Location Based Quiz – Application Documentation

Contents

System Aims

The aim of this system is to produce a location-based quiz, comprising both web and mobile applications. The web application allows an administrator to create quiz questions that are associated with a particular geographic location, and view the questions on a map. The question location is set by selecting a point on the map. These questions are then stored in a database hosted on a web server.

The mobile interface is a quiz activity that downloads these questions from the database and shows their locations on a map using point markers, in addition to the user's current location. The mobile application tracks the user's location using GPS and prompts them to answer questions when they are within 10 m of the associated marker. The user's answer is stored in the database on the web server, along with their user information, phone's IMEI, and final score.

This application uses a 3-teir web architecture comprising the web clients (web and mobile applications), web server, and database. Communication between the clients, server, and database is achieved using PHP.

This app is intended to encourage users to explore and learnt about their environment. Although it is currently set with questions in the vicinity of UCL, the app could be developed further to create a London-wide quiz for tourists.

1. Technical Documentation

1.1. Mobile Application

Application Task Specification

The mobile application implements the following tasks:

1. Obtain username and password from user, validate against existing users and save to database, retrieving resulting user ID from database and informing user of any errors.
2. Display map using Google Maps API, showing:
    a. The user's current location obtained using Location services, altering the map view to keep user's position in the centre.
    b. Point markers indicating the position of a question: These are downloaded from the web server as a GeoJSON file. Points are coloured azure as default.
3. Track the user's location at all times using GPS, whilst the GPS receiver is active.
4. When a user is within 20 m of a point, allow user to click on the point to answer the question.
5. When a user is within 10 m of a point, prompt user to answer the associated question.
6. Validate user's answer against the correct answer, and inform the user of the result.
7. Upload user's answers to database, along with the phone's IMEI number.
8. Display user's score beneath the map
9. Update question maker colours dynamically:
    - Purple: within 20 m of question
    - Green: question answered correctly
    - Red: question answered incorrectly
10. When the user has answered all questions, inform user of their final score, upload it to the database, and close the application.
11. User broadcast receivers to inform user when GPS receiver is enabled or disabled.
12. Use intents to pass information between activity classes

Installation

The mobile application can be installed through Android Studio. Open the project, allowing the gradle build to finish. Connect the device via USB and click 'run'.

Hardware and Software

This application was developed for a device with the following specifications. Running the application on a lower-spec device is not recommended.

Screen	5.0", touchscreen, 720 x 1280 pixels
RAM	1 GB
CPU	1..4 GHz
GPS	✓
Wi-Fi	802.11
Network (if no wi-fi)	HSPA/HSPA+/LTE
Storage	50 MB free
OS	Android 5.1 (minimum)

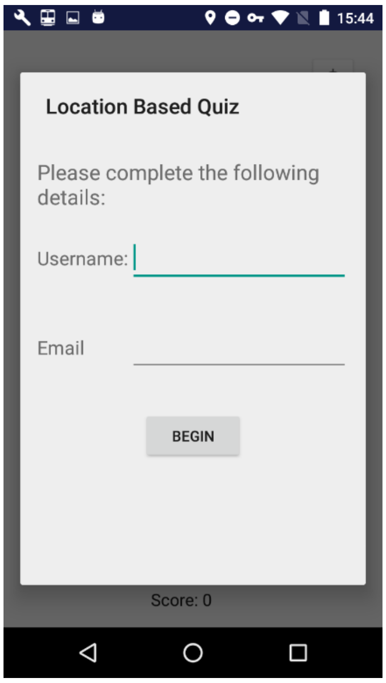
Development Software Requirements

Android Studio 2.2.2 with SDK 22 (minimum 21)

APIs: Google Play Services, Google Maps

Activity Architecture

The mobile app comprises 3 main activities. The following details the way in which users will navigate through the interface:



Location Based Quiz

Please complete the following details:

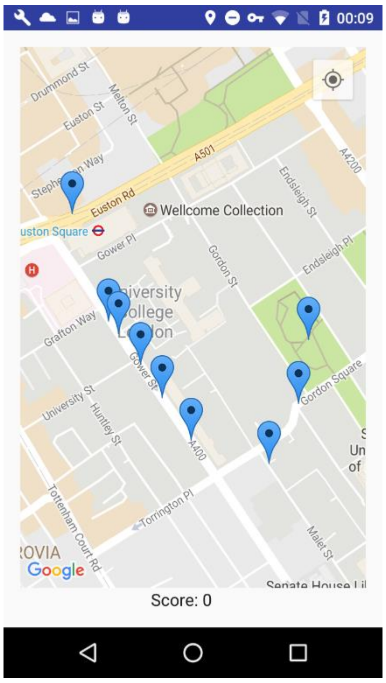
Username:

Email

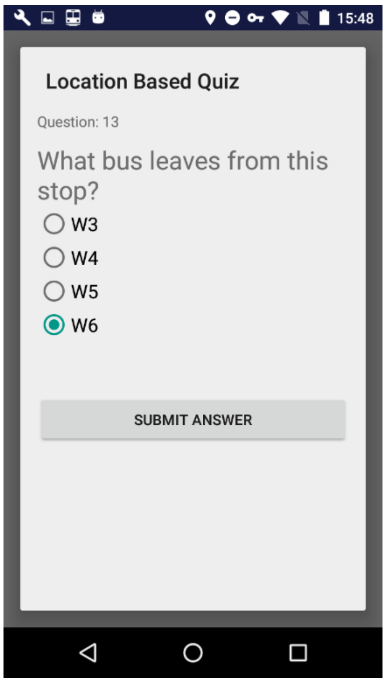
BEGIN

Score: 0

User Information



Map



Location Based Quiz

Question: 13

What bus leaves from this stop?

W3

W4

W5

[✓ W6]

SUBMIT ANSWER

Question

1. User Information:
    a. Select username
    b. Enter email
    c. Begin quiz
2. Map:
    a. View current position: this is tracked and updated in real-time
    b. View question locations and identify which questions have yet to be answered
    c. Select question to answer by either approaching the position or selecting the marker when in proximity
    d. View number of correctly answered questions (score)
    e. When quiz is complete, view final score and exit application
3. Question:
    a. Read question and select one answer from four possibilities
    b. Submit answer
    c. View result
    d. Exit question activity and return to map

Future Updates

1. *Add option to reset the quiz:* There is currently no option to re-start the quiz using the same user details. In its current state, the user can only attempt the quiz once using their email address.
2. *Add option to share score with friends:* This could be via email or social media.

Overview of Mobile Application Code

The application comprises the following components. Please see JavaDoc files or in-code comments for more detailed descriptions of the classes and their associated methods and variables.

Classes

Class	**Function**
MapActivity	• This is the main activity class for this application. It is associated with the activity_map.xml layout. • The questions are downloaded from the web server as GeoJSON points which are converted to Question objects, and stored in an ArrayList. • The user ID obtained from UserInfoActivity is used to create a User object, which is also stored in this activity. • Information is passed between this activity and the other activities using Intents. • A CustomLocationListener tracks the position of the user, and calculates the distance between the user and the questions every time their location changes. • User's location is received using the LocationManager service. • Questions and user position are displayed on a map created using Google Maps API. • The marker colours are refreshed as the user changes location, if the user is within 20 m of a point, it will turn purple. • Clicking on a question marker displays an alert dialog if the user is either too far away from the question marker, or has already answered the question. • Clicking on a purple question marker launches QuestionActivity and the user is prompted to answer a question. • If all questions are flagged isAnswered == true, a dialog is triggered informing the user of their final score, from which the user can exit the application.
CustomLocationListener	• Extends MapActivity. • Monitors changes in the user's location and calculates the distance between the user and question markers. • Broadcast receivers notify the user if their GPS receiver status has changed, i.e. has been disabled or enabled. • If user is within 20 m of the point, the isInProximity Boolean variable for the question is set to *true,* and the markers updated in MapActivity.
UserInfoActivity	• Activity is launched upon initial launch of MapActivity, setting the view to activity_user_info.xml. • The activity is launched using theme Theme.AppCompat.Light.Dialog.Alert, which creates the appearance of floating dialogue box. • User is prompted to enter username and email address using form. • If username or email already exist in the database, the user is asked to try different credentials.

	• Usernames must be between 6 and 30 characters, emails must be at least 6 characters and contain an '@' symbol.
QuestionActivity	• Activity is launched from MapActivity, setting the view to activity_question.xml. • The activity is launched using theme Theme.AppCompat.Light.Dialog.Alert, which creates the appearance of floating dialogue box. • Question information is passed to this activity through intents, and the resulting score sent back to MapActivity when the question has been answered. • Question information is extracted from the intent via a Bundle object • Answer is validated against correct answer, and uploaded to database along with the user's id and phone's IMEI number.
User	• Defines User object and implements setters and getters to access User variables in other classes • User's score is updated when they answer a question correctly.
Question	• Defines Question object and implements setters and getters to access Question variables in other classes. • Boolean variables isInProximity, isAnswered, inInProgress, and isCorrect indicate whether the user is within 20 m of the point, has answered the question, if they are currently answering the question, and if the question was answered correctly respectively.

Information is passed between the client and web server using HTTP requests, which are processed by KML files written in PHP, located on the server.

Layout Files

File	**Function**
activity_user_info.xml	Form allowing user to enter username and email address, and send these details to the database on the web server.
activity_map.xml	View current location of user, along with question markers.
activity_question.xml	View and answer question. Answer is validated against the correct answer and the user informed of the result.

Manifest

All activity class files are declared in the Android Manifest. The class MapActivity is declared as the MAIN and LAUNCHER class, and as such is the activity launched upon starting the app.

Permissions

The following permissions are required for the application to function:

```
<permission
    android:name="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.MAPS_RECEIVE"
    android:protectionLevel="signature" />

<uses-permission
android:name="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.MAPS_RECEIVE" />

<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission
```

```
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>

<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

Dependencies

The following dependencies are required in the gradle file:

