.route(request, function(response, status) {
    if (status == google.maps.DirectionsStatus.OK) {
```

Figure 5.5 Function to get direction

All datas are stored in txt file, therefore it uses get JSON in select option to get data. Code $.getJSON is used to get data from other file where data is stored. JSON generally contains data attribute in array. Code "$ ('#start').append($('<option/>') means dropdown select with ID="start" contains option that is read from option.title. Code .text(option.title)); means option select displays title data attributes. Code .attr("value",option.lat + ", " + option.lng) is used to read data latitude and longitude.

```
125     <div id="control">
126     <b>Start: </b>
127     <select id="start" onchange="calcRoute();">
128    <script type="text/javascript">
129         $(document).ready(function(){
130             $.getJSON('data.txt', function(data) {
131
132               $.each(data, function(i, option){
133                     console.log(option.title)
134                     $('#start').append($('<option/>')
135                                         .attr("value",option.lat + ", " + option.lng)
136                                         .text(option.title));
137               });
138             });
139         });
140     </script>
141     </select>
142     <b>End: </b>
143     <select id="end" onchange="calcRoute();">
144    <script type="text/javascript">
145         $(document).ready(function(){
146             $.getJSON('data.txt', function(data) {
147
148               $.each(data, function(i, option){
149                     console.log(option.title)
150                     $('#end').append($('<option/>')
151                                         .attr("value",option.lat + ", " + option.lng)
152                                         .text(option.title));
153               });
154             });
155         });
156     </script>
157     </select>
158     </div>
```

Figure 5.6 Code to make select option and get data from JSON

Input data menu is used to add data to JSON. In this menu, data can be added by filling all the form manually or clicked on the map. Clicking on the map is easier way to input data because latitude and longitude are filled automatically. This is code to make form info window when the marker clicked. Code google.maps.event.addListener(map, "click", function(event) { marker = new google.maps.Marker is used to make marker when map clicked.

Code google.maps.event.addListener(marker, "click", function() { infowindow.open(map, marker);

is used to make info window appears when the marker clicked.

Code google.maps.event.addListener(infowindow,'domready',function() { document.getElementById('lat').value=marker.getPosition().lat();

document.getElementById('lng').value=marker.getPosition().lng(); }); is used to fill latitude and longitude automatically when the marker clicked.

```
function initialize() {
  var latlng = new google.maps.LatLng(-6.9667, 110.41677);
  var options = {
    zoom: 13,
    center: latlng,
    mapTypeId: google.maps.MapTypeId.ROADMAP
  }
  var map = new google.maps.Map(document.getElementById("map"), options);
  var html = "<form action='save_json.php' method='post' onsubmit='target_popup(this)'>" +
             "Nama tempat: <input type='text' name='title' id='title' /></br>" +
             "Latitude: <input type='text' name='lat' id='lat' /></br>" +
             "Longitude: <input type='text' name='lng' id='lng' /></br>" +
             "Deskripsi: <input type='text' name='description' id='description' /></br>" +
             "Kategori: <input type='text' name='category' id='category' /></br>" +
             "Gambar(url): <input type='text' name='image' id='image' /></br>" +
             "<input type='submit' id='submit' value='Submit' /></br>" +
            "</form>";
infowindow = new google.maps.InfoWindow({
 content: html
});

google.maps.event.addListener(map, "click", function(event) {
    marker = new google.maps.Marker({
      position: event.latLng,
      map: map
    });
    google.maps.event.addListener(marker, "click", function() {
      infowindow.open(map, marker);
      google.maps.event.addListener(infowindow,'domready',function() {
         document.getElementById('lat').value=marker.getPosition().lat();
         document.getElementById('lng').value=marker.getPosition().lng();
      });
    });
});
}
```

Figure 5.7 Code to add data by clicking on the map

To process data storing, there is form action for this form. In this form, form action='save_json.php'.

Code $filetxt = '/opt/lampp/htdocs/project/data.txt'; is directory path where data is located.

Code if(isset($_POST['(attribute)']) will return TRUE only *if* all of the parameters are set.

Code if(empty($_POST['(attribute)']) will check whether a variable is *empty* or not.

