INSTITUTE of TECHNOLOGY CARLOW

Institiúid Teicneolaíochta Cheatharlach

TECHNICAL MANUAL

Detecting Disease in Plants

Lecturer

Nigel Whyte

Submission date

12/02/19

(Student) – Darran Gahan – C00098391

C00098391@itcarlow.ie

Contents

Mobile Application

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.c00098391.planttracker">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

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

    <uses-feature android:name="android.hardware.camera2.full" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:largeHeap="true"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme"
        tools:ignore="GoogleAppIndexingWarning">
        <activity
            android:name=".InputDetails"
            android:label="Create Experiment"
            android:screenOrientation="portrait"/>
        <activity
            android:name=".ContinueExperiment"
            android:label="Select Experiment"
            android:screenOrientation="portrait" />
        <activity
            android:name=".DiseaseResult"
            android:label="Disease Results"
            android:screenOrientation="portrait" />
        <activity
            android:name=".DiseaseAnalysis"
            android:label="Disease Analysis"
            android:screenOrientation="portrait" />
        <activity
            android:name=".DetectDisease"
            android:label="Detect Disease"
            android:screenOrientation="portrait" />
        <activity
            android:name=".TextResults"
            android:label="Text Results"
            android:screenOrientation="portrait" />
        <activity
            android:name=".CaptureText"
            android:label="Text Reader"
            android:screenOrientation="portrait" />
        <activity android:name=".LoginActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".MainActivity"
            android:label="Plant Tracker"
            android:screenOrientation="portrait" />
    </application>

</manifest>
```

MainActivity.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * author Darran Gahan
5    *
6    * Class for MainActivity for Plant Tracker app
7    * Class gives the user the option to Start New Experiment, Continue Experiment.
8    */
9
10  import android.Manifest;
11  import android.content.Context;
12  import android.content.Intent;
13  import android.content.pm.PackageManager;
14  import android.os.AsyncTask;
15  import android.support.annotation.NonNull;
16  import android.support.v4.app.ActivityCompat;
17  import android.support.v4.content.ContextCompat;
18  import android.support.v7.app.AppCompatActivity;
19  import android.os.Bundle;
20  import android.util.Log;
21  import android.view.View;
22  import android.widget.Button;
23  import android.widget.Toast;
24
25  import org.json.JSONArray;
26  import org.json.JSONException;
27  import org.json.JSONObject;
28
29  import java.io.BufferedReader;
30  import java.io.BufferedWriter;
31  import java.io.InputStream;
32  import java.io.InputStreamReader;
33  import java.io.OutputStream;
34  import java.io.OutputStreamWriter;
35  import java.net.HttpURLConnection;
36  import java.net.URL;
37  import java.net.URLEncoder;
38  import java.nio.charset.StandardCharsets;
39  import java.util.ArrayList;
40  import java.util.Iterator;
41  import java.util.List;
42
43  public class MainActivity extends AppCompatActivity {
44
45      // Attributes for main activity
46      Button btnStartExp, btnContinue;
47
48      private static final int PERMISSIONS_REQUEST = 1;
49
50      // Instance variables for GetUserExperimentData (Async class)
51      static InputStream inputStream = null;
52      static String json;
53      static JSONObject jObj = null;
54      static String error = "";
55      ArrayList<String> exps = new ArrayList<>();
56      ArrayList<String> expIds = new ArrayList<>();
57
58      UserData userData = new UserData();
59      // User details
60      String userId;
61      String username;
62
63
64      @Override
65      protected void onCreate(Bundle savedInstanceState) {
66          super.onCreate(savedInstanceState);
67          setContentView(R.layout.activity_main);
68
69          chechAndRequestPermissions();
70
71          btnStartExp = findViewById(R.id.btnStartExp);
72          btnContinue = findViewById(R.id.btnContinue);
73
```

```
74          final String username = getIntent().getStringExtra("username");
75
76
77          // UserData class
78          userData.setUsername(username);
79
80
81
82          String[] user = new String[1];
83          user[0] = username;
84
85          GetUserExperimentData userData = new GetUserExperimentData();
86          userData.execute(user);
87
88
89          // OnClick for Start New Experiment
90          btnStartExp.setOnClickListener(new View.OnClickListener() {
91              @Override
92              public void onClick(View view) {
93                  Intent intent = new Intent(MainActivity.this,
94                          com.c00098391.planttracker.CaptureText.class);
95                  //intent.putExtra("username", username);
96                 // intent.putExtra("userid", userId);
97                  startActivity(intent);
98              }
99          });
100
101         // OnClick for Continue Experiment
102         btnContinue.setOnClickListener(new View.OnClickListener() {
103             @Override
104             public void onClick(View view) {
105                 Intent intent = new Intent(MainActivity.this,
106                         com.c00098391.planttracker.ContinueExperiment.class);
107                 intent.putExtra("username", username);
108                 intent.putExtra("exps", exps);
109                 intent.putExtra("userid", userId);
110                 intent.putExtra("expids", expIds);
111                 startActivity(intent);
112             }
113         });
114
115     }
116
117
118     // Async task for getting user experiment data, this is done here because it was
        causing
119     // problems when done in ContinueExperiment.java.
120     public class GetUserExperimentData extends AsyncTask<String, Void, JSONObject> {
121
122         @Override
123         protected JSONObject doInBackground(String... args){
124
125             try{
126                 URL url = new
                    URL("http://www.c0009839.candept.com/API/getExperimentDetails.php");
127
128                 // put params in a JSON Object
129                 JSONObject dataParams = new JSONObject();
130                 dataParams.put("username", args[0]);
131
132                 // Set up connection
133                 HttpURLConnection conn = (HttpURLConnection) url.openConnection();
134                 conn.setReadTimeout(15000);
135                 conn.setConnectTimeout(15000);
136                 conn.setRequestMethod("POST");
137                 conn.setDoInput(true);
138                 conn.setDoOutput(true);
139
140                 //send data
141                 OutputStream os = conn.getOutputStream();
142                 BufferedWriter writer = new BufferedWriter(
143                         new OutputStreamWriter(os, StandardCharsets.UTF_8));
144                 writer.write(getPostDateString(dataParams));
```

```
145
146                 writer.flush();
147                 writer.close();
148                 os.close();
149
150                 // Get Response
151                 int responseCode = conn.getResponseCode();
152                 error = String.valueOf(conn.getResponseCode());
153
154                 if (responseCode == HttpURLConnection.HTTP_OK){
155                     inputStream = conn.getInputStream();
156                     BufferedReader in  = new BufferedReader(new
                        InputStreamReader(inputStream));
157                     StringBuilder sb = new StringBuilder();
158                     String line;
159
160                     while(null!= (line = in.readLine())){
161                         sb.append(line).append("\n");
162                     }
163                     in.close();
164                     inputStream.close();
165                     json = sb.toString();
166                     Log.i("API Camera: ", json);
167                 }
168                 else{
169                     Log.e("Buffer Error", "Error Getting Result " +responseCode);
170                 }
171                 try{
172                     jObj = new JSONObject(json);
173                     jObj.put("error_code", error);
174                 }catch(JSONException e){
175                     Log.e("JSON Parser", "Error Parsing Data " + e.toString());
176                 }
177             }catch(Exception e){
178                 Log.e("Exception: ", "Overall Try Block " + e.toString());
179             }
180             return jObj;
181         }// end of doInBackground
182
183         @Override
184         protected void onPostExecute(JSONObject result){
185
186             try {
187
188                 if (result != null){
189
190                     String dataFound = result.getString("message");
191                     if (dataFound.equals("Data found")){
192
193                         userId = result.getString("userid");
194                         userData.setUserid(userId);
195                         // username = result.getString("username");
196
197                         JSONArray jsonArray = result.getJSONArray("exp");
198
199                         // Loop gets details to display to user and stores them in
                            the exps array
200                         // also gets the experiment ids and stores them in the
                            expIds array to
201                         // enable me to retrieve correctly the correct experiment
                            details
202                         for (int i = 0; i <jsonArray.length();i++){
203                             String rep =
                                jsonArray.getJSONObject(i).getString("replicant");
204                             String expt =
                                jsonArray.getJSONObject(i).getString("expt");
205                             String treatment =
                                jsonArray.getJSONObject(i).getString("treatment");
206                             String expId = jsonArray.getJSONObject(i).getString("id");
207                             exps.add(rep +  " " + treatment + " " + expt);
208                             expIds.add(expId);
209                         }
210                     }else{
```

```
                        Toast.makeText(getApplicationContext(), result.getString(
                                "message"), Toast.LENGTH_LONG).show();
                    }
                }else{
                    Toast.makeText(getApplicationContext(),
                            "Unable to retrieve data from the server",
                            Toast.LENGTH_LONG).show();
                }
            }catch(JSONException e){
                e.printStackTrace();
            }
        }// End of onPostExecute
    }// End of GetUserExperimentData

    // Turn json object to string for post
    public String getPostDateString(JSONObject params) throws Exception{

        StringBuilder result = new StringBuilder();
        boolean first = true;
        Iterator<String> itr = params.keys();

        while(itr.hasNext()){
            String key = itr.next();
            Object value = params.get(key);
            if(first){
                first = false;
            }else{
                result.append("&");
            }
            result.append(URLEncoder.encode(key, "UTF-8"));
            result.append("=");
            result.append(URLEncoder.encode(value.toString(), "UTF-8"));
        }
        return result.toString();
    }

    // Static block to load the OpenCv lib.
    static {
        System.loadLibrary("opencv_java3");
    }

    // Check if required permissions have been granted
    private boolean chechAndRequestPermissions(){

        int permissionCAMERA = ContextCompat.checkSelfPermission(
                this, Manifest.permission.CAMERA);

        int writeStoragePermission = ContextCompat.checkSelfPermission(
                this, Manifest.permission.WRITE_EXTERNAL_STORAGE);

        int readStoragePermission = ContextCompat.checkSelfPermission(
                this, Manifest.permission.READ_EXTERNAL_STORAGE);

        int fineLocationPermission = ContextCompat.checkSelfPermission(
                this, Manifest.permission.ACCESS_FINE_LOCATION);

        int courseLocationPermission = ContextCompat.checkSelfPermission(
                this, Manifest.permission.ACCESS_COARSE_LOCATION);


        List<String> listPermissionsNeeded = new ArrayList<>();
        if (permissionCAMERA != PackageManager.PERMISSION_GRANTED){
            listPermissionsNeeded.add(Manifest.permission.CAMERA);

        }
        if (writeStoragePermission != PackageManager.PERMISSION_GRANTED) {
            listPermissionsNeeded.add(Manifest.permission.WRITE_EXTERNAL_STORAGE);
        }
        if (readStoragePermission != PackageManager.PERMISSION_GRANTED) {
            listPermissionsNeeded.add(Manifest.permission.READ_EXTERNAL_STORAGE);
        }
        if(fineLocationPermission != PackageManager.PERMISSION_GRANTED){
            listPermissionsNeeded.add(Manifest.permission.ACCESS_FINE_LOCATION);
```

```
283             }
284             if(courseLocationPermission != PackageManager.PERMISSION_GRANTED){
285                 listPermissionsNeeded.add(Manifest.permission.ACCESS_COARSE_LOCATION);
286             }
287
288             if(!listPermissionsNeeded.isEmpty()){
289                 ActivityCompat.requestPermissions(
290                         this, listPermissionsNeeded.toArray(new
                            String[listPermissionsNeeded.size()]),
291                         PERMISSIONS_REQUEST);
292                 return false;
293             }
294             return true;
295         }
296
297         @Override
298         public void onRequestPermissionsResult(int requestCode, @NonNull String[]
            permissions,
299                                                @NonNull int[] grantResults){
300             if (requestCode == PERMISSIONS_REQUEST) {
301                 if (grantResults[0] == PackageManager.PERMISSION_DENIED
302                         || grantResults[1] == PackageManager.PERMISSION_DENIED
303                         || grantResults[2] == PackageManager.PERMISSION_DENIED) {
304                     Toast.makeText(MainActivity.this,
305                             "Sorry, you cannot use this app without granting all the
                                permissions", Toast.LENGTH_SHORT)
306                             .show();
307                     finish();
308                 }
309                 else{
310                     Toast.makeText(MainActivity.this,
311                             "Permissions Granted", Toast.LENGTH_LONG)
312                             .show();
313                 }
314             }
315         }
316     }
317
```

LoginActivity.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * Student Name: Darran Gahan
5    * Student Number: C00098391
6    *
7    * Class is to handle the login of a user.
8    */
9
10  import android.annotation.SuppressLint;
11  import android.content.Intent;
12  import android.os.AsyncTask;
13  import android.support.v7.app.AppCompatActivity;
14  import android.os.Bundle;
15  import android.text.TextUtils;
16  import android.util.Log;
17  import android.view.View;
18  import android.widget.Button;
19  import android.widget.EditText;
20  import android.widget.Toast;
21
22  import org.json.JSONException;
23  import org.json.JSONObject;
24
25  import java.io.BufferedReader;
26  import java.io.BufferedWriter;
27  import java.io.InputStream;
28  import java.io.InputStreamReader;
29  import java.io.OutputStream;
30  import java.io.OutputStreamWriter;
31  import java.net.HttpURLConnection;
32  import java.net.URL;
33  import java.net.URLEncoder;
34  import java.nio.charset.StandardCharsets;
35  import java.util.Iterator;
36
37  public class LoginActivity extends AppCompatActivity {
38
39      // Async class variables
40      static InputStream inputStream = null;
41      static String json;
42      static JSONObject jObj = null;
43      static String error = "";
44
45      // View attributes
46      Button btnSignIn;
47      EditText etName, etPassword;
48
49      @Override
50      protected void onCreate(Bundle savedInstanceState) {
51          super.onCreate(savedInstanceState);
52          setContentView(R.layout.activity_login);
53
54          // Edit Texts
55          etName = findViewById(R.id.etName);
56          etPassword = findViewById(R.id.etPassword);
57          // Buttons
58          btnSignIn = findViewById(R.id.btnSignIn);
59
60          // onClick listener
61          btnSignIn.setOnClickListener(new View.OnClickListener() {
62              @Override
63              public void onClick(View view) {
64
65                  if (TextUtils.isEmpty(etName.getText()) &&
                    TextUtils.isEmpty(etPassword.getText())){
66                      Toast.makeText(getApplicationContext(),
67                              "Please enter a username and password",
                                Toast.LENGTH_LONG).show();
68                  }
69                  else if (TextUtils.isEmpty((etName.getText()))){
70                      Toast.makeText(getApplicationContext(),
71                              "Please enter a username", Toast.LENGTH_LONG).show();
```

```
                }
                else if (TextUtils.isEmpty((etPassword.getText()))){
                    Toast.makeText(getApplicationContext(),
                            "Please enter a password", Toast.LENGTH_LONG).show();
                }else {
                    String[] loginInfo = new String[2];
                    loginInfo[0] = etName.getText().toString();
                    loginInfo[1] = etPassword.getText().toString();
                    LoginRequest lr = new LoginRequest();
                    lr.execute(loginInfo);
                }

            }
        });// sign in on click

    }

    // Async task to post login data
    @SuppressLint("StaticFieldLeak")
    public class LoginRequest extends AsyncTask<String, Void, JSONObject>{

        @Override
        protected JSONObject doInBackground(String... args) {

            try{
                URL url = new
                URL("http://c0009839.candept.com/API/loginregister.php");

                // put params in a Json Object
                JSONObject loginParams = new JSONObject();
                loginParams.put("username", args[0]);
                loginParams.put("password", args[1]);
                Log.i("LOGINPARAMS", loginParams.toString());

                // Set up connection
                HttpURLConnection conn = (HttpURLConnection) url.openConnection();
                conn.setReadTimeout(15000);
                conn.setConnectTimeout(15000);
                conn.setRequestMethod("POST");
                conn.setDoInput(true);
                conn.setDoOutput(true);

                // Send data
                OutputStream os = conn.getOutputStream();
                BufferedWriter writer = new BufferedWriter(
                        new OutputStreamWriter(os, StandardCharsets.UTF_8));
                writer.write(getPostDataString(loginParams));

                writer.flush();
                writer.close();
                os.close();

                // Get response
                int responseCode = conn.getResponseCode();
                error = String.valueOf(conn.getResponseCode());

                if (responseCode == HttpURLConnection.HTTP_OK){
                    inputStream = conn.getInputStream();
                    BufferedReader in = new BufferedReader(new
                    InputStreamReader(inputStream));
                    StringBuilder sb = new StringBuilder();
                    String line;

                    while (null != (line = in.readLine())){
                        sb.append(line).append("\n");
                    }
                    in.close();
                    inputStream.close();
                    json = sb.toString();
                    Log.i("APT: ", json);

                }else{
                    Log.e("Buffer Error", "Error Getting Result " +responseCode);
```

```
                }
                try{
                    jObj = new JSONObject(json);
                    jObj.put("error_code", error);
                }catch(JSONException e){
                    Log.e("JSON Parser", "Error Parsing Data " + e.toString());
                }
            }catch(Exception e){
                Log.e("Exception: ", "Overall Try Block " + e.toString());
            }
            return jObj;
        }// end doInBackground

        // Interact with returned json object
        @Override
        protected void onPostExecute(JSONObject result){

            try{
                if(result != null){
                    String isAuthUser = result.getString("message");
                    if (isAuthUser.equals("Successfully logged in")){

                     Intent intent = new Intent(LoginActivity.this,
                             com.c00098391.planttracker.MainActivity.class);

                     intent.putExtra("username", etName.getText().toString());
                     startActivity(intent);
                    }else{
                        Toast.makeText(getApplicationContext(), result.getString(
                                "message"), Toast.LENGTH_SHORT).show();
                    }
                }else{
                    Toast.makeText(getApplicationContext(),
                            "Unable to retrieve data from the server",
                            Toast.LENGTH_LONG).show();
                }
            }catch(JSONException e){
                e.printStackTrace();
            }
        }
    }

    // Turn json object to string for post
    public String getPostDataString(JSONObject params) throws Exception {

        StringBuilder result = new StringBuilder();
        boolean first = true;
        Iterator<String> itr = params.keys();

        while(itr.hasNext()){
            String key = itr.next();
            Object value = params.get(key);
            if(first){
                first = false;
            }else{
                result.append("&");
            }
            result.append(URLEncoder.encode(key, "UTF-8"));
            result.append("=");
            result.append(URLEncoder.encode(value.toString(), "UTF-8"));
        }
        return result.toString();
    }
}

```

CaptureText.java

```java
package com.c00098391.planttracker;

/**
 * Created by Darran Gahan
 * Student Number: C00098391
 *
 *
 * Class to allow a user to capture an image containing text before
 * moving to a view activity to ensure the text that was detected
 * is correct.
 */

import android.Manifest;
import android.content.Context;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.ImageFormat;
import android.graphics.SurfaceTexture;
import android.hardware.camera2.CameraAccessException;
import android.hardware.camera2.CameraCaptureSession;
import android.hardware.camera2.CameraCharacteristics;
import android.hardware.camera2.CameraDevice;
import android.hardware.camera2.CameraManager;
import android.hardware.camera2.CameraMetadata;
import android.hardware.camera2.CaptureRequest;
import android.hardware.camera2.TotalCaptureResult;
import android.hardware.camera2.params.StreamConfigurationMap;
import android.media.Image;
import android.media.ImageReader;
import android.os.Handler;
import android.os.HandlerThread;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.util.Size;
import android.util.SparseIntArray;
import android.view.Surface;
import android.view.TextureView;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;

import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.nio.ByteBuffer;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

public class CaptureText extends AppCompatActivity {

    // Over lay grid image
    private ImageView gridImage;
    private static final String TAG = "CaptureText";
    private Button btnCaptureText;
    private TextureView textureView;
    private static final SparseIntArray ORIENTATIONS = new SparseIntArray();
    static {
        ORIENTATIONS.append(Surface.ROTATION_0, 90);
        ORIENTATIONS.append(Surface.ROTATION_90, 0);
        ORIENTATIONS.append(Surface.ROTATION_180, 270);
        ORIENTATIONS.append(Surface.ROTATION_270, 180);
    }

    private String cameraId;
```

```
74      protected CameraDevice cameraDevice;
75      protected CameraCaptureSession cameraCaptureSessions;
76      protected CaptureRequest.Builder captureRequestBuilder;
77      private Size imageDimension;
78      private ImageReader imageReader;
79      private File file;
80
81      protected String fileLoc;
82      private static final int REQUEST_CAMERA_PERMISSION = 200;
83      private Handler mBackGroundHandler;
84      private HandlerThread mBackgroundThread;
85
86      String username;
87      String userId;
88
89      @Override
90      protected void onCreate(Bundle savedInstanceState) {
91          super.onCreate(savedInstanceState);
92          setContentView(R.layout.activity_capture_text);
93
94          gridImage = findViewById(R.id.gridImage);
95          textureView = findViewById(R.id.texture);
96          assert textureView != null;
97          textureView.setSurfaceTextureListener(textureListener);
98          btnCaptureText = findViewById(R.id.btnCaptureText);
99          assert btnCaptureText != null;
100
101         // Get user info
102         username = getIntent().getStringExtra("username");
103         userId = getIntent().getStringExtra("userid");
104
105
106         btnCaptureText.setOnClickListener(new View.OnClickListener() {
107             @Override
108             public void onClick(View view) {
109
110                 captureText();
111             }
112         });
113
114
115     }// END OF ON CREATE
116
117     TextureView.SurfaceTextureListener textureListener = new
        TextureView.SurfaceTextureListener() {
118         @Override
119         public void onSurfaceTextureAvailable(SurfaceTexture surfaceTexture, int i,
            int i1) {
120             openCamera();
121         }
122
123         @Override
124         public void onSurfaceTextureSizeChanged(SurfaceTexture surfaceTexture, int
            i, int i1) {
125
126         }
127
128         @Override
129         public boolean onSurfaceTextureDestroyed(SurfaceTexture surfaceTexture) {
130             return false;
131         }
132
133         @Override
134         public void onSurfaceTextureUpdated(SurfaceTexture surfaceTexture) {
135
136         }
137     };
138
139     private final CameraDevice.StateCallback stateCallback = new
        CameraDevice.StateCallback() {
140         @Override
141         public void onOpened(@NonNull CameraDevice camera) {
142             Log.e(TAG, "In onOpened");
```

