CEGEG077 - Web and Mobile GIS

Location Based Quiz

Author: Jack Philpott, 17122213

Contact: jack.philpott.17@ucl.ac.uk

Date: 05.05.2018

Summary:
The overall objective from the assignment is to create a mobile and web app in order to perform a location-based quiz. This was implemented using multiple different programming languages, such as: HTML, CSS, JavaScript, SQL. The predominant software used to generate the applications were PhoneGap, Notepad++ and Yummy. The application was not built with the intention of commercial use, rather, it was designed to demonstrate an understanding of various programming languages and the application of the material learnt in class.

For more information and documentation please visit: 'https://github.com/JackPhilpott'

Jack Philpott
17122213

A Location Based Quiz

The overriding aim of the project is to generate a web and mobile application, designed to allow the user to partake in a location-based quiz. The quiz aims to provide insightful knowledge and allow the user to develop a greater understanding of both; the historical legacy encompassing UCL's past, and the modern-day architecture surrounding the UCL central campus. Therefore, the aforementioned quiz will be considered both to be within two-minute walking distance of the UCL grounds. The web application can be used by anyone of any age. The user does not need to have any prior experience in order to use the applications.

The project consists of two parts. Firstly, a mobile application will be generated for Android devices, which is why the application has been developed using a Motorola, Model XT1793. In order to create the mobile application, the mobile hardware application PhoneGap was used. The mobile application built within this project will enable geolocation tracking of the users movements. When they are within a specified distance of a pre-defined point, the user will be prompted to answer a multiple-choice question relative to the given location. Once the user selects an appropriate answer from one of the multiple options presented, the system will then inform the user if they have answered correctly, and consequently display the correct answer. This operation can then be repeated throughout the different locations on the map, where different questions will be asked at different corresponding locations. Subsequently, the answers submitted by the user, accompanied by the user's mobile phone ID, will be submitted to a database created in combination with the mobile application. This allows the performance of both the users and the mobile application to be monitored.

The second part of the project involved developing a supplementary Web Application to work in conjunction with the Mobile Application. A web application was created in order to allow users to generate new questions by selecting locations on a Google map, and appropriately inputting their own individual question with four possible corresponding answers. The user-created questions are stored in a database, and may be requested from the mobile app.

Technical Specifications

The second section of the report describes a brief technical guide to the background and functionality of the software and hardware packages used within this project.

General Overview:
The method for creating the mobile web app infrastructure on to a device was performed by generating a local web app server on the device to be used in response to the application being started on the phone device (Redpath, 2015). The local web app server allows a proxy URL (Uniform Resource Locator) address to implement functionalities associated with the given application. The aforementioned method has been adapted and implemented in this project as this is a common architecture to be running more than one data API at any given time.

This is done by connecting via the Node Server plugin with a file called httpServer.js, in order to connect to both the mobile and web applications using the PhoneGap Serve plugin, and then following to my personal PhoneGap proxy URL: 'http://developer.cege.ucl.ac.uk:31302'. This allowed a secure connection from the files contained with the local disk server, to be accessed and performed within the web and mobile applications. The httpServer.js file allows the connection to the database. Where the PhoneGap server is providing connection to the HTML, JavaScript, and CSS files that are needed for the applications. This is done by using a two-way communication channel between the web page and the mobile application, allowing the parent (web-page) app is able to trigger any JavaScript functions and reach to call native APIs such as the mobile application. Therefore, within this project this methodology is used to allow the storing of questions in to the database, that are then outputted in the mobile application in order to allow user performability.

PhoneGap:
PhoneGap is a mobile application development framework created by Adobe Systems, other names for PhoneGap is Apache Cordova. The application allows users to program the frameworks software in order to create mobile device applications. The framework provides functionality for programming using HTML5, JavaScript, CSS3. PhoneGap is the mobile application used in this project to perform the quiz on.

Yummy:
Yummy FTP Pro is a Final Transfer Protocol (FTP) client designed for functionality with macOS and made by Yummy Software. The Yummy software package was deemed most suitable for the task as it allows quick user interface with macOS, as well as, providing functionality for FTP, SFTP-FTP over an SSH (Secure Shell) connection, FTPS (FTPS over an SSL (Secure Sockets Layer) connection, and WebDAV (Web-based FTP).

Html files:
All html-files can be divided in to a header and body part. The header presents the structure setting for the entire body, as well as, the author and description of the file.
The body of a html file contains both the structure and content of the page, yet without any design elements. The predominant aesthetic functionalities of the web and mobile applications were performed using the Leaflet library within the index.html files for each respective app. A comprehensive list of the functionalities that the web and mobile applications are able to perform are outlaid on their respective user guides.

CSS – files:
CSS (Cascading Style Sheets) files are documents which define the style of a webpage through dynamic web application design (Gardner, 2011). Without a stylesheet implemented, webpages would be displayed very differently.

The main beneficiary of using pre-designed stylesheets from external sources, is that this single file and be used multiple times. As opposed to repeatedly defining settings for each application, each html-file can access the same CSS-file (stylesheet). The stylesheets were adapted.

JavaScript – files:
Javascript is a file extension for a JavaScript source code file format. Javascript is widely used in a lot of Web development applications due to its inherent ability to efficiently allow a user to perform multiple simple and complex functionalities. JS files are predominantly performed to the purpose of running client-side JavaScript code on webpage and mobile applications.

Database

Two tables were created in SQL script to store both the question and answer data created within the web mobile applications. The database software used in this project was PGadmin4. The following code for these tables are as follows:

```
CREATE TABLE public.questions
(
   questionlock character varying(100) COLLATE pg_catalog."default",
   question character varying(100) COLLATE pg_catalog."default",
   answera character varying(100) COLLATE pg_catalog."default",
   answerb character varying(100) COLLATE pg_catalog."default",
   answerc character varying(100) COLLATE pg_catalog."default",
   answerd character varying(100) COLLATE pg_catalog."default",
   rightans character varying(100) COLLATE pg_catalog."default",
   location geometry
)
WITH (
   OIDS = FALSE
)
TABLESPACE teaching2018tablespace;

ALTER TABLE public.questions
   OWNER to user73;
```

```
CREATE TABLE public.answers
(
   question character varying(100) COLLATE pg_catalog."default",
   answer character varying(100) COLLATE pg_catalog."default",
   rightans character varying(100) COLLATE pg_catalog."default"
)
WITH (
   OIDS = FALSE
)
TABLESPACE teaching2018tablespace;

ALTER TABLE public.answers
   OWNER to user73;
```

The web application is able to outsource the information inputted within the web application question form via the httpServer.js file which bridges a connection between the different server APIs. This information stored in the 'questions' table is then able to be retrieved using JavaScript functions performed in the mobile application. After this users in the mobile application are then able to submit data in to the answers table within PGadmin.