```
compile fileTree(dir: 'libs', include: ['*.jar'])
androidTestCompile 'junit:junit:4.12'
testCompile 'junit:junit:4.12'
compile 'com.android.support:appcompat-v7:22.2.1'
compile 'com.google.android.gms:play-services:8.1.0'
compile 'com.google.maps.android:android-maps-utils:0.4'
```

Error handling

Error handling has been implemented throughout this application using try-catch statements. Logcat has been utilised to record these errors should they occur, and report them to the developer through Android Studio. Errors encountered with the webserver are also handled, and any communication errors reported to the user.

1.2. Web Application

The web application comprises both a browser interface created using HTML, JavaScript, and PHP, for creation of quiz questions, and a collection of PHP files for communication between the mobile application and the database located on the web server. The database contains three tables for storing the data creating using this app: users, questions, and answers. The mobile application communicates with the web server through HTTP requests.

Application Requirements

The web interface meets the following requirements:

Visualised through Browser Interface

1. View map using Google Map API in web browser, along with HTML form for question attributes.
2. Enter details of quiz question via HTML form. The following details are required:
    - Question ID
    - Point name
    - Question
    - 4 x potential answers
    - Correct answer
    - Coordinates of point associated with question, entered separately as ‘Latitude’ and ‘Longitude’
3. Select point on map displayed using Google Maps API, and populate form with coordinates.
4. Submit question to database.
5. Clear all values from form.

6. Select point on map and populate form with associated question variables.

PHP Only

7. Retrieve username and email values from mobile application and submit values to database, providing a response to the application containing the resulting user ID.
8. Convert questions to GeoJSON format (KML)
9. Submit user's answer to database, along with user ID and mobile phone's IMEI
10. Update user information to include final score.

Installation

This application is run through a web browser, and so no installation is required.

Creation of the necessary tables in the database should be carried out by running the SQL code: Database_table_creation.sql.

Hardware and Software

The web interface must be run using an HTML5 enabled browser. The interface will render most accurately using the latest version of Google Chrome or Mozilla Firefox.

Development Software Requirements

Notepad++ or similar text editor is required to develop the PHP files.

BitViseSSH Client to access web server.

Overview of Mobile Application Code (Task Architecture)

View and Add Questions (manageQuestions.php)

The user enters the question information into the form and selects the position of the question marker on the map.

Send Question To Database (processQuestion.php)

Pressing "Submit" sends the question information to the user. The user receives information as to whether the process was successful.

Create GeoJSON (createQuestionsGeoJSON.php)

Transforms the point locations stored in the *question* table in the database into GeoJSON format. This file is then requested by the mobile application via an HTTP request.

Process User Info (processUserInfo.php)

Sends the username and email entered by the mobile application user to the database. The resulting user ID is returned to the application. If there is an error, error codes are sent to the mobile application to enable the application to inform the user of the precise error.

Submit Answer (submitAnswer.php)

Sends the answer given by the user to the database, including the question ID, question, whether or not the answer was correct, user ID, and phone's IMEI number.

Submit Score (submitScore.php)

Updates the user's record in the database to reflect their final score.

Future Updates

The browser interface is currently very limited and so many improvements can be made to increase functionality. Some of the most essential upgrades are:

1. Allow for deletion of questions
2. Allow for updating of questions
3. Display server responses on the same webpage as the map and form.
4. Allow users to drag markers to desired position
5. Show clicked location on map

1.3. Testing

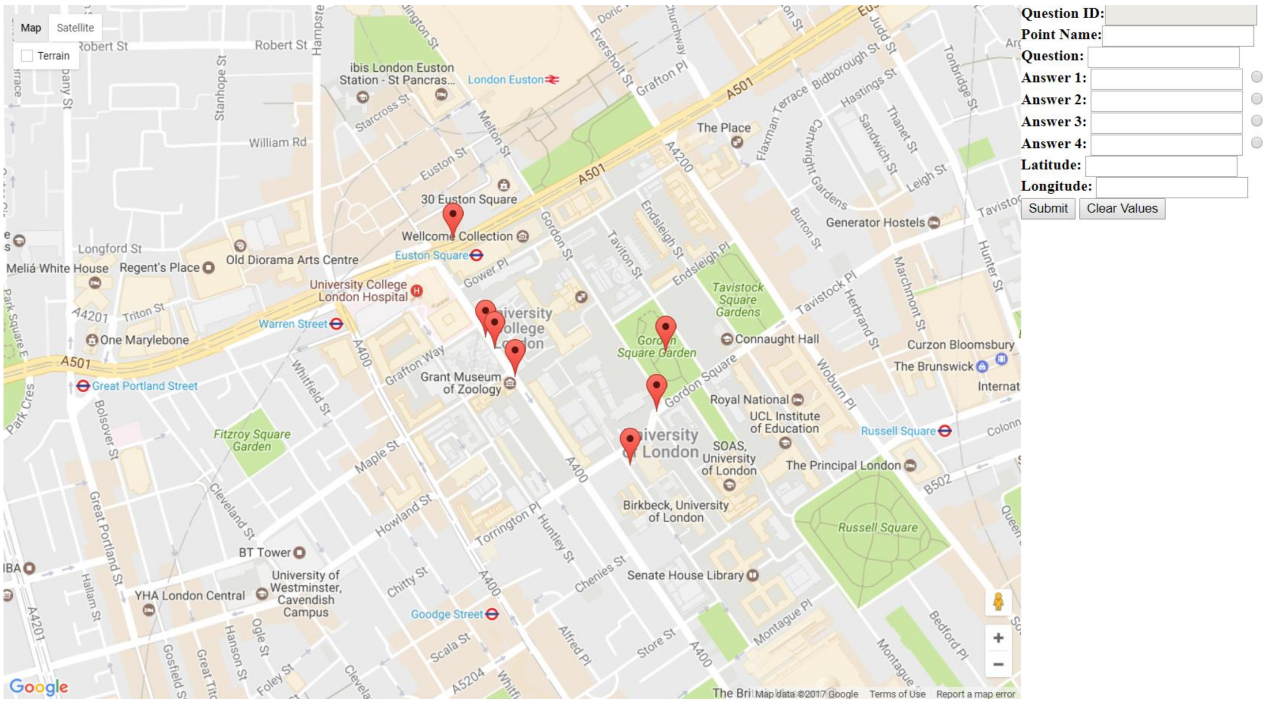
The web and mobile applications have been tested extensively in two locations around London: UCL, and Wood Green. Logcat has been utilised to identify and fix almost all known errors, with many validation steps added to ensure error-free usage of the quiz application. The application has been run successfully from beginning to end (completion of all questions) more than once. The only know error that has yet to be fixed is that if the user loses internet connectivity, the app will continue to run, despite the answers not being uploaded to the database. The app has not been used for longer than 10 consecutive minutes, and so extended use may have an effect on functionality.

2. Web Application User Guide

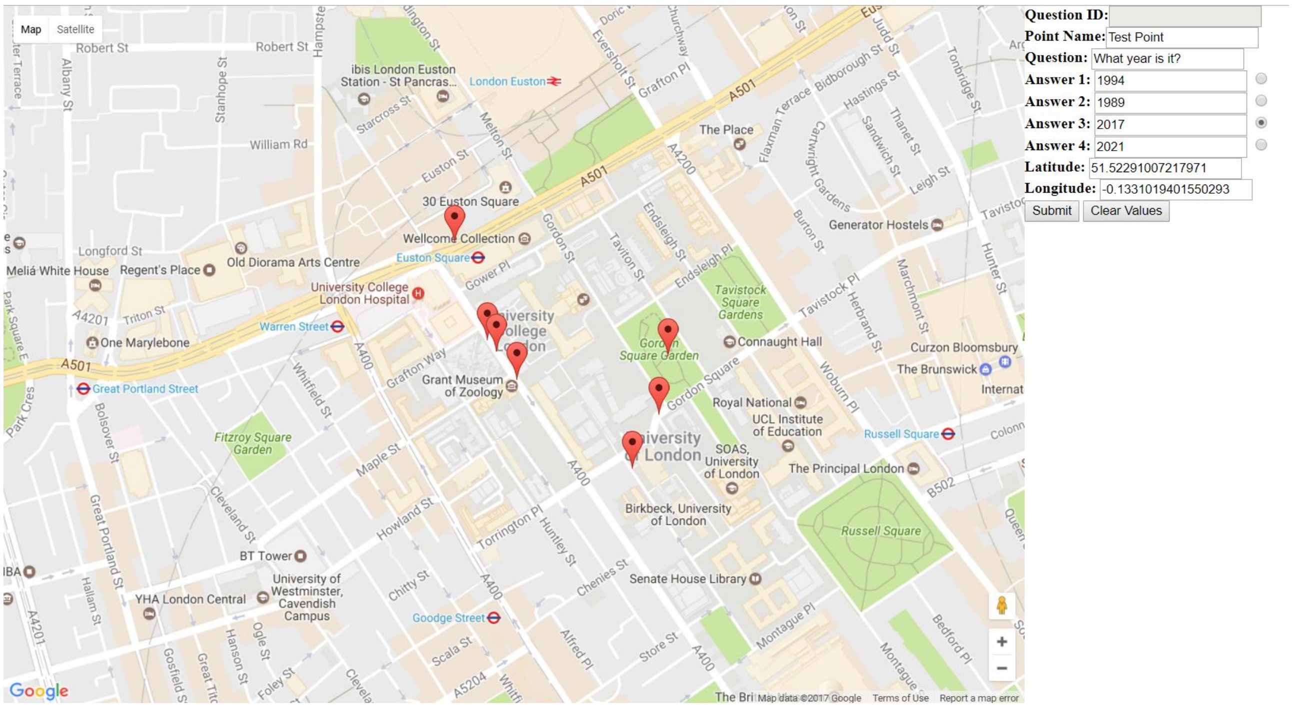
1) Open the following link in either Chrome or Firefox:

http://developer.cege.ucl.ac.uk:30522/teaching/user16/manageQuestions.php

This will open the following webpage:



2) To create a new question, populate the form with the desired values and select the correct answer using the radio buttons:
   a) Click on the map in the location you would like your question to appear, this will populate the 'Latitude' and 'Longitude' fields.
   b) Note that 'Question ID' cannot be populated manually. This number is assigned by the database.