```
143             cameraDevice = camera;
144             createCameraPreview();
145         }
146
147         @Override
148         public void onDisconnected(@NonNull CameraDevice cameraDevice) {
149             cameraDevice.close();
150         }
151
152         @Override
153         public void onError(@NonNull CameraDevice cameraDevice, int i) {
154             cameraDevice.close();
155          //   cameraDevice = null;
156         }
157     };
158
159     final CameraCaptureSession.CaptureCallback captureCallBackListener = new
        CameraCaptureSession.CaptureCallback() {
160         @Override
161         public void onCaptureCompleted(CameraCaptureSession session, CaptureRequest
            request, TotalCaptureResult result){
162             super.onCaptureCompleted(session, request, result);
163             Toast.makeText(CaptureText.this, "Saved: " + file,
                Toast.LENGTH_LONG).show();
164             createCameraPreview();
165         }
166     };
167
168     protected void startBackgroundThread(){
169         mBackgroundThread = new HandlerThread("Camera Background");
170         mBackgroundThread.start();
171         mBackGroundHandler = new Handler(mBackgroundThread.getLooper());
172     }
173
174     protected void stopBackgroundThread(){
175         mBackgroundThread.quitSafely();
176         try{
177             mBackgroundThread.join();
178             mBackgroundThread = null;
179             mBackGroundHandler = null;
180         }catch(InterruptedException e){
181             e.printStackTrace();
182         }
183     }
184
185     protected void captureText(){
186         if (null == cameraDevice){
187             Log.e(TAG, "Camera Device is null");
188             return;
189         }
190
191         CameraManager manager = (CameraManager)
            getSystemService(Context.CAMERA_SERVICE);
192         try{
193             CameraCharacteristics characteristics =
                manager.getCameraCharacteristics(cameraDevice.getId());
194             Size[] jpegSizes = null;
195             if (characteristics != null){
196                 jpegSizes =
                    characteristics.get(CameraCharacteristics.SCALER_STREAM_CONFIGURATION_
                    MAP)
197                         .getOutputSizes(ImageFormat.JPEG);
198             }
199             int width = 640;
200             int height = 640;
201             if(jpegSizes != null && 0 < jpegSizes.length){
202                 width = jpegSizes[0].getWidth();
203                 height = jpegSizes[0].getHeight();
204             }
205             ImageReader reader = ImageReader.newInstance(width, height,
                ImageFormat.JPEG, 1);
206             List<Surface> outputSurfaces = new ArrayList<>(2);
207             outputSurfaces.add(reader.getSurface());
```

```
208             outputSurfaces.add(new Surface(textureView.getSurfaceTexture()));
209             final CaptureRequest.Builder captureBuilder
210                     =
                        cameraDevice.createCaptureRequest(CameraDevice.TEMPLATE_STILL_CAPT
                        URE);
211             captureBuilder.addTarget(reader.getSurface());
212             captureBuilder.set(CaptureRequest.CONTROL_MODE,
                CameraMetadata.CONTROL_MODE_AUTO);
213
214             // Orientation
215             int rotation = getWindowManager().getDefaultDisplay().getRotation();
216             captureBuilder.set(CaptureRequest.JPEG_ORIENTATION,
                ORIENTATIONS.get(rotation));
217
218
219             String outPic = Constants.SCAN_IMAGE_LOCATION
220                     + File.separator + Utilities.generateFilename();
221             FolderUtil.createDefaultFolder(Constants.SCAN_IMAGE_LOCATION);
222             fileLoc = outPic;
223
224             final File file = new File(outPic);
225
226
227             ImageReader.OnImageAvailableListener readerListener = new
                ImageReader.OnImageAvailableListener(){
228                 @Override
229                 public void onImageAvailable(ImageReader reader){
230                     Image image = null;
231
232                     try {
233                         image = reader.acquireLatestImage();
234                         ByteBuffer buffer = image.getPlanes()[0].getBuffer();
235                         byte[] bytes = new byte[buffer.capacity()];
236                         buffer.get(bytes);
237
238                         BitmapFactory.Options options = new BitmapFactory.Options();
239                         // Bitmap sample size is set to 4 to ensure app does not tun
                            otu of menory
240                         options.inSampleSize = 4;
241                         Bitmap bmp = BitmapFactory.decodeByteArray(bytes, 0,
                            bytes.length,  options);
242
243                       //  Bitmap bmp = BitmapFactory.decodeByteArray(bytes, 0,
                          bytes.length);
244                         bmp = bmp.copy(Bitmap.Config.ARGB_8888, true);
245                         ByteArrayOutputStream stream = new ByteArrayOutputStream();
246                         bmp.compress(Bitmap.CompressFormat.JPEG, 70, stream);
247
248                         byte[] byteOut = stream.toByteArray();
249
250                         Intent intent = new Intent(CaptureText.this,
251                                 com.c00098391.planttracker.TextResults.class);
252                         intent.putExtra("username", username);
253                         intent.putExtra("userid", userId);
254                         intent.putExtra("image", byteOut);
255                         startActivity(intent);
256
257                         // Uncomment to allow image saving to device.
258                         // save(byteOut);
259                     // }catch(IOException e){
260                     //      e.printStackTrace();;
261                     }finally{
262                         if(image != null){
263                             image.close();
264                         }
265                     }
266                 }
267
268                 private void save(byte[] bytes) throws IOException{
269                     OutputStream output = null;
270                     try{
271                         output = new FileOutputStream(file);
272                         output.write(bytes);
```

```
                    }finally{
                        if(null != output){
                            output.close();
                        }
                    }
                }
            };

            reader.setOnImageAvailableListener(readerListener, mBackGroundHandler);
            final CameraCaptureSession.CaptureCallback captureListener
                    = new CameraCaptureSession.CaptureCallback() {
                @Override
                public void onCaptureCompleted(CameraCaptureSession session,
                CaptureRequest request, TotalCaptureResult result) {
                    super.onCaptureCompleted(session, request, result);

                }
            };

            cameraDevice.createCaptureSession(outputSurfaces, new
            CameraCaptureSession.StateCallback(){
                @Override
                public void onConfigured(CameraCaptureSession session){
                    try{
                        session.capture(captureBuilder.build(), captureListener,
                        mBackGroundHandler);
                    }catch(CameraAccessException e){
                        e.printStackTrace();
                    }
                }

                @Override
                public void onConfigureFailed(@NonNull CameraCaptureSession
                cameraCaptureSession){

                }
            }, mBackGroundHandler);
        }catch(CameraAccessException e){
            e.printStackTrace();
        }
    }

    protected void createCameraPreview(){
        try{
            SurfaceTexture texture = textureView.getSurfaceTexture();
            assert texture != null;
            texture.setDefaultBufferSize(imageDimension.getWidth(),
            imageDimension.getHeight()-100);
            Surface surface = new Surface(texture);

            captureRequestBuilder =
            cameraDevice.createCaptureRequest(CameraDevice.TEMPLATE_PREVIEW);
            captureRequestBuilder.addTarget(surface);
            cameraDevice.createCaptureSession(Arrays.asList(surface), new
            CameraCaptureSession.StateCallback(){
                @Override
                public void onConfigured(@NonNull CameraCaptureSession
                cameraCaptureSession){
                    if(null == cameraDevice){
                        return;
                    }

                    cameraCaptureSessions = cameraCaptureSession;
                    updatePreview();
                }

                @Override
                public void onConfigureFailed(@NonNull CameraCaptureSession
                cameraCaptureSession){
                    Toast.makeText(CaptureText.this, "Configuration changed",
                            Toast.LENGTH_SHORT).show();
                }
            }, null);
```

```
337             }catch(CameraAccessException e){
338                 e.printStackTrace();
339             }
340         }
341
342         private void openCamera(){
343             CameraManager manager = (CameraManager)
                getSystemService(Context.CAMERA_SERVICE);
344             Log.e(TAG, "in openCamera()");
345             try{
346                 cameraId = manager.getCameraIdList()[0];
347                 CameraCharacteristics characteristics =
                    manager.getCameraCharacteristics(cameraId);
348                 StreamConfigurationMap map
349                         =
                            characteristics.get(CameraCharacteristics.SCALER_STREAM_CONFIGURAT
                            ION_MAP);
350                 assert map != null;
351                 imageDimension = map.getOutputSizes(SurfaceTexture.class)[0];
352
353
354                 if (ActivityCompat.checkSelfPermission(this, Manifest.permission.CAMERA)
355                         != PackageManager.PERMISSION_GRANTED
356                         && ActivityCompat.checkSelfPermission(this,
357                         Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
                            PackageManager.PERMISSION_GRANTED){
358
359                     ActivityCompat.requestPermissions(CaptureText.this, new String[]{
360                             Manifest.permission.CAMERA,
361                             Manifest.permission.WRITE_EXTERNAL_STORAGE},
362                             REQUEST_CAMERA_PERMISSION);
363                     return;
364                 }
365                 manager.openCamera(cameraId, stateCallback, null);
366             }catch(CameraAccessException e){
367             e.printStackTrace();
368             }
369             Log.e(TAG, "in Open Camera (permissions");
370         }
371
372         protected void updatePreview(){
373             if(null == cameraDevice){
374                 Log.e(TAG, "Update Preview Error..");
375             }
376             captureRequestBuilder.set(CaptureRequest.CONTROL_MODE,
                CameraMetadata.CONTROL_MODE_AUTO);
377             try{
378                 cameraCaptureSessions.setRepeatingRequest(captureRequestBuilder.build(),
379                         null, mBackGroundHandler);
380             }catch(CameraAccessException e){
381                 e.printStackTrace();
382             }
383         }
384
385         private void closeCamera(){
386             if(null != cameraDevice){
387                 cameraDevice.close();
388                 cameraDevice = null;
389             }
390             if (null != imageReader){
391                 imageReader.close();
392                 imageReader = null;
393             }
394         }
395
396         @Override
397         public void onRequestPermissionsResult(int requestCode, @NonNull String[]
            permissions,
398                                                @NonNull int[] grantResults){
399             if (requestCode == REQUEST_CAMERA_PERMISSION) {
400                 if (grantResults[0] == PackageManager.PERMISSION_DENIED) {
401                     // Close app
402                     Toast.makeText(CaptureText.this,
```

```
                        "Sorry, you cannot use this app without granting
                        permission", Toast.LENGTH_LONG)
                        .show();
                finish();
            }
        }
    }

    @Override
    protected void onResume(){
        super.onResume();
        Log.e(TAG, "in onResume");
        startBackgroundThread();
        if(textureView.isAvailable()){
            openCamera();
        }else{
            textureView.setSurfaceTextureListener(textureListener);
        }
    }
    @Override
    protected void onPause(){
        Log.e(TAG, "in OnPause");
        closeCamera();
        stopBackgroundThread();
        super.onPause();
    }

}

```

TestResults.java

```
1   package com.c00098391.planttracker;
2
3   import android.annotation.SuppressLint;
4   import android.content.Context;
5   import android.content.Intent;
6   import android.graphics.Bitmap;
7   import android.graphics.BitmapFactory;
8   import android.os.AsyncTask;
9   import android.support.annotation.NonNull;
10  import android.support.v7.app.AppCompatActivity;
11  import android.os.Bundle;
12  import android.util.Base64;
13  import android.util.Log;
14  import android.view.View;
15  import android.widget.Button;
16  import android.widget.ImageView;
17  import android.widget.TextView;
18  import android.widget.Toast;
19
20  import com.google.android.gms.tasks.OnFailureListener;
21  import com.google.android.gms.tasks.OnSuccessListener;
22  import com.google.firebase.ml.vision.FirebaseVision;
23  import com.google.firebase.ml.vision.common.FirebaseVisionImage;
24  import com.google.firebase.ml.vision.text.FirebaseVisionText;
25  import com.google.firebase.ml.vision.text.FirebaseVisionTextRecognizer;
26
27  import org.json.JSONException;
28  import org.json.JSONObject;
29
30  import java.io.BufferedInputStream;
31  import java.io.BufferedReader;
32  import java.io.BufferedWriter;
33  import java.io.ByteArrayOutputStream;
34  import java.io.File;
35  import java.io.FileInputStream;
36  import java.io.FileNotFoundException;
37  import java.io.IOException;
38  import java.io.InputStream;
39  import java.io.InputStreamReader;
40  import java.io.OutputStream;
41  import java.io.OutputStreamWriter;
42  import java.io.RandomAccessFile;
43  import java.net.HttpURLConnection;
44  import java.net.URL;
45  import java.net.URLEncoder;
46  import java.nio.charset.StandardCharsets;
47  import java.nio.file.Files;
48  import java.nio.file.Paths;
49  import java.text.SimpleDateFormat;
50  import java.util.Arrays;
51  import java.util.Calendar;
52  import java.util.Iterator;
53  import java.util.List;
54  import java.util.concurrent.ExecutionException;
55
56  public class TextResults extends AppCompatActivity {
57
58      ImageView ivText;
59      TextView tvText;
60      Button btnShowText, btnInputdetailsMaually;
61      String data;
62
63      // Async variables
64      static InputStream inputStream = null;
65      static String json;
66      static JSONObject jObj = null;
67      static String error = "";
68
69      String username;
70      String userId;
71      String rep;
72      String treatment;
73      String expt;
```

```
74
75
76          // Location variables
77          String lat = "";
78          String lon = "";
79
80          // Weather variables
81          private static final String APP_ID = "b11dc521fd3aecc6374e2e331dc090e3";
82          String units = "metric";
83          String url;
84          String weather = "";
85
86          @SuppressLint("ClickableViewAccessibility")
87          @Override
88          protected void onCreate(Bundle savedInstanceState) {
89              super.onCreate(savedInstanceState);
90              setContentView(R.layout.activity_text_results);
91
92              ivText = findViewById(R.id.ivText);
93              tvText = findViewById(R.id.tvText);
94              btnShowText = findViewById(R.id.btnShowText);
95              btnInputdetailsMaually = findViewById(R.id.btnInputDetailsMaually);
96
97              // Start Location service and get lat and lon
98              startService(new Intent(TextResults.this,
99                      com.c00098391.planttracker.GPS.class));
100             final GPS gps = new GPS(TextResults.this);
101
102
103             lat = Double.toString(gps.getLatitude());
104             lon = Double.toString(gps.getLongitude());
105
106             // url for weather data
107             url =
                "http://api.openweathermap.org/data/2.5/weather?lat="+lat+"&lon="+lon+"&units=
                "+units+"&appid="+APP_ID;
108
109             username = getIntent().getStringExtra("username");
110             userId = getIntent().getStringExtra("userid");
111
112
113             BitmapFactory.Options options = new BitmapFactory.Options();
114             final byte[] byteArray = getIntent().getByteArrayExtra("image");
115             Bitmap bm = BitmapFactory.decodeByteArray(byteArray, 0, byteArray.length);
116
117             // Call to get detect text from image
118             runTextRecognition(bm);
119
120             // encoded image
121             String encodedImg = Base64.encodeToString(byteArray, Base64.DEFAULT);
122
123             // Set format for time and date
124             SimpleDateFormat df = new SimpleDateFormat("dd-MM-yyyy");
125             SimpleDateFormat tf = new SimpleDateFormat("hh:mm:ss");
126
127             // Create strings for time and date
128             final String date = df.format(Calendar.getInstance().getTime());
129             final String time = tf.format(Calendar.getInstance().getTime());
130
131             String weatherData = null;
132             try {
133                 weatherData = new TextResults.GetWeatherTask(weather).execute(url).get();
134             } catch (InterruptedException e) {
135                 e.printStackTrace();
136             } catch (ExecutionException e) {
137                 e.printStackTrace();
138             }
139
140             // String array for experiment details
141             final String [] expDetails = new String[11];
142             expDetails[0] = date;
143             expDetails[1] = time;
144             expDetails[2] = encodedImg;
```

```
145             expDetails[3] = username;
146             expDetails[4] = userId;
147             expDetails[5] = weatherData;
148             expDetails[6] = lat;
149             expDetails[7] = lon;
150
151             ivText.setImageBitmap(bm);
152
153             // Onclick to upload text results
154             btnShowText.setOnClickListener(new View.OnClickListener() {
155                 @Override
156                 public void onClick(View view) {
157
158                     String textResults = getTextData();
159                     String [] parts = textResults.split(" ");
160                     rep = parts[1];
161                     treatment = parts[2];
162                     expt = parts[4] +" " + parts[5];
163
164                     expDetails[8] = rep;
165                     expDetails[9] = expt;
166                     expDetails[10] = treatment;
167
168                     UploadData ud = new UploadData();
169                     ud.execute(expDetails);
170                 }
171             });
172
173             // button to move to input details manually
174             btnInputdetailsMaually.setOnClickListener(new View.OnClickListener() {
175                 @Override
176                 public void onClick(View view) {
177                     Intent intent = new Intent(TextResults.this,
                        com.c00098391.planttracker.InputDetails.class);
178                     intent.putExtra("image", byteArray);
179                     intent.putExtra("date", date);
180                     intent.putExtra("time", time);
181                     intent.putExtra("username", username);
182                     intent.putExtra("userid", userId);
183                     intent.putExtra("weather", weather);
184                     intent.putExtra("lat", lat);
185                     intent.putExtra("lon", lon);
186
187                     startActivity(intent);
188                 }
189             });
190
191         } // end of on create
192
193         public void setTextData(String data){
194             this.data = data;
195         }
196         public String getTextData(){
197             return this.data;
198         }
199
200         // Method to detect text from image
201         private void runTextRecognition(Bitmap b){
202             FirebaseVisionImage image = FirebaseVisionImage.fromBitmap(b);
203             FirebaseVisionTextRecognizer recognizer = FirebaseVision.getInstance()
204                     .getOnDeviceTextRecognizer();
205             btnShowText.setEnabled(false);
206             recognizer.processImage(image)
207                     .addOnSuccessListener(new OnSuccessListener<FirebaseVisionText>() {
208                         @Override
209                         public void onSuccess(FirebaseVisionText texts) {
210                             btnShowText.setEnabled(true);
211                             processTextRecognitionResult(texts);
212
213                         }
214                     })
215                     .addOnFailureListener(
216                             new OnFailureListener() {
```

```
                            @Override
                            public void onFailure(@NonNull Exception e) {
                                btnShowText.setEnabled(true);
                                e.printStackTrace();
                            }
                        }
                );
    }

    // Method to get detected text
    private void processTextRecognitionResult(FirebaseVisionText texts){
        List<FirebaseVisionText.TextBlock> blocks = texts.getTextBlocks();
        if(blocks.size() == 0){
            Toast.makeText(TextResults.this, "No text found",
            Toast.LENGTH_LONG).show();
            //return "";
        }

        StringBuilder sb = new StringBuilder();
        Boolean first = true;
        String s = "";

        for (int i = 0; i < blocks.size(); i++){
            List<FirebaseVisionText.Line> lines = blocks.get(i).getLines();
            for (int j = 0; j < lines.size(); j++){
                List<FirebaseVisionText.Element> elements =
                lines.get(j).getElements();

                for (int k = 0; k< elements.size(); k++){
                    //  Graphic textGraphic = new TextGraphic(mGraphicOverlay,
                    elements.get(k));
                    // mGraphicOverlay.add(textGraphic);

                    sb.append(elements.get(k).getText()+ " ");

                    s = s + elements.get(k).getText() + " ";
                }
            }
        }

        tvText.setText(s);

        setTextData(s);
    }

    // Async task for sending data i.e. image date and time....
    public class UploadData extends AsyncTask<String, Void, JSONObject> {


        @Override
        protected JSONObject doInBackground(String... args){

            try{
                URL url = new
                URL("http://www.c0009839.candept.com/API/CreateExperiment.php");

                // put params in a JSON Object
                JSONObject dataParams = new JSONObject();
                dataParams.put("date", args[0]);
                dataParams.put("time", args[1]);
                dataParams.put("image", args[2]);
                dataParams.put("username", args[3]);
                dataParams.put("userid", args[4]);
                dataParams.put("weather", args[5]);
                dataParams.put("lat", args[6]);
                dataParams.put("lon", args[7]);
                dataParams.put("rep",args[8]);
                dataParams.put("expt", args[9]);
                dataParams.put("treatment", args[10]);