Possible improvements

- A marker radius from which the questions could be set (i.e. a radius delineating a two-minute walking radius from the campus), where no questions could be set outside this specified area.
- The map could contain further detail (street name, building name etc)
- Error handling is crucial. The code should check whether all forms are filled or not. Leaving only one form blank leads to an error (change wording). i.e. the app should not allow the user to submit a question in to the database if any section is not filled

References:

Gardner, B.S., 2011. Responsive web design: Enriching the user experience. *Sigma Journal: Inside the Digital Ecosystem*, *11*(1), pp.13-19.

Redpath, R., International Business Machines Corp, 2015. *Mobile web app infrastructure*. U.S. Patent 9,077,770.

Appendixes: Code

```
//Javascript file which defines the data and functions used in the index map box

// Loading tile layers in to seperate variables, to give multiple map layer options
var Light =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy26u500262roa9yk2e7r0/tile
s/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
                                        {maxZoom: 18,
                                        attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
                                        '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
                                        'Imagery © <a href="http://mapbox.com">Mapbox</a>',
                                        id: 'mapbox.streets'}),

          Cream =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy8du3001q2qmzwl3id3f9/tile
s/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
                                        {maxZoom: 18,
                                        attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
                                        '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
                                        'Imagery © <a href="http://mapbox.com">Mapbox</a>',
                                        id: 'mapbox.streets'}),

          Satellite =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxyaclj00002ro8ehqsjf7b/tiles/
256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlwM
zBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
                                        {maxZoom: 18,
                                        attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
                                        '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
                                        'Imagery © <a href="http://mapbox.com">Mapbox</a>',
                                        id: 'mapbox.streets'}),

          Basic =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy6f9400272roa7rk37us5/tiles
/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
                                        {maxZoom: 18,
                                        attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
                                        '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
                                        'Imagery © <a href="http://mapbox.com">Mapbox</a>',
                                        id: 'mapbox.streets'}),

          Dark =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy5x0n000t2rn8sjo97h8k/tiles
/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
                                        {maxZoom: 18,
                                        attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
                                        '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
                                        'Imagery © <a href="http://mapbox.com">Mapbox</a>',
                                        id: 'mapbox.streets'});


//Load default map, zoomed to University College of London (UCL)
var mymap = L.map('mapid', {
   center: [51.5246, -0.1340],
   zoom: 13,
   layers: [Cream]
});

//Create basemap varaibles
var baseMaps = {
   "Default": Cream,
   "Light": Light,
          "Dark": Dark,
          "Satellite": Satellite,
          "Basic": Basic
};
// Add the layers to map
L.control.layers(baseMaps).addTo(mymap);
```

```
// Create draggable marker which is present when opening web page. Coordinates are
found by dragging the marker to desired location.
var marker = L.marker([51.5246, -0.1340], {draggable:true});
marker.addTo(mymap);
marker.on('dragend', function(e) {
                              document.getElementById("lat").value = marker.getLatLng().lat;
                              document.getElementById("lng").value = marker.getLatLng().lng;
                    });



// Function links to Clear Question button in index.html file to reset question input
function resetForm() {
          document.getElementById("location").value = "";
          document.getElementById("question").value = "";
          document.getElementById("answera").value = "";
          document.getElementById("answerb").value = "";
          document.getElementById("answerc").value = "";
          document.getElementById("answerd").value = "";
          document.getElementById("lat").value = "";
          document.getElementById("lng").value = "";
}



/// -- Create geoJSON layer that is return all submitted questions from the database using
the View Questions button in index.html file --///
//Create variables used to create questions layer
// Variable that holds the XMLHttpRequest()
var qvariable;
// Variable that holds the layer of questions
var questionsLayer;
// Global marker variable using AwesomeMarkers to be used as marker for questions layer
var QuestionMarker = L.AwesomeMarkers.icon({
          markerColor: 'purple'
          });

// Get the question data using an XMLHttpRequest
function getQuestions() {
          qvariable = new XMLHttpRequest();
```

```
        qvariable.open('GET','http://developer.cege.ucl.ac.uk:30302/getquestions');
        qvariable.onreadystatechange = questionResponse;
        qvariable.send();
}



// Receive response from the data server
function questionResponse() {
        if (qvariable.readyState == 4) {
                var questionData = qvariable.responseText;
                loadQuestionLayer(questionData);
        }
}



// Convert the received non-text data received in to JSON format and then add layer to map
created from data
function loadQuestionLayer(questionData) {
        // Convert text to JSON
        var questionJSON = JSON.parse(questionData);
        // Load geoJSON layer
        var questionsLayer = L.geoJson(questionJSON,
                {
                // Point to layer creates the question points
                pointToLayer: function (feature, latlng)
                {
                        // Includes a pop-up describing the location description (location) and
question name (question)
                        return L.marker(latlng,
{icon:QuestionMarker}).bindPopup("<b>"+feature.properties.questionlock +"</b>" + "<p>"
+ feature.properties.question + "</b>");
                        },
                }).addTo(mymap);
        // The map is then fitted to the extent of the questions layer
        mymap.fitBounds(questionsLayer.getBounds());
}



// JavaScript

//Adapted from: https://github.com/claireellul/cegeg077-week6formcode

//Create function to ensure all question boxes are filled
function validateQuestion() {
```

```
// define question boxes
	var qlocal=document.getElementById("questionlock").value;
	var qu=document.getElementById("question").value;
	var ansA=document.getElementById("answera").value;
	var ansB=document.getElementById("answerb").value;
	var ansC=document.getElementById("answerc").value;
	var ansD=document.getElementById("answerd").value;