   c) To clear all data from the form, click "Clear Values".



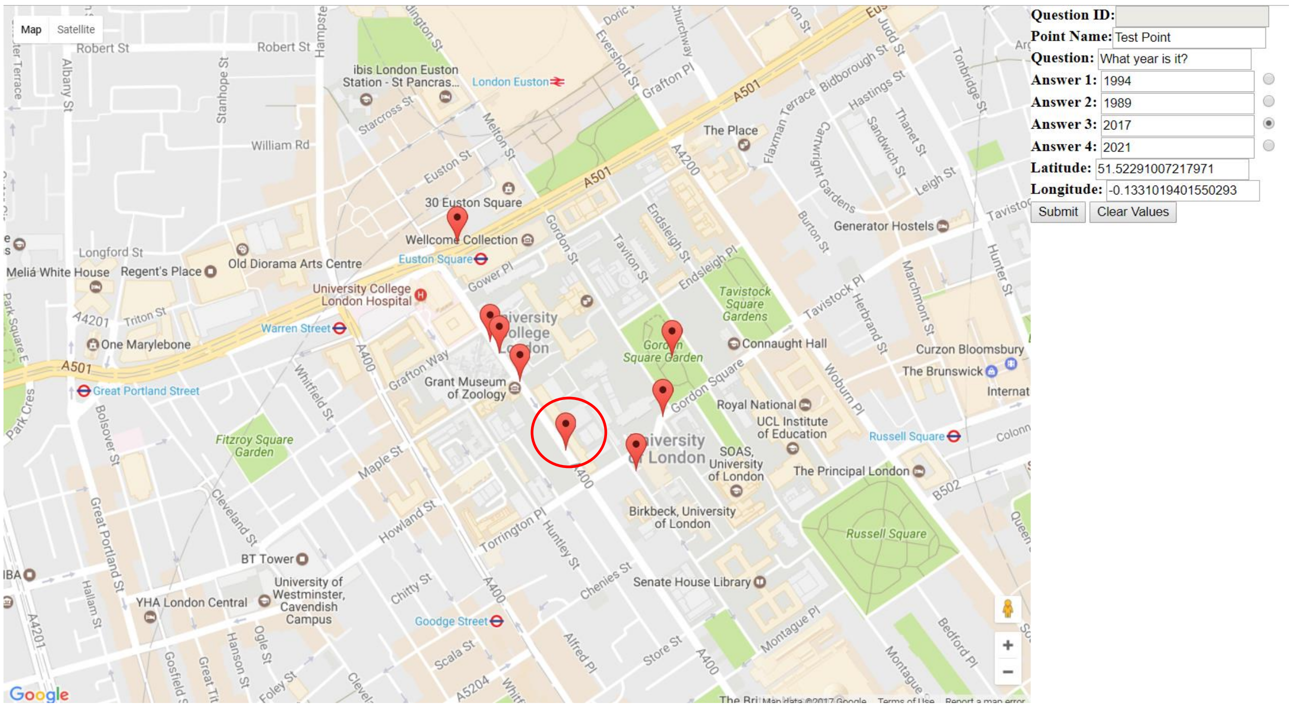
Question ID:	
Point Name:	Test Point
Question:	What year is it?
Answer 1:	1994
Answer 2:	1989
Answer 3:	2017 [✓]
Answer 4:	2021
Latitude:	51.52291007217971
Longitude:	-0.1331019401550293

3) Click "Submit" to save the question to the database. You will be shown either of the following messages:

Your data has been saved to the database. Please click the back button to continue adding questions to the database

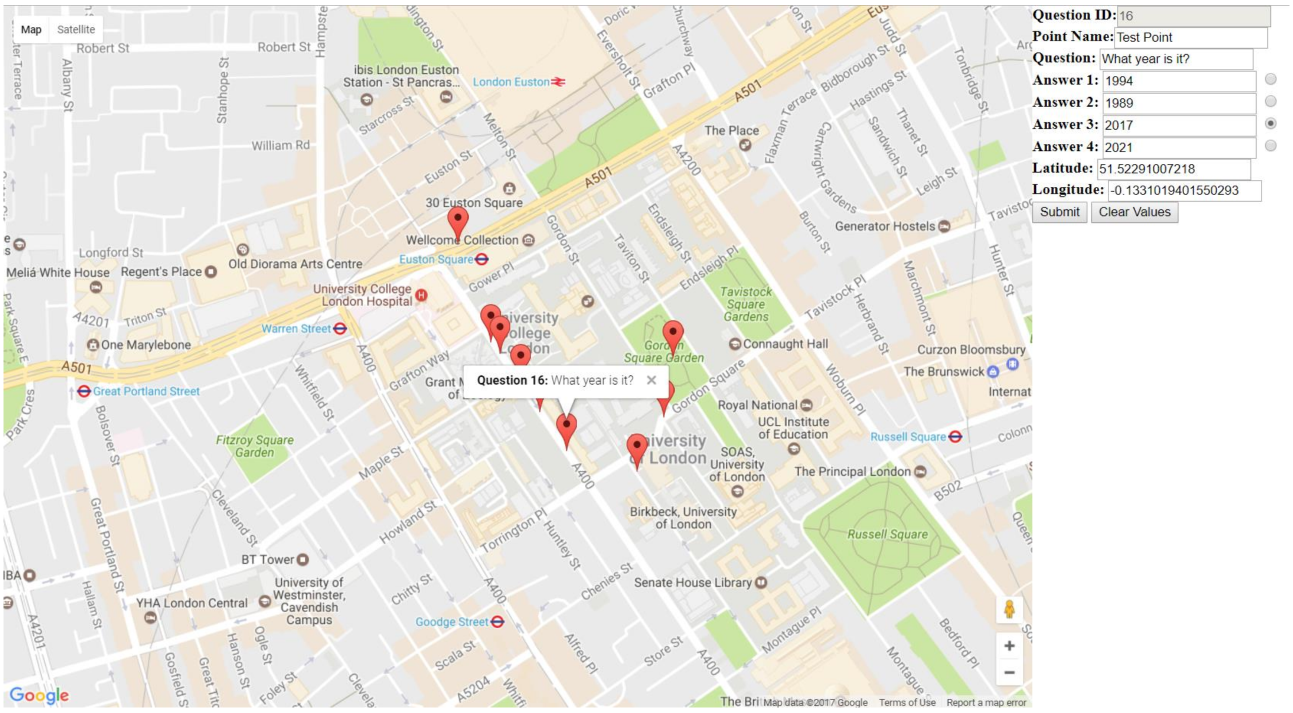
There was an error saving your data. Please click the back button and retry. Ensure your point name is unique.

4) Click the back button in the browser. If the data was saved successfully, the new question should be seen on the map in the selected location:



Question ID:	
Point Name:	Test Point
Question:	What year is it?
Answer 1:	1994
Answer 2:	1989
Answer 3:	2017 [✓]
Answer 4:	2021
Latitude:	51.52291007217971
Longitude:	-0.1331019401550293

5) Clicking on the point shows a pop-up, and populates the form with the question attributes. At this time, it is not possible to update a question's attributes using the web application.



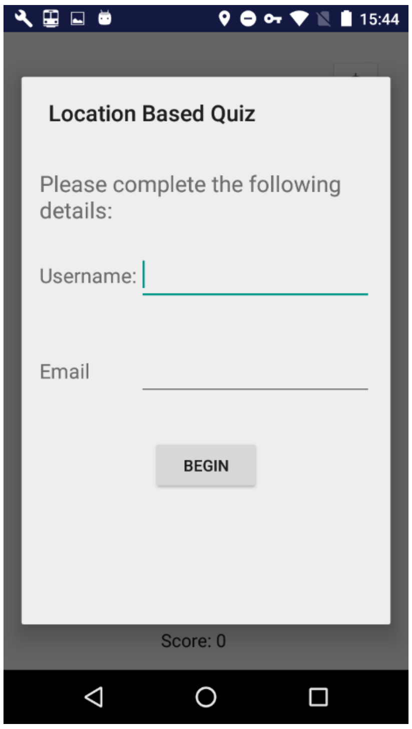
Question ID:	16
Point Name:	Test Point
Question:	What year is it?
Answer 1:	1994
Answer 2:	1989
Answer 3:	2017 [✓]
Answer 4:	2021
Latitude:	51.52291007218
Longitude:	-0.1331019401550293

3. Mobile Application User Guide

The screenshots are numbered according to their associated user information.

3.1. Starting the Application

a) Upon starting the app, the user is prompted to enter a username, and email address.
b) If the user enters no information (b.i) or incorrect information (b.ii), the user will be asked to correct their values.
c) Usernames must be between 6 and 30 characters, email addresses must contain an '@' symbol.
d) If the username or email address already exists in the database, the user will be asked to try again.

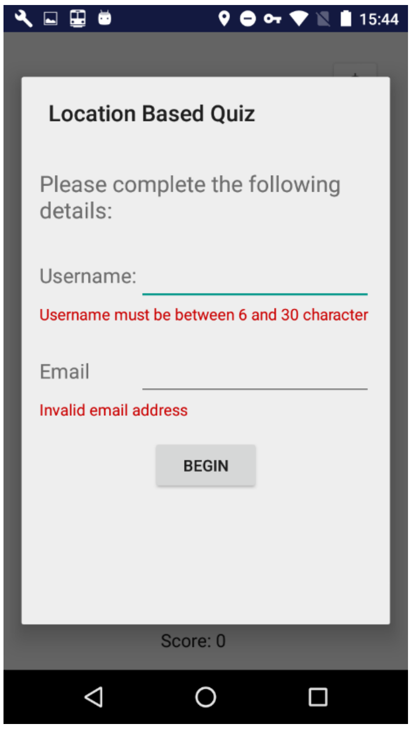

a)

Location Based Quiz

Please complete the following details:

Username:

Email


b.i)

Location Based Quiz

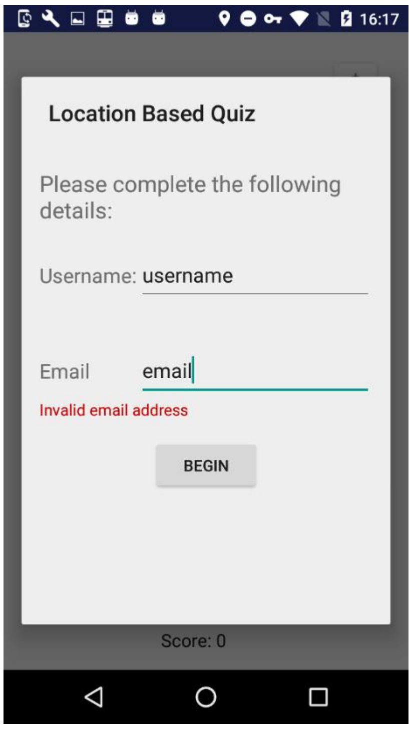
Please complete the following details:

Username:

Username must be between 6 and 30 character

Email

Invalid email address

b.ii), c)

16:17

Location Based Quiz

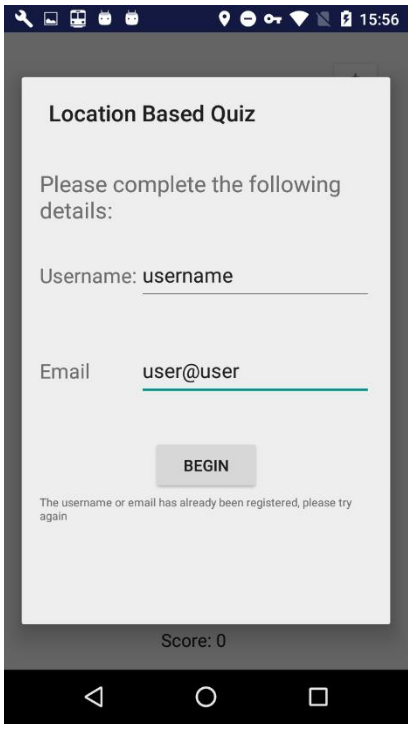
Please complete the following details:

Username: username

Email email

Invalid email address

Score: 0

d)

15:56

Location Based Quiz

Please complete the following details:

Username: username

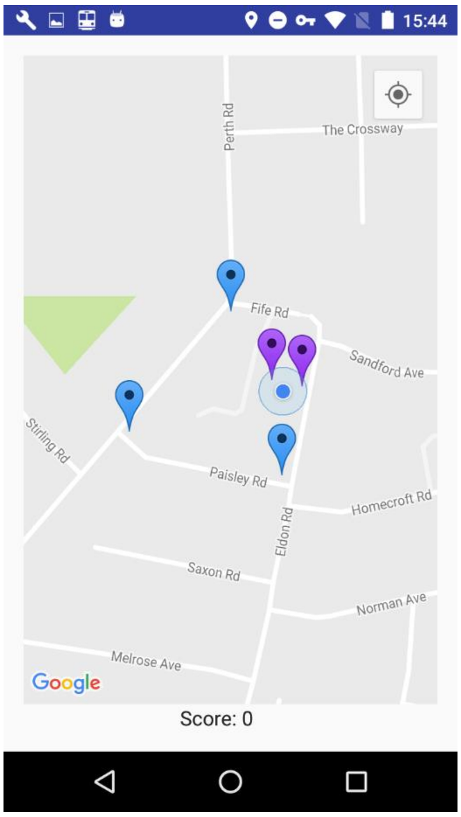
Email user@user

The username or email has already been registered, please try again

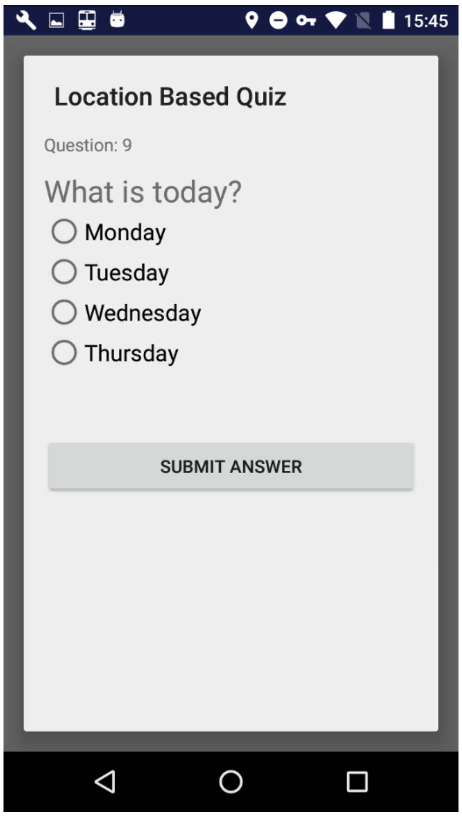
Score: 0

3.2. Answering Questions

e) One the user has entered their details, the user is shown a map with their current location, indicated by a blue dot, and the surrounding question markers. If the user's location is not shown immediately, ensure the device's GPS receiver is turned on, and the user is located outside. Once a GPS fix is established, the map will centre on the user's position.
f) Approach a point to answer a question: once the user is within 10 m of the point, the user will be prompted to answer a question.
g) Select answer and click "SUBMIT ANSWER" button.

e) 

Score: 0

f) 

Location Based Quiz

Question: 9

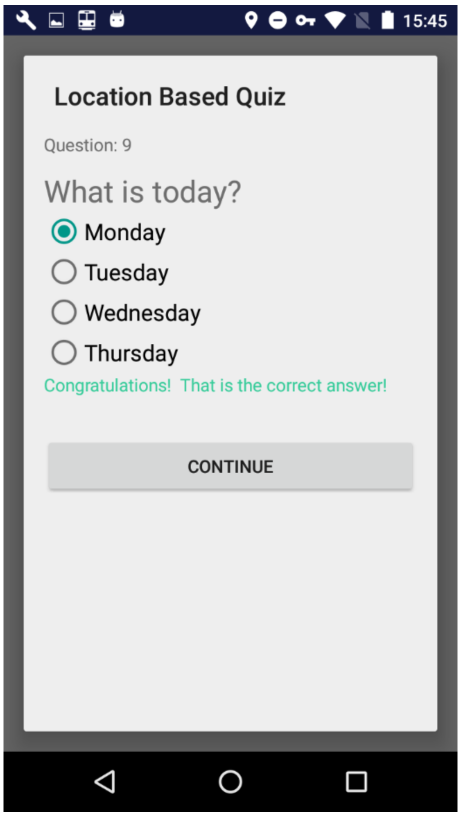
What is today?

- Monday
- Tuesday
- Wednesday
- Thursday

SUBMIT ANSWER

h) The user will be informed if their answer is correct (h.i) or incorrect (h.ii).

i) Click “CONTINUE” to return to the map, or press the “back” button.

h.i) 

Location Based Quiz

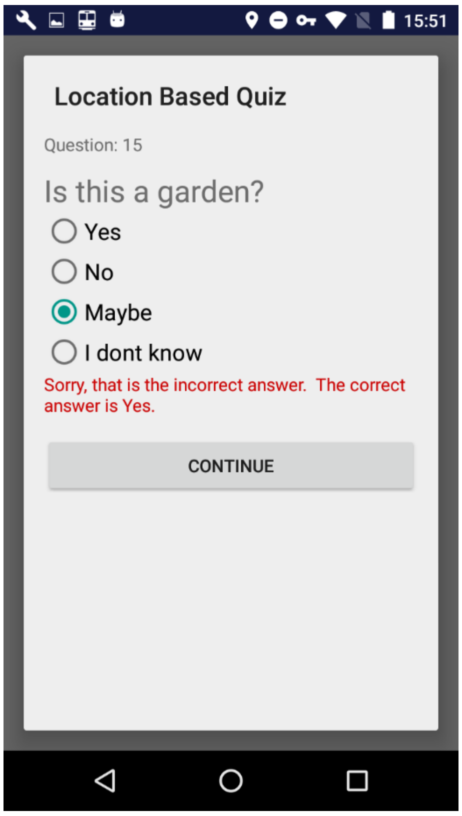
Question: 9

What is today?

[✓ Monday] Tuesday Wednesday Thursday

Congratulations! That is the correct answer!

CONTINUE

h.ii) 

Location Based Quiz

Question: 15

Is this a garden?

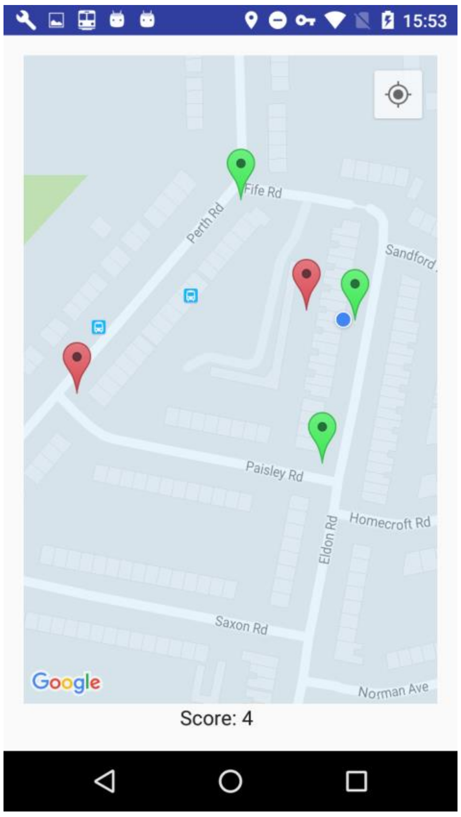
Yes No [✓ Maybe] I dont know

Sorry, that is the incorrect answer. The correct answer is Yes.

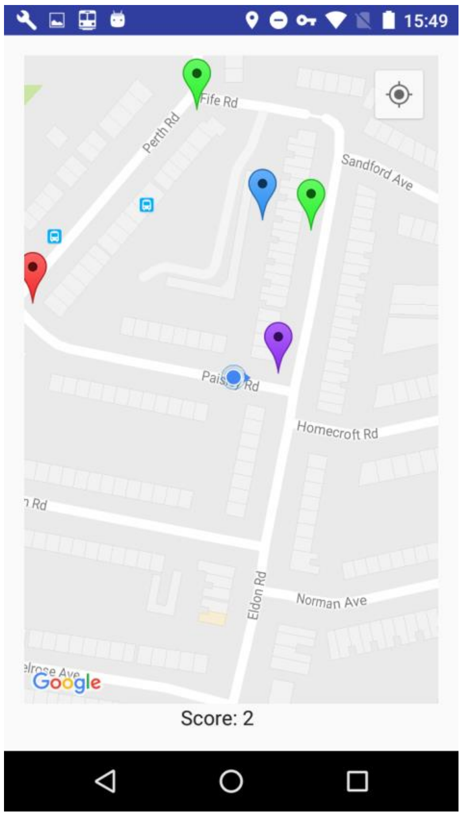
CONTINUE

j) Every time a question is answered, the question marker colour is updated to reflect whether or not the question was answered correctly. Green: question correct; Red: question incorrect. The 'Score' indicates the total number of correct answers.

k) Users can also answer questions by clicking on the question marker when it is purple. Purple markers indicate that the user is within 20 m of the point.



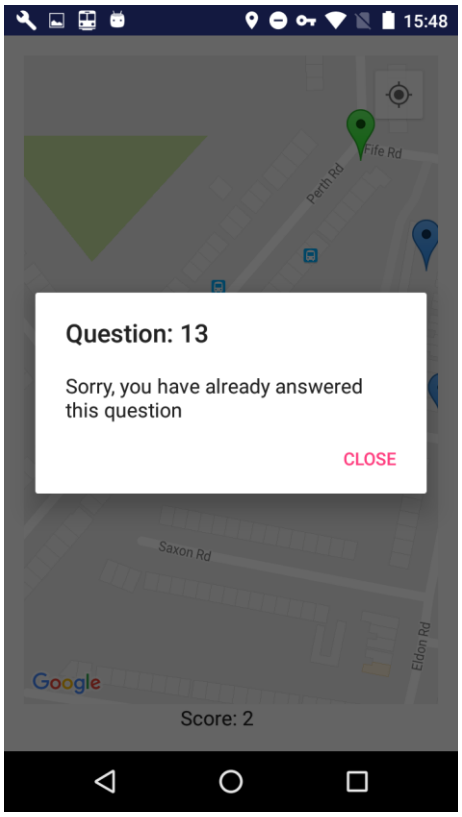
j)



k)

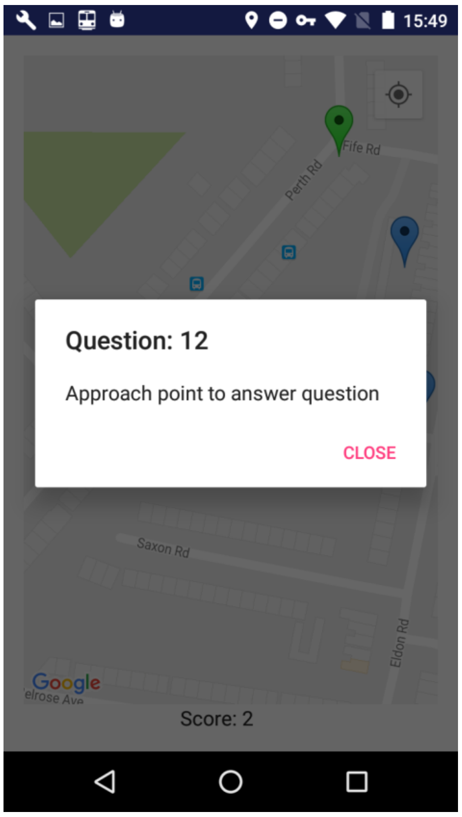
l) If a user selects a question that has already been answered, they will receive an alert message informing them of this.

m) If a user selects a question that has not been answered, but they are not within 20 m of the point, they will also receive an alert message informing them of this.


Question: 13

Sorry, you have already answered this question

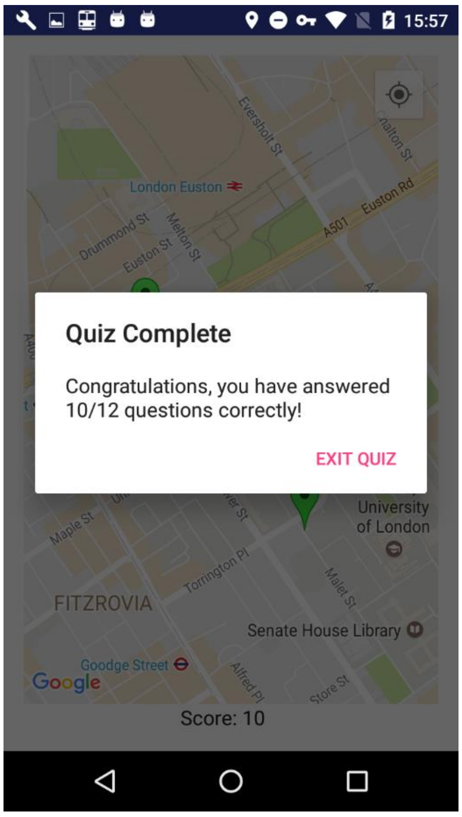
l)


Question: 12

Approach point to answer question

m)

n) Once the user has completed all questions, they will be informed of their score and prompted to exit the application.

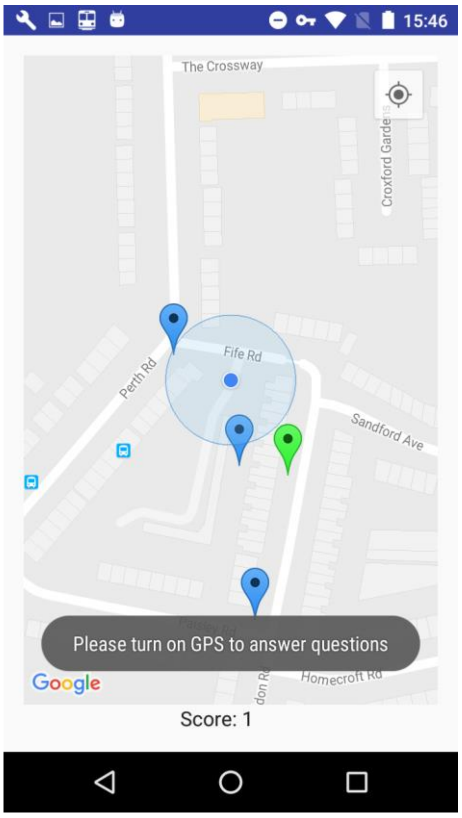

Quiz Complete

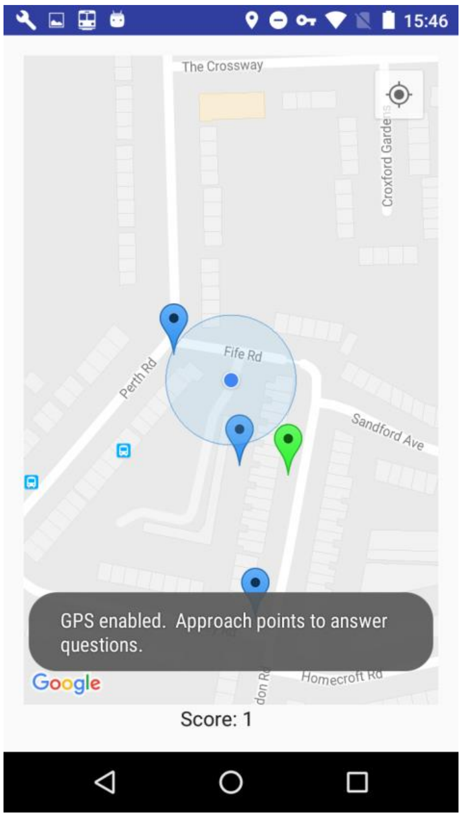
Congratulations, you have answered 10/12 questions correctly!

n)

3.3. GPS Warnings

o) An active GPS receiver is required to utilise this application. If the GPS receiver is turned on or off during use, the user will be informed of this.

o) 
Please turn on GPS to answer questions


GPS enabled. Approach points to answer questions.

1. Appendix A – Web Application Code

Database_table_creation.sql

