



**CS 458**  
**Software Verification & Validation**  
**2019-2020 Spring**

**PROJECT #3**

Berk Yıldız 21502040

## Table of Contents

<b>1. Implementation of Website .....</b>	<b>3</b>
<b>2. Test Codes &amp; Main Codes.....</b>	<b>6</b>
• <b>Enter Coordinates Page .....</b>	<b>6</b>
• <b>Current Location Page.....</b>	<b>11</b>
• <b>Distance to Earth Center (Manuel) .....</b>	<b>13</b>
• <b>Distance to Earth Center (GPS) .....</b>	<b>17</b>
<b>3. Code Refactoring.....</b>	<b>19</b>
<b>4. Evaluation of TDD Experience .....</b>	<b>20</b>

## 1. Implementation of Website

- The website is implemented by JavaScript and HTML.
- Website is scalable for desktop, mobile and tablet devices.

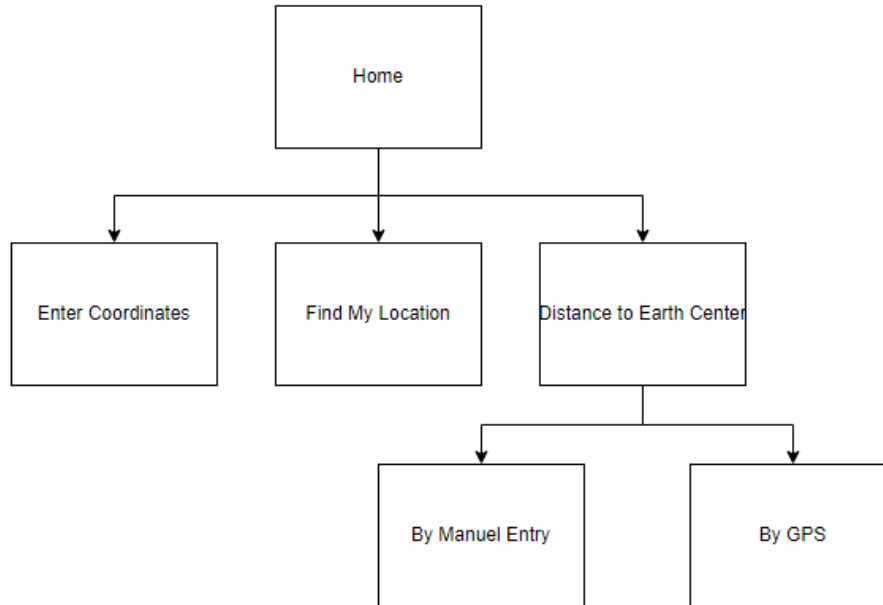


Figure 1: Sitemap

### Welcome to City Finder

[Enter Coordinates](#)

[Find My Place](#)

[Distance to Earth Center](#)