	if (qlocal==null || qlocal=="",qu==null || qu=="",ansA==null || ansA=="",ansB==null
|| ansB=="",ansC==null || ansC=="",ansD==null || ansD=="")
      {
         alert("Please populate all fields in question form"); //alert when field not filled
         return false;
      }
      else
// if all boxes are filled begin data upload process
      {
          DataStartUp()
      }
}
```

```
//Create variables for questions submission in to the database. These directly relate to the
fields in the Web App that submit a question.
function DataStartUp() {
	alert ("Question submitted to database!"); //return alert when question is
submitted, linked with the Submit button in the index.html file


	var lat = document.getElementById("lat").value;
	var lng = document.getElementById("lng").value;
	var questionlock = document.getElementById("questionlock").value;

	var question = document.getElementById("question").value;
	var answera = document.getElementById("answera").value;
	var answerb = document.getElementById("answerb").value;
	var answerc = document.getElementById("answerc").value;
	var answerd = document.getElementById("answerd").value;
//Create string to query and ensure one of the radio button is selected
	var postString = "questionlock="+questionlock +"&question="+question
+"&answera="+answera +"&answerb="+answerb +"&answerc="+answerc+
"&answerd="+answerd;
	if (document.getElementById("Opt1").checked) {
      postString=postString+"&rightans=1";
   }
   if (document.getElementById("Opt2").checked) {
	postString=postString+"&rightans=2";
   }
	if (document.getElementById("Opt3").checked) {
		postString=postString+"&rightans=3";
	}
	if (document.getElementById("Opt4").checked) {
		postString=postString+"&rightans=4";
	}
	postString = postString + "&lat=" + lat + "&lng=" + lng; //string latitude and longitude
	processData(postString); //processdata
}




// Create variable to retrieve the data uploaded to the database using the string above
var retrieveQ; //variable to hold XMLHttpRequest()
function processData(postString) {
  retrieveQ = new XMLHttpRequest();
  retrieveQ.open('POST','http://developer.cege.ucl.ac.uk:30302/SubmitData',true);
  retrieveQ.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
  retrieveQ.onreadystatechange = UpData; //uploaded data
  retrieveQ.send(postString);
}


// Wait for response from the data server and then process once received
function UpData() {
  if (retrieveQ.readyState == 4) {
   document.getElementById("dataUploadResult").innerHTML = retrieveQ.responseText;
   }
}


<!doctype html>

<!--  Html file contatining visual layout and features for the website app -->
```

```
<!--  Adapted from: https://github.com/claireellul/cegeg077-
week5app/blob/master/ucfscde/www/index.html -->

<!--
  Material Design Lite
  Copyright 2015 Google Inc. All rights reserved.
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at
      https://www.apache.org/licenses/LICENSE-2.0
  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License
-->
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, minimum-
scale=1.0">
    <title>UCL Question App - SWQW1</title> <!-- set page title-->



    <link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Roboto:regular,bold,italic,thin,light,bolditali
c,black,medium&amp;lang=en">
    <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
    <link rel="stylesheet" href="https://code.getmdl.io/1.3.0/material.blue_grey-
cyan.min.css" />
    <link rel="stylesheet" href="styles.css">

		<!--https://leafletjs.com/examples/quick-start/ -->
		<!-- the following links add the CSS and Javascript required for the Leaflet Map -->
		<link rel="stylesheet" href="https://unpkg.com/leaflet@1.1.0/dist/leaflet.css"
		integrity="sha512-
wcw6ts8Anuw10Mzh9Ytw4pylW8+NAD4ch3lqm9lzAsTxg0GFeJgoAtxuCLREZSC5lUXdVyo/7yf
sqFjQ4S+aKw=="crossorigin=""/>
		<script src="https://unpkg.com/leaflet@1.1.0/dist/leaflet.js" integrity="sha512-
mNqn2Wg7tSToJhvHcqfzLMU6J4mkOImSPTxVZAdo+lcPlk+GhZmYgACEe0x35K7YzW1zJ7XyJ
V/TT1MrdXvMcA=="   crossorigin=""></script>

		<!-- the following links add the CSS and Javascript required for the custom icons -->
		<link rel ="stylesheet" href="ionicons.min.css">
```

```
        <link rel="stylesheet" href="leaflet.awesome-markers.css">
        <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.0.12/css/all.css" integrity="sha384-
G0fIWCsCzJIMAVNQPfjH08cyYaUtMwjJwqiRKxxE/rx96Uroj1BtIQ6MLJuheaO9"
        crossorigin="anonymous">
        <script src="js/leaflet.awesome-markers.js"></script>
  </head>








  <body>
    <div class="demo-layout mdl-layout mdl-js-layout mdl-layout--fixed-drawer mdl-layout--
fixed-header">
      <header class="demo-header mdl-layout__header mdl-color--cyan-A100 mdl-color-text--
purple-400">
        <div class="mdl-layout__header-row">
          <span class="mdl-layout-title">UCL Trivia Quiz - Question Generator</span>
          <div class="mdl-layout-spacer"></div>

                        <!-- line below inserts github icon from fontawesome.com and modifies it
using the id to link to custom spec in leaflet.awesome_markers.css stylesheet -->
                    <button class="mdl-button mdl-js-button mdl-js-ripple-effect mdl-button--
icon" id="hdrbtn"><i class="fab fa-github" id="git"></i>
                    </button>
                    <ul class="mdl-menu mdl-js-menu mdl-color--purple-400 mdl-color-text--cyan-
A100 mdl-js-ripple-effect mdl-menu--bottom-right" for="hdrbtn">
                                        <li class="mdl-menu__item"><a
href="https://github.com/JackPhilpott">Repository</a></li>
                                        <li class="mdl-menu__item"><a href="#"
onclick="window.open('QuestionsUserGuide.html', '_system');">User Guide</a></li>
                                        <li class="mdl-menu__item"><a href="#"
onclick="window.open('TechnicalGuide.txt', '_system');">Technical Guide</a></li>
                    </ul>
                  </div>
      </header>


      <div class="demo-drawer mdl-layout__drawer mdl-color--cyan-A100 mdl-color-text--
purple-400">
        <header class="demo-drawer-header">
          <img src="img/LogoMakr_4LA1cL.png" class="demo-avatar" alt="UCL Trivia Quiz logo"
width="250" height="250">
```

```
        </header>
                        <br></br>
                        <br></br>
        <nav class="demo-navigation mdl-navigation mdl-color--cyan-A100 mdl-color-text--
purple-400">
          <a class="mdl-navigation__link" href=""><i class="mdl-color-text--purple-400 material-
icons" role="presentation">home</i>Home</a>
          <a class="mdl-navigation__link" href="" onclick='trackLocation();return false;'><i
class="mdl-color-text--purple-400 material-icons" role="presentation">local_offer</i>Track
                                Location (Mobile only)</a>
          <a class="mdl-navigation__link" id="show" href="" onclick='getQuestions();return
false;'><i class="mdl-color-text--purple-400 material-icons"
role="presentation">forum</i>View                       Questions</a>
          <a class="mdl-navigation__link" id="clear" href="" onclick='resetForm();return
false;'><i class="mdl-color-text--purple-400 material-icons"
role="presentation">delete</i>Clear                             Question</a>
          <div class="mdl-layout-spacer"></div>
        </nav>
      </div>




      <main class="mdl-layout__content mdl-color--grey-100">
    <div class="demo-charts mdl-color--grey-50 mdl-cell mdl-cell--12-col mdl-grid" >

                        <!-- the mapid div holds the map -->
                        <div id="mapid" style="width: 850px; height: 750px; float: left;"></div>
        <div style="width: 20px; height: 800px; float: right;">  </div>