                Log.i("DATAPARAS", dataParams.toString());

                // Set up connection
```

```
286                 HttpURLConnection conn = (HttpURLConnection) url.openConnection();
287                 conn.setReadTimeout(15000);
288                 conn.setConnectTimeout(15000);
289                 conn.setRequestMethod("POST");
290                 conn.setDoInput(true);
291                 conn.setDoOutput(true);
292
293                 //send date
294                 OutputStream os = conn.getOutputStream();
295                 BufferedWriter writer = new BufferedWriter(
296                         new OutputStreamWriter(os, StandardCharsets.UTF_8));
297                 writer.write(getPostDateString(dataParams));
298
299                 writer.flush();
300                 writer.close();
301                 os.close();
302
303                 // Get Response
304                 int responseCode = conn.getResponseCode();
305                 error = String.valueOf(conn.getResponseCode());
306
307                 if (responseCode == HttpURLConnection.HTTP_OK){
308                     inputStream = conn.getInputStream();
309                     BufferedReader in  = new BufferedReader(new
                        InputStreamReader(inputStream));
310                     StringBuilder sb = new StringBuilder();
311                     String line;
312
313                     while(null!= (line = in.readLine())){
314                         sb.append(line).append("\n");
315                     }
316                     in.close();
317                     inputStream.close();
318                     json = sb.toString();
319                     Log.i("API Camera: ", json);
320                 }
321                 else{
322                     Log.e("Buffer Error", "Error Getting Result " +responseCode);
323                 }
324                 try{
325                     jObj = new JSONObject(json);
326                     jObj.put("error_code", error);
327                 }catch(JSONException e){
328                     Log.e("JSON Parser", "Error Parsing Data " + e.toString());
329                 }
330             }catch(Exception e){
331                 Log.e("Exception: ", "Overall Try Block " + e.toString());
332             }
333             return jObj;
334         }// end of doInBackground
335
336         @Override
337         protected void onPostExecute(JSONObject result){
338
339             try {
340
341                 if (result != null){
342
343                     String uploadSuccess = result.getString("message");
344                     if (uploadSuccess.equals("Successfully created experiment")){
345                         Toast.makeText(getApplicationContext(), result.getString(
346                                 "message"), Toast.LENGTH_LONG).show();
347
348
349                         String expId = result.getString("expid");
350
351
352                         Intent intent = new Intent(TextResults.this,
353                                 com.c00098391.planttracker.DetectDisease.class);
354                         intent.putExtra("username", username);
355                         intent.putExtra("userid", userId);
356                         intent.putExtra("rep", rep);
357                         intent.putExtra("treatment", treatment);
```

```
                        intent.putExtra("expt", expt);
                        intent.putExtra("expid", expId);
                        startActivity(intent);

                    }else{
                        Toast.makeText(getApplicationContext(),
                                "error", Toast.LENGTH_LONG).show();
                    }
                }else{
                    Toast.makeText(getApplicationContext(),
                            "Unable to retrieve data from the server",
                            Toast.LENGTH_LONG).show();
                }
            }catch(JSONException e){
                e.printStackTrace();
            }
        }
    }

    // Turn json object to string for post
    public String getPostDateString(JSONObject params) throws Exception{

        StringBuilder result = new StringBuilder();
        boolean first = true;
        Iterator<String> itr = params.keys();

        while(itr.hasNext()){
            String key = itr.next();
            Object value = params.get(key);
            if(first){
                first = false;
            }else{
                result.append("&");
            }
            result.append(URLEncoder.encode(key, "UTF-8"));
            result.append("=");
            result.append(URLEncoder.encode(value.toString(), "UTF-8"));
        }
        return result.toString();
    }

    // Async task to get weather details
    @SuppressLint("StaticFieldLeak")
    private class GetWeatherTask extends AsyncTask<String, Void, String> {

        private String weather;

        public GetWeatherTask(String weather){
            this.weather = weather;
        }

        @Override
        protected  String doInBackground(String... strings){
            String weather = "UNDEFINED";

            try {
                URL url = new URL(strings[0]);
                HttpURLConnection urlConnection = (HttpURLConnection)
                url.openConnection();

                InputStream stream = new
                BufferedInputStream(urlConnection.getInputStream());
                BufferedReader bufferedReader = new BufferedReader(new
                InputStreamReader(stream));
                StringBuilder builder = new StringBuilder();

                String inputString;
                while ((inputString = bufferedReader.readLine()) != null){
                    builder.append(inputString);
                }

                JSONObject topLevel = new JSONObject(builder.toString());
                JSONObject main = topLevel.getJSONObject("main");
```

```
427                     String temp = String.valueOf(main.getDouble("temp"));
428
429                     String overview = topLevel.getJSONArray("weather")
430                             .getJSONObject(0).get("main").toString();
431                     String desc = topLevel.getJSONArray("weather")
432                             .getJSONObject(0).get("description").toString();
433
434                     weather = temp + "C, " + overview + "(" + desc + ")";
435
436                     urlConnection.disconnect();
437                 }catch (IOException | JSONException e){
438                     e.printStackTrace();
439                 }
440                 return weather;
441             }
442
443             @Override
444             protected void onPostExecute(String temp) {
445                 weather = "Current Weather " + temp;
446             }
447         }
448
449     }
450
```

InputDetails.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * Student Name: Darran Gahan
5    * Student Number: C00098391
6    *
7    * Class is used to allow user to input details manually if text detection does not
     work
8    */
9
10  import android.annotation.SuppressLint;
11  import android.content.Intent;
12  import android.os.AsyncTask;
13  import android.support.v7.app.AppCompatActivity;
14  import android.os.Bundle;
15  import android.util.Base64;
16  import android.util.Log;
17  import android.view.View;
18  import android.widget.Button;
19  import android.widget.EditText;
20  import android.widget.Toast;
21
22  import org.json.JSONException;
23  import org.json.JSONObject;
24
25  import java.io.BufferedInputStream;
26  import java.io.BufferedReader;
27  import java.io.BufferedWriter;
28  import java.io.IOException;
29  import java.io.InputStream;
30  import java.io.InputStreamReader;
31  import java.io.OutputStream;
32  import java.io.OutputStreamWriter;
33  import java.net.HttpURLConnection;
34  import java.net.URL;
35  import java.net.URLEncoder;
36  import java.nio.charset.StandardCharsets;
37  import java.util.Iterator;
38  import java.util.concurrent.ExecutionException;
39
40  public class InputDetails extends AppCompatActivity {
41
42      // Async class variables
43      static InputStream inputStream = null;
44      static String json;
45      static JSONObject jObj = null;
46      static String error = "";
47
48
49      Button btnCreateExp;
50      EditText etRep, etTreat, etExpt;
51
52      // Experiment variables
53      String rep;
54      String treatment;
55      String expt;
56
57      // Intent varaible
58      String userId;
59      String username;
60      String time;
61      String date;
62
63      // Location variables
64      String lat = "";
65      String lon = "";
66
67      // Weather variables
68      private static final String APP_ID = "b11dc521fd3aecc6374e2e331dc090e3";
69      String weather = "";
70      String units = "metric";
71      String url;
72
```

```
73      @Override
74      protected void onCreate(Bundle savedInstanceState) {
75          super.onCreate(savedInstanceState);
76          setContentView(R.layout.activity_input_details);
77
78          // Start Location service and get lat and lon
79          startService(new Intent(InputDetails.this,
80                  com.c00098391.planttracker.GPS.class));
81          final GPS gps = new GPS(InputDetails.this);
82          lat = Double.toString(gps.getLatitude());
83          lon = Double.toString(gps.getLongitude());
84          url =
            "http://api.openweathermap.org/data/2.5/weather?lat="+lat+"&lon="+lon+"&units=
            "+units+"&appid="+APP_ID;
85
86          btnCreateExp = findViewById(R.id.btnCreateExp);
87          etRep = findViewById(R.id.etRep);
88          etTreat = findViewById(R.id.etTreat);
89          etExpt = findViewById(R.id.etExpt);
90
91          // Get data from intent
92          byte[] byteArray = getIntent().getByteArrayExtra("image");
93          date = getIntent().getStringExtra("date");
94          time = getIntent().getStringExtra("time");
95          username = getIntent().getStringExtra("username");
96          userId = getIntent().getStringExtra("userid");
97
98          // Get weather
99          String weatherData = null;
100         try {
101             weatherData = new GetWeatherTask(weather).execute(url).get();
102         } catch (InterruptedException e) {
103             e.printStackTrace();
104         } catch (ExecutionException e) {
105             e.printStackTrace();
106         }
107
108         String encodedImg = Base64.encodeToString(byteArray, Base64.DEFAULT);
109
110         // Array for experiment details
111         final String [] expDetails = new String[11];
112         expDetails[0] = date;
113         expDetails[1] = time;
114         expDetails[2] = encodedImg;
115         expDetails[3] = username;
116         expDetails[4] = userId;
117         expDetails[5] = weatherData;
118         expDetails[6] = lat;
119         expDetails[7] = lon;
120
121
122         // button to create create experiment in database
123         btnCreateExp.setOnClickListener(new View.OnClickListener() {
124             @Override
125             public void onClick(View view) {
126
127                 rep = etRep.getText().toString();
128                 expt = etExpt.getText().toString();
129                 treatment = etTreat.getText().toString();
130
131                 expDetails[8] = rep;
132                 expDetails[9] = expt;
133                 expDetails[10] = treatment;
134
135                 CreateExperiment ce = new CreateExperiment();
136                 ce.execute(expDetails);
137
138             }
139         });
140     }
141
142     // Async task for sending data
143     public class CreateExperiment extends AsyncTask<String, Void, JSONObject> {
```

```
144
145
146         @Override
147         protected JSONObject doInBackground(String... args){
148
149             try{
150                 URL url = new
                    URL("http://www.c0009839.candept.com/API/CreateExperiment.php");
151
152                 // put params in a JSON Object
153                 JSONObject dataParams = new JSONObject();
154                 dataParams.put("date", args[0]);
155                 dataParams.put("time", args[1]);
156                 dataParams.put("image", args[2]);
157                 dataParams.put("username", args[3]);
158                 dataParams.put("userid", args[4]);
159                 dataParams.put("weather", args[5]);
160                 dataParams.put("lat", args[6]);
161                 dataParams.put("lon", args[7]);
162                 dataParams.put("rep",args[8]);
163                 dataParams.put("expt", args[9]);
164                 dataParams.put("treatment", args[10]);
165
166                 Log.i("DATAPARAS", dataParams.toString());
167
168                 // Set up connection
169                 HttpURLConnection conn = (HttpURLConnection) url.openConnection();
170                 conn.setReadTimeout(15000);
171                 conn.setConnectTimeout(15000);
172                 conn.setRequestMethod("POST");
173                 conn.setDoInput(true);
174                 conn.setDoOutput(true);
175
176                 //send date
177                 OutputStream os = conn.getOutputStream();
178                 BufferedWriter writer = new BufferedWriter(
179                         new OutputStreamWriter(os, StandardCharsets.UTF_8));
180                 writer.write(getPostDateString(dataParams));
181
182                 writer.flush();
183                 writer.close();
184                 os.close();
185
186                 // Get Response
187                 int responseCode = conn.getResponseCode();
188                 error = String.valueOf(conn.getResponseCode());
189
190                 if (responseCode == HttpURLConnection.HTTP_OK){
191                     inputStream = conn.getInputStream();
192                     BufferedReader in  = new BufferedReader(new
                        InputStreamReader(inputStream));
193                     StringBuilder sb = new StringBuilder();
194                     String line;
195
196                     while(null!= (line = in.readLine())){
197                         sb.append(line).append("\n");
198                     }
199                     in.close();
200                     inputStream.close();
201                     json = sb.toString();
202                     Log.i("API Camera: ", json);
203                 }
204                 else{
205                     Log.e("Buffer Error", "Error Getting Result " +responseCode);
206                 }
207                 try{
208                     jObj = new JSONObject(json);
209                     jObj.put("error_code", error);
210                 }catch(JSONException e){
211                     Log.e("JSON Parser", "Error Parsing Data " + e.toString());
212                 }
213             }catch(Exception e){
214                 Log.e("Exception: ", "Overall Try Block " + e.toString());
```

```
215             }
216             return jObj;
217         }// end of doInBackground
218
219         @Override
220         protected void onPostExecute(JSONObject result){
221
222             try {
223
224                 if (result != null){
225
226                     String uploadSuccess = result.getString("message");
227                     if (uploadSuccess.equals("Successfully created experiment")){
228                         Toast.makeText(getApplicationContext(), result.getString(
229                                 "message"), Toast.LENGTH_LONG).show();
230
231                         String expId = result.getString("expid");
232
233
234                         Intent intent = new Intent(InputDetails.this,
235                                 com.c00098391.planttracker.DetectDisease.class);
236                         intent.putExtra("username", username);
237                         intent.putExtra("userid", userId);
238                         intent.putExtra("rep", rep);
239                         intent.putExtra("expt", expt);
240                         intent.putExtra("treatment", treatment);
241                         intent.putExtra("expid", expId);
242
243                         startActivity(intent);
244
245
246
247                     }else{
248                         Toast.makeText(getApplicationContext(), result.getString(
249                                 "message"), Toast.LENGTH_LONG).show();
250                     }
251                 }else{
252                     Toast.makeText(getApplicationContext(),
253                             "Unable to retrieve data from the server",
                                Toast.LENGTH_LONG).show();
254                 }
255             }catch(JSONException e){
256                 e.printStackTrace();
257             }
258         }
259     }
260
261     // Turn json object to string for post
262     public String getPostDateString(JSONObject params) throws Exception{
263
264         StringBuilder result = new StringBuilder();
265         boolean first = true;
266         Iterator<String> itr = params.keys();
267
268         while(itr.hasNext()){
269             String key = itr.next();
270             Object value = params.get(key);
271             if(first){
272                 first = false;
273             }else{
274                 result.append("&");
275             }
276             result.append(URLEncoder.encode(key, "UTF-8"));
277             result.append("=");
278             result.append(URLEncoder.encode(value.toString(), "UTF-8"));
279         }
280         return result.toString();
281     }
282
283
284     // Async task to get weather info
285     @SuppressLint("StaticFieldLeak")
286     private class GetWeatherTask extends AsyncTask<String, Void, String> {
```

```
287
288         private String weather;
289
290         public GetWeatherTask(String weather){
291             this.weather = weather;
292         }
293
294         @Override
295         protected  String doInBackground(String... strings){
296             String weather = "UNDEFINED";
297
298             try {
299                 URL url = new URL(strings[0]);
300                 HttpURLConnection urlConnection = (HttpURLConnection)
                    url.openConnection();
301
302                 InputStream stream = new
                    BufferedInputStream(urlConnection.getInputStream());
303                 BufferedReader bufferedReader = new BufferedReader(new
                    InputStreamReader(stream));
304                 StringBuilder builder = new StringBuilder();
305
306                 String inputString;
307                 while ((inputString = bufferedReader.readLine()) != null){
308                     builder.append(inputString);
309                 }
310
311                 JSONObject topLevel = new JSONObject(builder.toString());
312                 JSONObject main = topLevel.getJSONObject("main");
313                 String temp = String.valueOf(main.getDouble("temp"));
314                 String humidity = String.valueOf(main.getInt("humidity"));
315
316                 String overview = topLevel.getJSONArray("weather")
317                         .getJSONObject(0).get("main").toString();
318                 String desc = topLevel.getJSONArray("weather")
319                         .getJSONObject(0).get("description").toString();
320
321                 weather = temp + "C, " + overview + "(" + desc + ")" + ", Humidity:
                    " +humidity;
322
323                 urlConnection.disconnect();
324             }catch (IOException | JSONException e){
325                 e.printStackTrace();
326             }
327             return weather;
328         }
329
330         @Override
331         protected void onPostExecute(String temp) {
332             weather = "Current Weather " + temp;
333         }
334     }
335 }
336
```

DetectDIsease.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * Created by Student: Darran Gahan
5    * Student Number: C00098391
6    *
7    * Class is used to capture an image and send it to DiseaseAnalysis class to be
     analysed
8    * class implements camera2 Api to perform this and also records the GPS location
     and the Weather
9    *
10   */
11
12  import android.Manifest;
13  import android.content.Context;
14  import android.content.Intent;
15  import android.content.pm.PackageManager;
16  import android.graphics.Bitmap;
17  import android.graphics.BitmapFactory;
18  import android.graphics.ImageFormat;
19  import android.graphics.SurfaceTexture;
20  import android.hardware.camera2.CameraAccessException;
21  import android.hardware.camera2.CameraCaptureSession;
22  import android.hardware.camera2.CameraCharacteristics;
23  import android.hardware.camera2.CameraDevice;
24  import android.hardware.camera2.CameraManager;
25  import android.hardware.camera2.CameraMetadata;
26  import android.hardware.camera2.CaptureRequest;
27  import android.hardware.camera2.TotalCaptureResult;
28  import android.hardware.camera2.params.StreamConfigurationMap;
29  import android.media.Image;
30  import android.media.ImageReader;
31  import android.os.AsyncTask;
32  import android.os.Handler;
33  import android.os.HandlerThread;
34  import android.support.annotation.NonNull;
35  import android.support.v4.app.ActivityCompat;
36  import android.support.v7.app.AppCompatActivity;
37  import android.os.Bundle;
38  import android.util.Base64;
39  import android.util.Log;
40  import android.util.Size;
41  import android.util.SparseIntArray;
42  import android.view.Surface;
43  import android.view.TextureView;
44  import android.view.View;
45  import android.widget.Button;
46  import android.widget.ImageView;
47  import android.widget.Toast;
48
49  import org.json.JSONException;
50  import org.json.JSONObject;
51
52  import java.io.BufferedInputStream;
53  import java.io.BufferedReader;
54  import java.io.ByteArrayOutputStream;
55  import java.io.File;
56  import java.io.FileOutputStream;
57  import java.io.IOException;
58  import java.io.InputStream;
59  import java.io.InputStreamReader;
60  import java.io.OutputStream;
61  import java.net.HttpURLConnection;
62  import java.net.URL;
63  import java.nio.ByteBuffer;
64  import java.text.SimpleDateFormat;
65  import java.util.ArrayList;
66  import java.util.Arrays;
67  import java.util.Calendar;
68  import java.util.List;
69  import java.util.concurrent.ExecutionException;
70
71  public class DetectDisease extends AppCompatActivity {
```

```
72
73      // Over lay grid image
74      private ImageView gridImage;
75      private static final String TAG = "CaptureText";
76      private Button btnDetectDisease, btnEndExperiment;
77
78      private TextureView textureView;
79
80      private static final SparseIntArray ORIENTATIONS = new SparseIntArray();
81      static {
82          ORIENTATIONS.append(Surface.ROTATION_0, 90);
83          ORIENTATIONS.append(Surface.ROTATION_90, 0);
84          ORIENTATIONS.append(Surface.ROTATION_180, 270);
85          ORIENTATIONS.append(Surface.ROTATION_270, 180);
86      }
87
88      private String cameraId;
89      protected CameraDevice cameraDevice;
90      protected CameraCaptureSession cameraCaptureSessions;
91      protected CaptureRequest.Builder captureRequestBuilder;
92      private Size imageDimension;
93      private ImageReader imageReader;
94      private File file;
95
96      // Location variables
97      String lat = "";
98      String lon = "";
99
100     // Weather variables
101     private static final String APP_ID = "b11dc521fd3aecc6374e2e331dc090e3";
102     String weather = "";
103     String units = "metric";
104     String url =
        "http://api.openweathermap.org/data/2.5/weather?lat="+lat+"&lon="+lon+"&units="+un
        its+"&appid="+APP_ID;
105
106     protected String fileLoc;
107     private static final int REQUEST_CAMERA_PERMISSION = 200;
108     private Handler mBackGroundHandler;
109     private HandlerThread mBackgroundThread;
110
111     UserData userData = new UserData();
112     // variables for experiment info and user
113     String expt;
114     String treatment;
115     String rep;
116     String username;
117     String userId;
118     String expId;
119
120     @Override
121     protected void onCreate(Bundle savedInstanceState) {
122         super.onCreate(savedInstanceState);
123         setContentView(R.layout.activity_detect_disease);
124
125         gridImage = findViewById(R.id.gridImage);
126         textureView = findViewById(R.id.texture);
127         assert textureView != null;
128         textureView.setSurfaceTextureListener(textureListener);
129         btnDetectDisease = findViewById(R.id.btnDetectDisease);
130         assert btnDetectDisease != null;
131         btnEndExperiment = findViewById(R.id.btnEndExperiment);
132
133         // Get user info
134         //username = getIntent().getStringExtra("username");
135         //userId = getIntent().getStringExtra("userid");
136         rep = getIntent().getStringExtra("rep");
137         expt = getIntent().getStringExtra("expt");
138         treatment = getIntent().getStringExtra("treatment");
139         expId = getIntent().getStringExtra("expid");
140
141         /**
142          */
```