```
create table public.users (
                              uid serial not null primary key,
                              username character varying (30) not null,
                              email character varying (50) not null,
                              score integer,

                              constraint username_unique unique (username),
                              constraint uid_unique unique (uid)

                                  );

create table public.questions (
                                  qid serial not null primary key,
                                  point_name character varying(50) not null,
                                  question character varying(200) not null,
                                  answer1 character varying (30) not null,
                                  answer2 character varying (30) not null,
                                  answer3 character varying (30) not null,
                                  answer4 character varying (30) not null,
                                  answer_correct integer not null,
                                  coordinates geometry not null,

                              constraint questions_unique unique (question,
coordinates),
                              constraint point_unique unique (coordinates),
                              constraint id_unique unique (point_name),

                              constraint answer_correct_check check
(answer_correct > 0 and answer_correct <= 4)
                                  );

insert into public.questions (point_name, question, answer1, answer2, answer3,
answer4, answer_correct, coordinates) values ('Euston Square', 'What year did
Euston Square station open?', '1863', '1909', '1925', '1963', '1',
st_geomfromtext('POINT(-0.13553202152252197 51.52579379667926)'));
insert into public.questions (point_name, question, answer1, answer2, answer3,
answer4, answer_correct, coordinates) values ('UCL Main Entrance', 'How many
students are enrolled at UCL?', '25,658', '39,473', '52,135', '37,253', '2',
st_geomfromtext('POINT(-0.13458788394927979 51.52426851607867)'));
insert into public.questions (point_name, question, answer1, answer2, answer3,
answer4, answer_correct, coordinates) values ('Cruciform', 'What year was the
crufiform building built?', '1919', '1889', '1901', '1906', '4',
st_geomfromtext('POINT(-0.13479173183441162 51.52442872300703)'));
insert into public.questions (point_name, question, answer1, answer2, answer3,
answer4, answer_correct, coordinates) values ('Gordon Square Gardens', 'Who is
Gordon Square Gardens named after?', 'Lady Georgine Gordon', 'Lord George
Gordon', 'Lady Georgiana Gordon', 'Lord Geoffrey Gordon', '3',
st_geomfromtext('POINT(-0.13070940971374512 51.52419508771479)'));
insert into public.questions (point_name, question, answer1, answer2, answer3,
answer4, answer_correct, coordinates) values ('Euston Church', 'What grade of
listed building is Euston Church?', 'I', 'II', 'II*', 'III', '1',
st_geomfromtext('POINT(-0.13091862201690674 51.523380692467136)'));
insert into public.questions (point_name, question, answer1, answer2, answer3,
answer4, answer_correct, coordinates) values ('Students Union', 'What is the
postcode of Student Central?', 'WC1E 7GZ', 'WC1E 7DW', 'WC1E 7AB', 'WC1E 7HY',
'4', st_geomfromtext('POINT(-0.13151943683624268 51.52261968702276)'));
```

```
create table public.answers (
                            aid serial not null primary key,
                            user_id integer not null,
                            question_id integer not null,
                            question character varying (100) not null,
                            answer character varying (30) not null,
                            correct_boolean boolean not null,
                            imei character varying(20) not null,

                            foreign key (user_id) references public.users (uid),
                            foreign key (question_id) references
public.questions (qid)
                            );
```

manageQuestions.php

```
<!DOCTYPE html>
<html>
<!-- taken from:
http://code.google.com/apis/maps/documentation/javascript/examples/map-
simple.html -->
  <head>
    <title>Google Maps JavaScript API v3 Example: Map Simple</title>
    <meta name="viewport"
        content="width=device-width, initial-scale=1.0, user-scalable=no">
    <meta charset="UTF-8">
    <style type="text/css">
      html, body, #map_canvas {
        margin: 0;
        padding: 0;
        height: 100%;
      }
    </style>
    <script type="text/javascript"
        src="http://maps.googleapis.com/maps/api/js?sensor=false"></script>

    <script type="text/javascript">
      var map;

      // this is the function to initialise the map when the page loads - it is
called by the 'addDomListener' call below
      function initialize() {
            // create Google maps centred on UCL
            var myOptions = {
              zoom: 16,
              center: new google.maps.LatLng(51.524213444816844, -
0.1341801881790161),
              mapTypeId: google.maps.MapTypeId.ROADMAP
            };

            // create the new map
            map = new google.maps.Map(document.getElementById('map_canvas'),
                myOptions);

            // load geoJSON data from KML
            map.data.loadGeoJson(

'http://developer.cege.ucl.ac.uk:30522/teaching/user16/createQuestionsGeoJSON.ph
p'
            )
            document.getElementById("questionID").disabled = true;

            // create info window in which to display question number and
question
            var infoWindow = new google.maps.InfoWindow();

            // add listener to get point and question attributes
            map.data.addListener('click', function(event) {
                var point = event.feature;


                document.getElementById("questionID").value =
point.getProperty("qid");
                document.getElementById("pointname").value =
point.getProperty("point_name");
                document.getElementById("question").value =
point.getProperty("question");
                document.getElementById("answer1").value =
point.getProperty("answer1");
                document.getElementById("answer2").value =
point.getProperty("answer2");
                document.getElementById("answer3").value =
point.getProperty("answer3");
                document.getElementById("answer4").value =
point.getProperty("answer4");


                var geom = JSON.stringify(point.getGeometry().get());
                var lat =
geom.substring(geom.lastIndexOf("t")+3,geom.lastIndexOf(","));
                var lng =
geom.substring(geom.lastIndexOf("g")+3,geom.lastIndexOf("}"));
                document.getElementById("latitude").value = lat;
                document.getElementById("longitude").value = lng;
                //document.getElementById("longitude").value = geom.get();

                var answerCorrect = point.getProperty("answer_correct");
                //document.getElementById("troubleshoot").value = answerCorrect;

                switch (answerCorrect) {

                    case 1:
                        //document.getElementById("troubleshoot").value =
answerCorrect;
                        document.getElementById("ra1").checked = true;
                        break;
                    case 2:
                        //document.getElementById("troubleshoot").value =
answerCorrect;
                        document.getElementById("ra2").checked = true;
                        break;
                    case 3:
                        //document.getElementById("troubleshoot").value =
answerCorrect;
                        document.getElementById("ra3").checked = true;
                        break;
                    case 4:
                        //document.getElementById("troubleshoot").value =
answerCorrect;
                        document.getElementById("ra4").checked = true;
                        break;
                    default:
```

```
                                break;
                        };

                        var qID = point.getProperty("qid");
                        var question = point.getProperty("question");

                        // set info window content to question ID and question
                        infoWindow.setContent("<strong>Question "+qID + ": </strong>" +
question);
                        infoWindow.setPosition(point.getGeometry().get());
                        infoWindow.setOptions({pixelOffset: new google.maps.Size(0,-
30)});
                        infoWindow.open(map);

                        });

                // listener to detect click on map and set lat lng in corresponding
boxes
                google.maps.event.addListener(map, 'click', function(point) {
                    document.getElementById("latitude").value = point.latLng.lat();
                    document.getElementById("longitude").value = point.latLng.lng();
                    });

        } // end of the initialize function

    function resetForm() {
        document.getElementById("questionForm").reset();
    };

        // call the initialize method to create the map
        // this listener is called when the window is first loaded
        google.maps.event.addDomListener(window, 'load', initialize);
    </script>
  </head>
  <body>
    <div id="map_canvas" style="width:70%;float:left"></div>
    <div id="lat_lng" style="width:30%;float:right">
    <form id="questionForm" action="processQuestion.php" method="post">
        <strong>Question ID:</strong><input type="text" name="questionID"
id="questionID" /><br />
        <strong>Point Name:</strong><input type="text" name="pointname"
id="pointname" size="match_parent" /><br />
        <strong>Question: </strong><input type="text" name="question"
id="question"/><br />
        <strong>Answer 1:</strong> <input type="text" name="answer1"
id="answer1" /> <input type="radio" name="answercorrect" value="1" id="ra1"> <br
/>
        <strong>Answer 2:</strong> <input type="text" name="answer2"
id="answer2" /> <input type="radio" name="answercorrect" value="2" id="ra2"> <br
/>
        <strong>Answer 3:</strong> <input type="text" name="answer3"
id="answer3" />  <input type="radio" name="answercorrect" value="3" id="ra3">
<br />
        <strong>Answer 4:</strong> <input type="text" name="answer4"
id="answer4" /> <input type="radio" name="answercorrect" value="4" id="ra4"><br
/>

        <strong>Latitude:</strong> <input type="text" name="latitude"
id="latitude" /><br />
        <strong>Longitude:</strong> <input type="text" name="longitude"
id="longitude" /><br />