Code $data = array is used to send data to array where data is added.

```
1 <?php
2 $filetxt = '/opt/lampp/htdocs/project/data.txt';
3
4 if(isset($_POST['title']) && isset($_POST['lat']) && isset($_POST['lng']) && isset($_POST['description']) && isset($_POST['category']) && isset($_POST
  ['image'])) {
5   if(empty($_POST['title']) || empty($_POST['lat']) || empty($_POST['lng']) || empty($_POST['description']) || empty($_POST['category']) || empty($_POST
  ['image'])) {
6     echo 'Semua field harus diisi';
7   }
8   else {
9     $data = array(
10      'title'=> $_POST['title'],
11      'lat'=> (float) $_POST['lat'],
12      'lng'=> (float) $_POST['lng'],
13      'description'=> $_POST['description'],
14      'category'=> $_POST['category'],
15      'image' => $_POST['image'],
16    );
17    $filetxt = '/opt/lampp/htdocs/project/data.txt';
18    $arr_data = array();
19    if(file_exists($filetxt)) {
20      $jsondata = file_get_contents($filetxt);
21      $arr_data = json_decode($jsondata, true);
22    }
23
24    $arr_data[] = $data;
25    $jsondata = json_encode($arr_data, JSON_PRETTY_PRINT);
26
27    if(file_put_contents('/opt/lampp/htdocs/project/data.txt', $jsondata)) echo 'Data berhasil disimpan';
28    else echo 'Tidak dapat menyimpan data di "/opt/lampp/htdocs/project/data.txt"';
29  }
30 }
31 else echo 'Form tidak tersirim';
32 ?>
```

Figure 5.8 PHP Code to add data in txt file

5.2 Testing

In this website there are four menus, those are home, category, direction and input data. In home page, it is introduction about this website, there are images about point of interests in Semarang. This is appearance of website home page.

Figure 5.9 Home page