```
143
144             username = userData.getUsername();
145             userId = userData.getUserId();
146
147         Toast.makeText(DetectDisease.this, username + " DATA FOUNd "  + userId,
            Toast.LENGTH_LONG).show();
148
149
150         // Start Location service and get lat and lon
151         startService(new Intent(DetectDisease.this,
152                 com.c00098391.planttracker.GPS.class));
153         final GPS gps = new GPS(DetectDisease.this);
154         lat = Double.toString(gps.getLatitude());
155         lon = Double.toString(gps.getLongitude());
156
157         btnEndExperiment.setOnClickListener(new View.OnClickListener() {
158             @Override
159             public void onClick(View view) {
160                 Intent intent = new Intent(DetectDisease.this,
161                         com.c00098391.planttracker.MainActivity.class);
162                 startActivity(intent);
163             }
164         });
165
166         btnDetectDisease.setOnClickListener(new View.OnClickListener() {
167             @Override
168             public void onClick(View view) {
169                 detectDisease();
170             }
171         });
172
173     }// END OF ON CREATE
174
175     TextureView.SurfaceTextureListener textureListener = new
        TextureView.SurfaceTextureListener() {
176         @Override
177         public void onSurfaceTextureAvailable(SurfaceTexture surfaceTexture, int i,
            int i1) {
178             openCamera();
179         }
180
181         @Override
182         public void onSurfaceTextureSizeChanged(SurfaceTexture surfaceTexture, int
            i, int i1) {
183
184         }
185
186         @Override
187         public boolean onSurfaceTextureDestroyed(SurfaceTexture surfaceTexture) {
188             return false;
189         }
190
191         @Override
192         public void onSurfaceTextureUpdated(SurfaceTexture surfaceTexture) {
193
194         }
195     };
196
197     private final CameraDevice.StateCallback stateCallback = new
        CameraDevice.StateCallback() {
198         @Override
199         public void onOpened(@NonNull CameraDevice camera) {
200             Log.e(TAG, "In onOpened");
201             cameraDevice = camera;
202             createCameraPreview();
203         }
204
205         @Override
206         public void onDisconnected(@NonNull CameraDevice cameraDevice) {
207             cameraDevice.close();
208         }
209
210         @Override
```

```
211             public void onError(@NonNull CameraDevice cameraDevice, int i) {
212                 cameraDevice.close();
213                // cameraDevice = null; // not used
214             }
215         };
216 
217         final CameraCaptureSession.CaptureCallback captureCallBackListener = new
            CameraCaptureSession.CaptureCallback() {
218             @Override
219             public void onCaptureCompleted(CameraCaptureSession session, CaptureRequest
                request, TotalCaptureResult result){
220                 super.onCaptureCompleted(session, request, result);
221                 Toast.makeText(DetectDisease.this, "Saved: " + file,
                    Toast.LENGTH_LONG).show();
222                 createCameraPreview();
223             }
224         };
225 
226         protected void startBackgroundThread(){
227             mBackgroundThread = new HandlerThread("Camera Background");
228             mBackgroundThread.start();
229             mBackGroundHandler = new Handler(mBackgroundThread.getLooper());
230         }
231 
232         protected void stopBackgroundThread(){
233             mBackgroundThread.quitSafely();
234             try{
235                 mBackgroundThread.join();
236                 mBackgroundThread = null;
237                 mBackGroundHandler = null;
238             }catch(InterruptedException e){
239                 e.printStackTrace();
240             }
241         }
242 
243         protected void detectDisease(){
244             if (null == cameraDevice){
245                 Log.e(TAG, "Camera Device is null");
246                 return;
247             }
248 
249             CameraManager manager = (CameraManager)
                getSystemService(Context.CAMERA_SERVICE);
250             try{
251                 CameraCharacteristics characteristics =
                    manager.getCameraCharacteristics(cameraDevice.getId());
252                 Size[] jpegSizes = null;
253                 if (characteristics != null){
254                     jpegSizes =
                        characteristics.get(CameraCharacteristics.SCALER_STREAM_CONFIGURATION_
                        MAP)
255                             .getOutputSizes(ImageFormat.JPEG);
256                 }
257                 int width = 640;
258                 int height = 640;
259                 if(jpegSizes != null && 0 < jpegSizes.length){
260                     width = jpegSizes[0].getWidth();
261                     height = jpegSizes[0].getHeight();
262                 }
263                 ImageReader reader = ImageReader.newInstance(width, height,
                    ImageFormat.JPEG, 1);
264                 List<Surface> outputSurfaces = new ArrayList<>(2);
265                 outputSurfaces.add(reader.getSurface());
266                 outputSurfaces.add(new Surface(textureView.getSurfaceTexture()));
267                 final CaptureRequest.Builder captureBuilder
268                         =
                            cameraDevice.createCaptureRequest(CameraDevice.TEMPLATE_STILL_CAPT
                            URE);
269                 captureBuilder.addTarget(reader.getSurface());
270                 captureBuilder.set(CaptureRequest.CONTROL_MODE,
                    CameraMetadata.CONTROL_MODE_AUTO);
271 
272                 // Orientation
```

```
            int rotation = getWindowManager().getDefaultDisplay().getRotation();
            captureBuilder.set(CaptureRequest.JPEG_ORIENTATION,
            ORIENTATIONS.get(rotation));

            String outPic = Constants.SCAN_IMAGE_LOCATION
                    + File.separator + Utilities.generateFilename();
            FolderUtil.createDefaultFolder(Constants.SCAN_IMAGE_LOCATION);
            fileLoc = outPic;

            final File file = new File(outPic);

            String fn = file.toString();
            Log.i("FILE TEST", "File Path after create.... " + fn);

            ImageReader.OnImageAvailableListener readerListener = new
            ImageReader.OnImageAvailableListener(){
                @Override
                public void onImageAvailable(ImageReader reader){
                    Image image = null;

                    try {
                        // Get weather
                        String units = "metric";
                        String url =
                        "http://api.openweathermap.org/data/2.5/weather?lat="+lat+"&lo
                        n="+lon+"&units="+units+"&appid="+APP_ID;
                        String weatherData =  new
                        GetWeatherTask(weather).execute(url).get();

                        image = reader.acquireLatestImage();
                        ByteBuffer buffer = image.getPlanes()[0].getBuffer();
                        byte[] bytes = new byte[buffer.capacity()];
                        buffer.get(bytes);


                        // Create bitmap and scale it to reduce amount of memory used.
                        BitmapFactory.Options options = new BitmapFactory.Options();
                        options.inSampleSize = 3;
                        Bitmap bmp = BitmapFactory.decodeByteArray(bytes, 0,
                        bytes.length,  options);
                        bmp = bmp.copy(Bitmap.Config.ARGB_8888, true);
                        ByteArrayOutputStream stream = new ByteArrayOutputStream();
                        bmp.compress(Bitmap.CompressFormat.JPEG, 70, stream);
                        byte[] byteOut = stream.toByteArray();

                        Intent intent = new Intent(DetectDisease.this,
                                DiseaseAnalysis.class);
                        intent.putExtra("username", username);
                        intent.putExtra("image", byteOut);
                        intent.putExtra("weather", weatherData);
                        intent.putExtra("lat", lat);
                        intent.putExtra("lon", lon);
                        intent.putExtra("userid", userId);
                        intent.putExtra("rep", rep);
                        intent.putExtra("treatment", treatment);
                        intent.putExtra("expt", expt);
                        intent.putExtra("expid", expId);
                        startActivity(intent);

                        //To enable saving to the device following line need to be
                        uncommented
                        // save(byteOut);
                        // }catch(IOException e){
                        //      e.printStackTrace();;
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    } catch (ExecutionException e) {
                        e.printStackTrace();
                    } finally{
                        if(image != null){
                            image.close();
                        }
                    }
```

```
339                 }
340
341                 private void save(byte[] bytes) throws IOException {
342                     OutputStream output = null;
343                     try{
344                         output = new FileOutputStream(file);
345                         output.write(bytes);
346                     }finally{
347                         if(null != output){
348                             output.close();
349                         }
350                     }
351                 }
352             };
353
354             reader.setOnImageAvailableListener(readerListener, mBackGroundHandler);
355             final CameraCaptureSession.CaptureCallback captureListener
356                     = new CameraCaptureSession.CaptureCallback() {
357                 @Override
358                 public void onCaptureCompleted(CameraCaptureSession session,
                    CaptureRequest request, TotalCaptureResult result) {
359                     super.onCaptureCompleted(session, request, result);
360
361                 }
362             };
363
364             cameraDevice.createCaptureSession(outputSurfaces, new
                CameraCaptureSession.StateCallback(){
365                 @Override
366                 public void onConfigured(CameraCaptureSession session){
367                     try{
368                         session.capture(captureBuilder.build(), captureListener,
                            mBackGroundHandler);
369                     }catch(CameraAccessException e){
370                         e.printStackTrace();
371                     }
372                 }
373
374                 @Override
375                 public void onConfigureFailed(@NonNull CameraCaptureSession
                    cameraCaptureSession){
376
377                 }
378             }, mBackGroundHandler);
379         }catch(CameraAccessException e){
380             e.printStackTrace();
381         }
382     }
383
384     protected void createCameraPreview(){
385         try{
386             SurfaceTexture texture = textureView.getSurfaceTexture();
387             assert texture != null;
388             texture.setDefaultBufferSize(imageDimension.getWidth(),
                imageDimension.getHeight()-100);
389             Surface surface = new Surface(texture);
390
391             captureRequestBuilder =
                cameraDevice.createCaptureRequest(CameraDevice.TEMPLATE_PREVIEW);
392             captureRequestBuilder.addTarget(surface);
393             cameraDevice.createCaptureSession(Arrays.asList(surface), new
                CameraCaptureSession.StateCallback(){
394                 @Override
395                 public void onConfigured(@NonNull CameraCaptureSession
                    cameraCaptureSession){
396                     if(null == cameraDevice){
397                         return;
398                     }
399
400                     cameraCaptureSessions = cameraCaptureSession;
401                     updatePreview();
402                 }
403
```

```
404                 @Override
405                 public void onConfigureFailed(@NonNull CameraCaptureSession
                    cameraCaptureSession){
406                     Toast.makeText(DetectDisease.this, "Configuration changed",
407                             Toast.LENGTH_SHORT).show();
408                 }
409             }, null);
410         }catch(CameraAccessException e){
411             e.printStackTrace();
412         }
413     }
414
415     private void openCamera(){
416         CameraManager manager = (CameraManager)
            getSystemService(Context.CAMERA_SERVICE);
417         Log.e(TAG, "in openCamera()");
418         try{
419             cameraId = manager.getCameraIdList()[0];
420             CameraCharacteristics characteristics =
                manager.getCameraCharacteristics(cameraId);
421             StreamConfigurationMap map
422                     =
                    characteristics.get(CameraCharacteristics.SCALER_STREAM_CONFIGURAT
                    ION_MAP);
423             assert map != null;
424             imageDimension = map.getOutputSizes(SurfaceTexture.class)[0];
425
426
427             if (ActivityCompat.checkSelfPermission(this, Manifest.permission.CAMERA)
428                     != PackageManager.PERMISSION_GRANTED
429                     && ActivityCompat.checkSelfPermission(this,
430                     Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
                    PackageManager.PERMISSION_GRANTED){
431
432                 ActivityCompat.requestPermissions(DetectDisease.this, new String[]{
433                                 Manifest.permission.CAMERA,
434                                 Manifest.permission.WRITE_EXTERNAL_STORAGE},
435                         REQUEST_CAMERA_PERMISSION);
436                 return;
437             }
438             manager.openCamera(cameraId, stateCallback, null);
439         }catch(CameraAccessException e){
440             e.printStackTrace();
441         }
442         Log.e(TAG, "in Open Camera (permissions");
443     }
444
445     protected void updatePreview(){
446         if(null == cameraDevice){
447             Log.e(TAG, "Update Preview Error..");
448         }
449         captureRequestBuilder.set(CaptureRequest.CONTROL_MODE,
            CameraMetadata.CONTROL_MODE_AUTO);
450         try{
451             cameraCaptureSessions.setRepeatingRequest(captureRequestBuilder.build(),
452                     null, mBackGroundHandler);
453         }catch(CameraAccessException e){
454             e.printStackTrace();
455         }
456     }
457
458     private void closeCamera(){
459         if(null != cameraDevice){
460             cameraDevice.close();
461             cameraDevice = null;
462         }
463         if (null != imageReader){
464             imageReader.close();
465             imageReader = null;
466         }
467     }
468
469     @Override
```

```
    public void onRequestPermissionsResult(int requestCode, @NonNull String[]
    permissions,
                                           @NonNull int[] grantResults){
        if (requestCode == REQUEST_CAMERA_PERMISSION) {
            if (grantResults[0] == PackageManager.PERMISSION_DENIED) {
                // Close app
                Toast.makeText(DetectDisease.this,
                        "Sorry, you cannot use this app without granting
                        permission", Toast.LENGTH_LONG)
                        .show();
                finish();
            }
        }
    }

    @Override
    protected void onResume(){
        super.onResume();
        Log.e(TAG, "in onResume");
        startBackgroundThread();
        if(textureView.isAvailable()){
            openCamera();
        }else{
            textureView.setSurfaceTextureListener(textureListener);
        }
    }
    @Override
    protected void onPause(){
        Log.e(TAG, "in OnPause");
        closeCamera();
        stopBackgroundThread();
        super.onPause();
    }

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

        private String weather;

        public GetWeatherTask(String weather){
            this.weather = weather;
        }

        @Override
        protected  String doInBackground(String... strings){
            String weather = "UNDEFINED";

            try {
                URL url = new URL(strings[0]);
                HttpURLConnection urlConnection = (HttpURLConnection)
                url.openConnection();

                InputStream stream = new
                BufferedInputStream(urlConnection.getInputStream());
                BufferedReader bufferedReader = new BufferedReader(new
                InputStreamReader(stream));
                StringBuilder builder = new StringBuilder();

                String inputString;
                while ((inputString = bufferedReader.readLine()) != null){
                    builder.append(inputString);
                }

                JSONObject topLevel = new JSONObject(builder.toString());
                JSONObject main = topLevel.getJSONObject("main");
                String temp = String.valueOf(main.getDouble("temp"));
                String humidity = String.valueOf(main.getInt("humidity"));

                String overview = topLevel.getJSONArray("weather")
                        .getJSONObject(0).get("main").toString();
                String desc = topLevel.getJSONArray("weather")
                        .getJSONObject(0).get("description").toString();

                weather = temp + "C, " + overview + "(" + desc + ")" + ", Humidity:
```

```
                    " +humidity;
538
539                 urlConnection.disconnect();
540             }catch (IOException | JSONException e){
541                 e.printStackTrace();
542             }
543             return weather;
544         }
545
546         @Override
547         protected void onPostExecute(String temp) {
548             weather = "Current Weather " + temp;
549         }
550     }
551 }
552
```

DiseaseAnalysis.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * Created by Student: Darran Gahan
5    * Student Number: C00098391
6    *
7    * Class is used to analyse a bitmap image and detect a colour selected by a user
8    * by touching the location on the screen.
9    */
10
11  import android.annotation.SuppressLint;
12  import android.content.DialogInterface;
13  import android.content.Intent;
14  import android.graphics.Bitmap;
15  import android.graphics.BitmapFactory;
16  import android.graphics.Color;
17  import android.graphics.Rect;
18  import android.graphics.drawable.BitmapDrawable;
19  import android.graphics.drawable.Drawable;
20  import android.os.AsyncTask;
21  import android.support.constraint.ConstraintLayout;
22  import android.support.v7.app.AppCompatActivity;
23  import android.os.Bundle;
24  import android.util.DisplayMetrics;
25  import android.util.Log;
26  import android.view.MotionEvent;
27  import android.view.View;
28  import android.widget.Button;
29  import android.widget.ImageView;
30  import android.widget.LinearLayout;
31  import android.widget.TextView;
32  import android.widget.Toast;
33
34  import com.google.android.gms.flags.Flag;
35
36  import org.json.JSONException;
37  import org.json.JSONObject;
38  import org.opencv.android.Utils;
39  import org.opencv.core.Mat;
40  import org.opencv.core.Size;
41  import org.opencv.imgproc.Imgproc;
42
43  import java.io.BufferedReader;
44  import java.io.BufferedWriter;
45  import java.io.InputStream;
46  import java.io.InputStreamReader;
47  import java.io.OutputStream;
48  import java.io.OutputStreamWriter;
49  import java.net.HttpURLConnection;
50  import java.net.URL;
51  import java.net.URLEncoder;
52  import java.nio.charset.StandardCharsets;
53  import java.util.HashMap;
54  import java.util.Iterator;
55
56  public class DiseaseAnalysis extends AppCompatActivity {
57
58      // Attributes for view
59      ImageView imgView;
60      TextView tvAnalysis;
61      Button btnUploadAnalysis;
62
63      // Variables for use with intent
64      String username;
65      String userId;
66      String lat = "";
67      String lon = "";
68      String weather = "";
69      String rep;
70      String treatment;
71      String expt;
72      String expId;
73
```

```
74      UserData userData = new UserData();
75
76
77      @SuppressLint("ClickableViewAccessibility")
78      @Override
79      protected void onCreate(Bundle savedInstanceState) {
80          super.onCreate(savedInstanceState);
81
82          setContentView(R.layout.activity_disease_analysis);
83
84          btnUploadAnalysis = findViewById(R.id.btnUploadAnalysis);
85          tvAnalysis = findViewById(R.id.tvAnalysis);
86
87          imgView = findViewById(R.id.imgView);
88
89          imgView.getHeight();
90          imgView.getWidth();
91
92          imgView.setDrawingCacheEnabled(true);
93          imgView.buildDrawingCache(true);
94
95          username = userData.getUsername();
96          userId = userData.getUserId();
97
98
99          // All variables passed from the detect stage
100         expt = getIntent().getStringExtra("expt");
101         //username = getIntent().getStringExtra("username");
102         //userId = getIntent().getStringExtra("userid");
103         weather = getIntent().getStringExtra("weather");
104         lat = getIntent().getStringExtra("lat");
105         lon = getIntent().getStringExtra("lon");
106         rep = getIntent().getStringExtra("rep");
107         treatment = getIntent().getStringExtra("treatment");
108         expt = getIntent().getStringExtra("expt");
109         expId = getIntent().getStringExtra("expid");
110
111         final byte[] byteArray = getIntent().getByteArrayExtra("image");
112         final Bitmap bm = BitmapFactory.decodeByteArray(byteArray, 0,
            byteArray.length);
113
114         imgView.setImageBitmap(bm);
115
116         Mat mat = new Mat();
117         Utils.bitmapToMat(bm, mat);
118         final Mat hsvMat = mat;
119
120         // OnTouch to get location to in turn get color at that location
121         imgView.setOnTouchListener(new View.OnTouchListener() {
122             @Override
123             public boolean onTouch(View view, MotionEvent motionEvent) {
124
125                 //convert location on image view to location on bitmap
126                 float xRatio = (float)bm.getWidth() / imgView.getWidth();
127                 float xPos = motionEvent.getX() * xRatio;
128                 float yRatio = (float)bm.getHeight() / imgView.getHeight();
129                 float yPos = motionEvent.getY() * yRatio;
130
131                 int xP = (int) xPos;
132                 int yP = (int) yPos;
133
134
135                 // pixel at onTouch location
136                 int pixel = bm.getPixel(xP, yP);
137
138                 // Get RGB values of pixel at given location
139                 int redValue = Color.red(pixel);
140                 int blueValue = Color.blue(pixel);
141                 int greenValue = Color.green(pixel);
142
143                 // Get the average color around the selected pixel
144                 // this takes a 9 X 9 grid around the selected location
145                 // and averages the color.
```

```
146                 int red;
147                 int green;
148                 int blue;
149
150                 int totalRed = 0;
151                 int totalGreen = 0;
152                 int totalBlue = 0;
153
154                 for (int i = yP-4; i <= yP + 4; i++){
155
156                     for (int k = xP - 4; k <= xP + 4; k ++){
157
158                         int f = bm.getPixel(i, k);
159                         red = Color.red(f);
160                         green = Color.green(f);
161                         blue = Color.blue(f);
162
163                         totalRed += red;
164                         totalGreen += green;
165                         totalBlue += blue;
166                     }
167                 }
168
169                 int finalRed = totalRed / 81;
170                 int finalGreen = totalGreen / 81;
171                 int finalBlue = totalBlue / 81;
172
173                 // scales down size the Mat object of the image to
174                 // speed up the analysis
175                 Size sz = new Size(400 ,400);
176                 Imgproc.resize(hsvMat, hsvMat, sz);
177                 Imgproc.cvtColor(hsvMat, hsvMat, Imgproc.COLOR_RGB2BGR);
178                 Imgproc.cvtColor(hsvMat, hsvMat, Imgproc.COLOR_BGR2HLS);
179
180                 // array of the color for onTouch
181                 int [] hsl = new int[3];
182
183                 // Get the HSL values from the RGB values
184                 // HSL uses lightness which allows us to identify white very easily
185                 rgb2hsl(finalRed, finalGreen, finalBlue, hsl);
186
187
188                 HashMap<String, Integer> colorList = new HashMap<>();
189
190                 colorList.put("bad", 0);
191
192                 int hsvHeight = hsvMat.rows();
193                 int hsvWidth = hsvMat.cols();
194                 int total = hsvHeight * hsvWidth;
195
196                 String color = "";
197                 int colorCount = 0;
198                 double L = hsl[2] / 2.0; // L is divied by to because l of hsl works
                    on a 360 degreescale
199                                          // degree scale but opencv uses 180 decrees
                                             scale..
200
201                 // Set upper and lower bounds for the color range.
202                 double lowerH = hsl[0] -15;
203                 double upperH = hsl[0] + 15;
204                 double lowerS = hsl[1] - 15;
205                 double upperS = hsl[1] + 15;
206                 double lowerL = L - 18.0;
207                 double upperL = L + 18.0;
208
209                 for (int i = 0; i < hsvHeight; i++){
210                     for (int k = 0; k < hsvWidth; k++) {
211
212                         double [] hslMat = hsvMat.get(i, k);
213                         double h = hslMat[0]; // open cv uses 180 not 360 for memory
                            reasons
214                         double s = hslMat[1];
215                         double l = hslMat[2];
```

```
216
217                         // White
218                         if ((l >= 90) && (l <= 100)) {
219                             color = "white";
220                         }
221                         // Bad
222                         else if ((h >= lowerH && h <= upperH) && (s >= lowerS && s
                            <= upperS) && (l >= lowerL && l <= upperL)) {
223                             color = "bad";
224                         }
225
226                         /**
227                          * Have white and bad ....
228                          */
229                         if (colorList.containsKey(color)) {
230                             colorCount = colorList.get(color);
231                             colorCount++;
232                             colorList.put(color, colorCount);
233
234                         } else {
235                             colorCount = 1;
236                             colorList.put(color, colorCount);
237                         }
238
239                     }
240                 }
241                 // Anything that's not white and not bad == leaf.
242
243                 double disease;
244                 int whiteCount = colorList.get("white");
245                 int badCount = colorList.get("bad");
246                 double leaf = total - whiteCount;
247                 disease = (badCount / leaf) * 100.0;
248
249                 String analysis = String.format("%.2f", disease);
250
251                 Intent intent = new Intent(DiseaseAnalysis.this,
252                         com.c00098391.planttracker.DiseaseResult.class);
253                 intent.putExtra("disease", analysis);
254                 intent.putExtra("image", byteArray);
255                 intent.putExtra("lat", lat);
256                 intent.putExtra("lon", lon);
257                 intent.putExtra("weather", weather);
258                 intent.putExtra("username", username);
259                 intent.putExtra("userid", userId);
260                 intent.putExtra("rep", rep);
261                 intent.putExtra("treatment", treatment);
262                 intent.putExtra("expt", expt);
263                 intent.putExtra("expid", expId);
264                 intent.putExtra("red", redValue);
265                 intent.putExtra("blue", blueValue);
266                 intent.putExtra("green", greenValue);
267                 startActivity(intent);
268
269                 return false;
270             }
271         });
272
273     }
274
275     /**
276      * Method to convert color from RGB to HSL color space
277      * Method takes an input for rgb and returns the hsl values
278      */
279     private void rgb2hsl(int r, int g, int b, int hsl[]) {
280
281         float var_R = ( r / 255f );
282         float var_G = ( g / 255f );
283         float var_B = ( b / 255f );
284
285         float var_Min;    //Min. value of RGB
286         float var_Max;    //Max. value of RGB
287         float del_Max;    //Delta RGB value
```

```
288
289             if (var_R > var_G)
290             { var_Min = var_G; var_Max = var_R; }
291             else
292             { var_Min = var_R; var_Max = var_G; }
293
294             if (var_B > var_Max) var_Max = var_B;
295             if (var_B < var_Min) var_Min = var_B;
296
297             del_Max = var_Max - var_Min;
298
299             float H = 0, S, L;
300             L = ( var_Max + var_Min ) / 2f;
301
302             if ( del_Max == 0 ) { H = 0; S = 0; } // gray
303             else {                                //Chroma
304                 if ( L < 0.5 )
305                     S = del_Max / ( var_Max + var_Min );
306                 else
307                     S = del_Max / ( 2 - var_Max - var_Min );
308
309                 float del_R = ( ( ( var_Max - var_R ) / 6f ) + ( del_Max / 2f ) ) /
                    del_Max;
310                 float del_G = ( ( ( var_Max - var_G ) / 6f ) + ( del_Max / 2f ) ) /
                    del_Max;
311                 float del_B = ( ( ( var_Max - var_B ) / 6f ) + ( del_Max / 2f ) ) /
                    del_Max;
312
313                 if ( var_R == var_Max )
314                     H = del_B - del_G;
315                 else if ( var_G == var_Max )
316                     H = ( 1 / 3f ) + del_R - del_B;
317                 else if ( var_B == var_Max )
318                     H = ( 2 / 3f ) + del_G - del_R;
319                 if ( H < 0 ) H += 1;
320                 if ( H > 1 ) H -= 1;
321             }
322             hsl[0] = (int)(360*H);
323             hsl[1] = (int)(S*100);
324             hsl[2] = (int)(L*100);
325
326         }
327     }
328
```