Figure 2: Home Page

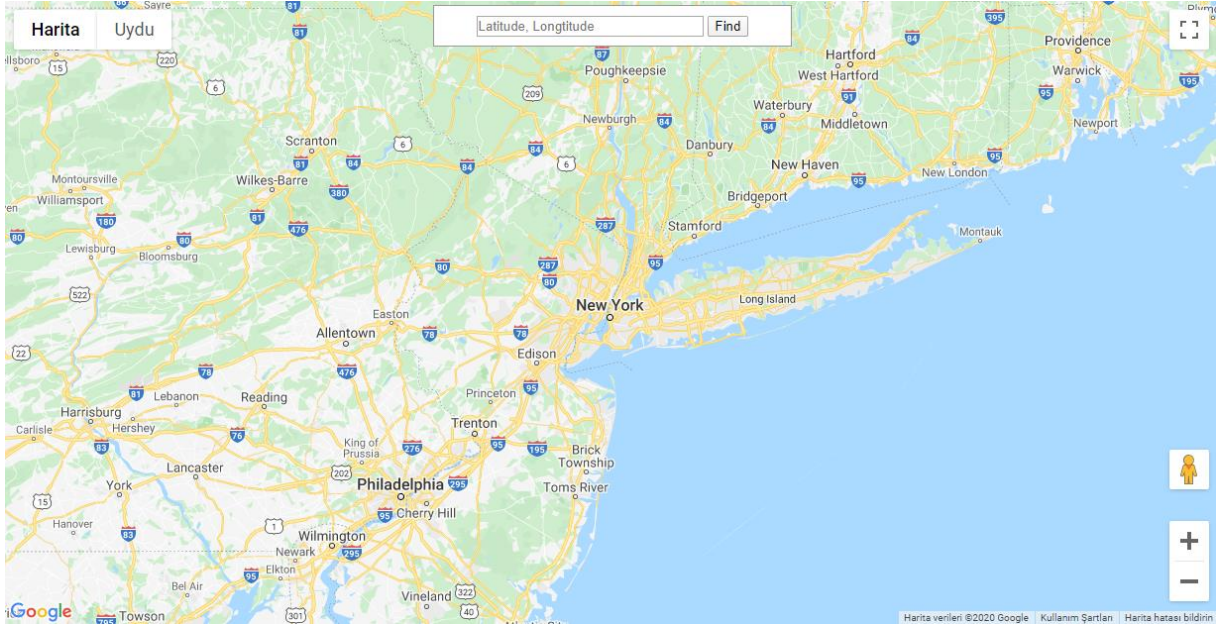


Figure 3: Enter Coordinates Page

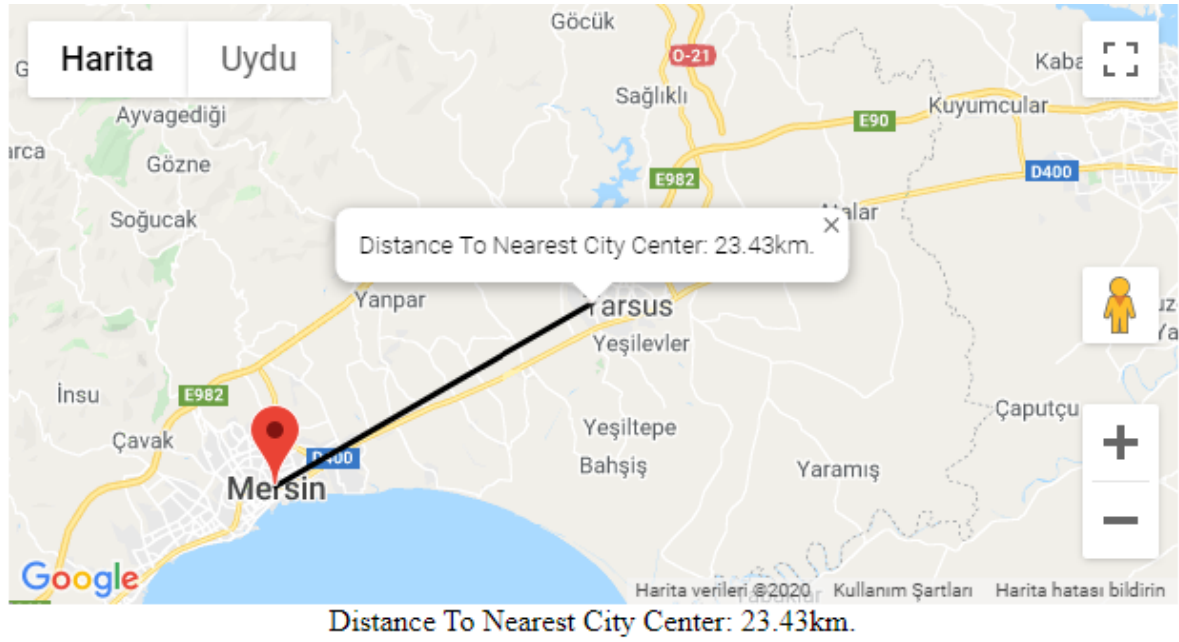


Figure 4: Find My Location Page

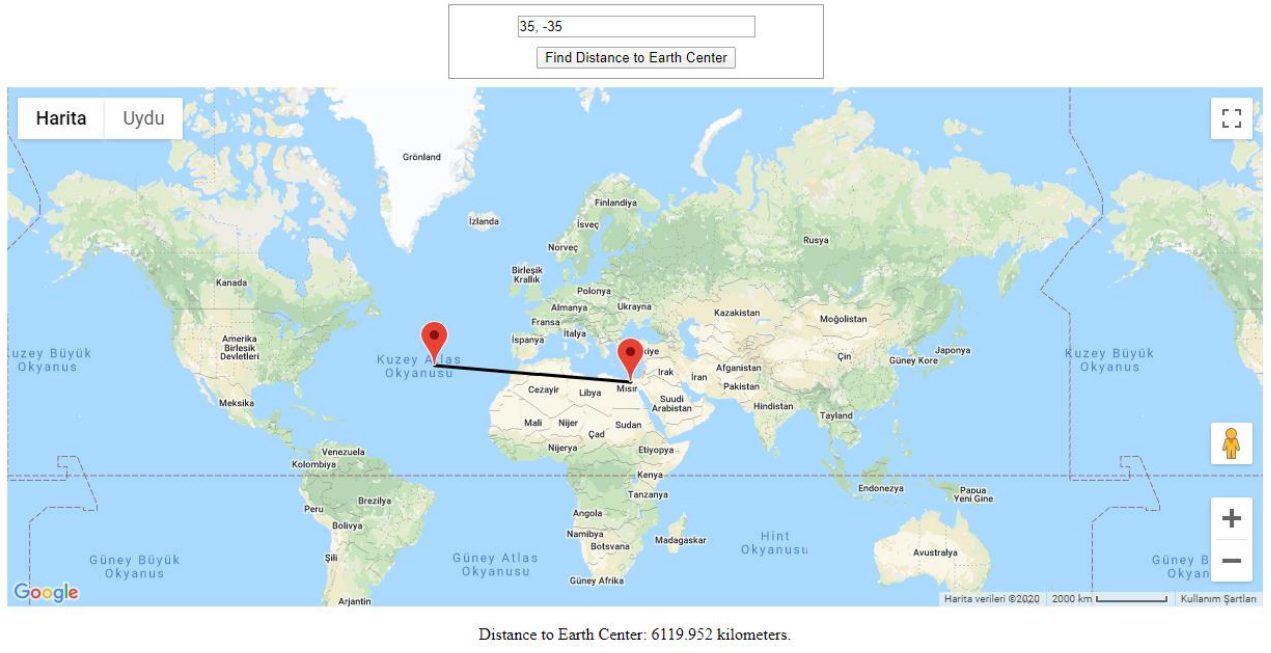


Figure 5: Distance to Earth Center (Manuel) Page



Figure 6: Distance to Earth Center (GPS) Page

## 2. Test Codes & Main Codes

- Enter Coordinates Page

*Verify if coordinate input area and Find button exist on the screen.*

```
driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/location/enterCoordinates.html");
boolean exist1 =
driverChrome.findElement(By.cssSelector("#latlng")).isDisplayed();
boolean exist2 =
driverChrome.findElement(By.cssSelector("#submit")).isDisplayed();
boolean exist3 =
driverChrome.findElement(By.cssSelector("#map")).isDisplayed();
if(exist1 && exist2 && exist3) System.out.println("Existence Of Elements: Passed");
else System.out.println("Existence Of Elements: Failed");
```

```
<body>
  <div id="floating-panel">
    <input id="latlng" type="text" placeholder="Latitude, Longitude" value="">
    <input id="submit" type="button" value="Find">
  </div>
</body>
</html>
```

*Enter valid latitude & longitude value and click Find. Verify if correct info window shows up.*

```
driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("40, 40");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message1 =
driverChrome.findElement(By.xpath("//*[ @id=\\"map\\" ]/div/div/div[1]/div[3]/div/div[4]/div/div/div/div")).getText();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if (message.equals("Erzincan") || message1.equals("Erzincan"))
System.out.println("Valid Input Test: Passed");
else System.out.println("Valid Input Test: Failed");