        <input type="submit" value="Submit" />
        <input type="reset" value="Clear Values" onclick="resetForm()" /><br />
    </form>
    </div><!-- end of the lat-lng div -->
  </body>
</html>
```

processQuestion.php

```
<?php


// connect to the database
    $db = pg_connect('host=localhost user=user16 port=5432 dbname=user16db
password=user16password');

    // check if the connection worked, if not, exit
    if (!$db) {
      echo "An error occurred connecting to the database.\n";
      exit;
    }

// extract the values from the form using $_REQUEST, since this will work
regardless of whether POST or GET is used.
$pointname = $_REQUEST['pointname'];
$question = $_REQUEST['question'];
$answer1 = $_REQUEST['answer1'];
$answer2 = $_REQUEST['answer2'];
$answer3 = $_REQUEST['answer3'];
$answer4 = $_REQUEST['answer4'];
$answercorrect = $_REQUEST['answercorrect'];
$latitude = $_REQUEST['latitude'];
$longitude = $_REQUEST['longitude'];

// create WKT point for database
$coordinates ="ST_geomfromtext('POINT(".$longitude." ".$latitude.")')";

// build query to insert data into database
$query = "insert into public.questions (point_name, question, answer1, answer2,
answer3, answer4, answer_correct, coordinates) values (";
$query =
$query."'".$pointname."','".$question."','".$answer1."','".$answer2."','".$answe
r3."','".$answer4."','".$answercorrect."',".$coordinates.")";


// run the query and inform the user of the result.
if (pg_query($db,$query)) {
    echo("Your data has been saved to the database.  Please click the back
button to continue adding questions to the database");
}
else {
    echo("There was an error saving your data.  Please click the back button and
retry.  Ensure your point name is unique.");
}

?>
```

processUserInfo.php

```
<?php

// connect to the database
    $db = pg_connect('host=localhost user=user16 port=5432 dbname=user16db
password=user16password');

    // check if connection worked
    if (!$db) {
      //echo "An error occurred connecting to the database.\n";
      echo (-1); // return error code for identification
      exit;
    }

// extract the values from the form using $_REQUEST, since this will work
regardless of whether POST or GET is used.
$username = $_REQUEST['username'];
$email = $_REQUEST['email'];

// build insert query
$query = "insert into public.users (username, email) values (";
$query = $query."'".$username."','".$email."')";

// run query and if successful, run select query to retrieve user ID and return
to mobile app
if (pg_query($db,$query)) {
    //echo("Query successful");
    $select = "select uid from public.users where username = '".$username."'";
    $result = pg_query($db,$select) or die('Query failed: ' . pg_last_error());

    while ($row = pg_fetch_row($result)) {
        echo($row[0]);
    }
    exit;
} else {
    // error code -999 for query unsuccessful
    echo (-999);
    exit;
}


pg_close($db);

?>
```

createQuestionsGeoJSON.php

```
<?php
/* Note:  This code is adapted from source code found here:
http://stackoverflow.com/questions/17775627/creating-a-geojson-in-php-from-
mysql-to-use-with-mapbox-javascript-api
   Accessed: 14th February 2016
 */

// connect to database
    $db = pg_connect('host=localhost user=user16 port=5432 dbname=user16db
password=user16password');
```

```
    // exit if database not responding
    if (!$db) {
      echo "An error occurred connecting to the database.\n";
      exit;
    }


    // retreive data from database table
    $query = "select qid, point_name, question, answer1, answer2, answer3, 
answer4, answer_correct, ST_astext(coordinates) as coordinates from 
public.questions";
    $result = pg_query($db,$query);

    // check if it worked, and if not exit
    if (!$result) {
      echo "An error occurred running the query.\n";
      exit;
    }

    // Build GeoJSON feature collection array
    $geojson = array(
       'type'       => 'FeatureCollection',
       'features'   => array()
    );

    $id=0;
    // Loop through rows to build feature arrays
    while ($row = pg_fetch_array($result))   {

        // get attributes
        $properties = $row;

        // remove coordinates for now
        unset($properties['coordinates']);

        // handle coords to extract lat and lng
        $coordinates = $row['coordinates'];

        // remove 'POINT'
        $coordinates = str_replace('POINT(','',$coordinates);
        // remove )
        $coordinates = str_replace(')','',$coordinates);
        // extract the $lat and $lng values from string using explode
        $coordvalues = explode(' ',$coordinates);

        $lat = $coordvalues[0];
        $lng = $coordvalues[1];
        $id = $id + 1;
        // build feature array
        $feature = array(
            'type' => 'Feature',
            'properties' => $properties,
            'geometry' => array(
                'type' => 'Point',
                'coordinates' => array(
                    $lat,
                    $lng
                )
            ),
        'id'=>$id
        );
        // Add feature arrays to feature collection array
        array_push($geojson['features'], $feature);
    }


    // echo data out
    header('Content-type: application/json');
    echo json_encode($geojson, JSON_NUMERIC_CHECK);

    // disconnect from the database - this is good practice
    $db = NULL;
?>
```

submitAnswer.php

```
<?php

echo("This is your data<br>");
print_r($_REQUEST);
echo("<br><br>");

// connect to the database
    $db = pg_connect('host=localhost user=user16 port=5432 dbname=user16db
password=user16password');

    //check database connection
    if (!$db) {
      echo "An error occurred connecting to the database.\n";
      exit;
    }

// extract the values from the form
// we use $_REQUEST as this works no matter whether the user has used POST or
GET
$qid = $_REQUEST['qid'];
$question = $_REQUEST['question'];
$answer = $_REQUEST['answer'];
$correct= $_REQUEST['correct'];
$imei= $_REQUEST['imei'];
$uid = $_REQUEST['uid'];

// build query to insert information into database
$query = "insert into public.answers (user_id, question_id, question, answer,
correct_boolean, imei) values (";
$query =
$query.$uid.",".$qid.",'".$question."','".$answer."',".$correct.",'".$imei."')";
//$comment = "You have chosen: ";
//$comment = $answer.";

echo($query);
echo("<br/><br/>");


// run the insert query
// if the result is TRUE it has worked
if (pg_query($db,$query)) {
    echo("Your data has been saved to the database");
}
else {
    echo("There was an error saving your data");
}
```

```
?>
```

submitScore.php

```
<?php

echo("This is your data<br>");
print_r($_REQUEST);
echo("<br><br>");

    // connect to the database
    $db = pg_connect('host=localhost user=user16 port=5432 dbname=user16db 
password=user16password');

    // check database connection
    if (!$db) {
      echo "An error occurred connecting to the database.\n";
      exit;
    }

// extract the values from the form
// we use $_REQUEST as this works no matter whether the user has used POST or 
GET
$uid = $_REQUEST['uid'];
$score = $_REQUEST['score'];

$query = "update public.users set score = ";
$query = $query.$score." where uid = ".$uid;

echo($query);
echo("<br/><br/>");


// run the insert query
// if the result is TRUE it has worked
if (pg_query($db,$query)) {
    echo("Your data has been saved to the database");
}
else {
    echo("There was an error saving your data");
}

?>
```

2. Appendix B – Mobile Application Code

AndroidManifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz">

    <permission

android:name="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.MAPS_RECEIVE"
        android:protectionLevel="signature" />

    <uses-permission
android:name="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.MAPS_RECEIVE" />

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission
android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>
    <!-- The following two permissions are not required to use
         Google Maps Android API v2, but are recommended. -->
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE" />

    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true"/>


    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity android:name=".MapActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>


        <activity
android:name="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.QuestionActivity"
            android:theme="@style/Theme.AppCompat.Light.Dialog.Alert"
            android:label="@string/app_name">

        </activity>

        <activity
android:name="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.UserInfoActivity"
            android:theme="@style/Theme.AppCompat.Light.Dialog.Alert"
            android:label="@string/app_name">

        </activity>


        <meta-data
```

```
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyD9O0su-RAw4bS7uiV4gcwl4E45Vh0Jr7w"/>


    </application>

</manifest>
```

Gradle

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 22
    buildToolsVersion "22.0.1"
    defaultConfig {
        applicationId "uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz"
        minSdkVersion 21
        targetSdkVersion 22
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-
rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile 'junit:junit:4.12'
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:22.2.1'
    compile 'com.google.android.gms:play-services:8.1.0'
    compile 'com.google.maps.android:android-maps-utils:0.4'
}
```

activity_map.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"

tools:context="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.QuestionActivity">

    <LinearLayout
        android:orientation="vertical"
        android:id ="@+id/mainwrapper"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true"
        android:weightSum="1">

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="460dp"
            android:layout_weight="1">

            <fragment
                android:id="@+id/map"
                android:layout_width="match_parent"
                class="com.google.android.gms.maps.SupportMapFragment"
                android:name="com.google.android.gms.maps.MapFragment"
                android:layout_height="match_parent" />

        </LinearLayout>

        <LinearLayout
            android:orientation="horizontal"
            android:layout_width="match_parent"
            android:layout_height="wrap_content">

            <TextView
                android:layout_width="173dp"
                android:text="Score: "
                android:textSize="16sp"
                android:textColor="@color/primary_material_dark"
                android:layout_height="wrap_content"
                android:gravity="bottom|right" />

            <TextView
                android:id="@+id/tvScore"
                android:layout_width="match_parent"
                android:textSize="16sp"
                android:textColor="@color/primary_material_dark"
                android:layout_height="match_parent"
                android:gravity="bottom|left"
                android:text="0" />
        </LinearLayout>
    </LinearLayout>
</RelativeLayout>
```

activity_user_info.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"

tools:context="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.QuestionActivity"
    android:layout_width="match_parent">

    <LinearLayout
        android:orientation="vertical"
        android:id ="@+id/mainwrapper"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true">
        <TextView xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textSize="20sp"
            tools:layout_height="wrap_content"
            android:text="Please complete the following details:"
            android:layout_marginBottom="20dp"
            android:layout_marginTop="20dp">


        </TextView>

        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="100px"
            >
            <TextView android:layout_width="175px" android:text="Username:"
android:layout_height="match_parent"
android:textAppearance="?android:attr/textAppearanceMedium"
android:id="@+id/textView1"
                android:gravity="left|center_vertical"></TextView>
            <EditText android:tag="username" android:name="username"
android:layout_width="wrap_content" android:id="@+id/username"
android:layout_height="match_parent" android:layout_weight="1"
                android:inputType="text">
                <requestFocus></requestFocus>
            </EditText>
        </LinearLayout>
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="35dp"
            android:id="@+id/tvUsername" />
        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="100px"
            >
            <TextView android:layout_width="175px" android:text="Email"
android:layout_height="match_parent"
android:textAppearance="?android:attr/textAppearanceMedium"
android:id="@+id/textView2"
                android:gravity="left|center_vertical"></TextView>
            <EditText android:tag="email" android:name="email"
android:layout_width="wrap_content" android:id="@+id/email"
android:layout_height="match_parent" android:layout_weight="1"
                android:inputType="text">
            </EditText>
        </LinearLayout>

        <TextView
            android:layout_width="fill_parent"
            android:layout_height="35dp"
            android:id="@+id/tvEmail" />

        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="100px"
            >

            <Button
                android:id="@+id/buttonBegin"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="Begin"
                android:layout_marginLeft="100dp"
                android:layout_marginRight="100dp" />
        </LinearLayout>

        <TextView
            android:id="@+id/webResponse"
            android:layout_width="fill_parent"
            android:text="@string/debug"
            android:layout_height="100dp"
            android:textSize="10sp" />

    </LinearLayout>
</RelativeLayout>
```

activity_question.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"

tools:context="uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.QuestionActivity">

    <LinearLayout
        android:orientation="vertical"
        android:id ="@+id/mainwrapper"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignParentTop="true"
        android:layout_alignParentStart="true">

        <TextView
            android:layout_width="match_parent"
            android:layout_height="30dp"
            android:orientation="horizontal"
            android:id="@+id/tvID"
            android:textSize="14sp">

        </TextView>

        <TextView xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:id="@+id/tvQuestion"
            android:layout_height="wrap_content"
            android:textSize="24sp"
            tools:layout_height="wrap_content">


        </TextView>

        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            >

            <RadioGroup
                android:id="@+id/radioGroup"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content" >

                <RadioButton
                    android:id="@+id/radioAnswer1"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="@string/radioAnswer1"
                    android:textSize="18sp" />

                <RadioButton
                    android:id="@+id/radioAnswer2"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="@string/radioAnswer2"
                    android:textSize="18sp" />

                <RadioButton
                    android:id="@+id/radioAnswer3"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="@string/radioAnswer3"
                    android:textSize="18sp" />

                <RadioButton
                    android:id="@+id/radioAnswer4"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="@string/radioAnswer4"
                    android:layout_weight="1"
                    android:textSize="18sp" />

            </RadioGroup>
        </LinearLayout>

        <TextView
            android:id="@+id/answerResponse"
            android:layout_width="fill_parent"
            android:layout_height="50dp" />

        <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:orientation="horizontal"
            android:layout_width="fill_parent"
            android:layout_height="100px"
            >

            <Button
                android:id="@+id/buttonSubmit"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="Submit Answer" />
        </LinearLayout>

        <TextView
            android:id="@+id/webResponse"
            android:layout_width="fill_parent"
            android:layout_height="100dp"
            android:textSize="10sp"
            android:text="@string/debug" />

    </LinearLayout>
</RelativeLayout>
```