<!-- Create instructional text box -->
                                <div>
                                        <!-- Create boxes for latitude ad logitude values -->
                            <em><font color="purple">Drag <i class="fas fa-map-marker-alt"></i>
icon on map to input coordinates</font color="purple"></em><br>
                        <label for="lat"><b>Latitude:</b></label><br><input type="text" size="32"
id="lat"/><br/>
                        <label for="lng"><b>Longitude:</b></label><br><input type="text"
size="32" id="lng"/><br/>
```

```
                        <label for="questionlock"><b>Location Name:</b></label><br><input
type="text" size="32" id="questionlock" /><br/>
                                                       <br/>




<!-- Create the labels and answer boxes for the question and answers -->
                                                       <br></br>
                        <em><font color="purple">Enter a question</font color="purple"></em>
                                                       <br>
            <label for="question"><b>Question: </b></label><br><input type="text" size="32"
id="question" /><br/>
            <label for="answera"><b>A:</b></label><br><input type="text" size="32"
id="answera" /><br/>
            <label for="answerb"><b>B:</b></label><br><input type="text" size="32"
id="answerb" /><br/>
            <label for="answerc"><b>C:</b></label><br><input type="text" size="32"
id="answerc" /><br/>
            <label for="answerd"><b>D:</b></label><br><input type="text" size="32"
id="answerd" /><br/>
          <br/>

<!-- Create the radio selector to define the correct answer -->
                                 <br></br>
            <b>Which is the correct answer?</b>
            <br>
      1: <input type="radio" name="answer" id = Opt1 value="1" checked="yes" /><br />
      2: <input type="radio" name="answer" id = Opt2 value="2"/><br />
      3: <input type="radio" name="answer" id = Opt3 value="3"/><br />
      4: <input type="radio" name="answer" id = Opt4 value="4"/><br />
              <br/>
                       <button class="mdl-button mdl-button--raised mdl-button--colored"
id="startUpload" onclick="validateQuestion();return false;"><i class="mdl-color-text--
purple-400                                     material-icons" role="presentation">flag</i>Submit
Question</button>




       </div>
                       <!-- Call javascript files and API data into the index -->
     </main>
     <script src="https://code.getmdl.io/1.3.0/material.min.js"></script>
           <script src="js/appActivity.js"></script>
           <script src="js/uploadData.js"></script>
           </body>
</html>
```

```
/*
Author: L. Voogdt
License: MIT
Version: 1.0
*/

/* Create custom variables for the github icon on the Web page*/
#git {
	color: #9C27B0;
	font-size: 2.5em;
}
#hdrbtn{
	width: 80px;
	height: 80px;
	min-width: initial;
}

/* Marker setup */
.awesome-marker {
  background: url('images/markers-soft.png') no-repeat 0 0;
  width: 35px;
  height: 46px;
  position:absolute;
  left:0;
  top:0;
  display: block;
  text-align: center;
}

.awesome-marker-shadow {
  background: url('images/markers-shadow.png') no-repeat 0 0;
  width: 36px;
  height: 16px;
}

/* Retina displays */
@media (min--moz-device-pixel-ratio: 1.5),(-o-min-device-pixel-ratio: 3/2),
(-webkit-min-device-pixel-ratio: 1.5),(min-device-pixel-ratio: 1.5),(min-resolution: 1.5dppx) {
 .awesome-marker {
  background-image: url('images/markers-soft@2x.png');
  background-size: 720px 46px;
 }
 .awesome-marker-shadow {
  background-image: url('images/markers-shadow@2x.png');
  background-size: 35px 16px;
 }
}
```

```
.awesome-marker i {
  color: #333;
  margin-top: 10px;
  display: inline-block;
  font-size: 14px;
}

.awesome-marker .icon-white {
  color: #fff;
}

/* Colors */
.awesome-marker-icon-red {
  background-position: 0 0;
}

.awesome-marker-icon-darkred {
  background-position: -180px 0;
}

.awesome-marker-icon-lightred {
  background-position: -360px 0;
}

.awesome-marker-icon-orange {
  background-position: -36px 0;
}

.awesome-marker-icon-beige {
  background-position: -396px 0;
}

.awesome-marker-icon-green {
  background-position: -72px 0;
}

.awesome-marker-icon-darkgreen {
  background-position: -252px 0;
}

.awesome-marker-icon-lightgreen {
  background-position: -432px 0;
}

.awesome-marker-icon-blue {
  background-position: -108px 0;
}

.awesome-marker-icon-darkblue {
  background-position: -216px 0;
}

.awesome-marker-icon-lightblue {
  background-position: -468px 0;
}

.awesome-marker-icon-purple {
  background-position: -144px 0;
}

.awesome-marker-icon-darkpurple {
  background-position: -288px 0;
}

.awesome-marker-icon-pink {
  background-position: -504px 0;
}

.awesome-marker-icon-cadetblue {
  background-position: -324px 0;
}

.awesome-marker-icon-white {
  background-position: -574px 0;
}

.awesome-marker-icon-gray {
  background-position: -648px 0;
}

.awesome-marker-icon-lightgray {
  background-position: -612px 0;
}

.awesome-marker-icon-black {
  background-position: -682px 0;
}


/*
  Leaflet.AwesomeMarkers, a plugin that adds colorful iconic markers for Leaflet, based on
the Font Awesome icons
  (c) 2012-2013, Lennard Voogdt
  http://leafletjs.com
  https://github.com/lvoogdt
*/

/*global L*/

(function (window, document, undefined) {
    "use strict";
    /*
     * Leaflet.AwesomeMarkers assumes that you have already included the Leaflet library.
     */

    L.AwesomeMarkers = {};

    L.AwesomeMarkers.version = '2.0.1';

    L.AwesomeMarkers.Icon = L.Icon.extend({
        options: {
            iconSize: [35, 45],
            iconAnchor:   [17, 42],
            popupAnchor: [1, -32],
            shadowAnchor: [10, 12],
            shadowSize: [36, 16],
            className: 'awesome-marker',
            prefix: 'glyphicon',
            spinClass: 'fa-spin',
            extraClasses: '',
            icon: 'home',
            markerColor: 'blue',
            iconColor: 'white'
        },

        initialize: function (options) {
            options = L.Util.setOptions(this, options);
        },

        createIcon: function () {
            var div = document.createElement('div'),
                options = this.options;

            if (options.icon) {
                div.innerHTML = this._createInner();
            }

            if (options.bgPos) {
                div.style.backgroundPosition =
                    (-options.bgPos.x) + 'px ' + (-options.bgPos.y) + 'px';
            }

            this._setIconStyles(div, 'icon-' + options.markerColor);
            return div;
        },

        _createInner: function() {
            var iconClass, iconSpinClass = "", iconColorClass = "", iconColorStyle = "", options =
this.options;

            if(options.icon.slice(0,options.prefix.length+1) === options.prefix + "-") {
                iconClass = options.icon;
            } else {
                iconClass = options.prefix + "-" + options.icon;
            }

            if(options.spin && typeof options.spinClass === "string") {
                iconSpinClass = options.spinClass;
            }

            if(options.iconColor) {
                if(options.iconColor === 'white' || options.iconColor === 'black') {
                    iconColorClass = "icon-" + options.iconColor;
                } else {
                    iconColorStyle = "style='color: " + options.iconColor + "' ";
                }
            }

            return "<i " + iconColorStyle + "class='" + options.extraClasses + " " + options.prefix +
" " + iconClass + " " + iconSpinClass + " " + iconColorClass + "'></i>";
        },

        _setIconStyles: function (img, name) {
            var options = this.options,