```

```
<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <title>Enter Coordinates</title>
```

```

</head>
<body>
  <div id="floating-panel">
    <input id="latlng" type="text" placeholder="Latitude, Longitude" value="">
    <input id="submit" type="button" value="Find">
  </div>
  <div id="map"></div>
  <div id="msg"></div>
  <script>
    function initMap() {
      var map = new google.maps.Map(document.getElementById('map'),);
      var geocoder = new google.maps.Geocoder;
      var infowindow = new google.maps.InfoWindow;

      document.getElementById('submit').addEventListener('click', function() {
        geocodeLatLng(geocoder, map, infowindow);
      });
    }

    function geocodeLatLng(geocoder, map, infowindow) {
      var input = document.getElementById('latlng').value;
      var latlngStr = input.split(',', 2);
      var latlng = {lat: parseFloat(latlngStr[0]), lng: parseFloat(latlngStr[1])};
      geocoder.geocode({'location': latlng}, function(results, status) {
        if (status === 'OK') {
          if (results[0]) {
            map.setZoom(10);
            var marker = new google.maps.Marker({
              position: latlng,
              map: map
            });
            infowindow.setContent(results[0].address_components[3].long_name);
            console.log(results[0]);
            infowindow.open(map, marker);
            document.getElementById('msg').innerHTML = results[0].address_components[3].long_name;
          } }
        }
      });
    }
  </script>
  <script async defer
    src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDl5V0Iht6xQKcLk_chezV-f_syogLDl8k&callback=initMap">
  </script>
</body>