MapActivity

```
package uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz;

/*============================
Note that code in this file has been adapted from:
https://github.com/googlemaps/android-maps-
utils/blob/master/demo/src/com/google/maps/android/utils/demo/GeoJsonDemoActivity.ja
va
Note also that putting KML on an Android Google Map functionality is in Beta so
there may be some issues
*/

/**
 * <h2> MapActivity </h2>
 *
 * Main activity class for Location Based Quiz
 * Controls behaviour for activity_map.xml
 *
 * @author Lucille Ablett
 *
 * Adapted from code provided as part of CEGEG077 Web and Mobile GIS
 * Also adapted from: https://github.com/googlemaps/android-maps-
utils/blob/master/demo/src/com/google/maps/android/utils/demo/GeoJsonDemoActivity.ja
va
 *
 */

import android.app.Activity;
import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.location.LocationManager;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v4.app.FragmentActivity;
import android.util.Log;
import android.widget.TextView;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.GoogleMap.OnMarkerClickListener;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptor;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.maps.android.geojson.GeoJsonFeature;
import com.google.maps.android.geojson.GeoJsonLayer;
import com.google.maps.android.geojson.GeoJsonPoint;
import com.google.maps.android.geojson.GeoJsonPointStyle;

import org.json.JSONException;
import org.json.JSONObject;

import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;
import java.util.ArrayList;


public class MapActivity extends FragmentActivity implements OnMapReadyCallback,
OnMarkerClickListener {

    /**
     * Interface instance variables:
     * @param mLayer map layer containing GeoJson information
     * @param map Google Map
     * @param mLogTag logcat message tag for debugging purposes
     * @param tvScore TextView in which user's score is displayed
     */
    private static GeoJsonLayer mLayer;
    protected GoogleMap map;
    private final static String mLogTag = "GeoJsonDemo";
    //private final Context context = this;
    public static MapActivity mapActivity = null;
    private static TextView tvScore;

    /**
     * Proximity tracking variables
     * @param MINIMUM_DISTANCECHANGE_FOR_UPDATE minimum distance from previous
location before LocationListener triggered
     * @param MINIMUM_TIME_BETWEEN_UPDATE minimum rate at which location change
updates must be registered
     * @param locationManager location manager instance
     * @param questionList ArrayList to which quiz questions will be added
     * @param currentLat user's current latitude
     * @param currentLng user's current longitude
     */
    private static final long MINIMUM_DISTANCECHANGE_FOR_UPDATE = 1; // in Meters
    private static final long MINIMUM_TIME_BETWEEN_UPDATE = 1000; // in Milliseconds
    private LocationManager locationManager;
    public ArrayList<Question> questionList = new ArrayList<Question>();
    public double currentLat;
    public double currentLng;

    /**
    * @param user User object for current user
    */
    public User user = null;

    /**
     * @param user GET_QUESTION_REQUEST request code for intent that triggers
QuestionActivity
     * @param user GET_USERINFO_REQUEST request code for intent that triggers
UserInfoActivity
     */
    public static final int GET_QUESTION_REQUEST = 1;  // The request code
    public static final int GET_USERINFO_REQUEST = 2;  // The request code

    /**
     * Called when the activity is first created.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        //this.instance = this;
        setContentView(R.layout.activity_map);  // set the view to activity_map.xml
        tvScore = (TextView) findViewById(R.id.tvScore);
        this.mapActivity = this;
        getUserInfo();
        initLocationListener();

        // assign the map fragment to a variable so that we can manipulate it
        SupportMapFragment mapFragment = ((SupportMapFragment)
getSupportFragmentManager().findFragmentById(R.id.map));

        // Call onMapReady once the map is drawn
        mapFragment.getMapAsync(this);

    }

    /**
    * Create new map using user's current location, then retrieve GeoJSON points
from server.
    * @param newMap
    *
    */
    @Override
    public void onMapReady(final GoogleMap newMap) {
        //LatLngBounds UCL = new LatLngBounds(new LatLng(51.5223, -0.14), new
LatLng(51.528744085048615, -0.1254415512084961));
        LatLng mapCentre = new LatLng(currentLat, currentLng); // set map centre
        map = newMap;
        map.setMyLocationEnabled(true); // display user's location on map
        //map.moveCamera(CameraUpdateFactory.newLatLngBounds(UCL, 4, 4, 0));
        map.moveCamera(CameraUpdateFactory.newLatLngZoom(mapCentre,4)); // move map
camera to user's current position.  If GPS is off this will be (0,0).
        retrieveFileFromUrl();
        map.setOnMarkerClickListener((OnMarkerClickListener) this);

    }

    /**
     * Trigger UserInfoActivity class
     */
    public void getUserInfo() {
        // Using intents, begin class UserInfoActivity and request result.
        Intent intent = new Intent();
        intent.setClassName("uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz",
"uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.UserInfoActivity");
        this.startActivityForResult(intent, GET_USERINFO_REQUEST); // result request
code is passed to new activity
    }

    /**
     * Update tvScore to show user's current score
     */
    public void showScore(){
        tvScore.setText(Integer.toString(user.getScore()));
    }

    public User getUser(){
        return user;
    }


    /**
     * Check the number of questions a user has answered.
     * When user has answered all questions, trigger alert dialog to inform them how
many questions they answered correctly.
     * Submit score to database, and exit application.
     *
     * Adapted from:
     * // and http://stackoverflow.com/questions/13377300/how-to-show-dialog-from-a-
static-method
```

```
     */
    public void checkQuestionCount(){
        Log.i("questionCount", Integer.toString(user.getQuestionCount()) + " " +
questionList.size());
        if (user.getQuestionCount() == questionList.size()){ // questionList.size()
= total number of questions
            Log.i("quizComplete", "Quiz Complete");
            AlertDialog.Builder quizComplete = new AlertDialog.Builder(this);
            quizComplete.setTitle("Quiz Complete");
            quizComplete.setMessage("Congratulations, you have answered " +
user.getScore() + "/" + questionList.size() + " questions correctly!");
            quizComplete.setNegativeButton(R.string.button_exit, (
                    new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface quizComplete, int id) {
                            submitScorePost();  // submit score to database
                            finish(); // exit application
                        };
                    }
            ));
            quizComplete.show();  // show dialog
        }
    }

    /**
     * initialise the custom location listener to track user's position
     */
    private void initLocationListener(){
        Log.i("listener", "listener enabled");
        // set up location listener
        locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
        CustomLocationListener customLL = new CustomLocationListener();
        customLL.parentActivity = this;
        currentLat = customLL.currentLat;
        currentLng = customLL.currentLng;

        Log.i("currentlat", Double.toString(currentLat));
        Log.i("currentlng", Double.toString(currentLng));

        // set up the location manager and listener
        locationManager.requestLocationUpdates(
                LocationManager.GPS_PROVIDER,
                MINIMUM_TIME_BETWEEN_UPDATE,
                MINIMUM_DISTANCECHANGE_FOR_UPDATE,
                customLL
        );
    }

    /**
     * retrieve GeoJSON file from server
     */
    private void retrieveFileFromUrl() {
        String mGeoJsonUrl
                //=
"http://developer.cege.ucl.ac.uk:30522/teaching/user16/createGeoJSON.php";
                =
"http://developer.cege.ucl.ac.uk:30522/teaching/user16/createQuestionsGeoJSON.php";
        DownloadGeoJsonFile downloadGeoJsonFile = new DownloadGeoJsonFile();
        downloadGeoJsonFile.execute(mGeoJsonUrl);
    }

    /**
     * Defines actions when marker is clicked:
     *      If user is in proximity to question but question has not been answered,
trigger getQuestion()
     *      If user has answered question, inform them of this using a dialog
     *      If neither is true, tell user to approach the point to answer the
question
     * @param marker marker clicked
     * @return
     */
    @Override
    public boolean onMarkerClick(final Marker marker) {

        Log.i("marker", marker.getId());

        AlertDialog.Builder dialog = new AlertDialog.Builder(this);
        dialog.setTitle("Question: " + marker.getTitle());
        dialog.setNegativeButton(R.string.button_close, null);

        int qID = Integer.parseInt(marker.getTitle());

        for (Question q : questionList) {
            if (q.getId() == qID) {
                if (q.getProximity() == true && q.getAnswered() == false) {
                    Log.i("getQuestion", Integer.toString(qID) + " " +
Boolean.toString(q.getAnswered()));
                    q.setInProgress(true);
                    initQuestion(q);
                    break;
                } if (q.getAnswered() == true) {
                    Log.i("getQuestion", Integer.toString(qID) + " " +
Boolean.toString(q.getAnswered()));
                    dialog.setMessage("Sorry, you have already answered this
question");
                    dialog.show();
                } else {
                    dialog.setMessage("Approach point to answer question");
                    dialog.show();
                }

            }
        }

        return true;
    }

    /**
     * Extract question information from GeoJSON feature and add to ArrayList
questionList
     * @param feature GeoJSON feature
     * @return q question object
     */
    private Question addQuestion(GeoJsonFeature feature) {
        // extract properties from feature and assign to correct local variables
        int id = Integer.parseInt(feature.getProperty("qid"));
        String pointName = feature.getProperty("point_name");
        String question = feature.getProperty("question");
        String answer1 = feature.getProperty("answer1");
        String answer2 = feature.getProperty("answer2");
        String answer3 = feature.getProperty("answer3");
        String answer4 = feature.getProperty("answer4");
        int answerCorrect = Integer.parseInt(feature.getProperty("answer_correct"));
        Log.i("addQuestionCorrect", feature.getProperty("answer_correct"));
        double lat = ((GeoJsonPoint)
feature.getGeometry()).getCoordinates().latitude;
        double lng = ((GeoJsonPoint)
feature.getGeometry()).getCoordinates().longitude;

        // create new question object from feature properties
        Question q = new Question(id, pointName, question, answer1, answer2,
answer3, answer4, answerCorrect, lat, lng);
```

```
        questionList.add(q); // add question to ArrayList
        return q;
    }


    /**
     * Get question and associated variables and pass them to QuestionActivity using
intents
     * @param q question object
     *
     * Adapted from
https://developer.android.com/training/basics/intents/result.html#ReceiveResult
     */
    protected void initQuestion(Question q) {
        Intent intent = new Intent();
        intent.setClassName("uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz",
"uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.QuestionActivity");
        Log.i("userIDma", Integer.toString(user.getID()));
        intent.putExtra("qid", Integer.toString(q.getId()));
        intent.putExtra("uid", user.getID());
        intent.putExtra("question", q.getQuestion());
        intent.putExtra("answer1", q.getAnswer1());
        intent.putExtra("answer2", q.getAnswer2());
        intent.putExtra("answer3", q.getAnswer3());
        intent.putExtra("answer4", q.getAnswer4());
        intent.putExtra("answerCorrect", q.getAnswerCorrect());

        Log.i("getQuestionID", Integer.toString(q.getId()));
        Log.i("getQuestionAnswer", q.getAnswerCorrect());

        this.startActivityForResult(intent, GET_QUESTION_REQUEST);  // start
activity, requesting result using result code
    }

    /**
     * Triggered by method setResult() method in activity started using
startActivityForResult()
     * @param requestCode request identifier
     * @param resultCode result identifier
     * @param data data received
     *
     * Adapted from
https://developer.android.com/training/basics/intents/result.html#ReceiveResult
     */
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        // Check which request the method is responding to
        // QuestionActivity response
        if (requestCode == GET_QUESTION_REQUEST) {
            // Make sure request was successful
            if(resultCode == Activity.RESULT_OK){
                String result = data.getStringExtra("result");
                Log.i("activityResultAnswer", result);

                //  if user answered question correctly, increase user score by 1
                if (Boolean.parseBoolean(result) == true){
                    user.setScore(1);
                    Log.i("userScore", Integer.toString(user.getScore()));
                    showScore();
                }

                if (resultCode == Activity.RESULT_CANCELED) {
                Log.i("activityResult", "result failed");
                }

                user.setQuestionCount(1); // increase user question count by 1
                checkQuestionCount(); // check to see whether or not user has
answered all questions
            }
        }

        // UserInfoActivity response
        if (requestCode == GET_USERINFO_REQUEST) {
            int userID = data.getIntExtra("result", -10); // get userID generated by
database
            Log.i("activityResultUser", Integer.toString(userID));
            user = new User(userID); // create new user using userID
        }
    }

    /**
     * initialise the question markers from GeoJSON features
     */
    private void initialiseMarkers() {
        // Iterate over all the features stored in the layer
        for (GeoJsonFeature feature : mLayer.getFeatures()) {
            Log.i("markerInit", "initialised");
            // Check if qid (questionID) property exists
            if (feature.hasProperty("qid")) {
                Question q = addQuestion(feature); // call method addQuestion on
feature

                // get questionID and create marker
                Integer id = Integer.parseInt(feature.getProperty("qid"));
                BitmapDescriptor pointIcon;
                pointIcon =
BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_AZURE);

                // Create a new point style
                GeoJsonPointStyle pointStyle = new GeoJsonPointStyle();

                // Set options for the point style
                pointStyle.setIcon(pointIcon);
                pointStyle.setTitle(Integer.toString(id));

                // Assign the point style to the feature
                feature.setPointStyle(pointStyle);

                // refresh marker colours to reflect proximity flags
                updateMarkerColour();
            }
        }
    }

    /**
     * Update the marker colours based on the following:
     *  Azure:  question not answered, user not in proximity
     *  Purple:  question not answered, user in proximity
     *  Red:  question answered, user answered incorrectly
     *  Azure:  question answered, user answered correctly
     */
    public static void updateMarkerColour() {
        // iterate through GeoJSON features on map
        for (GeoJsonFeature feature : mLayer.getFeatures()) {
            BitmapDescriptor pointIcon;
            //Log.i("markerColours", feature.toString());
            if (feature.hasProperty("qid")) {
                Log.i("markerColours", feature.getProperty("qid"));
                // find question associated with marker and check flags
                for (Question q : mapActivity.questionList) {
                    if (Integer.parseInt(feature.getProperty("qid")) == q.getId()) {
                        Log.i("markerColours", "found question");
                        if (q.getAnswered() == true && (q.getCorrect() == true)) {
                            // Get the icon for the feature
                            pointIcon =
BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_GREEN);
                        } else if (q.getAnswered() == true && (q.getCorrect() ==
false)) {
                            pointIcon =
BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_RED);
                        } else if (q.getAnswered() == false && (q.getProximity() ==
true)) {
                            pointIcon =
BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_VIOLET);
                        } else {
                            pointIcon =
BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_AZURE);
                        }
                        // Get marker point style
                        GeoJsonPointStyle pointStyle = feature.getPointStyle();

                        // Update the point style icon
                        pointStyle.setIcon(pointIcon);

                        // Assign the point style to the feature
                        feature.setPointStyle(pointStyle);
                    }
                }
            }
        }
    }

    /**
     * Submit the user's score to the database
     */
    private void submitScorePost() {
        // create an asynchronous operation that will take these values
        // and send them to the server

        MapActivity.SendHttpRequestTask sfd = new MapActivity.SendHttpRequestTask();
        try {

            String urlParameters =
                    "uid=" + URLEncoder.encode(Integer.toString(user.getID()), "UTF-
8") +
                            "&score=" +
URLEncoder.encode(Integer.toString(user.getScore()), "UTF-8");

            sfd.execute(urlParameters);
        }
        catch             (UnsupportedEncodingException e){}

    }

    /**
     * Download GeoJSON file
     */
    private class DownloadGeoJsonFile extends AsyncTask<String, Void, JSONObject> {
        protected JSONObject doInBackground(String... params) {
            try {
                // Open a stream from the URL
                InputStream stream = new URL(params[0]).openStream();

                String line;
                StringBuilder result = new StringBuilder();
                BufferedReader reader = new BufferedReader(new
InputStreamReader(stream));
```

```
                while ((line = reader.readLine()) != null) {
                    Log.i("writeLine", "line written");
                    // Read and save each line of the stream
                    result.append(line);
                    Log.i("result", result.toString());
                }

                // Close the stream
                reader.close();
                stream.close();

                // Convert result to JSONObject
                Log.i("JSONresult", result.toString());
                return new JSONObject(result.toString());
            } catch (IOException e) {
                Log.e(mLogTag, "GeoJSON file could not be read");
            } catch (JSONException e) {
                Log.e(mLogTag, "GeoJSON file could not be converted to a
JSONObject");
            }
            return null;
        }

        /**
         * Once download executed, create new GeoJsonLayer
         * @param jsonObject
         */
        @Override
        protected void onPostExecute(JSONObject jsonObject) {
            if (jsonObject != null) {
                // Create a new GeoJsonLayer, pass in downloaded GeoJSON file as
JSONObject
                mLayer = new GeoJsonLayer(map, jsonObject);
                // Add the layer onto the map
                initialiseMarkers();
                mLayer.addLayerToMap();
            }
        }
    }

    /**
     * Class SendHttpRequestTask
     * Communicates with web server to connect with database and upload score
     */
    private class SendHttpRequestTask extends AsyncTask<String, Void, String> {

        @Override
        protected void onPreExecute() {

        }

        @Override
        protected String doInBackground(String... params) {
            URL url;
            String urlParams = params[0];
            String
targetURL="http://developer.cege.ucl.ac.uk:30522/teaching/user16/submitScore.php";

            HttpURLConnection connection = null;
            try {
                //Create connection
                url = new URL(targetURL);
                connection = (HttpURLConnection) url.openConnection();
                connection.setRequestMethod("POST");
                connection.setRequestProperty("Content-Type",
                        "application/x-www-form-urlencoded");

                connection.setRequestProperty("Content-Length", "" +
                        Integer.toString(urlParams.getBytes().length));
                connection.setRequestProperty("Content-Language", "en-US");

                connection.setUseCaches(false);
                connection.setDoInput(true);
                connection.setDoOutput(true);

                //Send request
                DataOutputStream wr = new DataOutputStream(
                        connection.getOutputStream());
                wr.writeBytes(urlParams);
                wr.flush();
                wr.close();

                //Get Response
                InputStream is = connection.getInputStream();
                BufferedReader rd = new BufferedReader(new InputStreamReader(is));
                String line;
                StringBuffer response = new StringBuffer();
                while ((line = rd.readLine()) != null) {
                    response.append(line);
                    response.append('\r');
                }
                rd.close();
                connection.disconnect();
                return response.toString();

            } catch (Exception e) {

                e.printStackTrace();
                return null;

            } finally {

                if (connection != null) {
                    connection.disconnect();
                }
            }
        }

        @Override
        protected void onPostExecute(String response) {


        }
    }
}
```