                size = L.point(options[name === 'shadow' ? 'shadowSize' : 'iconSize']),
                anchor;

            if (name === 'shadow') {
                anchor = L.point(options.shadowAnchor || options.iconAnchor);
            } else {
                anchor = L.point(options.iconAnchor);
            }

            if (!anchor && size) {
                anchor = size.divideBy(2, true);
            }
```

```
            img.className = 'awesome-marker-' + name + ' ' + options.className;

            if (anchor) {
                img.style.marginLeft = (-anchor.x) + 'px';
                img.style.marginTop  = (-anchor.y) + 'px';
            }

            if (size) {
                img.style.width  = size.x + 'px';
                img.style.height = size.y + 'px';
            }
        },

        createShadow: function () {
            var div = document.createElement('div');

            this._setIconStyles(div, 'shadow');
            return div;
      }
    });

    L.AwesomeMarkers.icon = function (options) {
        return new L.AwesomeMarkers.Icon(options);
    };

}(this, document));
```

```
<p> UCL Trivia Quiz Technical Guide</p>
<p>The UCL Trivia Quiz contains three repositories, ecnompassing one server script and two phonegap apps.</p>

<p>* The mobile quiz app is located at : https://github.com/JackPhilpott/quiz</p>
<p>* The server is found is located at : https://github.com/JackPhilpott/server</p>
<p>* The question creation app is located at : https://github.com/JackPhilpott/questions</p>


<p> Server</p>
<p>The httpServer.js file must be running in order for each app to use their full fucntionality:</p>

<p> *'git clone https://github.com/JackPhilpott/server.git' allows a user to download the repository to their local files</p>
<p>* In local files navigate to repository using 'cd server'</p>
<p>* In order to rude the httpServer.js file, insert 'node httpServer.js &'</p>
```

```
<p>When the server connection has been established, the user can then connect to either
of the phonegap apps.</p>

<p> Create the Question Map</p>

<p>* Use 'git clone https://github.com/JackPhilpott/questions.git' to download the
repository.</p>
<p>* To get to the questions repository, insert 'cd questions/ucesjlp'</p>
<p>* Run quiz app using phonegap serve</p>

<p> Use the Quiz App </p>

<p>* Use 'git clone https://github.com/JackPhilpott/quiz.git' to download the repository.
</p>
<p>* To get to the questions repository, insert 'cd quiz/ucesjlp'</p>
<p>* Run quiz app using phonegap serve</p>

<p> Troubleshooting </p>

<p>* Currently neither of the apps are compatible with windows or apple phones .</p>
<p>* Depedent on the age of model of android phone in use, bugs may be experienced. This
is particularly noted in older generations.</p>
<p>* Both the quiz app and question map are designed to be fully optimised functionally in
their respective designed views. i.e. the quiz mobile app provides the best user experience
when ran on a phone, however when viewed on a web page it reduces in user experience
and functionality. Same is applied to the webpage app respectively. </p>

<!doctype html>
<html>
<html lang="en">

        <head>
                <title> Question Form User Guide </title>
                <h1> Question Form User Guide </h1>
        </head>
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.0.12/css/all.css"
integrity="sha384-
G0fIWCsCzJIMAVNQPfjH08cyYaUtMwjJwqiRKxxE/rx96Uroj1BtIQ6MLJuheaO9"
        crossorigin="anonymous">


<body>

<p> <b> Welcome! <b> This web application is the first step in enabling you to build your
very own mobile application. </p>
```

```
<p> In order to do this, you can use this application to create user-specified questions at
desired locations which will appear on the mobile app in their respective locations. </p>
<br></br>



<h2> <b> Key information: <b></h2>
<p> - Please ensure the httpServer.js is running. </p>
<p> - Please ensure all fields within the question form are populated. </p>
<p> - Please find more basemap layer selections available by selecting the layers button at
the top right of the map</p>
<p> - When selecting the 'View Questions' button, ensure the blue location marker has been
changed from previously submitted question location otherwise you will not be able to
select the marker to drag it further. Additionally, selecting the 'Home' button will reset the
map and therefore remove the 'View Questions' layer. </p>
<br></br>
<br></br>



<h2> <b> Generating a new question: <b></h2>
<p> 1: Firstly, the location of the question can be selected by clicking and dragging the blue
marker in the map to a desired location. </p>
<p> 2: Once location is selected, release the marker and you will see the coordinates in the
latitude and lontidue fields to the right of the map have been automatically filled with the
coordinates corresponding to the marker position. </p>
<p> 3: You can now populate the remaining fields within the form to the right of the map
(i.e. Questions, and Answers etc.) </p>
<p> 4: That question may now be submitted using the "Submit" button below.  </p>
<p>                 - Please ensure all fields in the form and populated, otherwise the question
will not be allowed to submit. </p>
<p> 5: Once the question has been submitted, you can now select the "View Questions"
button to the left of the map, creating a layer of all questions successfully submitted to the
database. </p>
<p> 6: You can now use the "Clear Question" button on the left of the map to reset the
question form and restart the process to submitting another question. </p>
</body>
</html>








<!doctype html>
<html>
<html lang="en">
```

```
        <head>
                <title> Mobile App User Guide </title>
                <h1> Mobile App User Guide </h1>
        </head>
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.0.12/css/all.css"
integrity="sha384-
G0fIWCsCzJIMAVNQPfjH08cyYaUtMwjJwqiRKxxE/rx96Uroj1BtIQ6MLJuheaO9"
        crossorigin="anonymous">


<body>
<p> <b> Welcome! <b> This is a mobile application that is made to enable you to participate
in a geolocation enabled interactive quiz. </p>
<p> <b> Enjoy!! <b> </p>
<br></br>



<h2> <b> Key information: <b></h2>
<p> - Please ensure the httpServer.js is running. </p>
<p> - Please ensure all fields within the answer submit are populated. </p>
<p> - Please find more basemap layer selections available by selecting the layers button at
the top right of the map</p>
<br></br>
<br></br>



<h2> <b> Generating a new question: <b></h2>
<p> 1: Firstly, select the <i class="mdl-color-text--purple-400 material-icons"
role="presentation">flag</i>. Click 'ok' on the alert and you will see your location. </p>
<p> 2: Next select the <i class="mdl-color-text--purple-400 material-icons"
role="presentation">input_antenna</i>. This inserts an interactive layer of questions. </p>
<p> 3: Now the user moves to a location where the question marker is within the blue
radius circle. </p>
<p>     4: Now select <i class="mdl-color-text--purple-400 material-icons"
role="presentation">tap_and_play</i>. If the marker returns orange then the marker is
within the radius, if not the marker remains pink</p>
<p> 5: Select the orange marker and a answer form will appear. Fill in answer form and click
submit.  </p>
<p>             - Please ensure all fields in the form and populated, otherwise the answer
will not be allowed to submit. </p>
<p> 6: Once the answer has been submitted, an alert will indicate if you answered right or
wrong. If right then the respective question marker is returned green, if wrong then
returned red </p>
<p> 7: You can now move on to the next question by following the map to repeat the
process above and complete the quiz </p>