```



**\*\*\* Test scenarios below are just passing by the same source code because of the implementation of Google Maps API. They use same components.**

*Enter invalid coordinate and click Find. Verify if “Geocoder failed due to: INVALID\_REQUEST” message shows up.*

```
driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("aaa, aaa");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.switchTo().alert().getText();
if(message.equals("Geocoder failed due to: INVALID_REQUEST"))
System.out.println("Invalid Input Test: Passed");
else System.out.println("Invalid Input Test: Failed");
```

*Enter 0, 0 to as coordinates input. Verify if “Geocoder failed due to: ZERO\_RESULTS” message shows up.*

```
driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("0, 0");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.switchTo().alert().getText();
if(message.equals("Geocoder failed due to: ZERO_RESULTS"))
System.out.println("Zero Input Test: Passed");
else System.out.println("Zero Input Test: Failed");
```

*Leave coordinate input blank. Verify if “Geocoder failed due to: INVALID\_REQUEST” message shows up.*

```
driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.switchTo().alert().getText();
if(message.equals("Geocoder failed due to: INVALID_REQUEST"))
System.out.println("Blank Input Test: Passed");
else System.out.println("Blank Input Test: Failed");
```

*Enter valid latitude value, leave longitude blank and click Find. Verify if “Geocoder failed due to: INVALID\_REQUEST” message shows up.*

```
driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("40, ");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
```



```

message = driverChrome.switchTo().alert().getText();
if(message.equals("Geocoder failed due to: INVALID_REQUEST"))
System.out.println("Blank Longitude Test: Passed");
else System.out.println("Blank Longitude Test: Failed");

```

*Leave latitude blank, enter valid longitude value and click Find button. Verify if correct if “Geocoder failed due to: INVALID\_REQUEST” message shows up.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys(", 40");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.switchTo().alert().getText();
if(message.equals("Geocoder failed due to: INVALID_REQUEST"))
System.out.println("Blank Latitude Test: Passed");
else System.out.println("Blank Latitude Test: Failed");

```

*Enter invalid latitude value, valid longitude value and click Find. Verify if “Geocoder failed due to: INVALID\_REQUEST” message shows up.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("bb, 40 ");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.switchTo().alert().getText();
if(message.equals("Geocoder failed due to: INVALID_REQUEST"))
System.out.println("Invalid Latitude Test: Passed");
else System.out.println("Invalid Latitude Test: Failed");

```

*Enter valid latitude value, invalid longitude value and click Find. Verify if “Geocoder failed due to: INVALID\_REQUEST” message shows up.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/enterCoordinates.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("40, bb ");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.switchTo().alert().getText();
if(message.equals("Geocoder failed due to: INVALID_REQUEST"))
System.out.println("Invalid Longitude Test: Passed");
else System.out.println("Invalid Longitude Test: Failed");

```

```

<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <title>Enter Coordinates</title>

```

```

</head>
<body>
  <div id="floating-panel">
    <input id="latlng" type="text" placeholder="Latitude, Longitude" value="">
    <input id="submit" type="button" value="Find">
  </div>
  <div id="map"></div>
  <div id="msg"></div>

  <script>
  function initMap() {
    var map = new google.maps.Map(document.getElementById('map'), {
      zoom: 8,
      center: {lat: 40.731, lng: -73.997}
    });
    var geocoder = new google.maps.Geocoder;    var infowindow = new
google.maps.InfoWindow;

    document.getElementById('submit').addEventListener('click', function() {
      geocodeLatLng(geocoder, map, infowindow);
    });
  }

  function geocodeLatLng(geocoder, map, infowindow) {
    var input = document.getElementById('latlng').value;
    var latlngStr = input.split(',', 2);

    var latlng = {lat: parseFloat(latlngStr[0]), lng: parseFloat(latlngStr[1])};
    geocoder.geocode({'location': latlng}, function(results, status) {
      if (status === 'OK') {
        if (results[0]) {
          map.setZoom(10);
          var marker = new google.maps.Marker({
            position: latlng,
            map: map
          });
          infowindow.setContent(results[0].address_components[3].long_name);

          console.log(results[0]);
          infowindow.open(map, marker);
          document.getElementById('msg').innerHTML = results[0].address_components[3].long_name;
        } else {
          window.alert('No results found');
        }
      }
    });
  }
  </script>

```