UserInfoActivity

```
package uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz;

import android.app.Activity;
import android.content.Intent;
import android.graphics.Color;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;


/**
 * <h2> UserInfoActivity </h2>
 *
 * Activity class for Location Based Quiz
 * Controls behaviour for activity_user_info.xml
 *
 * @author Lucille Ablett
 *
 * Adapted from code provided as part of CEGEG077 Web and Mobile GIS
 *
 */


public class UserInfoActivity extends AppCompatActivity implements
View.OnClickListener {

    /**
     * @param username username of user
     * @param email email of user
     * @param tv TextView for error messages
     */
    private String username;
    private String email;
    private TextView tv;

    //private MapActivity mapActivity = null;


    /**
     * Called when the activity is first created.
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_user_info); // set view to
activity_user_info.xml
        tv = (TextView) findViewById(R.id.webResponse);
        Log.i("userinfo", "loaded");
        ((Button) findViewById(R.id.buttonBegin)).setOnClickListener(this);

        //mapActivity = new MapActivity().getInstance();
    }

    /**
     * When click is registered in activity, this class is called to determine what
action to take
     * @param view current view
     */
    public void onClick(View view) {
        Log.i("click", "begin clicked");
        try {
            switch (view.getId())
            {
                case R.id.buttonBegin: // if buttonBegin is clicked, validate the
input
                    if (validateInput() == true){
                        Log.i("usersubmit", "submit");
                        submitDataPost(); // sumbit data to database
                    };
                    break;
                default:
                    Log.i("view",Integer.toString(view.getId()));
                    break;
            }
        } catch (Exception e){
            Log.i("click", "Error");
        }
    }

    /**
     * validate username and email inputs
     * @return
     */
    private boolean validateInput() {
        boolean valid = true;  // assume information is valid
        // retrieve information from form
        username = ((EditText) findViewById(R.id.username)).getText().toString();
        Log.i("username", username);

        email = (String) ((EditText) findViewById(R.id.email)).getText().toString();
        Log.i("email", email);

        // check validity
        if (username.length() < 6) {
            ((TextView) findViewById(R.id.tvUsername)).setText("Username must be
between 6 and 30 character");
            ((TextView)
findViewById(R.id.tvUsername)).setTextColor(Color.parseColor("#cc0000"));
            valid = false;  // not valid
        }

        if (!(email.contains("@"))){
            ((TextView) findViewById(R.id.tvEmail)).setText("Invalid email
address");
            ((TextView)
findViewById(R.id.tvEmail)).setTextColor(Color.parseColor("#cc0000"));
            valid = false;  // not valid
        }

        return valid;
    }

    /**
     *  Asynchronous operation that takes values and sends them to the server
     */
    private void submitDataPost() {

        SendHttpRequestTask sfd = new SendHttpRequestTask();
        try {

            String urlParameters =
                "username=" + URLEncoder.encode(username, "UTF-8") +
                "&email=" + URLEncoder.encode(email, "UTF-8");


            sfd.execute(urlParameters);
        }
        catch             (UnsupportedEncodingException e){}
    }

    /**
     * Class SendHttpRequestTask
     * Communicates with web server to connect with database and upload user
information
     */
    private class SendHttpRequestTask extends AsyncTask<String, Void, String> {

        @Override
        protected void onPreExecute() {

        }

        @Override
        protected String doInBackground(String... params) {
            URL url;
            String urlParams = params[0];
            String
targetURL="http://developer.cege.ucl.ac.uk:30522/teaching/user16/processUserInfo.php
";

            HttpURLConnection connection = null;
            try {
                //Create connection
                url = new URL(targetURL);
                connection = (HttpURLConnection) url.openConnection();
                connection.setRequestMethod("POST");
                connection.setRequestProperty("Content-Type",
                        "application/x-www-form-urlencoded");

                connection.setRequestProperty("Content-Length", "" +
                        Integer.toString(urlParams.getBytes().length));
                connection.setRequestProperty("Content-Language", "en-US");

                connection.setUseCaches(false);
                connection.setDoInput(true);
                connection.setDoOutput(true);

                //Send request
                DataOutputStream wr = new DataOutputStream(
                        connection.getOutputStream());
                wr.writeBytes(urlParams);
                wr.flush();
                wr.close();

                //Get Response
                InputStream is = connection.getInputStream();
                BufferedReader rd = new BufferedReader(new InputStreamReader(is));
                String line;
                StringBuffer response = new StringBuffer();
                while ((line = rd.readLine()) != null) {
                    response.append(line);
                    response.append('\r');
                }
                rd.close();
                connection.disconnect();
                return response.toString();

            } catch (Exception e) {

                e.printStackTrace();
                return null;

            } finally {
                if (connection != null) {
                    connection.disconnect();
                }
            }
        }

        /**
         * After server communication complete, validate response code.
         * @param response
         */
        @Override
        protected void onPostExecute(String response) {
            try {
                Log.i("response", response);
                response = response.trim();

                if (response.contains("-")) {
                    int r = Integer.parseInt(response);
                    if (r == -1) {
                        tv.setText("There was an error connecting to the server,
please try again later"); // database communication error
                    } else if (r == -999) {
                        tv.setText("The username or email has already been
registered, please try again");  //  username or password exists
                    }

                } else {

                    Intent returnIntent = new Intent();
                    returnIntent.putExtra("result",Integer.parseInt(response));
                    setResult(Activity.RESULT_OK,returnIntent);  // send userID back
to MapActivity

                    //MapActivity.showScore();
                    finish();
                }
            }
            catch (Exception e) {
                Log.e("userinfo_fail", e.toString());
                tv.setText("Please ensure you are connected to the internet and try
again");
            }

        }
    }
}
```

QuestionActivity

```
package uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.graphics.Color;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.telephony.TelephonyManager;
import android.util.Log;
import android.view.View;
import android.webkit.WebView;
import android.widget.Button;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;

import static
uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.MapActivity.mapActivity;
//import static
uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.MapActivity.questionList;
import static
uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz.MapActivity.updateMarkerColour;


/**
 * <h2> QuestionActivity </h2>
 *
 * Activity class for Location Based Quiz
 * Controls behaviour for activity_question.xml
 *
 * @author Lucille Ablett
 *
 * Adapted from code provided as part of CEGEG077 Web and Mobile GIS
 *
 */
public class QuestionActivity extends AppCompatActivity implements
View.OnClickListener {

    private TextView tv;
    private WebView wv;
    /*
    private TextView tvID;
    private TextView tvQuestion;
    private RadioGroup radioGroup;
    private RadioButton radioAnswer1;
    private RadioButton radioAnswer2;
    private RadioButton radioAnswer3;
    private RadioButton radioAnswer4;
    private Button buttonSubmit;*/

    /**
     * Instance variables:
     * @param answer answer to question as given by user
     * @param answerCorrect correct answer to question
     * @param correct whether or not the user answered correctly
     * @param questionID question ID
     * @param userID user ID
     */
    private String answer;
    private String answerCorrect;
    private boolean correct = false;
    private String questionID;
    private int userID;


    //private MapActivity mapActivity = new MapActivity().getInstance();

    /**
     * Called when the activity is first created.
```

```
      */
     @Override
     public void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);

         setContentView(R.layout.activity_question); // set view to
activity_question.xml
         tv = (TextView) findViewById(R.id.webResponse);
         // get information from intent and assign to appropriate variables/layout
features
         Bundle extras = getIntent().getExtras();
         questionID = (String)extras.get("qid");
         userID = extras.getInt("uid");
         Log.i("userIDqa", Integer.toString(userID));
         // assign question properties to correct layout features
         ((TextView) findViewById(R.id.tvID)).setText("Question: " + questionID);
         ((TextView)
findViewById(R.id.tvQuestion)).setText((String)extras.get("question"));

((RadioButton)findViewById(R.id.radioAnswer1)).setText((String)extras.get("answer1")
);

((RadioButton)findViewById(R.id.radioAnswer2)).setText((String)extras.get("answer2")
);

((RadioButton)findViewById(R.id.radioAnswer3)).setText((String)extras.get("answer3")
);

((RadioButton)findViewById(R.id.radioAnswer4)).setText((String)extras.get("answer4")
);

         answerCorrect = (String)extras.get("answerCorrect");

         ((Button) findViewById(R.id.buttonSubmit)).setOnClickListener(this);

     }

     /**
      * When click is registered in activity, this class is called to determine what
action to take
      * @param view current view
      */
     public void onClick(View view) {
         try {
             switch (view.getId()) {
                 case R.id.buttonSubmit:
                     //Log.i("buttontext", (String) ((Button)
findViewById(R.id.buttonSubmit)).getText());
                     //if buttonSubmit is clicked, and it's text is set to "Submit
Answer", call method submitDataPost()
                     if (((Button)
findViewById(R.id.buttonSubmit)).getText().equals("Submit Answer")) {
                         Log.i("submit", "submit1");
                         submitDataPost();
                         //Log.i("submit", "submit");

                     // if buttonSubmit is clicked, and it's text is set to
"Continue"
                     } else if (((Button)
findViewById(R.id.buttonSubmit)).getText().equals("Continue")) {
                         Intent returnIntent = new Intent();
                         returnIntent.putExtra("result",Boolean.toString(correct));
                         setResult(Activity.RESULT_OK,returnIntent);  // send correct
back to MapActivity

                         finish();  // end activity
```

```
                    }
                    break;
                }
            } catch (Exception e) {
                Log.e("submitfail", e.toString());
                }
        }

        /**
         * Validate the answer given by user
         */
        public void validateAnswer(){

            // get selected radio button from radioGroup
            RadioGroup radioGroup = (RadioGroup) findViewById(R.id.radioGroup);
            int selectedID = radioGroup.getCheckedRadioButtonId();

            // find the radio button by returned id
            RadioButton radioButton = (RadioButton) findViewById(selectedID);
            answer = (String)radioButton.getText();
            Log.i("answer", answer);
            Log.i("answercorrect", answerCorrect);
            Log.i("answercheck", Boolean.toString(answer.equals(answerCorrect)));

            // validate answer against correct answer, and inform user of the result
            if (answer.equals(answerCorrect)){
                correct = true;
                ((TextView) findViewById(R.id.answerResponse)).setText("Congratulations!
That is the correct answer!");
                ((TextView)
findViewById(R.id.answerResponse)).setTextColor(Color.parseColor("#33cc99")); //
change colour of text

            } else {
                correct = false;
                ((TextView) findViewById(R.id.answerResponse)).setText("Sorry, that is
the incorrect answer.  The correct answer is " + answerCorrect + ".");
                ((TextView)
findViewById(R.id.answerResponse)).setTextColor(Color.parseColor("#cc0000"));  //
change colour of text
            }
        }

        /**
         * submit data to server using asynchronous operation
         *
https://developer.android.com/reference/android/telephony/TelephonyManager.html
         */
        private void submitDataPost() {
            //Log.i("submit", "submit3");

            SendHttpRequestTask sfd = new SendHttpRequestTask();
            Log.i("submit", "submit5");
            try {
                //Log.i("submit", "submit4");
                String qid = questionID;
                //Log.i("questionIDa", qid);

                String question = (String) ((TextView)
findViewById(R.id.tvQuestion)).getText();
                //Log.i("questionA", question);

                validateAnswer();

                // get the imei number of the phone using TelephonyManager
                TelephonyManager telephonyManager =
    (TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);

                String imei = (String) telephonyManager.getDeviceId();

                // place values into url string for sending to server
                String urlParameters =
                    "uid=" + URLEncoder.encode(Integer.toString(userID), "UTF-8") +
                    "&qid=" + URLEncoder.encode(qid, "UTF-8") +
                    "&question=" + URLEncoder.encode(question, "UTF-8") +
                    "&answer=" + URLEncoder.encode(answer, "UTF-8") +
                    "&correct=" +
    URLEncoder.encode(Boolean.toString(correct).toUpperCase(), "UTF-8") +
                    "&imei=" + URLEncoder.encode(imei, "UTF-8");

                updateQuestion();  // update question to reflect that it has been
    answered
                updateMarkerColour();  // refresh markers to reflect whether or not
    question was answered correctly
                ((Button) findViewById(R.id.buttonSubmit)).setText("Continue");  // set
    text on buttonSubmit to "Continue"

                sfd.execute(urlParameters);
            }
            catch             (UnsupportedEncodingException e){}

        }

        /**
         * update the Answered and Correct flags for the question that was just answered
         */
        private void updateQuestion(){
            for (int i = 0; i < mapActivity.questionList.size(); i++) {
                Question q = mapActivity.questionList.get(i);
                if (q.getId() == Integer.parseInt(questionID)){
                    Log.i("updatingQuestion", "question " + questionID + " updated");
                    q.setIsAnswered(true);
                    q.setIsCorrect(correct);
                    Log.i("updatingQuestion",
    Boolean.toString(mapActivity.questionList.get(i).getAnswered()) +
    Boolean.toString(mapActivity.questionList.get(i).getCorrect()));
                    break;
                } else {
                    Log.i("updatingQuestion", "No questions updated");
                }
            }
        }


        private class SendHttpRequestTask extends AsyncTask<String, Void, String> {

            @Override
            protected void onPreExecute() {

            }

            /**
             * retrieve submitAnswer.php from web
             * @param params
             * @return
             */
            @Override
            protected String doInBackground(String... params) {
                URL url;
                String urlParams = params[0];
                String
    targetURL="http://developer.cege.ucl.ac.uk:30522/teaching/user16/submitAnswer.php";
```

```
            HttpURLConnection connection = null;
            try {
                //Create connection
                url = new URL(targetURL);
                connection = (HttpURLConnection) url.openConnection();
                connection.setRequestMethod("POST");
                connection.setRequestProperty("Content-Type",
                        "application/x-www-form-urlencoded");

                connection.setRequestProperty("Content-Length", "" +
                        Integer.toString(urlParams.getBytes().length));
                connection.setRequestProperty("Content-Language", "en-US");

                connection.setUseCaches(false);
                connection.setDoInput(true);
                connection.setDoOutput(true);

                //Send request
                DataOutputStream wr = new DataOutputStream(
                        connection.getOutputStream());
                wr.writeBytes(urlParams);
                wr.flush();
                wr.close();

                //Get Response
                InputStream is = connection.getInputStream();
                BufferedReader rd = new BufferedReader(new InputStreamReader(is));
                String line;
                StringBuffer response = new StringBuffer();
                while ((line = rd.readLine()) != null) {
                    response.append(line);
                    response.append('\r');
                }
                rd.close();
                connection.disconnect();
                return response.toString();

            } catch (Exception e) {

                e.printStackTrace();
                return null;

            } finally {

                if (connection != null) {
                    connection.disconnect();
                }
            }
        }

        @Override
        protected void onPostExecute(String response) {
            // used for debugging purposes
            //tv.setText(response);
        }
    }
}
```