```

```
<p> <b> Good luck!! <b> </p>
</body>
</html>

// Code adapted from: https://github.com/claireellul/cegeg077-
week5server/blob/master/httpServer.js


// Create global variables
var express = require('express'); //server forms a part of nodejs program
var path = require("path");
var app = express();
var bodyParser = require('body-parser');

app.use(bodyParser.urlencoded({
  extended: true
}));
app.use(bodyParser.json());


// Additional functionality when PhoneGap has a server in use, supporting cross-domain
queries
app.use(function(req, res, next) {
	res.setHeader("Access-Control-Allow-Origin", "*");
	res.setHeader("Access-Control-Allow-Headers", "X-Requested-With");
	res.setHeader('Access-Control-Allow-Methods', 'GET,PUT,POST,DELETE');
	next();
});



///------ SUBMITTING and RETRIEVING question answers -----///
// Upload the submitted form data as inputted from user in web application
app.post('/SubmitData',function(req,res){
    console.dir(req.body);
    pool.connect(function(err,retrieveQ,done) {
        if(err){
        console.log("no connection available"+ err);
        res.status(400).send(err);
        }

// Create string for the geometry components latitude and longitude to bring together as
one variable
var geometrystring = "st_geomfromtext('POINT(" + req.body.lng + " " + req.body.lat + ")'";

//Create string for other table components and their corresponding answers to be used in
upload Answer
var querystring = "INSERT into questions
(questionlock,question,answera,answerb,answerc,answerd,rightans,location) values ('";
querystring = querystring + req.body.questionlock + "','" + req.body.question + "','" +
req.body.answera+"','" + req.body.answerb+"','" + req.body.answerc+"','" +
req.body.answerd+"','" + req.body.rightans+"'," + geometrystring +"))";
                    console.log(querystring);
                    retrieveQ.query( querystring,function(err,result) {
                              done();
                              if(err){
                                        console.log(err);
                                        res.status(400).send(err);
                              }
                              res.status(200).send("Question submitted to database");
                    });
          });
});
```

```
// Upload the stored answers from the database to be retrieved in the mobile application
app.post('/UpAns',function(req,res){ //UpAns = Uploaded Answers
     console.dir(req.body);
     pool.connect(function(err,retrieveQ,done) {
          if(err){
          console.log("no connection available"+ err);
          res.status(400).send(err);
          }
var querystring = "INSERT into answers (question,answer,rightans) values ('";
querystring = querystring + req.body.question + "','" + req.body.answer +"','" +
req.body.AnsRight+"')";
                    console.log(querystring);
      retrieveQ.query( querystring,function(err,result) {
                              done();
                              if(err){
           console.log(err);
           res.status(400).send(err);
        }
        res.status(200).send("Answer uploaded!");
     });
          });
});
```

```
// The QuesGet function allows both web and mobile applications to retrieve the question
data (question location etc.) from the database
app.get('/getQuestions', function (req,res) { //QuesGet
   pool.connect(function(err,retrieveQ,done) {
     if(err){
       console.log("no connection available"+ err);
       res.status(400).send(err);
     }
      // The geoJSON functionality inbuilt within system is used transform location data in
the geometry column of the database table
      /* The desired geoJSON format was created using a query adapted from:
                    http://www.postgresonline.com/journal/archives/267-Creating-GeoJSON-
Feature-Collections-with-JSON-and-PostGIS-functions.html, accessed 2nd May 2018*/

          var querystring = " SELECT 'FeatureCollection' As type, array_to_json(array_agg(f)) As
features  FROM ";
          querystring = querystring + "(SELECT 'Feature' As type     ,
ST_AsGeoJSON(lg.location)::json As geometry, ";
          querystring = querystring + "row_to_json((SELECT l FROM (SELECT questionlock,
question, answera, answerb, answerc, answerd, rightans) As l      )) As properties";
          querystring = querystring + "   FROM questions  As lg limit 100  ) As f ";
          console.log(querystring);
          retrieveQ.query(querystring,function(err,result){

                              done(); //release client back to the pool
       if(err){
         console.log(err);
         res.status(400).send(err);
       }
       res.status(200).send(result.rows);
     });
   });
});

<doctype html>

 <!--  Html file contatining visual layout and features for the phone quiz app -->


 <!--  Adapted from: https://github.com/claireellul/cegeg077-
week5app/blob/master/ucfscde/www/index.html -->

<!--
 Material Design Lite
 Copyright 2015 Google Inc. All rights reserved.

 Licensed under the Apache License, Version 2.0 (the "License");
```

```
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

      https://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License
-->
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, minimum-
scale=1.0">
    <title>UCL Trivia Quiz - SWQW1</title> <!-- set page title-->



    <!-- Add to homescreen for Chrome on Android -->
    <meta name="mobile-web-app-capable" content="yes">
    <link rel="icon" sizes="192x192" href="images/android-desktop.png">

    <!-- Add to homescreen for Safari on iOS -->
    <meta name="apple-mobile-web-app-capable" content="yes">
    <meta name="apple-mobile-web-app-status-bar-style" content="black">
    <meta name="apple-mobile-web-app-title" content="Material Design Lite">
    <link rel="apple-touch-icon-precomposed" href="images/ios-desktop.png">

    <link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Roboto:regular,bold,italic,thin,light,bolditali
c,black,medium&amp;lang=en">
    <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">
    <link rel="stylesheet" href="https://code.getmdl.io/1.3.0/material.blue_grey-
cyan.min.css" />
    <link rel="stylesheet" href="styles.css">

            <!--https://leafletjs.com/examples/quick-start/ -->
            <!-- the following links add the CSS and Javascript required for the Leaflet Map -->
            <link rel="stylesheet" href="https://unpkg.com/leaflet@1.1.0/dist/leaflet.css"
            integrity="sha512-
wcw6ts8Anuw10Mzh9Ytw4pylW8+NAD4ch3lqm9lzAsTxg0GFeJgoAtxuCLREZSC5lUXdVyo/7yf
sqFjQ4S+aKw=="crossorigin=""/>
```

```
	<script src="https://unpkg.com/leaflet@1.1.0/dist/leaflet.js" integrity="sha512-
mNqn2Wg7tSToJhvHcqfzLMU6J4mkOImSPTxVZAdo+lcPlk+GhZmYgACEe0x35K7YzW1zJ7XyJ
V/TT1MrdXvMcA=="   crossorigin=""></script>

	<!-- the following links add the CSS and Javascript required for the custom icons -->
	<link rel ="stylesheet" href="ionicons.min.css">
	<link rel="stylesheet" href="leaflet.awesome-markers.css">
	<link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.0.12/css/all.css" integrity="sha384-
G0fIWCsCzJIMAVNQPfjH08cyYaUtMwjJwqiRKxxE/rx96Uroj1BtIQ6MLJuheaO9"
	crossorigin="anonymous">
	<script src="js/leaflet.awesome-markers.js"></script>
  </head>


  <body>
  <!-- Set top bar and title -->
    <div class="demo-layout mdl-layout mdl-js-layout mdl-layout--fixed-drawer mdl-layout--
fixed-header">
      <header class="demo-header mdl-layout__header mdl-color--cyan-A100 mdl-color-text--
purple-400">
        <div class="mdl-layout__header-row">

				<button class="mdl-button mdl-js-button mdl-js-ripple-effect mdl-
button--icon" id="b1" href=""onclick='trackLocation();return false;'><i class="mdl-color-text-
-purple-400							material-icons"