DiseaseResults.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * Created by Student: Darran Gahan
5    * Student Number: C00098391
6    *
7    * Class is used to display the analysis to the user it displays the image, the
     analysis
8    * and the color that the user selected.
9    */
10
11  import android.content.Intent;
12  import android.graphics.Bitmap;
13  import android.graphics.BitmapFactory;
14  import android.graphics.Color;
15  import android.os.AsyncTask;
16  import android.support.v7.app.AppCompatActivity;
17  import android.os.Bundle;
18  import android.util.Base64;
19  import android.util.Log;
20  import android.view.View;
21  import android.widget.Button;
22  import android.widget.ImageView;
23  import android.widget.TextView;
24  import android.widget.Toast;
25
26  import org.json.JSONException;
27  import org.json.JSONObject;
28
29  import java.io.BufferedInputStream;
30  import java.io.BufferedReader;
31  import java.io.BufferedWriter;
32  import java.io.IOException;
33  import java.io.InputStream;
34  import java.io.InputStreamReader;
35  import java.io.OutputStream;
36  import java.io.OutputStreamWriter;
37  import java.net.HttpURLConnection;
38  import java.net.URL;
39  import java.net.URLEncoder;
40  import java.nio.charset.StandardCharsets;
41  import java.text.SimpleDateFormat;
42  import java.util.Calendar;
43  import java.util.Iterator;
44  import java.util.concurrent.ExecutionException;
45
46  public class DiseaseResult extends AppCompatActivity {
47
48      // Vew Attributes
49      ImageView imgView;
50      Button btnUploadAnalysis, btnEndExperiment;
51      TextView tvAnalysis, tvColor;
52
53      // Variables for uploading
54      static InputStream inputStream = null;
55      static String json;
56      static JSONObject jObj = null;
57      static String error = "";
58
59      UserData userData = new UserData();
60
61      // variables for analysis information
62      String username;
63      String userId;
64      String rep;
65      String treatment;
66      String expt;
67      String expId;
68      int red;
69      int green;
70      int blue;
71      String lat = "";
72      String lon = "";
```

```
    String weather = "";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_disease_result);

        imgView = findViewById(R.id.ivAnalysis);
        btnUploadAnalysis = findViewById(R.id.btnUploadAnalysis);
        tvAnalysis = findViewById(R.id.tvAnalysis);
        tvColor = findViewById(R.id.tvColor);


        username = UserData.getUsername();
        userId = UserData.getUserId();

        String analysis = getIntent().getStringExtra("disease");
        lat = getIntent().getStringExtra("lat");
        lon = getIntent().getStringExtra("lon");
        weather = getIntent().getStringExtra("weather");
       // username = getIntent().getStringExtra("username");
        //userId = getIntent().getStringExtra("userid");
        rep = getIntent().getStringExtra("rep");
        treatment = getIntent().getStringExtra("treatment");
        expt = getIntent().getStringExtra("expt");
        expId = getIntent().getStringExtra("expid");
        red = getIntent().getIntExtra("red", 0);
        green = getIntent().getIntExtra("green", 0);
        blue=getIntent().getIntExtra("blue", 0);

        // Set textview to selected color
        tvColor.setBackgroundColor(Color.rgb(red, green,blue));
        Toast.makeText(DiseaseResult.this, username + " DATA FOUNd "  + userId,
        Toast.LENGTH_LONG).show();

        final byte[] byteArray = getIntent().getByteArrayExtra("image");
        Bitmap bm = BitmapFactory.decodeByteArray(byteArray, 0, byteArray.length);

        tvAnalysis.setText(analysis);

        imgView.setImageBitmap(bm);

        // Base64 encode image
        String encodedImg = Base64.encodeToString(byteArray, Base64.DEFAULT);

        // Set format for time and date
        final SimpleDateFormat df = new SimpleDateFormat("dd-MM-yyyy");
        final SimpleDateFormat tf = new SimpleDateFormat("hh:mm:ss");

        // Create strings for time and date
        String date = df.format(Calendar.getInstance().getTime());
        String time = tf.format(Calendar.getInstance().getTime());

        final String [] analysisDetails = new String[13];
        analysisDetails[0] = date;
        analysisDetails[1] = time;
        analysisDetails[2] = encodedImg;
        analysisDetails[3] = lat;
        analysisDetails[4] = lon;
        analysisDetails[5] = weather;
        analysisDetails[6] = analysis;
        analysisDetails[7] = userId;
        analysisDetails[8] = username;
        analysisDetails[9] = rep;
        analysisDetails[10] = treatment;
        analysisDetails[11] = expt;
        analysisDetails[12] = expId;

        // Upload analysis to database with onClick
        btnUploadAnalysis.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

```

```
                UploadAnalysis ud = new UploadAnalysis();
                ud.execute(analysisDetails);

            }
        });

    }

    // Async task for sending analysis
    public class UploadAnalysis extends AsyncTask<String, Void, JSONObject> {

        @Override
        protected JSONObject doInBackground(String... args){

            try{
                URL url = new
                URL("http://www.c0009839.candept.com/API/AnalysisUpload.php");

                // put params in a JSON Object
                JSONObject dataParams = new JSONObject();
                dataParams.put("date", args[0]);
                dataParams.put("time", args[1]);
                dataParams.put("image", args[2]);
                dataParams.put("lat", args[3]);
                dataParams.put("lon", args[4]);
                dataParams.put("weather", args[5]);
                dataParams.put("analysis", args[6]);
                dataParams.put("userid", args[7]);
                dataParams.put("username", args[8]);
                dataParams.put("rep", args[9]);
                dataParams.put("treatment", args[10]);
                dataParams.put("expt", args[11]);
                dataParams.put("expid", args[12]);

                // Set up connection
                HttpURLConnection conn = (HttpURLConnection) url.openConnection();
                conn.setReadTimeout(15000);
                conn.setConnectTimeout(15000);
                conn.setRequestMethod("POST");
                conn.setDoInput(true);
                conn.setDoOutput(true);

                //send date
                OutputStream os = conn.getOutputStream();
                BufferedWriter writer = new BufferedWriter(
                        new OutputStreamWriter(os, StandardCharsets.UTF_8));
                writer.write(getPostDateString(dataParams));

                writer.flush();
                writer.close();
                os.close();

                // Get Response
                int responseCode = conn.getResponseCode();
                error = String.valueOf(conn.getResponseCode());

                if (responseCode == HttpURLConnection.HTTP_OK){
                    inputStream = conn.getInputStream();
                    BufferedReader in  = new BufferedReader(new
                    InputStreamReader(inputStream));
                    StringBuilder sb = new StringBuilder();
                    String line;

                    while(null!= (line = in.readLine())){
                        sb.append(line).append("\n");
                    }
                    in.close();
                    inputStream.close();
                    json = sb.toString();
                    Log.i("API Camera: ", json);
                }
                else{
                    Log.e("Buffer Error", "Error Getting Result " +responseCode);
```

```
                }
                try{
                    jObj = new JSONObject(json);
                    jObj.put("error_code", error);
                }catch(JSONException e){
                    Log.e("JSON Parser", "Error Parsing Data " + e.toString());
                }
            }catch(Exception e){
                Log.e("Exception: ", "Overall Try Block " + e.toString());
            }
            return jObj;
        }// end of doInBackground

        @Override
        protected void onPostExecute(JSONObject result){

            try {

                if (result != null){

                    String uploadSuccess = result.getString("message");
                    if (uploadSuccess.equals("Successfully uploaded analysis")){
                        Toast.makeText(getApplicationContext(), result.getString(
                                "message"), Toast.LENGTH_LONG).show();

                        Intent intent = new Intent(DiseaseResult.this,
                                com.c00098391.planttracker.DetectDisease.class);
                        intent.putExtra("username", username);
                        intent.putExtra("userid", userId);
                        intent.putExtra("rep", rep);
                        intent.putExtra("expt", expt);
                        intent.putExtra("treatment", treatment);
                        intent.putExtra("expid", expId);
                        startActivity(intent);

                    }else{
                        Toast.makeText(getApplicationContext(), result.getString(
                                "message"), Toast.LENGTH_LONG).show();
                    }
                }else{
                    Toast.makeText(getApplicationContext(),
                            "Unable upload results", Toast.LENGTH_LONG).show();
                }
            }catch(JSONException e){
                e.printStackTrace();
            }
        }
    }

    // Turn json object to string for post
    public String getPostDateString(JSONObject params) throws Exception{

        StringBuilder result = new StringBuilder();
        boolean first = true;
        Iterator<String> itr = params.keys();

        while(itr.hasNext()){
            String key = itr.next();
            Object value = params.get(key);
            if(first){
                first = false;
            }else{
                result.append("&");
            }
            result.append(URLEncoder.encode(key, "UTF-8"));
            result.append("=");
            result.append(URLEncoder.encode(value.toString(), "UTF-8"));
        }
        return result.toString();
    }
}

```

ContinueExperiment.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * Created by Student: Darran Gahan
5    * Student Number: C00098391
6    *
7    * Class is used to allow a user to continue an experiment they previously created
8    *
9    */
10
11
12  import android.content.Intent;
13
14  import android.support.v7.app.AppCompatActivity;
15  import android.os.Bundle;
16
17  import android.view.View;
18  import android.widget.AdapterView;
19  import android.widget.ArrayAdapter;
20  import android.widget.ListView;
21  import android.widget.TextView;
22  import java.util.ArrayList;
23
24
25  public class ContinueExperiment extends AppCompatActivity {
26
27      // View attributes
28      TextView tvJson;
29      ListView lvExps;
30
31      // Array list for experiment details
32      ArrayList<String> exps = new ArrayList<>();
33      ArrayList<String> expIds = new ArrayList<>();
34
35      // user details
36  //    String username;
37  //    String userId;
38
39      // Static class for Userdata..
40      UserData userData = new UserData();
41
42      @Override
43      protected void onCreate(Bundle savedInstanceState) {
44          super.onCreate(savedInstanceState);
45          setContentView(R.layout.activity_continue_experiment);
46
47          tvJson = findViewById(R.id.tvJson);
48          lvExps = findViewById(R.id.lvExps);
49
50          //username = getIntent().getStringExtra("username");
51          //userId = getIntent().getStringExtra("userid");
52          exps = getIntent().getStringArrayListExtra("exps");
53          expIds = getIntent().getStringArrayListExtra("expids");
54
55          String username = UserData.getUsername();
56          String userId = UserData.getUserId();
57
58          final ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
59                  android.R.layout.simple_list_item_1, android.R.id.text1
60                  , exps);
61          lvExps.setAdapter(adapter);
62
63          lvExps.setOnItemClickListener(new AdapterView.OnItemClickListener() {
64              @Override
65              public void onItemClick(AdapterView<?> adapterView, View view, int i,
                long l) {
66
67                  // get selected item
68                  String value = adapter.getItem(i);
69                  // get id of selected item
70                  String expId = expIds.get(i);
71
72
```

```
                    // Details for next activity
                    String[] parts = value.split(" ");
                    String rep = parts[0];
                    String treatment = parts[1];
                    String expt = parts[2] + " " + parts[3];
                    Intent intent = new Intent(ContinueExperiment.this,
                            com.c00098391.planttracker.DetectDisease.class);
                    intent.putExtra("rep", rep);
                    intent.putExtra("expt", expt);
                    intent.putExtra("treatment", treatment);
                   // intent.putExtra("username", username);
                  //  intent.putExtra("userid", userId);
                    intent.putExtra("expid", expId);
                    startActivity(intent);

                }
            });


        } //  end on create

    }

```

FolderUtill.java

```
1   package com.c00098391.planttracker;
2
3   import java.io.File;
4
5   /**
6    * Created by Darran Gahan
7    *
8    * Class is used to check if a folder exists on the device for saving images
9    * and create it if not.
10   */
11
12  public class FolderUtil {
13
14      private FolderUtil(){
15
16      }
17
18      public static void createDefaultFolder(String dirPath){
19
20          File directory = new File(dirPath);
21          if (!directory.exists()){
22              directory.mkdir();
23          }
24      }
25
26      public boolean checkFileExists(String filePath){
27          File file = new File(filePath);
28          return file.exists();
29      }
30  }
```

Constants.java

```
1   package com.c00098391.planttracker;
2
3   import android.os.Environment;
4
5   import java.io.File;
6
7   /**
8    * Created by Darran Gahan
9    * Student Number: C00098391
10   *
11   * Class is used to store constants mainly for file naming purposes
12   */
13
14  public class Constants {
15
16      private Constants(){
17
18      }
19
20      public static final String SCAN_IMAGE_LOCATION =
        Environment.getExternalStorageDirectory() +
21              File.separator + "PlantTracker/";
22
23  }
24
```

Utilities.java

```
1  package com.c00098391.planttracker;
2
3  import android.annotation.SuppressLint;
4
5  import java.text.SimpleDateFormat;
6  import java.util.Date;
7
8  /**
9   * Student Name: Darran Gahan
10  * Student Number: C00098391;
11  *
12  * This class s used to generate a file name for captured image when saving to
13  * users device
14  */
15
16 public class Utilities {
17
18     private Utilities(){
19
20     }
21
22     public static String generateFilename(){
23         @SuppressLint("SimpleDateFormat")SimpleDateFormat sdf
24                 = new SimpleDateFormat("yyyy-MM-dd_HH-mm-ss");
25         return "PlantTracker" + sdf.format(new Date()) + ".jpg";
26     }
27 }
28
```

UserData.java

```
1  package com.c00098391.planttracker;
2
3
4
5  public class UserData {
6
7      private static String username;
8      private static String userId;
9
10     public static String getUsername() {
11         return username;
12     }
13
14     public static void setUsername(String username) {
15         UserData.username = username;
16     }
17
18     public static String getUserId() {
19         return userId;
20     }
21
22     public static void setUserid(String userId) {
23         UserData.userId = userId;
24     }
25 }
26
```