In category menu, there is map that shows marker datas which are divided by category. There is sidebar at right side. Sidebar display marker list in the map. When marker on the map or marker list in the sidebar is clicked, it shows information about point of interest. In the example, Gereja Blenduk in marker list is cliked, then info window appears. Info window contains about title, description, category and picture. In example, clicking Gereja Blenduk. Info window displays the title is Gereja Blenduk, description is Gereja Blenduk, category is sejarah and picture of Gereja Blenduk.

Figure 5.10 Category menu page

There is picture of point of interest when info window appears. Picture might be too small to view. Therefore, picture can be enlarged in new tab when clicked. This is display when picture is enlarged, this example is picture of Gereja Blenduk.

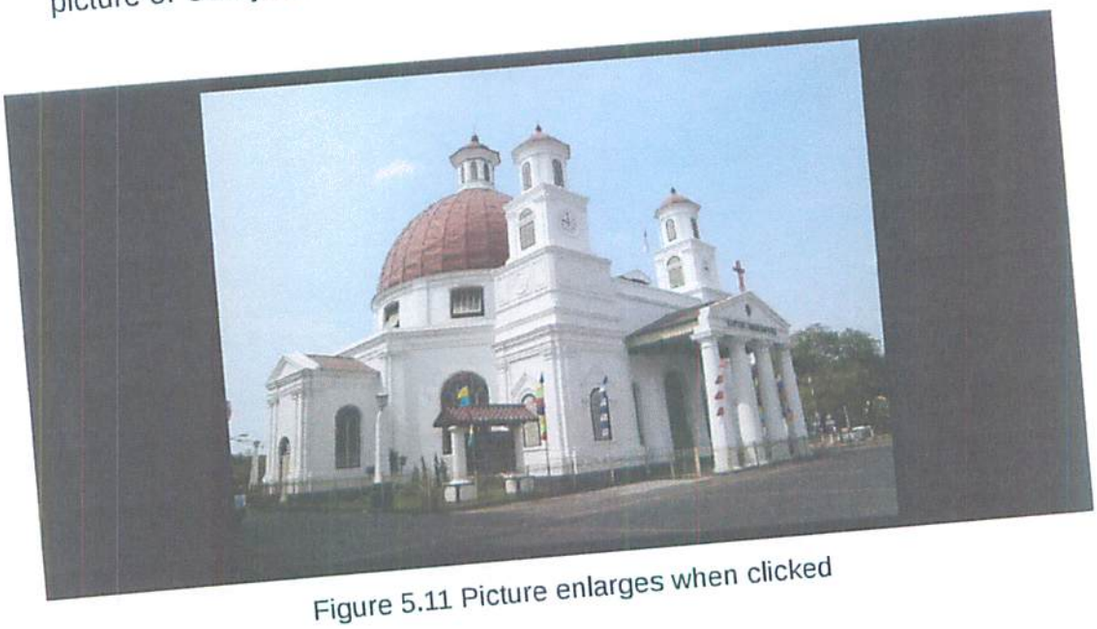

Figure 5.11 Picture enlarges when clicked

This is direction menu page. There are select list at start point to end point. It will show direction from a place to another place. It displays distance, estimation time and steps from start point to end point in the sidebar. In example, selecting Lawang Sewu at start point and Simpang Lima, Semarang at end point. It will tell that start at point A is located in Jalan Pemuda no.142 Semarang Tengah 50132, Indonesia. End at point B is located in Jalan Simpang Lima Semarang Selatan, Kota Semarang, Jawa Tengah 50241, Indonesia. It show distance 1,9km from Lawang Sewu to Simpang Lima and estimation time going there is about 2 minutes. It displays step from Lawang Sewu to Simpang Lima.

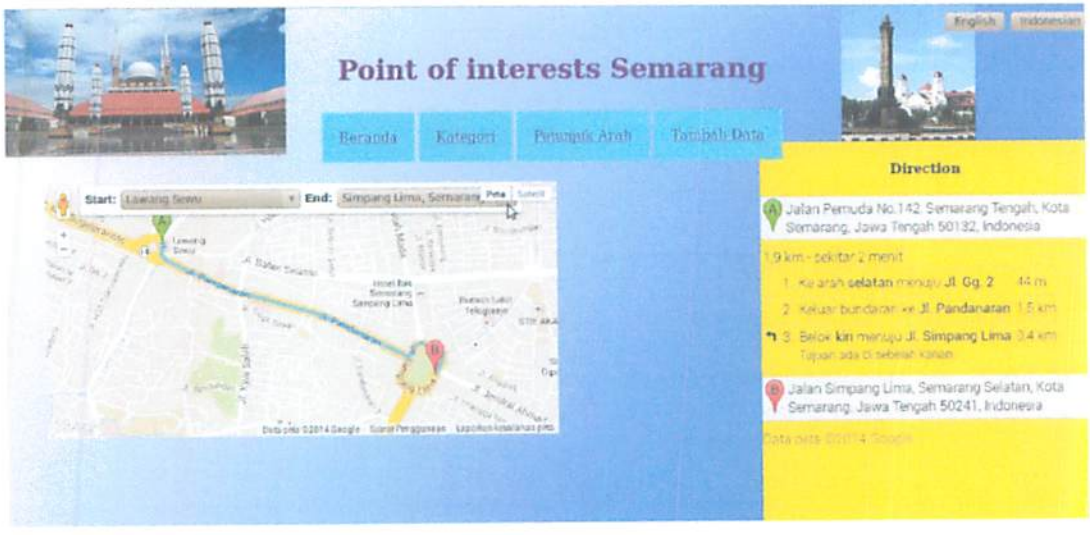

Figure 5.12 Direction menu page

This is input data menu. Input data menu is used to add data to JSON. In this menu, data can be added by filling all the form manually or clicked on the map. Clicking on the map is easier way to input data because latitude and longitude are filled automatically.