role="presentation">flag</i></button>

			<button class="mdl-button mdl-js-button mdl-js-ripple-effect mdl-button--
icon" id="b1" href=""onclick='getQuestions();return false;'><i class="mdl-color-text--purple-
400							material-icons"
role="presentation">input_antenna</i></button>

		<button class="mdl-button mdl-js-button mdl-js-ripple-effect mdl-button--icon"
id="b1" href=""onclick='availableQuestions();return false;'><i
			class="mdl-color-text--purple-400 material-icons"
role="presentation">tap_and_play</i></button>


				<div class="mdl-layout-spacer"></div>
					<span class="mdl-layout-title">UCL Trivia Quiz</span>
			<div class="mdl-layout-spacer"></div>


				<button class="mdl-button mdl-js-button mdl-js-ripple-effect mdl-
button--icon" id="hdrbtn"><i class="fab fa-github" id="git"></i></button>
```

```
                    <ul class="mdl-menu mdl-js-menu mdl-color--grey-700 mdl-js-ripple-effect
mdl-menu--bottom-right" for="hdrbtn">  <!-- Set the settings menu specification -->
                                        <li class="mdl-menu__item"><a
href="https://github.com/JackPhilpott">Repository</a></li>
                                        <li class="mdl-menu__item"><a href="#"
onclick="window.open('QuizUserGuide.html', '_system');">User Guide</a></li>
                                        <li class="mdl-menu__item"><a href="#"
onclick="window.open('TechnicalGuide.txt', '_system');">Technical Guide</a></li>
                  </ul>
                </div>
    </header>

<!-- Set map box size -->
<main class="mdl-layout__content mdl-color--grey-100">
          <div class="demo-charts mdl-color--white mdl-shadow--2dp mdl-cell mdl-cell--12-col
mdl-grid">
                    <div id="mapid" style="width: 100%; height: 100%;"></div>

   <!-- Set the active questions page layout -->
                    <div id="PopQuest" style ="display:none">
                              <h3 id = "questionHeader"> Choose an answer </h3>

                                        <textarea readonly id = "question" style="border: none;
width:100%" ></textarea>

                    <textarea readonly id = "answera"></textarea>
                     <input type="radio" name="answer" id = Opt1 value="1" checked="yes"
/><br />

           <textarea readonly id = "answerb"></textarea>
           <input type="radio" name="answer" id = Opt2 value="2"/><br />

           <textarea readonly id = "answerc"></textarea>
           <input type="radio" name="answer" id = Opt3 value="3"/><br />

           <textarea readonly id = "answerd"></textarea>
           <input type="radio" name="answer" id = Opt4 value="4"/><br />

           <button id="SubmitClick" onclick ="validateData()">Submit</button></div>
           </div>
          </main>
 <!-- Call javascript files and API data into the index -->
<script src="https://code.getmdl.io/1.3.0/material.min.js"></script>
<script src="js/appActivity.js"></script>
</body>
</html>
```

```
// Code adapted from: https://github.com/claireellul/cegeg077-
week5app/blob/master/ucfscde/www/js/appActivity.js

// Loading tile layers in to seperate variables, to give multiple map layer options
var Light =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy26u500262roa9yk2e7r0/tile
s/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
				{maxZoom: 18,
				attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
				'<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
				'Imagery © <a href="http://mapbox.com">Mapbox</a>',
				id: 'mapbox.streets'}),

	Cream =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy8du3001q2qmzwl3id3f9/tile
s/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
				{maxZoom: 18,
				attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
				'<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
				'Imagery © <a href="http://mapbox.com">Mapbox</a>',
				id: 'mapbox.streets'}),

	Satellite =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxyaclj00002ro8ehqsjf7b/tiles/
256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlwM
zBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
				{maxZoom: 18,
				attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
				'<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
				'Imagery © <a href="http://mapbox.com">Mapbox</a>',
				id: 'mapbox.streets'}),

	Basic =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy6f9400272roa7rk37us5/tiles
/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
				{maxZoom: 18,
				attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
                                '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
                                'Imagery © <a href="http://mapbox.com">Mapbox</a>',
                                id: 'mapbox.streets'}),

        Dark =
L.tileLayer('https://api.mapbox.com/styles/v1/puffer1210/cjgxy5x0n000t2rn8sjo97h8k/tiles
/256/{z}/{x}/{y}?access_token=pk.eyJ1IjoicHVmZmVyMTIxMCIsImEiOiJjamd4eHlrN3IyYmlw
MzBwMW05enRiejA0In0.HISDyJ-JxCB8SZSIsZdpig',
                                {maxZoom: 18,
                                attribution: 'Map data &copy; <a
href="http://openstreetmap.org">OpenStreetMap</a> contributors, ' +
                                '<a href="http://creativecommons.org/licenses/by-
sa/2.0/">CC-BY-SA</a>,' +
                                'Imagery © <a href="http://mapbox.com">Mapbox</a>',
                                id: 'mapbox.streets'});


//Load default map, zoomed to University College of London (UCL)
var mymap = L.map('mapid', {
  center: [51.5246, -0.1340],
  zoom: 13,
  layers: [Cream]
});

//Create basemap varaibles
var baseMaps = {
  "Default": Cream,
  "Light": Light,
        "Dark": Dark,
        "Satellite": Satellite,
        "Basic": Basic
};
// Add the layers to map
L.control.layers(baseMaps).addTo(mymap);




/* Code used to form tracker functionality for user position adapted from:
https://stackoverflow.com/questions/48651799/how-do-i-simply-get-the-current-
geolocation-using-leaflet-without-events?noredirect=1&lq=1
https://github.com/domoritz/leaflet-locatecontrol
https://leafletjs.com/examples/mobile/
https://stackoverflow.com/questions/10563789/how-to-locate-user-with-leaflet-locate */
// Track the location of the user
// If the phone is unable to find the users locations then an alert message is returned
expressing this. Error handling.
var TrackUser = true;
var LocalUser;
var LocRad;
var autoPan = false;

function trackLocation() {
        if (!TrackUser){
                mymap.fitBounds(LocalUser.getLatLng().toBounds(250));
                autoPan = true;
        } else {
                if (navigator.geolocation) {
                alert("Finding location!");
                navigator.geolocation.watchPosition(showPosition);
        } else {
                alert("Geolocation is not supported within this browser."); //Error handing
                }
        }
}




// Create variable to hold XMLHttpRequest()
var VarQuest;
function processData(postString) {
  VarQuest = new XMLHttpRequest();
  VarQuest.open('POST','http://developer.cege.ucl.ac.uk:30302/UpAns',true); //connect to
uploaded answers
  VarQuest.setRequestHeader("Content-type", "application/x-www-form-urlencoded");
  VarQuest.onreadystatechange = answerUploaded;
  VarQuest.send(postString);
}



// Receive response from the data server
function answerUploaded() {
        if (VarQuest.readyState == 4) {
                document.getElementById('PopQuest').style.display = 'none';
        document.getElementById('mapid').style.display = 'block';
                if (answerTrue) {PTClick.setIcon(CorrectQTM);
                } else {PTClick.setIcon(WrongQTM);
                }
   }
}
```

```
// Create variables for the XMLHttpRequest() and layer of questions