GPS.java

```
1   package com.c00098391.planttracker;
2
3   /**
4    * Student: Darran Gahan
5    * Student Number: C00098391
6    *
7    * Class is used to get the GPS location of the device
8    *
9    */
10
11  import android.annotation.SuppressLint;
12  import android.app.AlertDialog;
13  import android.app.Service;
14  import android.content.Context;
15  import android.content.DialogInterface;
16  import android.content.Intent;
17  import android.location.Location;
18  import android.location.LocationListener;
19  import android.location.LocationManager;
20  import android.os.Bundle;
21  import android.os.IBinder;
22  import android.provider.Settings;
23  import android.support.annotation.Nullable;
24  import android.util.Log;
25
26  public class GPS extends Service implements LocationListener {
27
28      private final Context mContext;
29      // Check GPS status
30      boolean isGPSEnabled = false;
31      // Check network status
32      boolean isNetworkEnabled = false;
33      // flag for GPS status
34      boolean canGetLocation = false;
35      Location location;
36      double latitude;
37      double longitude;
38
39      // Set the distance to update GPS in meters..
40      private static final long MIN_DISTANCE_CHANGE_FOR_UPDATE = 1;
41      // Thr minimum time between updates in milliseconds
42      private static final long MIN_TIME_BETWEEN_UPDATES = 1000 * 10; // 10 seconds, 1
        min = 1000 * 60 * 1...
43      // Declaring a Location Manager
44      protected LocationManager locationManager;
45
46      public GPS(Context context){
47          this.mContext = context;
48          getLocation();
49      }
50      @SuppressLint("MissingPermission")
51      public Location getLocation(){
52
53          try{
54              locationManager = (LocationManager)
                mContext.getSystemService(LOCATION_SERVICE);
55
56              // Get GPS Status
57              isGPSEnabled =
                locationManager.isProviderEnabled(LocationManager.GPS_PROVIDER);
58
59              // Get Network Status
60              isNetworkEnabled =
                locationManager.isProviderEnabled(LocationManager.NETWORK_PROVIDER);
61
62              if (!isGPSEnabled && isNetworkEnabled){
63                  // no provider enabled
64              }else{
65                  this.canGetLocation = true;
66                  if(isNetworkEnabled){
67                      locationManager.requestLocationUpdates(
68                              LocationManager.NETWORK_PROVIDER,
69                              MIN_TIME_BETWEEN_UPDATES,
```

```
                            MIN_DISTANCE_CHANGE_FOR_UPDATE,
                            this);
                    Log.d("Network Enabled", "Network Enabled");
                    if(locationManager != null){
                        location = locationManager.getLastKnownLocation(
                                LocationManager.NETWORK_PROVIDER);
                        if (location != null){
                            latitude = location.getLatitude();
                            longitude = location.getLongitude();
                        }
                    }
                }
                if(isGPSEnabled){
                    if (location == null){
                        locationManager.requestLocationUpdates(
                                LocationManager.GPS_PROVIDER,
                                MIN_TIME_BETWEEN_UPDATES,
                                MIN_DISTANCE_CHANGE_FOR_UPDATE,
                                this);
                        Log.d("GPS Enabled", "GPS Enabled");
                        if (locationManager != null){
                            location = locationManager.getLastKnownLocation(
                                    LocationManager.GPS_PROVIDER);
                            if (location != null){
                                latitude = location.getLatitude();
                                longitude = location.getLongitude();
                            }
                        }
                    }
                }
            }

        } catch (Exception e){
            e.printStackTrace();
        }
        return location;
    }

    // Stop using GPS listener
    public void stopUsingGPS(){
        if(locationManager != null){
            locationManager.removeUpdates(GPS.this);
        }
    }

    // Get Latitude
    public double getLatitude(){
        if (location != null){
            latitude = location.getLatitude();
        }
        return latitude;
    }

    // Get Longitude
    public double getLongitude(){
        if (location != null){
            longitude = location.getLongitude();
        }
        return longitude;
    }

    public boolean canGetLocation(){
        return this.canGetLocation;
    }

    public void showSettingsAlert(){
        AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);
        // Dialog title
        alertDialog.setTitle("GPS settings");
        // Dialog message
        alertDialog.setPositiveButton("Settings", new
        DialogInterface.OnClickListener() {
            @Override
```

```
            public void onClick(DialogInterface dialogInterface, int i) {
                Intent intent = new Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                mContext.startActivity(intent);
            }
        });
        // if cancel button is pressed
        alertDialog.setNegativeButton("Cancel", new
        DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialogInterface, int i) {
                dialogInterface.cancel();
            }
        });
        alertDialog.show();
    }

    @Nullable
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }

    @SuppressLint("MissingPermission")
    @Override
    public void onLocationChanged(Location location) {

        if(isNetworkEnabled){
            locationManager.requestLocationUpdates(
                    LocationManager.NETWORK_PROVIDER,
                    MIN_TIME_BETWEEN_UPDATES,
                    MIN_DISTANCE_CHANGE_FOR_UPDATE,
                    this);
            Log.d("Network Enabled", "Network Enabled");
            if(locationManager != null){
                location = locationManager.getLastKnownLocation(
                        LocationManager.NETWORK_PROVIDER);
                if (location != null){
                    latitude = location.getLatitude();
                    longitude = location.getLongitude();
                }
            }
        }
        if(isGPSEnabled){
            if (location == null){
                locationManager.requestLocationUpdates(
                        LocationManager.GPS_PROVIDER,
                        MIN_TIME_BETWEEN_UPDATES,
                        MIN_DISTANCE_CHANGE_FOR_UPDATE,
                        this);
                Log.d("GPS Enabled", "GPS Enabled");
                if (locationManager != null){
                    location = locationManager.getLastKnownLocation(
                            LocationManager.GPS_PROVIDER);
                    if (location != null){
                        latitude = location.getLatitude();
                        longitude = location.getLongitude();
                    }
                }
            }
        }

    }

    @Override
    public void onStatusChanged(String s, int i, Bundle bundle) {

    }

    @Override
    public void onProviderEnabled(String s) {

    }

```

```
214         @Override
215         public void onProviderDisabled(String s) {
216
217         }
218
219     }
220
```

Web Application

index.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
require 'db.php';


if($_SERVER['REQUEST_METHOD'] == 'POST'){

    require 'userLogin.php';
}

?>

<!DOCTYPE html>
<html>

    <head>
        <title>Plant Tracker Login</title>
        <?php include 'css/css.html';?>
    </head>
<body>

    <div class="form">


            <div id ="login">

                <h1>Plant Tracker Login</h1>

                <form action ="index.php" autocomplete="off" method="POST">

                    <div class="field-wrap">
                        <label>
                            Username<span class="req">*</span>
                        </label>
                        <input type="text" required name="username"
                        autocomplete="off"/>
                    </div>
                    <div class="field-wrap">
                        <label>
                            Password<span class="req">*</span>
                        </label>
                            <input type="password" required name="password"
                            autocomplete="off"/>
                    </div>

                        <button class="button button-block" name="login">Log
                        In</button>

                </form>
            </div>


    </div>

    <script
    src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
    <script src="js/index.js"></script>
</body>

</html>

```

userLogin.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


require 'db.php';

session_start();


    $username = $_POST["username"];
    $results = $mysqli->query("SELECT * FROM users WHERE username='$username'");

    $_SESSION['username'] = $username;



    // Check if user with given username exists in db
    if($results->num_rows == 0){
        $_SESSION['message'] = "Error loggin in user $username, please try again";
        header("location: error.php");

    }
    else{ // User exists

        $userDetails = $results->fetch_assoc();
        $pass = $_POST['password'];
        $hashPass = hash('sha256', $pass, false);
        $_SESSION['userId'] = $userDetails['id'];


        if ($hashPass == $userDetails['password']){

            $_SESSION['username'] = $userDetails['username'];

            // if account has been activated i.e. email verified
           // $_SESSION['active'] = $userDetails['active'];

            // show user is logged in
            $_SESSION['logged_in'] = true;

            header("location: profile.php");
        }
        else{
            $_SESSION['message'] = "Error loggin in user $username, please try again";
            header("location: error.php");
        }

    }
```

Profile.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


 require 'db.php';
 session_start();
// include("includes\header.php");

 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
}

?>
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>Profile <?= $username ?></title>
    <?php include 'css/css.html'; ?>
</head>

<body>


        <div class="nav-bar">
            <?php include '../PlantTracker/includes/header.php'; ?>
        </div>



        <p>
        <?php
        // Display message about account verfication link only once
        if(isset($_SESSION['message'])){
            echo $_SESSION['message'];

            // So more messages wont appear after page refresh
            unset($_SESSION['message']);
        }
        ?>
        </p>


        <div class="profile-container" id="profile-container">
        <h1>Plant Tracker Home<h1>
        <p>User: <?= $username ?></p>



            <div class="img-div" id="img-div">
                <!--<img src="img\logo.jpg" alt="profile image" height="200"
                width="200">-->
            </div>

            <div class="profile-info" id="profile-info">

            </div>
            <div class="games" id="games">

            </div>

        </div>
        <div class="profile-main" id="profile-main">
```

```
72
73              <div class = delete-div>
74                  <a href="generateReport.php"><button class="nav-button"
                    name="generateReport">Generate Report</button></a>
75
76                  <?php
77                      if($username == "admin"){
78                          echo '<a href="addUser.php"><button class="nav-button"
                            name="addUser">Add User</button></a>';
79                      }
80                  ?>
81          </div>
82
83
84      <form action="addUser.php" method="post" autocomplete="off">
85          <div class="delete-div">
86              <!--<p><a href ="profile.php"><?= $email ?></a></p>-->
87
88          </div>
89
90      </form>
91
92          </div>
93          <footer>
94
95              <div class="footer-div" id="footer-div">
96                  <?php include '../PlantTracker/includes/footer.php'; ?>
97              </div>
98          </footer>
99
100
101 </body>
102 </head>
103
104
105
```

Register.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */



$username = $_POST['username'];
$email = $_POST['email'];
$pass1 = $_POST['password1'];
$pass2 = $_POST['password2'];

if ($pass1 === $pass2){

    $pass = hash('sha256', $pass1, false);

    // chech if user exists..
    $results = $mysqli->query("SELECT * FROM users WHERE username = '$username'") or
    die($mysqli->error());

    // User exist
    if ($results->num_rows > 0){

        $_SESSION['message'] = "Username already exists";
        //header("location: addUser.php");
    }
    else{ // User does not exists

        $sql = "INSERT INTO users (username, password, email)" . "VALUES
        ('$username', '$pass', '$email')";
        $mysqli->query($sql);

        $_SESSION['message'] = "$username successfully registered.";

    }

}
else{
    $_SESSION['message'] = "Error, input passwords do not match";
}

```

db.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */



$username = $_POST['username'];
$email = $_POST['email'];
$pass1 = $_POST['password1'];
$pass2 = $_POST['password2'];

if ($pass1 === $pass2){

    $pass = hash('sha256', $pass1, false);

    // chech if user exists..
    $results = $mysqli->query("SELECT * FROM users WHERE username = '$username'") or
    die($mysqli->error());

    // User exist
    if ($results->num_rows > 0){

        $_SESSION['message'] = "Username already exists";
        //header("location: addUser.php");
    }
    else{ // User does not exists

        $sql = "INSERT INTO users (username, password, email)" . "VALUES
        ('$username', '$pass', '$email')";
        $mysqli->query($sql);

        $_SESSION['message'] = "$username successfully registered.";

    }

}
else{
    $_SESSION['message'] = "Error, input passwords do not match";
}

```

viewExperiments.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
include('includes/phpgraphlib.php');
require 'db.php';

 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
}


// Why charts did not work, DO NOT REMOVE.....
$_SESSION['disease'] = $_POST['disease'];
$_SESSION['date'] = $_POST['date'];

$disease = $_POST['disease'];
$date = $_POST['date'];

$graph = $_POST['chart'];

$exp_array = array();


if($username == "admin"){
    $sql = "SELECT * FROM expindex ORDER BY id ASC";
}else{
    $sql = "SELECT * FROM expindex WHERE username='$username' ORDER BY id ASC";
}
$result = $mysqli->query($sql);

//if($result){
//  while ($row = mysql_fetch_assoc($result)){
//      $id=$row['id'];
//      $replicant=$row['replicant'];
//      $exp_array[$id]=$replicant;
//
//  }
//  ksort($exp_array);
//}

?>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Report <?= $username ?></title>
<?php include 'css/css.html'; ?>
</head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>
    <p>
    </p>

<div class="profile-container" id="profile-container">
        <h1>Experiments Associated with <?= $username ?></h1>

<div class="profile-info" id="profile-info">
```

```
73
74  </div>
75
76
77  </div>
78  <div class="profile-main" id="profile-main">
79
80      <div class="about-analysis" id="about-analysis">
81
82          <?php
83              if($result->num_rows > 0){
84                  echo
                    "<table><tr><th>Id</th><th>Replicant</th><th>Expt</th><th>Treatment</t
                    h><th>View Individual Analysis</th><th>View Experiment
                    Label</th></tr>";
85                  while ($row = $result->fetch_assoc()){
86                      $id = $row["id"];
87                      echo
                        "<tr><td>".$row["id"]."</td><td>".$row["replicant"]."</td><td>".$r
                        ow["expt"]."</td><td>".$row["treatment"]."</td><td><a
                        href='Experiments.php?id=$id'>Analysis</a></td><td><a
                        href='viewExperimentLabel.php?id=$id'>View Experiment
                        Label</a></td></tr>";
88                  }
89                  echo "</table>";
90              }else {
91                  echo "No results";
92              }
93          ?>
94
95      </div>
96
97      <form action="addUser.php" method="post" autocomplete="off">
98          <div class="delete-div">
99
100             <!--<p><a href ="profile.php"><?= $email ?></a></p>-->
101
102         </div>
103
104     </form>
105 </div>
106
107
108     <div class="footer-div" id="footer-div">
109         <?php include 'includes/footer.php'; ?>
110     </div>
111
112
113 <script
    src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
114 <script src="js/index.js"></script>
115
116 </body>
117 </html>
```

viewExperimentLabel

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

session_start();
require 'db.php';


 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
}

if (isset($_GET['id'])){
    $getId = $_GET['id'];
}

$id = 1;
$sql = "SELECT * FROM expindex WHERE id =$getId";
$result = $mysqli->query($sql);
$data = $result->fetch_assoc();
$image = $data['image'];
$lat = $data['lat'];
$lon = $data['lon'];
$imagePath = $data['imagepath'];



?>

<!DOCTYPE html>
<html>

    <head>
        <title>Plant Tracker Experiment Label</title>
        <?php include 'css/css.html';?>
    </head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>
<p>
</p>

<div class="profile-main" id="profile-main">

    <div class="picture">


            <div id ="login">

                <h1>Plant Tracker Experiment Label</h1>
                <!-- Tag needed to display base64 image -->
                <!--<img src="<?php echo $image; ?>"/> -->

                <img src="<?php echo $imagePath;?>" alt="Image not found">

        </div>

</div>
    <div class="about-analysis" id="about-analysis">

        <?php
            if($result->num_rows > 0){
```

```
                    echo
                    "<table><tr><th>Id</th><th>Replicant</th><th>Expt</th><th>Treatment</t
                    h><th>Weather</th><th>Date</th><th>Time</th></tr>";
                        echo
                        "<tr><td>".$data["id"]."</td><td>".$data["replicant"]."</td><td>".
                        $data["expt"]."</td><td>".$data["treatment"]."</td><td>".$data["we
                        ather"]."</td><td>".$data["date"]."</td><td>".$data["time"]."</td>
                        </tr>";

                    echo "</table>";
                }else {
                    echo "No results";
                }
            ?>

        </div>
            <div class="picture">
                <div id ="login">
                        <div id="googleMap" style="width:100%;height:400px;"></div>
                </div>
        </div>

        </div>

    <div class="footer-div" id="footer-div">
        <?php include 'includes/footer.php'; ?>
    </div>

    <script>
    function myMap() {

    var lat = <?php echo $lat; ?>;
    var lon = <?php echo $lon; ?>;
     var myLatLng = {lat: lat, lng: lon};

    var mapProp= {

      center:new google.maps.LatLng(lat,lon),
      zoom:18,
      mapTypeId: 'satellite',
    };
    var map = new google.maps.Map(document.getElementById("googleMap"),mapProp);

    var marker = new google.maps.Marker({
        position: myLatLng,
        map: map,
        title: 'Hello World!'
      });

    }
    </script>

    <script
    src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDwKOoWZvBoSeYvm1G7egLLvqJBy7ZyNyw&callback=myMap"></script>



        <script
        src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
        <script src="js/index.js"></script>
    </body>

    </html>
```

viewAnalysis.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

session_start();
require 'db.php';


 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
}

if (isset($_GET['id'])){
    $getId = $_GET['id'];
}

$id = 1;
$sql = "SELECT * FROM analysis WHERE id =$getId";
$result = $mysqli->query($sql);
$data = $result->fetch_assoc();
$image = $data['image'];
$lat = $data['lat'];
$lon = $data['lon'];
$imagePath = $data['imagepath'];



?>

<!DOCTYPE html>
<html>

    <head>
        <title>Plant Tracker Individual Analysis</title>
        <?php include 'css/css.html';?>
    </head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>
<p>
</p>

<div class="profile-main" id="profile-main">

    <div class="picture">


            <div id ="login">

                <h1>Plant Tracker Individual Analysis</h1>
                <!-- Tag needed to display base64 image -->
                <!--<img src="<?php echo $image; ?>"/> -->

                <img src="<?php echo $imagePath;?>" alt="Image not found">

        </div>

</div>
    <div class="about-analysis" id="about-analysis">

        <?php
            if($result->num_rows > 0){
```

```
                    echo
                    "<table><tr><th>Id</th><th>Replicant</th><th>Expt</th><th>Treatment</t
                    h><th>Weather</th><th>Date</th><th>Time</th><th>Analysis</th></tr>";
                        echo
                        "<tr><td>".$data["id"]."</td><td>".$data["replicant"]."</td><td>".
                        $data["expt"]."</td><td>".$data["treatment"]."</td><td>".$data["we
                        ather"]."</td><td>".$data["date"]."</td><td>".$data["time"]."</td>
                        <td>".$data["analysis"]."</td></tr>";

                    echo "</table>";
                }else {
                    echo "No results";
                }
            ?>

        </div>
            <div class="picture">
                <div id ="login">
                        <div id="googleMap" style="width:100%;height:400px;"></div>
                </div>
        </div>

        </div>

    <div class="footer-div" id="footer-div">
        <?php include 'includes/footer.php'; ?>
    </div>

    <script>
    function myMap() {

    var lat = <?php echo $lat; ?>;
    var lon = <?php echo $lon; ?>;
     var myLatLng = {lat: lat, lng: lon};

    var mapProp= {

      center:new google.maps.LatLng(lat,lon),
      zoom:18,
      mapTypeId: 'satellite',
    };
    var map = new google.maps.Map(document.getElementById("googleMap"),mapProp);

    var marker = new google.maps.Marker({
        position: myLatLng,
        map: map,
        title: 'Hello World!'
      });

    }
    </script>

    <script
    src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDwK0oWZvBoSeYvm1G7egLLvqJBy7ZyNyw&callback=myMap"></script>
        <script
        src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
        <script src="js/index.js"></script>
    </body>

    </html>
```

Logout.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


// Logout, unsets and destroys session variables

session_start();
session_unset();
session_destroy();

?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Error</title>
    <?php include 'css/css.html'; ?>
</head>

<body>
    <div class="form">
        <h1>Plant Tracker Logout</h1>

        <p><?= 'You have been logged out'; ?></p>

        <a href="index.php"><button class="button button-block"/>Home</button></a>
    </div>
</body>
</html>
```

lineGraph.php

```php
<?php

/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

session_start();
include('includes/phpgraphlib.php');
require('db.php');


//$set1 = array (
//  1917 => 4011, 1918 => 4886, 1919 => 5411,
//  1920 => 5831, 1921 => 5865, 1922 => 5704, 1923 => 5337, 1924 => 5144,
//  1925 => 5018, 1926 => 4971, 1927 => 4630, 1928 => 4411, 1929 => 4287,
//  1930 => 4116, 1931 => 3940, 1932 => 3764, 1933 => 3592, 1934 => 3447,
//  1935 => 3280, 1936 => 3215, 1937 => 3366, 1938 => 3569, 1939 => 3598,
//  1940 => 4436, 1941 => 5939, 1942 => 7397, 1943 => 8855, 1944 => 9835,
//  1945 => 9998, 1946 => 10631, 1947 => 11340, 1948 => 11549, 1949 => 11642,
//  );

$replicant = $_SESSION['replicant'];
$treatment = $_SESSION['treatment'];
$experiment = $_SESSION['experiment'];


$link = mysql_connect('localhost', 'root', 'testtest')
   or die('Could not connect: ' . mysql_error());
mysql_select_db('project') or die('Could not select database');

$exp_array = array();

$query = "SELECT analysis, id FROM analysis WHERE (replicant='$replicant' AND
treatment='$treatment' AND expt='$experiment') ";

$result = mysql_query($query) or die('Query failed: ' . mysql_error());
if($result){
    while ($row = mysql_fetch_assoc($result)){
        $analysis=$row['analysis'];
        $id=$row['id'];
        $exp_array[$id]=$analysis;

    }
}

ksort($exp_array);
$graph = new PHPGraphLib(600, 400);
$graph->addData($exp_array);
$graph->setTitleLocation('left');
$graph->setTitle("Line Graph of level of disease for each plant in experiment");
$graph->setBars(false);
$graph->setLine(true);
$graph->setDataPoints(false);
$graph->setLineColor('blue', 'red');
$graph->setDataValues(false);
$graph->setXValuesInterval(5);
$graph->setDataValueColor('blue', 'red');
$graph->setLegend(true);
$graph->setLegendTitle("analysis", "set2");
$graph->createGraph();


?>
```

generateReport.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
require ('db.php');

// Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
    //$email = $_SESSION['email'];
//  $active = $_SESSION['active'];
}


if($_SERVER['REQUEST_METHOD'] == 'POST'){

    if($_POST['chart'] == 'bar'){
        require 'barGraph.php';
    }
    else if ($_POST['chart'] == 'line'){
        require 'lineGraph.php';
    }
}

?>

<!DOCTYPE html>
<html>

<head>
    <title>Plant Tracker Report Generator</title>
    <?php include 'css/css.html';?>
</head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>


<div class="form">

    <div id ="login">

        <h1>Select Data For Chart</h1>

        <form action ="createReport.php" autocomplete="off" method="POST">

            <h3>Chart</h3>
            <select class="chart" name="chart">Chart
                  <option value="bar">Bar</option>
                  <option value="line">Line</option>
            </select>


             <h3>Replicant<h3>
             <select class="chart" name="replicant">;

            <?php

            $query = "SELECT DISTINCT replicant FROM expindex ORDER BY id asc";
            $result = $mysqli->query($query);