CustomLocationListener

```
package uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz;

import android.app.Activity;
```

```java
import android.content.Intent;
import android.location.Location;
import android.location.LocationListener;
import android.os.Bundle;
import android.util.Log;
import android.widget.Toast;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.model.LatLng;

/**
 * <h2> Custom Location Listener </h2>
 *
 * Defines Custom Location Listener utilised in Location Based Quiz
 *
 * This class monitors the location of the user and calculates the distance between
the user and the location of the each quiz question.
 * When the user is within proximity of a question, the QuestionActivity is
triggered.
 *
 * @author Lucille Ablett
 *
 * Adapted from code provided as part of CEGEG077 Web and Mobile GIS
 *
 */


public class CustomLocationListener extends MapActivity implements LocationListener
{

    /**
     * Instance variables:
     * @param parentActivity instance of the MapActivity class, which is the parent
class
     * @param currentLat current latitude of the user
     * @param currentLng current longitude of the user
     */
    public MapActivity parentActivity;
    public double currentLat;
    public double currentLng;

    /**
     * Method called when change in position is detected by network provider
     * Monitors current position, and calculates the distance from this location to
each quiz question.
     * If the user is within 20 m of the point, the point will turn purple,
notifying the user that the question can be answered if the marker is clicked.
     * If the user is within 10 m of the point, and the question has not been
answered, QuestionActivity is automatically triggered for that question.
     *
     * @param location Current location of user
     */
    public void onLocationChanged(Location location) {

        //Toast.makeText(parentActivity.getBaseContext(), "You have moved to:
Latitude/longitude:"+location.getLatitude()+ " " + location.getLongitude(),
Toast.LENGTH_LONG).show();
        currentLat = location.getLatitude();
        currentLng = location.getLongitude();

        LatLng mapCentre = new LatLng(currentLat, currentLng); // set map centre
        mapActivity.map.moveCamera(CameraUpdateFactory.newLatLng(mapCentre));

        // measure distance between current and question locations
        for (int i = 0; i < parentActivity.questionList.size(); i++) {
            Question q = parentActivity.questionList.get(i);
            Location fixedLoc = new Location("one");

            double lat = q.getLatitude();
            double lng = q.getLongitude();

            fixedLoc.setLatitude(lat);
            fixedLoc.setLongitude(lng);
            Log.i("locationlat", lat + " " + location.getLatitude());
            Log.i("locationlng", lng + " " + location.getLongitude());

            // use Android method distanceTo() to calculate distance
            float distance = location.distanceTo(fixedLoc);
            Log.i("distance", Float.toString(distance));

            // if distance is <20, set the proximity variable for question q to
'true', then refresh the markers in MapActivity.
            if (distance < 20) {
                q.setProximity(true);
                updateMarkerColour();
            }

            // if distance is <10, and the user has not already answered the
question, trigger getQuestion() method on question q.
            if (distance < 10 && q.getAnswered()==false && mapActivity.getUser() !=
null && q.getInProgress()==false){
                q.setInProgress(true);
                mapActivity.initQuestion(q);

            }

            // if distance is >20, set the question proximity to 'false' and refresh
markers in MapActivity.
            if (distance > 20) {
                q.setInProgress(false);
                q.setProximity(false);
                updateMarkerColour();
            }
        }
    }

    /**
     * Create toast message when network location disabled
     * @param s
     */
    public void onProviderDisabled(String s) {
        Toast.makeText(parentActivity.getBaseContext(),"Please turn on GPS to answer
questions",Toast.LENGTH_LONG).show();
    }

    /**
     * Create toast message when network location enabled
     * @param s
     */
    public void onProviderEnabled(String s) {
        Toast.makeText(parentActivity.getBaseContext(),"GPS enabled.  Approach
points to answer questions.",Toast.LENGTH_LONG).show();
    }

    /**
     * Method required by class, but not utilised.
     */
    public void onStatusChanged(String arg0, int arg1, Bundle arg2) {
        // TODO Auto-generated method stub
    }
}
```

User

```
package uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz;

/**
 * <h2> User </h2>
 *
 * Defines User object utilised in Location Based Quiz
 *
 * @author Lucille Ablett
 *
 *
 */

public class User {

    /**
    * Instance variables:
    * @param uid user ID from database
    * @param score number of questions correctly answered
    * @param questionsAttempted number of questions attemted by user
    */
    private int uid;
    private int score = 0;
    private int questionsAttempted = 0;

    /**
     * Class constructor, creates new user instance
     * @param u constructor parameter passed to instance variable uid
     */
    public User(int u) {
        this.uid = u;
    }

    /**
     * returns userID number
     * @return uid
     */
    public int getID() {
        return uid;
    }

    /**
     * adds the number specified in the method parameters to the users's score
     * @param s number to increase score by
     */
    public void setScore(int s) {
        score = score + s;
    }

    /**
     * returns the user's score
     * @return
     */
    public int getScore() {
        return score;
    }

    /**
     * adds the number specified in the method paramteres to the users total number
of attempted questions
     * @param s number to increase questionsAttempted by
     */
    public void setQuestionCount(int s) {
        questionsAttempted = questionsAttempted + s;
    }

    /**
     * returns the number of questions the user has attempted
     * @return questionsAttempted
     */
    public int getQuestionCount() {
        return questionsAttempted;
    }
}
```

Question

```
package uk.ac.ucl.cege.cegeg077.uceslra.locationbasedquiz;

import android.util.Log;

/**
 * <h2> Question </h2>
 *
 * Defines Question object utilised in Location Based Quiz
 *
 * @author Lucille Ablett
 *
 *
 */

public class Question {

    /**
     * Instance variables:
     * @param id id number of quiz question
     * @param pointName alphanumeric question identifier
     * @param question quiz question
     * @param answer1 first potential answer to quiz question
     * @param answer2 second potential answer to quiz question
     * @param answer3 third potential answer to quiz question
     * @param answer4 fourth potential answer to quiz question
     * @param answerCorrect correct answer to quiz question
     * @param latitude latitude of quiz question
     * @param longitude longitude of quiz question
     * @param isInProximity whether or not the user is within 20 m of the quiz
question
     * @param IsAnswered whether or not the user has answered the qiz question
     * @param IsCorrect whether or not the user answered the question correctly
     */

    private int id;
    private String pointName;
    private String question;
    private String answer1;
    private String answer2;
    private String answer3;
    private String answer4;
    private int answerCorrect;
    private double latitude;
    private double longitude;
    private boolean isInProximity = false;
    private boolean isAnswered = false;
    private boolean isCorrect = false;
    private boolean inProgress = false;

    /**
```

```
     * Class constructor with the following parameters that are passed to instance
variables
     * @param qid id number of quiz question
     * @param name alphanumeric question identifier
     * @param q quiz question
     * @param a1 first potential answer to quiz question
     * @param a2 second potential answer to quiz question
     * @param a3 third potential answer to quiz question
     * @param a4 fourth potential answer to quiz question
     * @param aC correct answer to quiz question
     * @param lat latitude of quiz question
     * @param lng longitude of quiz question
     */
    public Question(int qid, String name, String q, String a1, String a2, String a3,
String a4, int aC, double lat, double lng){
        this.id = qid;
        this.pointName = name;
        this.question = q;
        this.answer1 = a1;
        this.answer2 = a2;
        this.answer3 = a3;
        this.answer4 = a4;
        this.answerCorrect = aC;
        this.latitude = lat;
        this.longitude = lng;
    }

    /**
     * returns quiz question ID
     * @return id
     */
    public int getId() {
        return id;
    }

    /**
     * returns quiz questions name
     * @return pointName
     */
    public String getName () {
        return pointName;
    }

    /**
     * returns quiz question
     * @return question
     */
    public String getQuestion() {
        return question;
    }

    /**
     * returns first potential answer
     * @return answer1
     */
    public String getAnswer1() {
        Log.i("answer1", answer1);
        return answer1;
    }

    /**
     * returns second potential answer
     * @return answer2
     */
    public String getAnswer2() {
        Log.i("answer2", answer2);
        return answer2;
    }

    /**
     * returns third potential answer
     * @return answer3
     */
    public String getAnswer3() {
        Log.i("answer3", answer3);
        return answer3;
    }

    /**
     * returns fourth potential answer
     * @return answer4
     */
    public String getAnswer4() {
        Log.i("answer4", answer4);
        return answer4;
    }

    /**
     * returns correct
     * @return answerCorrect
     */
    public String getAnswerCorrect() {

        String aC = "";
        // perform switch statement on answerCorrect to return text value of correct
answer
        switch (answerCorrect) {
            case 1:
                aC = answer1;
                break;
            case 2:
                aC = answer2;
                break;
            case 3:
                aC = answer3;
                break;
            case 4:
                aC = answer4;
                break;
            default:
                break;

        }
        return aC;
    }

    /**
     * return isInProximity boolean
     * @return isInProximity
     */
    public boolean getProximity() {
        return isInProximity;
    }

    /**
     * return isAnswered boolean
     * @return isAnswered
     */
    public boolean getAnswered() {
        return isAnswered;
    }
    /**
     * return whether or not question was answered correctly
     * @return isCorrect
     */
    public boolean getCorrect() {
        return isCorrect;
    }

    /**
     * return question longitude
     * @return longitude
     */
    public double getLongitude() {
        return longitude;
    }

    /**
     * return question latitude
     * @return latitude
     */
    public double getLatitude() {
        return latitude;
    }

    /**
     * set proximity flag for question
     * @param proximity if true, user is within 20 m of question point
     */
    public void setProximity(boolean proximity) {
        isInProximity = proximity;
    }

    /**
     * set answered flag
     * @param a if true, user has answered question
     */
    public void setIsAnswered(boolean a) {
        isAnswered = a;
    }

    /**
     * records whether or not user answered question correctly
     * @param c if true, user answered question correctly
     */
    public void setIsCorrect(boolean c) {
        isCorrect = c;
    }

    public boolean getInProgress() {
        return inProgress;
    }

    public void setInProgress(boolean p) {
        inProgress = p;
    }
}
```