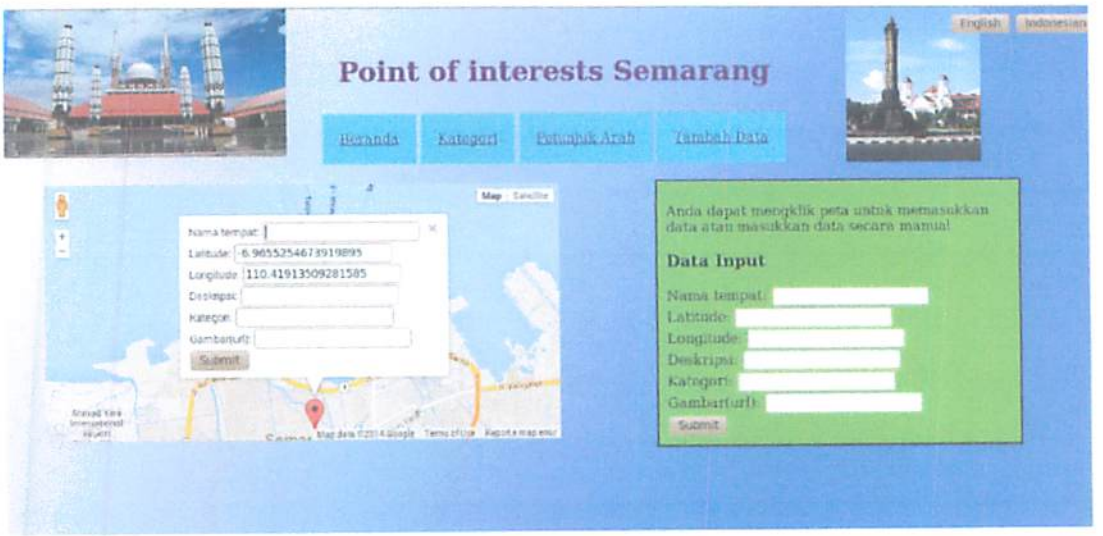

Figure 5.13 Input data menu page

All form has to be filled. If there is one empty form, data will not be submitted. It will appears pop-up window "Semua field harus diisi" means all field have to be filled. There will always be pop-up window that tell whether data submitted or not.

Figure 5.14 Input data when there is empty form

If data successfully submitted, it will be saved in txt file. In example, fill in title is “coba”, latitude and longitude is already filled, description is “coba”, category is “sejarah”, gambar (url) is “http://upload.wikimedia.org/wikipedia/commons/2/2f/Becak_Tugu_Muda_Semarang_Central_Java.jpg”. It will appears pop-up window “Data berhasil disimpan” means data has successfully saved.

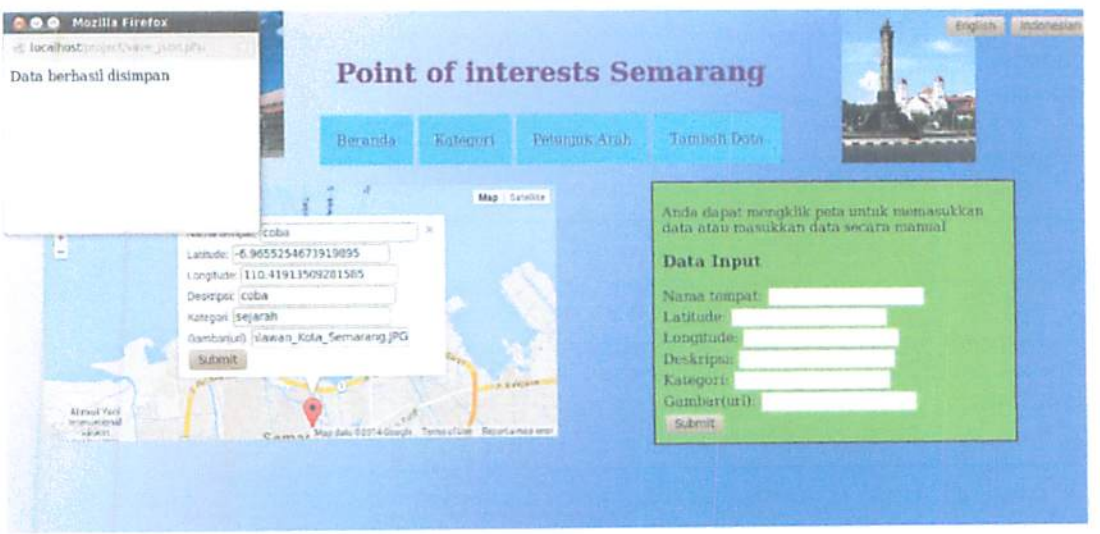

Figure 5.15 Input data when data successfully submitted