PTQuest = [];
var qvariable;
var questionsLayer;

// Create function to use an XMLHttpRequest to retrieve question data
function getQuestions() {
        qvariable = new XMLHttpRequest();
        qvariable.open('GET','http://developer.cege.ucl.ac.uk:30302/getQuestions');
        qvariable.onreadystatechange = questionResponse;
        qvariable.send();
}

// Get response from data server
function questionResponse() {
        if (qvariable.readyState == 4) {
                var questionData = qvariable.responseText;
                loadQuestionLayer(questionData);
        }
}

// Convert the received data to JSON format and add it to the map
function loadQuestionLayer(questionData) {
        var questionJSON = JSON.parse(questionData);        // Convert text to JSON
        var questionsLayer = L.geoJson(questionJSON,        // Load geoJSON layer
        {
        // Point to layer creates the question points
        pointToLayer: function (feature, latlng)
        {
                layer_marker = L.marker(latlng, {icon:QuestionsTM})                    //Every
question for database has pink marker
                layer_marker.bindPopup("<b>"+feature.properties.questionlock +"</b>");
                //Include popup with location description

                PTQuest.push(layer_marker);                    //Push markers to PTQuest
                return layer_marker;
        },
        }).addTo(mymap);

        // Map changes extent to fit the extent of questions layer
        mymap.fitBounds(questionsLayer.getBounds());
}
```

```
// Create global marker variables
var InitialTM = L.AwesomeMarkers.icon({markerColor: 'blue'});
var WrongQTM = L.AwesomeMarkers.icon({markerColor: 'red'});
var CorrectQTM = L.AwesomeMarkers.icon({markerColor: 'green'});
var QuestionsTM = L.AwesomeMarkers.icon({markerColor: 'pink'});
var QWithinRange = L.AwesomeMarkers.icon({markerColor: 'orange'});


// Create function for the intial show of the users position
function showPosition(position) {                // Blue marker used to display user intial
position
        if(!TrackUser){
                mymap.removeLayer(LocalUser);
                mymap.removeLayer(LocRad);
        }
        var radius = 20;                    // Blue marker has 20m radial circle surroudning
        LocalUser = L.marker([position.coords.latitude,position.coords.longitude],
{icon:InitialTM}).addTo(mymap);
        LocRad = L.circle([position.coords.latitude,position.coords.longitude],
radius).addTo(mymap);
        if(TrackUser){
                TrackUser = false;
                mymap.fitBounds(LocalUser.getLatLng().toBounds(250));              // Center
map on user
                autoPan = true;
        }else if (autoPan) {
                mymap.panTo(LocalUser.getLatLng());
        }
}


// Use fucntion to find distance from a marker and calculate the radius distance
function DistCheckQ(questionMarkers){
        latlng = LocalUser.getLatLng();           // User current location
        alert("Checking if within 20m from question");
                        for(var i=0; i<questionMarkers.length; i++) {
                                currentMarker = questionMarkers[i];
                                currentMarker_latlng = currentMarker.getLatLng();
```

```
                                              var distance = 
getDistanceFromLatLonInM(currentMarker_latlng.lat, currentMarker_latlng.lng, latlng.lat, 
latlng.lng);
                                                        if (distance <= 20) {
                                     questionMarkers[i].setIcon(QWithinRange);
                                                        questionMarkers[i].on('click', onClick);
                                                        } else {
                                              questionMarkers[i].setIcon(QuestionsTM);
                                                                 questionMarkers[i].bindPopup("Move 
closer to the question marker on map");
                            }
                  }
}




// This variable tells the user is they are correct
var answerTrue;

// Function and Process for user to iterate through in order to submit a question, with 
multiple error handling functionalities.
function startDataUpload() {
         alert ("Selected answer submitted");
         var AnsRight = PTClick.feature.properties.rightans;  // Question's rightanswer
         var question = document.getElementById("question").value; // Assign question
         var answer; // The users answer
         var postString = "question="+question;        // Uploads the data to database table

         // Get radio button from Opt
         if (document.getElementById("Opt1").checked) {answer = 
1;postString=postString+"&answer="+answer;}
   if (document.getElementById("Opt2").checked) {answer = 
2;postString=postString+"&answer="+answer;}
         if (document.getElementById("Opt3").checked) {answer 
=3;postString=postString+"&answer="+answer;}
         if (document.getElementById("Opt4").checked) {answer 
=4;postString=postString+"&answer="+answer;}

         if (answer == AnsRight) {alert("Correct!");answerTrue = true;                  // Is the 
answer correct?
         } else {
                  alert("Sorry, your answer of " +answer+" is incorrect! \n The rightanswer is: " 
+ AnsRight);
                  answerTrue = false;
         }
         postString = postString + "&AnsRight="+AnsRight;
```

```
        processData(postString);
}






/*Adapted from:
https://github.com/njj/haversine
https://www.movable-type.co.uk/scripts/latlong.html
https://andrew.hedges.name/experiments/haversine/ */
function getDistanceFromLatLonInM(lat1,lon1,lat2,lon2) {
  var R = 6371;
  var dLat = deg2rad(lat2-lat1);
  var dLon = deg2rad(lon2-lon1);
  var a =
    Math.sin(dLat/2) * Math.sin(dLat/2) +
    Math.cos(deg2rad(lat1)) * Math.cos(deg2rad(lat2)) *
    Math.sin(dLon/2) * Math.sin(dLon/2)
    ;
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
  var d = R * c;
  var d2 = d * 1000;
  return d2;
}

function deg2rad(deg) {
  return deg * (Math.PI/180)
}

function availableQuestions(){
        DistCheckQ(PTQuest);
}





// Create a global variable for the clicked marker
var PTClick;

function onClick(e) {
        QClickShow(this);
        PTClick = this;
}
```

```
// Create variables to iterate through the answers with the options chosen in the app
function QClickShow(QClicked) {
		document.getElementById('PopQuest').style.display = 'block';
		document.getElementById('mapid').style.display = 'none';

		document.getElementById("question").value = QClicked.feature.properties.question;
		document.getElementById("answera").value = QClicked.feature.properties.answera;
		document.getElementById("answerb").value = QClicked.feature.properties.answerb;
		document.getElementById("answerc").value = QClicked.feature.properties.answerc;
		document.getElementById("answerd").value = QClicked.feature.properties.answerd;

		document.getElementById("Opt1").checked = false;
		document.getElementById("Opt2").checked = false;
		document.getElementById("Opt3").checked = false;
		document.getElementById("Opt3").checked = false;
		PTClick = QClicked;
}


// This is another error handling method which makes the user select a radio button, i.e.
select an answer
function validateData() {
      var a=document.getElementById("Opt1").checked;
      var b=document.getElementById("Opt2").checked;
      var c=document.getElementById("Opt3").checked;
      var d=document.getElementById("Opt4").checked;
      if (a==false && b==false && c==false && d==false)
      {
         alert("Please select an answer.");
                              return false;
      }
      else
      {
         startDataUpload()
      }
}
```