```

        } else {
            window.alert('Geocoder failed due to: ' + status);
        }
    });
}
</script>
<script async defer
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDl5V0Iht6xQKcLk_
chezV-f_syogLDl8k&callback=initMap">
</script>
</body>
</html>

```

- **Current Location Page**

*Verify if correct info window shows up with correct output.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/currentLocation.html");
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message =
driverChrome.findElement(By.xpath("//*[@id=\"map\"]/div/div/div[1]/div[3]/div/d
iv[4]/div[1]/div/div/div/div/div")).getText();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message1 = driverChrome.findElement(By.cssSelector("#msg")).getText();
if (message.equals("Distance To Nearest City Center: 23.54km.") ||
message1.equals("Distance To Nearest City Center: 23.54km."))
System.out.println("Current Location Test: Passed");
else System.out.println("Correct Location Test: Failed");

```

```

<body>

<div id="map"></div>
<div id="msg"></div>
<script>
    var map,line, infoWindow, geocoder, nearestCity, dakota, pos, mk1, mk
2, nearestCityLatLng,Lat1,Lng1;

    function haversine_distance(mk1, mk2) {
        var R = 6371.0710;
        var rlat1 = mk1.position.lat() * (Math.PI/180);
        var rlat2 = mk2.position.lat() * (Math.PI/180);
        var difflat = rlat2-rlat1;
        var difflon = (mk2.position.lng()-
mk1.position.lng()) * (Math.PI/180);

        var d = 2 * R * Math.asin(Math.sqrt(Math.sin(difflat/2)*Math.sin(diff
lat/2)+Math.cos(rlat1)*Math.cos(rlat2)*Math.sin(difflon/2)*Math.sin(difflon
/2)));
    }

```

```

    return d;
}

function initMap() {
    map = new google.maps.Map(document.getElementById('map'), {
        center: {lat: -34.397, lng: 150.644},
        zoom: 6
    });
    infoWindow = new google.maps.InfoWindow;
    geocoder = new google.maps.Geocoder;
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(function(position) {
            pos = {
                lat: position.coords.latitude,
                lng: position.coords.longitude
            };

            infoWindow.setPosition(pos);

            infoWindow.open(map);
            map.setCenter(pos);
            console.log(pos);

            geocoder.geocode({'location': pos}, function(results, status) {
                if (status === 'OK') {
                    if (results[0]) {
                        map.setZoom(10);
                        var marker = new google.maps.Marker({
                            position: pos,
                            map: map
                        });

                        console.log(results[0]);
                        nearestCity = results[0].address_components[4].long_name;

                        geocoder.geocode({
                            'address': nearestCity
                        }, function(results, status) {
                            if (status == google.maps.GeocoderStatus.OK) {
                                Lat1 = results[0].geometry.location.lat();
                                Lng1 = results[0].geometry.location.lng();
                                console.log ("!!!!!" + Lat1 + " " + Lng1 );

                                nearestCityLatLng = {
                                    lat: Lat1,
                                    lng: Lng1
                                };

```

```

        mk1 = new google.maps.Marker({position: pos, map: map});
        mk2 = new google.maps.Marker({position: nearestCityLatLng ,
map: map});
        line = new google.maps.Polyline({path: [pos, nearestCityLat
Lng], map: map})
        var distance = haversine_distance(mk1, mk2);
        console.log("Distance: " + distance);
        infoWindow.setContent(

        "Distance To Nearest City Center: " + distance.toFixed(2) +
"km." );
        document.getElementById('msg').innerHTML = "Distance To Near
est City Center: " + distance.toFixed(2) + "km.";
</script>
<script async defer
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDl5V0Iht6xQKcLk_
chezV-f_syogLDl8k&callback=initMap">
</script>