```

```
73              while($row = $result->fetch_assoc()){
74                  echo "<option value='".$row['replicant']."'>" . $row['replicant'] .
                    "</option>";
75
76              }
77              ?>
78              </select>
79                 <h3>Treatment<h3>
80                  <select class="chart" name="treatment">;
81              <?php
82
83              $query = "SELECT DISTINCT treatment FROM expindex ORDER BY id asc";
84              $result = $mysqli->query($query);
85
86              while($row = $result->fetch_assoc()){
87
88                  echo "<option value='".$row['treatment']."'>" . $row['treatment'] .
                    "</option>";
89
90              }
91              ?>
92              </select>
93
94                  <h3>Expt</h3>
95              <select class="chart" name="experiment">;
96
97              <?php
98              $query = "SELECT DISTINCT expt FROM expindex ORDER BY id asc";
99              $result = $mysqli->query($query);
100
101             while($row = $result->fetch_assoc()){
102
103                 echo "<option value='".$row['expt']."'>".$row['expt']."</option>";
104             }
105             ?>
106             </select>
107
108
109
110
111
112         <!--
113             <select class="chart" name="rep">Chart
114                   <option value="bar">Bar</option>
115                   <option value="line">Line</option>
116             </select>
117             <br>
118             <select class="chart" name="expt">Chart
119                   <option value="bar">Bar</option>
120                   <option value="line">Line</option>
121             </select>
122             <br>
123             <select class="chart" name="expt">Chart
124                   <option value="bar">Bar</option>
125                   <option value="line">Line</option>
126             </select>
127
128
129
130             <div class="field-wrap">
131                 <label>
132                     Species Name<span class="req">*</span>
133                 </label>
134                 <input type="text" name="species" autocomplete="off"/>
135             </div>
136         -->
137
138             <button class="button button-block" name="login">Get Data</button>
139
140             <p>
141                 <?php
142                     if (isset($_SESSION['message']) AND !empty($_SESSION['message'])){
```

```
143                          echo $_SESSION['message'];
144                         // So more messages wont appear after page refresh
145                         unset($_SESSION['message']);
146                     }
147                 ?>
148             </p>
149         </form>
150     </div>
151
152 </div>
153
154
155 <footer>
156
157     <div class="footer-div" id="footer-div">
158         <?php include 'includes/footer.php'; ?>
159     </div>
160 </footer>
161
162
163 <script
    src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
164 <script src="js/index.js"></script>
165 </body>
166
167 </html>
```

generateCSV.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
include('includes/phpgraphlib.php');
require 'db.php';

$username =$_SESSION['username'];

//$_SESSION['disease'] = $_POST['disease'];
//$_SESSION['date'] = $_POST['date'];

$disease = $_POST['disease'];



$exp_array = array();

$sql = "SELECT * FROM experiments WHERE expname='$disease'";
$result = $mysqli->query($sql);

if($result){
    while ($row = mysql_fetch_assoc($result)){
        $analysis=$row['analysis'];
        $expindex=$row['expindex'];
        $exp_array[$expindex]=$analysis;

    }
    ksort($exp_array);
}

$sqlCsv = "SELECT * FROM experiments INTO OUTFILE '../PlantTracker/out.csv' FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n';"

$mysqli->query($sqlCsv);

?>
```

Experiments.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
include('includes/phpgraphlib.php');
require 'db.php';


 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
}

$username = $_SESSION['username'];


if(isset($_GET['id'])){
    $getId = $_GET['id'];
}

// Why charts did not work, DO NOT REMOVE.....
$_SESSION['disease'] = $_POST['disease'];
$_SESSION['date'] = $_POST['date'];

$disease = $_POST['disease'];
$date = $_POST['date'];


$exp_array = array();

$sql = "SELECT * FROM analysis WHERE expid='$getId'";
$result = $mysqli->query($sql);

if($result){
    while ($row = mysql_fetch_assoc($result)){
        $id=$row['id'];
        $replicant=$row['replicant'];
        $exp_array[$id]=$replicant;

    }
    ksort($exp_array);
}

?>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>All Analysis <?= $username ?></title>
<?php include 'css/css.html'; ?>
</head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>
    <p>
    </p>

<div class="profile-container" id="profile-container">
        <h1>Analysis Associated with <?= $username ?></h1>

```

```
73
74          <p><?= $getId ?></p>
75      <div class="profile-info" id="profile-info">
76
77      </div>
78
79
80      </div>
81      <div class="profile-main" id="profile-main">
82
83          <div class="about-analysis" id="about-analysis">
84
85              <?php
86                  if($result->num_rows > 0){
87                      echo
                        "<table><tr><th>Id</th><th>Replicant</th><th>Expt</th><th>Treatment</t
                        h><th>Analysis</th><th>View Individual Analysis</th></tr>";
88                      while ($row = $result->fetch_assoc()){
89                          $id = $row['id'];
90                          echo
                            "<tr><td>".$row["id"]."</td><td>".$row["replicant"]."</td><td>".$r
                            ow["expt"]."</td><td>".$row["treatment"]."</td><td>".$row["analysi
                            s"]."</td><td><a href='viewAnalysis.php?id=$id'>Individual
                            Analysis</a></td></tr>";
91                      }
92                      echo "</table>";
93                  }else {
94                      echo "No results";
95                  }
96              ?>
97
98          </div>
99
100         <form action="addUser.php" method="post" autocomplete="off">
101             <div class="delete-div">
102
103                 <!--<p><a href ="profile.php"><?= $email ?></a></p>-->
104
105             </div>
106
107         </form>
108     </div>
109
110
111         <div class="footer-div" id="footer-div">
112             <?php include 'includes/footer.php'; ?>
113         </div>
114
115
116     <script
        src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
117     <script src="js/index.js"></script>
118
119     </body>
120     </html>
```

Error.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();

//$_SESSION['message'] = "You must be logged in to view this page";

?>

<!DOCTYPE html>
<html>
    <head>
        <titlt>Error</titlt>
        <?php include 'css/css.html';?>
    </head>
    <body>


    <div class="form">


        <p>
            <?php
            if (isset($_SESSION['message']) AND !empty($_SESSION['message'])){
                echo $_SESSION['message'];

                // So more messages wont appear after page refresh
                unset($_SESSION['message']);
            }
            else{  // uncomment after testing will route straight to index if no
            session message is se
                echo $_SESSION['message'] = "You must be logged in to view this page";
                // header("location: index.php");
            }
            ?>

        </p>

        <a href="index.php"><button class="button button-block"/>Home</button></a>
    </div>

    </body>
</html>

```

Download.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();

$username = $_SESSION['username'];
$today = date("d.m.y");
$time = date("h:i:s");

$filename = "PlantTracker_".$username."_".$today."_".$time.".csv";
$export_data = unserialize($_POST['export_data']);

// Create file
$file = fopen($filename, "w");

foreach ($export_data as $line){
    fputcsv($file, $line);
}

fclose($file);

// download
header("Content-Description: File Transfer");
header("Content-Disposition: attachment; filename=".$filename);
header("Content-Type: application/csv; ");

readfile($filename);

// deleting file
unlink($filename);
exit();
unlink($filename);
exit();
```

csvData.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

session_start();
require ('db.php');

// Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
    //$email = $_SESSION['email'];
//  $active = $_SESSION['active'];
}


if($_SERVER['REQUEST_METHOD'] == 'POST'){

    if($_POST['chart'] == 'bar'){
        require 'barGraph.php';
    }
    else if ($_POST['chart'] == 'line'){
        require 'lineGraph.php';
    }
}


?>

<!DOCTYPE html>
<html>

<head>
    <title>Plant Tracker CSV Generator</title>
    <?php include 'css/css.html';?>
</head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>


<div class="form">

    <div id ="login">

        <h1>Select Data For CSV</h1>

        <form action ="createCSV.php" autocomplete="off" method="POST">


            <h3>Replicant<h3>
             <select class="chart" name="replicant">;

            <?php

            $query = "SELECT DISTINCT replicant FROM expindex ORDER BY id asc";
            $result = $mysqli->query($query);

            while($row = $result->fetch_assoc()){
                echo "<option value='".$row['replicant']."'>" . $row['replicant'] .
                "</option>";

            }
            ?>
```

```
            </select>
                <h3>Treatment<h3>
                 <select class="chart" name="treatment">;
            <?php

            $query = "SELECT DISTINCT treatment FROM expindex ORDER BY id asc";
            $result = $mysqli->query($query);

            while($row = $result->fetch_assoc()){

                echo "<option value='".$row['treatment']."'>" . $row['treatment'] .
                "</option>";

            }
            ?>
            </select>

                <h3>Expt</h3>
            <select class="chart" name="expt">;

            <?php
            $query = "SELECT DISTINCT expt FROM expindex ORDER BY id asc";
            $result = $mysqli->query($query);

            while($row = $result->fetch_assoc()){

                echo "<option value='".$row['expt']."'>".$row['expt']."</option>";
            }
            ?>
            </select>




        <!--
            <select class="chart" name="rep">Chart
                  <option value="bar">Bar</option>
                  <option value="line">Line</option>
            </select>
            <br>
            <select class="chart" name="expt">Chart
                  <option value="bar">Bar</option>
                  <option value="line">Line</option>
            </select>
            <br>
            <select class="chart" name="expt">Chart
                  <option value="bar">Bar</option>
                  <option value="line">Line</option>
            </select>


            <div class="field-wrap">
                <label>
                    Species Name<span class="req">*</span>
                </label>
                <input type="text" name="species" autocomplete="off"/>
            </div>
        -->

            <button class="button button-block" name="login">Get Data</button>

            <p>
                <?php
                    if (isset($_SESSION['message']) AND !empty($_SESSION['message'])){
                         echo $_SESSION['message'];
                        // So more messages wont appear after page refresh
                        unset($_SESSION['message']);
                    }
                ?>
            </p>
```

csvBuilder.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
include('includes/phpgraphlib.php');
require 'db.php';

$sql = "SELECT * FROM experiments ORDER BY id ASC";
$result = $mysqli->query($sql);

?>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Report <?= $username ?></title>
<?php include 'css/css.html'; ?>
</head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>
    <p>
    </p>

<div class="profile-container" id="profile-container">
        <h1>Report</h1>
        <p><?= $username ?></p>

</div>

    <div class="profile-leagues" id="profile-leagues">


    <form action ="download.php" autocomplete="off" method="POST">
        <!--<input type='submit' value='Export' name='Export'>-->
    <div class="about-analysis" id="about-analysis">
        <table>
            <tr>
                <th>Id</th>
                <th>expname</th>
                <th>date</th>
                <th>time</th>
                <th>location</th>
                <th>analysis</th>
                <th>userid</th>
                <th>expindex</th>
                <th>notes</th>
                <th>expindex</th>
            </tr>
        <?php

                $user_arr = array();
                $user_arr[] = array('Id', 'Exp Name', 'Date', 'Time', 'Location',
                'Analysis', 'User Id', 'Exp Index', 'Notes', 'Exp Index');
                while ($row = $result->fetch_array()){

                    $id = $row["id"];
                    $expname = $row["expname"];
                    $date = $row["date"];
                    $time = $row["time"];
                    $location = $row["location"];
                    $analysis = $row["analysis"];
                    $userid = $row["userid"];
                    $expindex = $row["expindex"];
                    $notes = $row["notes"];
```

```
72                      $expindexid = $row["expindex"];
73
74                      $user_arr[] = array($id, $expname, $date, $time, $location,
                        $analysis, $userid, $expindex, $notes, $expindexid);
75          ?>
76              <tr>
77                  <td><?php echo $id; ?></td>
78                  <td><?php echo $expname; ?></td>
79                  <td><?php echo $date; ?></td>
80                  <td><?php echo $time; ?></td>
81                  <td><?php echo $location; ?></td>
82                  <td><?php echo $analysis; ?></td>
83                  <td><?php echo $userid; ?></td>
84                  <td><?php echo $expindex; ?></td>
85                  <td><?php echo $notes; ?></td>
86                  <td><?php echo $expindexid; ?></td>
87              </tr>
88              <?php
89              }
90              ?>
91          </table>
92          <?php
93              $serialize_user_arr = serialize($user_arr);
94          ?>
95
96  <textarea name='export_data' style='display: none;'><?php echo $serialize_user_arr;
    ?></textarea>
97      </div>
98
99      <button class="button button-block" name="login">Generate CSV</button>
100     </form>
101
102 </div>
103
104
105     <div class="footer-div" id="footer-div">
106         <?php include 'includes/footer.php'; ?>
107     </div>
108
109
110 <script
    src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
111 <script src="js/index.js"></script>
112
113 </body>
114 </html>
```

createReport.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

session_start();
include('includes/phpgraphlib.php');
require 'db.php';

 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
}

// Why charts did not work, DO NOT REMOVE.....
$_SESSION['disease'] = $_POST['disease'];
$_SESSION['date'] = $_POST['date'];

$sql = "SELECT id FROM users where username='$username'";
$results = $mysqli->query($sql);
$userDetails = $results->fetch_assoc();
$userID = $userDetails['id'];

$_SESSION['replicant'] = $_POST['replicant'];
$_SESSION['treatment'] = $_POST['treatment'];
$_SESSION['experiment'] = $_POST['experiment'];

$replicant = $_POST['replicant'];
$treatment = $_POST['treatment'];
$experiment = $_POST['experiment'];


$graph = $_POST['chart'];



if ($graph === "bar"){
    $chart = "barGraph.php";
}
else if ($graph === "line"){
    $chart = "lineGraph.php";
}

$exp_array = array();

$sql = "SELECT analysis, time FROM analysis WHERE (replicant='$replicant' AND
treatment='$treatment' AND expt='$experiment')";
$result = $mysqli->query($sql);

if($result){
    while ($row = mysql_fetch_assoc($result)){
        $analysis=$row['analysis'];
        $time=$row['time'];
        $exp_array[$time]=$analysis;

    }
    ksort($exp_array);
}


?>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
```

```
72      <title>Report <?= $username ?></title>
73  <?php include 'css/css.html'; ?>
74  </head>
75  <body>
76
77  <div class="nav-bar">
78      <?php include 'includes/header.php'; ?>
79  </div>
80      <p>
81      </p>
82
83  <div class="profile-container" id="profile-container">
84          <h1>Report</h1>
85          <p><?= $username ?></p>
86
87  <div class="img-div" id="img-div" style="text-align: center;">
88      <img src=<?php echo $chart;?> />
89  </div>
90  <div class="profile-info" id="profile-info">
91
92  </div>
93
94
95  </div>
96  <div class="profile-main" id="profile-main">
97
98      <div class="about-analysis" id="about-analysis">
99
100
101
102     </div>
103
104     <form action="addUser.php" method="post" autocomplete="off">
105         <div class="delete-div">
106
107             <!--<p><a href ="profile.php"><?= $email ?></a></p>-->
108
109         </div>
110
111     </form>
112 </div>
113
114
115     <div class="footer-div" id="footer-div">
116         <?php include 'includes/footer.php'; ?>
117     </div>
118
119
120 <script src="js/index.js"></script>
121
122 </body>
123 </html>
```

createExperiment.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */



if($_SERVER['REQUEST_METHOD'] == 'POST'){

    require 'addExperiment.php';
}

?>

<!DOCTYPE html>
<html>

    <head>
        <title>Plant Tracker Login</title>
        <?php include 'css/css.html';?>
    </head>
<body>

    <div class="form">

            <div id ="login">

                <h1>Create Experiment</h1>

                <form action ="addExperiment.php" autocomplete="off" method="POST">

                    <div class="field-wrap">
                        <label>
                            Experiment Name<span class="req">*</span>
                        </label>
                        <input type="text" required name="experimentname"
                        autocomplete="off"/>
                    </div>
                        <button class="button button-block" name="login">Add
                        Experiment</button>
                    <p>
                        <?php
                            if (isset($_SESSION['message']) AND
                            !empty($_SESSION['message'])){
                                echo $_SESSION['message'];

                                // So more messages wont appear after page refresh
                                unset($_SESSION['message']);
                            }
                        ?>

                    </p>
                </form>
            </div>

    </div>

    <script
    src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
    <script src="js/index.js"></script>
</body>
```

createCSV.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
require ('db.php');


 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
}
// Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
    //$email = $_SESSION['email'];
//  $active = $_SESSION['active'];
}

if($_SERVER['REQUEST_METHOD'] == 'POST'){

    $_SESSION['username'] = $username;

    if($_POST['chart'] == 'bar'){
        require 'barGraph.php';
    }
//  else if ($_POST['chart'] == 'line'){
//      require 'lineGraph.php';
//  }
}

$sql = "SELECT id FROM users where username='$username'";
$results = $mysqli->query($sql);
$userDetails = $results->fetch_assoc();
$userID = $userDetails['id'];


$replicant = $_POST['replicant'];
$treatment = $_POST['treatment'];
$experiment = $_POST['expt'];

?>

<!DOCTYPE html>
<html>

<head>
    <title>Plant Tracker CSV Generator</title>
    <?php include 'css/css.html';?>
</head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>

<p>
</p>

```

```
<div class="profile-main">


    <div id ="login">


        <h1>Generate CSV</h1>

            <div class="img-div" id="img-div">

            </div>
  <div class="about-analysis" id="about-analysis">


        <form action ="download.php" autocomplete="off" method="POST">
             <input type="hidden" name="username" value="<? = $username.'.csv'?>"
             autocomplete="off"/>

             <button class="button button-block" name="login">Generate CSV</button>

        <input type='submit' value='Export' name='Export'>
            <table border='1' style='border-collapse:collapse;'>
    <tr>

        <th>Date</th>
        <th>Time</th>
        <th>Weather</th>
        <th>Analysis</th>
        <th>Replicant</th>
        <th>Expt</th>
        <th>Treatment</th>
        <th>Lat</th>
        <th>Lon</th>
    </tr>
    <?php
    if($username == "admin"){
        $query = "SELECT * FROM analysis WHERE (replicant='$replicant' AND
        treatment='$treatment' AND expt='$experiment')";
    }else{
        $query = "SELECT * FROM analysis WHERE (replicant='$replicant' AND
        treatment='$treatment' AND expt='$experiment' AND username='$username')";
    }
     $result = $mysqli->query($query);
     $user_arr = array();
     while($row = mysqli_fetch_array($result)){
        $date =$row['date'];
        $time = $row['time'];
        $weather = $row['weather'];
        $lat = $row['lat'];
        $lon = $row['lon'];
        $analysis = $row['analysis'];
        $rep = $row['replicant'];
        $treatment = $row['treatment'];
        $expt = $row['expt'];
        $user_arr[] =
        array($date,$time,$weather,$lat,$lon,$analysis,$rep,$treatment,$expt);
    ?>
       <tr>
        <td><?php echo $date; ?></td>
        <td><?php echo $time; ?></td>
        <td><?php echo $weather; ?></td>
        <td><?php echo $analysis; ?></td>
        <td><?php echo $rep; ?></td>
        <td><?php echo $expt; ?></td>
        <td><?php echo $treatment; ?></td>
        <td><?php echo $lat; ?></td>
        <td><?php echo $lon; ?></td>
       </tr>
    <?php
     }
    ?>
    </table>
```

```
141         <?php
142       $serialize_user_arr = serialize($user_arr);
143      ?>
144     <textarea name='export_data' style='display: none;'><?php echo
        $serialize_user_arr; ?></textarea>
145
146
147           </form>
148
149         </div>
150           <p>
151           </p>
152       </div>
153
154   </div>
155
156   <footer>
157
158       <div class="footer-div" id="footer-div">
159           <?php include 'includes/footer.php'; ?>
160       </div>
161   </footer>
162
163
164   <script
      src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
165   <script src="js/index.js"></script>
166   </body>
167
168   </html>
```

barGraph.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

session_start();
include('includes/phpgraphlib.php');
require('db.php');



//$name = $_POST['disease'];
//$date = $_POST['date'];

$graph = new PHPGraphLib(700, 550);

//$disease = "black spot";
//$date = '06.03.19';

$replicant = $_SESSION['replicant'];
$treatment = $_SESSION['treatment'];
$experiment = $_SESSION['experiment'];

$userId = 1;
//

$link = mysql_connect('localhost', 'root', 'testtest')
   or die('Could not connect: ' . mysql_error());
mysql_select_db('project') or die('Could not select database');

$exp_array = array();

$query = "SELECT analysis, time FROM analysis WHERE (replicant='$replicant' AND
treatment='$treatment' AND expt='$experiment')";

$result = mysql_query($query) or die('Query failed: ' . mysql_error());
if($result){
    while ($row = mysql_fetch_assoc($result)){
        $analysis=$row['analysis'];
        $expindex=$row['time'];
        $exp_array[$expindex]=$analysis;

    }
}

ksort($exp_array);
$graph->addData($exp_array);
$graph->setTitle('Analysis level for each plant in experiment');
$graph->setGradient('lime', 'green');
$graph->createGraph();


?>
```

bar.php

```php
<?php

include_once("config.php");
include_once("/includes/lib/inc/chartphp_dist.php");

$bar_chart_data =
 array(
    array(
        array("2010/12",48.25),
        array("2011/01",238.75),
        array("2011/02",95.50),
        array("2011/03",300.50),
        array("2011/04",286.80),
        array("2011/05",148.25),
        array("2011/06",128.75),
        array("2011/07",95.50),
        array("2011/08",200.50),
        array("2011/09",216.80),
        array("2011/10",248.25),
        array("2011/11",148.25),
        array("2011/12",318.75),
        array("2012/01",295.50),
        array("2012/02",30.50),
        array("2012/03",236.80),
        array("2012/04",367)
        )
    );

$p = new chartphp();

//include("../../example_data.php");
$p->data=$bar_chart_data;
$p->chart_type = "bar";

// Common Options
$p->title = "Bar Chart";
$p->xlabel = "Months";
$p->ylabel = "Purchase";
$p->show_xticks = true;
$p->show_yticks = true;
$p->show_point_label = true;
$p->targetx_start = "2010/12";
$p->targetx_end = "2012/04";
$p->targety_start = 250;
$p->targety_end = 250;
$p->targetline_color = "purple";
$p->targetline_width = 4;
$p->targetline_style = "dashdot";  //line

$out = $p->render('c1');

?>
/ includes / lib / js /
<!DOCTYPE html>
<html>
<head>
        <link rel="stylesheet" href="/includes/lib/js/chartphp.css">
        <script src="/includes/lib/js/jquery.min.js"></script>
        <script src="/includes/lib/js/chartphp.js"></script>
    <meta charset="UTF-8">
    <title>Report <?= $username ?></title>
<?php include 'css/css.html'; ?>
</head>
<body>

<div class="nav-bar">
    <?php include 'includes/header.php'; ?>
</div>

<p>
    <?php
        echo $disease;
```

```
            echo $date;
        ?>
    </p>

    <div class="profile-container" id="profile-container">
            <h1>Report</h1>
            <p><?= $username ?></p>

    <div class="img-div" id="img-div" style="text-align: center;">

       <?php echo $out; ?>
    </div>

    <div class="profile-info" id="profile-info">

    </div>


    </div>
    <div class="profile-leagues" id="profile-leagues">

        <div class="about-info" id="about-info">


        </div>

        <form action="addUser.php" method="post" autocomplete="off">
            <div class="delete-div">
                <!--<p><a href ="profile.php"><?= $email ?></a></p>-->

            </div>

        </form>
    </div>

    <footer>

        <div class="footer-div" id="footer-div">
            <?php include 'includes/footer.php'; ?>
        </div>
    </footer>

    <script
    src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
    <script src="js/index.js"></script>

    </body>
    </html>
```

addUser.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


session_start();
require 'db.php';

// Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
    //$email = $_SESSION['email'];
//  $active = $_SESSION['active'];
}


if($_SERVER['REQUEST_METHOD'] == 'POST'){

    require 'register.php';
}

?>

<!DOCTYPE html>
<html>

    <head>
        <title>Plant Tracker Login</title>
        <?php include 'css/css.html';?>
    </head>
<body>


    <div class="nav-bar">
        <?php include '../PlantTracker/includes/header.php'; ?>
    </div>



    <div class="form">

            <div id ="login">

                <h1>Add User</h1>

                <form action ="addUser.php" autocomplete="off" method="POST">

                    <div class="field-wrap">
                        <label>
                            Username<span class="req">*</span>
                        </label>
                        <input type="text" required name="username"
                        autocomplete="off"/>
                    </div>
                    <div class="field-wrap">
                        <label>
                            Email<span class="req">*</span>
                        </label>
                        <input type="email" required name="email" autocomplete="off"/>
                    </div>
                    <div class="field-wrap">
                        <label>
                            Password<span class="req">*</span>
                        </label>
```

```
72                          <input type="password" required name="password1"
                            autocomplete="off"/>
73                      </div>
74
75                      <div class="field-wrap">
76                          <label>
77                              Confirm password<span class="req">*</span>
78                          </label>
79                          <input type="password" required name="password2"
                            autocomplete="off"/>
80                      </div>
81
82                          <button class="button button-block" name="login"/>Add
                            User</button>
83
84                      <p>
85                          <?php
86                              if (isset($_SESSION['message']) AND
                                !empty($_SESSION['message'])){
87                                  echo $_SESSION['message'];
88
89                                  // So more messages wont appear after page refresh
90                                  unset($_SESSION['message']);
91                              }
92                          ?>
93
94                      </p>
95                  </form>
96              </div>
97      </div>
98
99          <footer>
100             <div class="footer-div" id="footer-div">
101                 <?php include '../PlantTracker/includes/footer.php'; ?>
102             </div>
103         </footer>
104
105     <script
        src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
106     <script src="js/index.js"></script>
107 </body>
108
109 </html>
```

about.php

```php
<?php
 /**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

 require 'db.php';
 session_start();
// include("includes\header.php");

 // Check user is logged in
if($_SESSION['logged_in'] != 1){
    $_SESSION['message'] = "You must be logged in to view the profile page";
    header("location: error.php");
}
else{

    $username = $_SESSION['username'];
    //$email = $_SESSION['email'];
//  $active = $_SESSION['active'];
}

?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Profile <?= $username ?></title>
    <?php include 'css/css.html'; ?>
</head>

<body>


        <div class="nav-bar">
            <?php include '../PlantTracker/includes/header.php'; ?>
        </div>

<p>
</p>



        <div class="profile-container" id="profile-container">
            <h1>About</h1>
            <p><?= $username; ?></p>
            <h1>Plant Tracker</h1>

            <div class="img-div" id="img-div">
                <!--<img src="img\logo.jpg" alt="profile image" height="200"
                width="200">-->
            </div>

            <div class="profile-info" id="profile-info">

            </div>
            <div class="games" id="games">

            </div>

        </div>
        <div class="profile-main" id="profile-main">
            <p>
                Plant Tracker is a final year project for a Software Development
                student at ITCarlow. Plant Tracker is an application that is made up
                of two parts a mobile application and a website. The mobile
                application is used to determine the level of disease present on
                leaves of plants through the use of computer vision by capturing an
                image on the mobile device which is then analysed by the mobile
                application. All analyses can be viewed and accessed on the website
                and then used to generate reports about the analysis.
```

```
65
66              </p>
67
68
69
70
71          </div>
72
73          <div class="footer-div" id="footer-div">
74              <?php include '../PlantTracker/includes/footer.php'; ?>
75          </div>
76
77
78
79  <script
    src='http://cdnjs.cloudflare.com/ajax/libs/jquery/2.1.3/jquery.min.js'></script>
80  <script src="js/index.js"></script>
81
82  </body>
83  </html>
84
85
86
```

API

Analysis.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


include_once 'dbConnect.php';

class Analysis {

    private $db;

    private $db_table = 'analysis';

    public function __construct(){

        $this->db = new DbConnect();
    }


    public function addAnalysisToDB($date, $time, $image, $lat, $lon, $weather,
    $analysis, $userId, $username, $rep, $treatment, $expt, $expId){


        $imageName = "analysis_rep ".$rep."_treatment
        ".$treatment."_".$expt."_".$date."_".$time."_"."_".$username;
        $imagePath = "../PlantTracker/uploads/AnalysisImages/$imageName";


        $query = "INSERT INTO ".$this->db_table." (date, time, imagepath, lat, lon,
        weather, analysis, username, userid, replicant, treatment, expt, expid)
                    VALUES ('$date', '$time', '$imagePath', '$lat', '$lon',
                    '$weather', '$analysis', '$username', '$userId', '$rep',
                    '$treatment', '$expt', '$expId')";

        $inserted = mysqli_query($this->db->getDb(), $query);

        if($inserted == 1){

            file_put_contents($imagePath, base64_decode($image));

            $json['success'] = 1;
            $json['message'] = "Successfully uploaded analysis";

        }else{
            $json['success'] = 0;
            $json['message'] = "Error in uploading data. MSG from PHP API";
        }
        return $json;
        //mysqli_close($this->db->getDb());
    }


}

?>
```

AnalysisUpload.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


require_once 'dbConnect.php';
require_once 'Analysis.php';

$date = "";
$time = "";
$image = "";
$lat = "";
$lon = "";


if (isset($_POST['date'])){
    $date = $_POST['date'];
}

if (isset($_POST['time'])){
    $time = $_POST['time'];
}

if (isset($_POST['image'])){
    $image = $_POST['image'];
}
if (isset($_POST['lat'])){
    $lat = $_POST['lat'];
}

if (isset($_POST['lon'])){
    $lon = $_POST['lon'];
}

if (isset($_POST['weather'])){
    $weather = $_POST['weather'];
}

if (isset($_POST['analysis'])){
    $analysis = $_POST['analysis'];
}

if (isset($_POST['userid'])){
    $userId = $_POST['userid'];
}

if (isset($_POST['username'])){
    $username = $_POST['username'];
}

if (isset($ POST['rep'])){
    $rep = $_POST['rep'];
}

if (isset($_POST['treatment'])){
    $treatment = $_POST['treatment'];
}

if (isset($_POST['expt'])){
    $expt = $_POST['expt'];
}

if (isset($_POST['expid'])){
    $expId = $_POST['expid'];
}
// need analysis, userid, username, rep, treatment, expt,
```

```
73  $analysisObj = new Analysis();
74
75  // Upload image
76  if(!empty($date) && !empty($time) && !empty($image) && !empty($lon) && !empty($lat)
    && !empty($analysis) && !empty($weather)
77     && !empty($userId) && !empty($username) && !empty($rep) && !empty($treatment) &&
       !empty($expt) && !empty($expId) ){
78
79      $json_array = $analysisObj->addAnalysisToDB($date, $time, $image, $lat, $lon,
        $weather, $analysis, $userId, $username, $rep, $treatment, $expt, $expId);
80      echo json_encode($json_array);
81  }
82
83
84  ?>
```

constants.php

```
1   <?php
2   /**
3    * Student Name: Darran Gahan
4    * Student Number: C00098391
5    *
6    * Configuration settings to connect to the database
7    *
8    */
9
10      define("host", "localhost");
11      define("user", "root");
12      define("password", 'testtest');
13      define("db", "project");
14
15  ?>
16
17
18
```

CreateExperiment.php

```php
<?php

/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

require_once 'dbConnect.php';
require_once 'Experiment.php';



$date = "";
$time = "";
//$image = "";
//$rep = "";
//$expt = "";
$username = "";
//$userId = "3";


if (isset($_POST['date'])){
    $date = $_POST['date'];
}

if (isset($_POST['time'])){
    $time = $_POST['time'];
}

if (isset($_POST['image'])){
    $image = $_POST['image'];
}

if(isset($_POST['username'])){
    $username = $_POST['username'];
}

if(isset($_POST['userid'])){
    $userId = $_POST['userid'];
}

if (isset($_POST['rep'])){
    $rep = $_POST['rep'];
}

if (isset($_POST['expt'])){
    $expt = $_POST['expt'];
}

if (isset($_POST['treatment'])){
    $treatment = $_POST['treatment'];
}

if (isset($_POST['weather'])){
    $weather = $_POST['weather'];
}

if (isset($_POST['lat'])){
    $lat = $_POST['lat'];
}

if (isset($_POST['lon'])){
    $lon = $_POST['lon'];
}




$experimentObj = new Experiment();

// Upload experiment
if(!empty($date) && !empty($time) && !empty($image)){
```

dbConnect.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 *
 * Connection to the database
 *
 */

    include_once 'constants.php';

    class DbConnect{

        private $connect;

        public function __construct(){

            $this->connect = mysqli_connect(host, user, password, db);

            if (mysqli_connect_errno($this->connect)){

                echo "Unable to connect to MySQL Database: " . mysqli_connect_error();
            }
        }

        public function getDb(){
            return $this->connect;
        }
    }

?>
```

Experiment.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


include_once 'dbConnect.php';

class Experiment {

    private $db;

    private $db_table = 'expindex';

    public function __construct(){

        $this->db = new DbConnect();
    }


    public function addExperimentToDB($date, $time, $image, $username, $userId,
    $rep, $expt, $treatment, $weather, $lat, $lon){

        //$sql = "SELECT id FROM users WHERE username='$username'";
        //$results = mysqli_query($this->db->getDb(), $sql);
        //$row = mysqli_fetch_row($results);
        //$userId = $row[0];


        $imageName = "rep ".$rep."_treatment
        ".$treatment."_".$expt."_".$date."_".$username;

        $imagePath = "../PlantTracker/uploads/ExpInfoImages/$imageName";



        $query = "INSERT INTO ".$this->db_table." (date, time, imagepath, username,
        userid, replicant, expt, treatment, weather, lat, lon)
                    VALUES ('$date', '$time', '$imagePath', '$username', '$userId',
                    '$rep', '$expt', '$treatment', '$weather', '$lat', '$lon')";

        $inserted = mysqli_query($this->db->getDb(), $query);


        $sql = "SELECT id FROM expindex WHERE time='$time'";
        $results = mysqli_query($this->db->getDb(), $sql);
        $row = mysqli_fetch_row($results);
        $expId = $row[0];


        if($inserted == 1){

            file_put_contents($imagePath, base64_decode($image));

            $json['success'] = 1;
            $json['message'] = "Successfully created experiment";
            //$json['userid'] = $userId;
            //$json['username'] = $username;
            $json['expid'] = $expId;
            //$json['expt'] = $exp;
            //$jsom['rep'] = $rep;
            //$json['treatment'] = $treatment;

        }else{
            $json['success'] = 0;
            $json['message'] = "Error in uploading data. MSG from PHP experiment API";
        }
        return $json;
        //mysqli_close($this->db->getDb());
    }


}

?>
```

ExperimentDetails.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */


include_once 'dbConnect.php';

class ExperimentDetails{

    private $db;
    private $db_table = 'experimentindex';

    public function __construct(){
        $this->db = new DbConnect();
    }

    public function expDetailsToUser($username){

        $userSql = "SELECT id FROM users WHERE username='$username'";
        $userResults = mysqli_query($this->db->getDb(), $userSql);
        $row = mysqli_fetch_assoc($userResults);
        $userId =$row['id'];


        $sql = "SELECT id, replicant, expt, treatment FROM expindex WHERE
        username='$username'";
        $results = mysqli_query($this->db->getDb(), $sql);

        if (mysqli_num_rows($results) > 0){
            $json['message'] = "Data found";
            while ($row = mysqli_fetch_assoc($results))
                $rows[] = $row;
        }else{
            $json['message'] = "No data found";
        }
        $json['exp'] = $rows;
        $json['userid'] = $userId;
        $json['username'] = $username;
        return $json;
    }
}

?>
```

getExperimentDetails.php

```php
<?php

/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

require_once 'dbConnect.php';
require_once 'ExperimentDetails.php';

//$username = "";

if(isset($_POST['username'])){
    $username = $_POST['username'];
}

$experimentDetailsObj = new ExperimentDetails();

if(!empty($username)){

    $json_array = $experimentDetailsObj->expDetailsToUser($username);
    echo json_encode($json_array);
}

?>
```

Image.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

include once 'dbConnect.php';

class Image {

    private $db;

    private $db_table = 'images';

    public function __construct(){

        $this->db = new DbConnect();
    }


    public function addImageToDB($date, $time, $image, $lat, $lon, $weather){

        $query = "INSERT INTO ".$this->db_table." (date, time, image, lat, lon,
        weather)
                    VALUES ('$date', '$time', '$image', '$lat', '$lon', '$weather')";

        $inserted = mysqli_query($this->db->getDb(), $query);

        if($inserted == 1){
            $json['success'] = 1;
            $json['message'] = "Successfully uploaded image";

        }else{
            $json['success'] = 0;
            $json['message'] = "Error in uploading data. MSG from PHP API";
        }
        return $json;
        //mysqli_close($this->db->getDb());
    }


       public function notImageTest($date, $time){

        $query = "INSERT INTO ".$this->db_table." (date, time)
                    VALUES ('$date', '$time')";

        $inserted = mysqli_query($this->db->getDb(), $query);

        if($inserted == 1){
            $json['success'] = 1;
            $json['message'] = "Successfully uploaded Date and Time";

        }else{
            $json['success'] = 0;
            $json['message'] = "Error in uploading data. MSG from PHP API";
        }
        return $json;
        //  mysqli_close($this->db->getDb());
    }

}

?>
```

imageUpload.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

require once 'dbConnect.php';
require once 'Image.php';

$date = "";
$time = "";
$image = "";
$lat = "";
$lon = "";



if (isset($_POST['date'])){
    $date = $_POST['date'];
}

if (isset($_POST['time'])){
    $time = $_POST['time'];
}

if (isset($_POST['image'])){
    $image = $_POST['image'];
}
if (isset($_POST['lat'])){
    $lat = $_POST['lat'];
}

if (isset($_POST['lon'])){
    $lon = $_POST['lon'];
}

if (isset($_POST['weather'])){
    $weather = $_POST['weather'];
}



$imageObj = new Image();

// Upload image
if(!empty($date) && !empty($time) && !empty($image) && !empty($lon) && !empty($lat)){

    $json_array = $imageObj->addImageToDB($date, $time, $image, $lat, $lon, $weather);
    echo json_encode($json_array);
}

/*
if(!empty($date) && !empty($time)){

    $json_array = $imageObj->notImageTest($date, $time);
    echo json_encode($json_array);
}
*/

?>
```

Loginregister.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

    require_once 'user.php';

    $username = "";
    $password = "";
    $email = "";

    // username, password and email sent from app

    if (isset($_POST['username'])){
        $username = $_POST['username'];
    }

    if (isset($_POST['password'])){
        $password = $_POST['password'];
    }

    if (isset($_POST['email'])){
        $email = $_POST['email'];
    }

    $userObject = new User();

    // Registration
    if(!empty($username) && !empty($password) && !empty($email)){

        $hashed_password = hash("sha256", $password, false);
        $json_registration = $userObject->createNewUser($username, $hashed_password,
        $email);
        $success =json_encode($json_registration);
        $ret = $success['message'];
        //echo $ret;
        echo json_encode($json_registration);
    }

    // Login
    if(!empty($username) && !empty($password) && empty($email)){

        $hashed_password = hash("sha256", $password, false);
        $json_array = $userObject->loginUsers($username, $hashed_password);

        echo json_encode($json_array);
    }
?>
```

test-connection.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 */

$host = 'localhost';
$user = 'root';
$password = 'testtest';
$con = mysqli_connect($host, $user, $password);

if(con){
    echo '<h1>Connected to MySQL Database</h1>';
}else {
    echo '<h1>MySQL Server is not connected</h1>';
}

?>
```

User.php

```php
<?php
/**
 * Student Name: Darran Gahan
 * Student Number: C00098391
 *
 * Register not used at present as there is no register funtion
 * on the mobile application, left here incase of future use...
 *
 */

    include_once 'dbConnect.php';

    class User {

        private $db;
        //$userId;

        private $db_table = 'users';

        public function __construct(){

            $this->db = new DbConnect();
        }

        public function isLoginExist($username, $password) {

            $query = "SELECT * FROM ".$this->db_table." WHERE username = '$username'
            AND password = '$password' LIMIT 1";
            $result = mysqli_query($this->db->getDb(), $query);

            if (mysqli_num_rows($result) > 0 ){

                mysqli_close($this->db->getDb());

                return true;
            }

            //$userDetails = $reaults->fetch_assoc();
            //$userId = $userDetails['userid'];

            mysqli_close($this->db->getDb());

            return false;
        }



        public function isEmailUsernameExist($username, $email){

            $query = "SELECT * FROM ".$this->db_table." WHERE username '$username'
            AND email = '$email'";

            $result = mysqli_query($this->db->getDb(), $query);

            if (mysqli_num_rows($result) > 0){
                mysqli_close($this->db->getDb());

                return true;
            }

            return false;

        }

        public function isValidEmail($email){
            return filter_var($email, FILTER_VALIDATE_EMAIL) !== false;
        }

        // Create new user

        public function createNewUser($username, $password, $email){

```

```
71              // Check if user already exists
72              $isExisting = $this->doesEmailUsernameExist($username, $email);
73
74              if($isExisting){
75
76                  $json['success'] = 0;
77                  $json['message'] = "Error registering. Probably the username/email
                    already exists.....";
78              }
79
80              else{
81
82                  // Check id email is valid
83                  $isValid = $this->isValidEmail($email);
84
85                  if($isValid) {
86
87                      $query = "INSERT INTO ".$this->db_table." (username, password,
                        email, created_at, updated_at)
88                      VALUES ('$username', '$password', '$email', NOW(), NOW())";
89
90                      $inserted = mysqli_query($this->db->getDb(), $query);
91
92                      if($inserted == 1){
93                          $json['success'] = 1;
94                          $json['message'] = "Successfully registered the user";
95                          $test = "Successfully registered the user";
96
97                      }else{
98                          $json['success'] = 0;
99                          $json['message'] = "Error in registering. Probably the
                            username/email already exists";
100                         $test = "Error in registering. Probably the username/email
                            already exists";
101                     }
102
103                     mysqli_close($this->db->getDb());
104                 }
105                 else{
106                     $json['success'] = 0;
107                     $json['message'] = "Error in registering. Email Address is not
                        valid";
108                     $test =  "Error in registering. Email Address is not valid";
109                 }
110             }
111             return $json;
112             //return $test
113         }
114
115         // Login user
116         public function loginUsers($username, $password){
117
118             $json = array();
119
120             $canUserLogin = $this->isLoginExist($username, $password);
121
122             if ($canUserLogin){
123
124                 $json['success'] = 1;
125                 $json['message'] = "Successfully logged in";
126             //  $json['userid'] = $userId;
127                 $test = "Successfully logged in";
128
129             }else{
130
131                 $json['success'] = 0;
132                 $json['message'] = "Incorrect login details";
133                 $test = "Incorrect login details";
134             }
135             return $json;
136         }
137     }


138 ?>
```