```

- **Distance to Earth Center (Manuel)**

*Verify if coordinate input area and Find Distance to Earth Center button exist on the screen.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
boolean exist1 =
driverChrome.findElement(By.cssSelector("#latlng")).isDisplayed();
boolean exist2 =
driverChrome.findElement(By.cssSelector("#submit")).isDisplayed();
boolean exist3 =
driverChrome.findElement(By.cssSelector("#map")).isDisplayed();
if(exist1 && exist2 && exist3) System.out.println("Existence Of Elements:
Passed");
else System.out.println("Existence Of Elements: Failed");

```

```

<div id="floating-panel">
    <input id="latlng" type="text" placeholder="Latitude, Longitude" valu
e="">
    <input id="submit" type="button" value="Find Distance to Earth Center
">
</div>
<div id="map"></div>
<div id="msg"></div>
<script>
function initMap() {
    // The map, centered on Central Park
    const center = {lat: 39.925533, lng: 32.866287};
    const options = {zoom: 2, scaleControl: true, center: center};
    map = new google.maps.Map(
        document.getElementById('map'), options);

```

```

// Locations of landmarks
document.getElementById('submit').addEventListener('click', function() {
    calculateDistance();
});
}
</script>

```

**\*\*\* Test scenarios below are just passing by the same source code because they use the same methods.**

*Leave coordinate input blank and click Find Distance to Earth Center button. Verify if correct output obtained.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if(message.equals("Distance to Earth Center: NaN kilometers."))
System.out.println("Blank Coordinate Test: Passed");
else System.out.println("Blank Coordinate Test: Failed");

```

*Enter invalid latitude & longitude value and click Find Distance to Earth Center button. Verify if correct output obtained.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("%+^, aaa");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if(message.equals("Distance to Earth Center: NaN kilometers."))
System.out.println("Invalid Coordinate Test: Passed");
else System.out.println("Invalid Coordinate Test: Failed");

```

*Enter valid coordinate and click Find Distance to Earth Center button. Verify if correct output obtained.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);

```



```

message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if(message.equals("Distance to Earth Center: 658.561 kilometers."))
System.out.println("Valid Coordinate Test: Passed");
else System.out.println("Valid Coordinate Test: Failed");

```

*Enter valid latitude value, leave longitude blank and click Find Distance to Earth Center button. Verify if correct output obtained.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("10, ");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if(message.equals("Distance to Earth Center: NaN kilometers."))
System.out.println("Blank Longitude Test: Passed");
else System.out.println("Blank Longitude Test: Failed");

```

*Leave latitude blank, enter valid longitude value and click Find Distance to Earth Center button. Verify if correct output obtained.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys(", 10");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if(message.equals("Distance to Earth Center: NaN kilometers."))
System.out.println("Blank Latitude Test: Passed");
else System.out.println("Blank Latitude Test: Failed");

```

*Enter valid latitude value, invalid longitude value and click Find Distance to Earth Center button. Verify if correct output obtained.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("25, aaa");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if(message.equals("Distance to Earth Center: NaN kilometers."))
System.out.println("Invalid Longitude Test: Passed");
else System.out.println("Invalid Longitude Test: Failed");

```

*Enter invalid latitude value, enter valid longitude value and click Find Distance to Earth Center button. Verify if correct output obtained.*

```

driverChrome.navigate().to("file:///C:/Users/BERKYILDIZ/Desktop/cs458/project3/
location/earthCenterManuel.html");
driverChrome.findElement(By.cssSelector("#latlng")).sendKeys("aaa, 10");
driverChrome.findElement(By.cssSelector("#submit")).click();
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
driverChrome.manage().timeouts().implicitlyWait(3000, TimeUnit.SECONDS);
message = driverChrome.findElement(By.cssSelector("#msg")).getText();
if(message.equals("Distance to Earth Center: NaN kilometers."))
System.out.println("Invalid Latitude Test: Passed");
else System.out.println("Invalid Latitude Test: Failed");

```

```

<!DOCTYPE html>
<html>
  <head>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
    <title>Distance to Earth Center (Manuel)</title>
  </head>
  <body>
    <div id="floating-panel">
      <input id="latlng" type="text" placeholder="Latitude, Longitude" value="">
      <input id="submit" type="button" value="Find Distance to Earth Center">
    </div>
    <div id="map"></div>
    <div id="msg"></div>
    <script>
// Initialize and add the map
var map;
function haversine_distance(mk1, mk2) {
  var R = 6371.0710;
  var rlat1 = mk1.position.lat() * (Math.PI/180);
  var rlat2 = mk2.position.lat() * (Math.PI/180);
  var difflat = rlat2-rlat1;
  var diff lon = (mk2.position.lng()-
mk1.position.lng()) * (Math.PI/180);

  var d = 2 * R * Math.asin(Math.sqrt(Math.sin(difflat/2)*Math.sin(diff
lat/2)+Math.cos(rlat1)*Math.cos(rlat2)*Math.sin(diff lon/2)*Math.sin(diff lon
/2)));
  return d;
}
function calculateDistance() {
  const earthCenter = {lat: 30.013056, lng: 31.208853};
  var input = document.getElementById('latlng').value;
  var latlngStr = input.split(',', 2);
  const latlng = {lat: parseFloat(latlngStr[0]), lng: parseFloat(latlngStr[
1])};

```

```

var mk1 = new google.maps.Marker({position: earthCenter, map: map});
var mk2 = new google.maps.Marker({position: latlng, map: map});
var line = new google.maps.Polyline({path: [earthCenter, latlng], map: map});
var distance = haversine_distance(mk1, mk2);
document.getElementById('msg').innerHTML = "<br>Distance to Earth Center: " + distance.toFixed(3) + " kilometers.";
}
function initMap() {
  const center = {lat: 39.925533, lng: 32.866287};
  const options = {zoom: 2, scaleControl: true, center: center};
  map = new google.maps.Map(
    document.getElementById('map'), options);

  document.getElementById('submit').addEventListener('click', function() {
    calculateDistance();
  });
}
</script>
<script async defer
  src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDl5V0Iht6xQKcLk_
  chezV-f_syogLDl8k&callback=initMap">
</script>
</body>
</html>

```

- **Distance to Earth Center (GPS)**

*Verify if correct output obtained.*

```

<!DOCTYPE html>
<html>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <head>
    <title>Distance to Earth Center (GPS)</title>
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no">
    <meta charset="utf-8">
  </head>
  <body>
    <div id="map"></div>
    <div id="msg">
  </div>
  <div id="map"></div>
  <script>
    var map, infoWindow, pos, mk1, mk2, dakota, line, distance;

```

```

function haversine_distance(mk1, mk2) {
    var R = 6371.0710;
    var rlat1 = mk1.position.lat() * (Math.PI/180);
    var rlat2 = mk2.position.lat() * (Math.PI/180);
    var difflat = rlat2-rlat1;
    var difflon = (mk2.position.lng()-
mk1.position.lng()) * (Math.PI/180);

    var d = 2 * R * Math.asin(Math.sqrt(Math.sin(difflat/2)*Math.sin(difflon/2)+Math.cos(rlat1)*Math.cos(rlat2)*Math.sin(difflon/2)*Math.sin(difflat/2)));
    return d;
}

function initMap() {
    map = new google.maps.Map(document.getElementById('map'), {
        center: {lat: -34.397, lng: 150.644},
        zoom: 6
    });
    infoWindow = new google.maps.InfoWindow;

    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(function(position) {
            pos = {
                lat: position.coords.latitude,
                lng: position.coords.longitude
            };
            map.setCenter(pos);
            map.setZoom(4);
            dakota = {lat: 30.013056, lng: 31.208853};
            mk1 = new google.maps.Marker({position: dakota, map: map});
            mk2 = new google.maps.Marker({position: pos, map: map});
            line = new google.maps.Polyline({path: [dakota, pos], map: map});

            distance = haversine_distance(mk1, mk2);
            document.getElementById('msg').innerHTML = "<br>Distance to Earth Center: " + distance.toFixed(2) + " kilometers.";

        })
    }
}
</script>
<script async defer
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDl5V0Iht6xQKcLk_chezV-f_syogLDl8k&callback=initMap">
</script>
</body>
</html>

```

### 3. Code Refactoring

The first code was pretty neat, not paying attention to spacing and indentation. The only goal when writing the code was just passing the specified tests. The default values of the parameters were irrelevant to the project. There was no improvement in the design of the pages. Also, there was no comment anywhere in the code.

In terms of refactoring, primarily page designs were improved and style sections were added to html files. Code snippets combined. Typos fixed. Indentation and spacing were organized. Comments have been added to the required lines in the code. The default values of the parameters were adjusted and the values related to the project were determined. While all these changes were made, there was no change in functionality.

**Codes written for test cases in section 2 are not refactored.** The HTML files that refactored are in the project file.

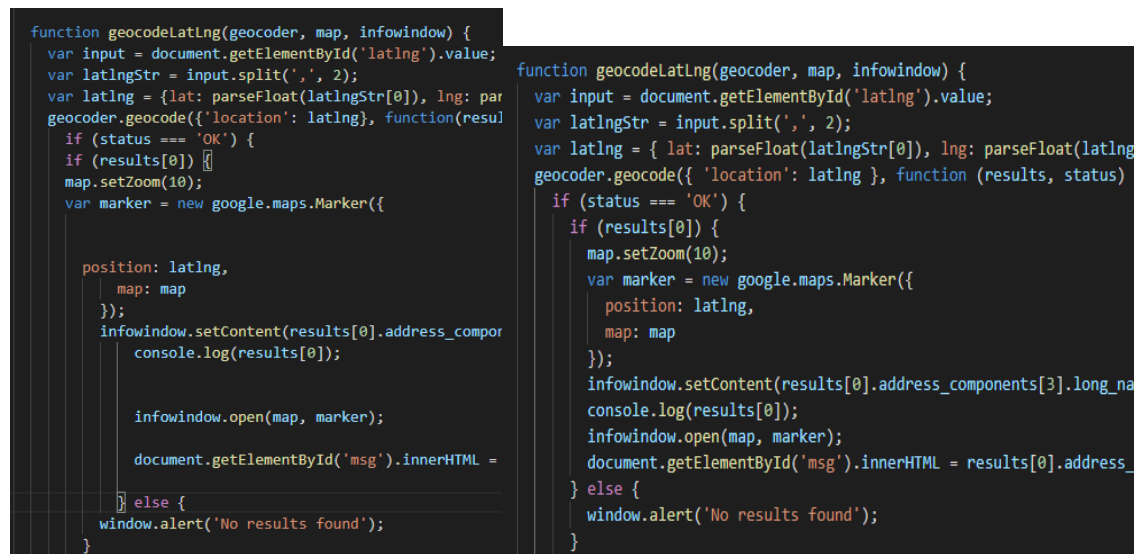
- **Some refactoring examples:**

Adding style to pages:

```
<style>
  #map {
    height: 80%;
    width: 100%;
  }
  #msg{
    text-align: center;
  }
  html, body {
    height: 100%;
    margin: 0;
    padding: 0;
  }
</style>
```

```
<style>
  #map {
    height: 50%;
    margin-left: 25%;
    margin-right: 25%;
  }
  #msg{
    text-align: center;
  }
  html, body {
    height: 100%;
    margin: 0;
    padding: 0;
  }
</style>
```

## Indentation and spacing:



The figure displays two side-by-side code snippets in a dark-themed editor, illustrating the 'Before' and 'After' states of a refactoring process. The 'Before' code on the left shows a function `geocodeLatLng` with a complex, deeply nested structure. It uses `document.getElementById('latlng').value` to get input, splits it, and then uses `parseFloat` to parse the latitude and longitude. The geocoding logic is wrapped in multiple nested `if` statements, including one for `results[0]` and another for `status === 'OK'`. The 'After' code on the right shows the same function but with a cleaner, more readable structure. It uses `document.getElementById('latlng').value` directly, and the `parseFloat` calls are moved to the point where the latitude and longitude are assigned to the `latlng` object. The `if` statements are also restructured to be more concise and easier to follow.

Figure 7: Before and after refactoring

## 4. Evaluation of TDD Experience

Test-Driven Development is the practice of writing a test for a piece of required functionality, before writing any implementation code. It relies on the repetition of a very short development cycle. Main advantage of TDD is that it provides fast feedback which helps developer to take faster action for solving problems. When we consider the development process, debugging and bug fixing take more time than writing the code. TDD reduces the time on debugging and rework by having a fast feedback mechanism because developer continues to coding by dealing with problems and fixing the bugs simultaneously. So TDD has advantages in terms of development velocity. Evaluation in terms of code quality can depend on the developer. Theoretically TDD serves very well for code quality by its refactor stage because before starting to another cycle, refactoring has to be completed. However, if developer decides to think refactoring as a second level work and delay it for focusing on developing main functionalities of the software, code quality would decrease and TDD can be disadvantageous in terms of code quality.