Technical Manual

Animal Tracker And Health Monitoring Application

Name: Gearoid Lacey

Student Number: C00183380

Due date: 5th April 2017

Code

Html Code

index.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
      <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
    </head>
    <body class="body">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <script type="text/javascript" src="cordova.js"></script>
                <a href="Login.html" class="btn btn-default btn-lg a"
id="loginButton">Login</a>

                <a href="register.html" class="btn btn-default btn-lg a"
id="registerButton">Register</a>

                <a href="changePassword.html" class="btn btn-default btn-lg a"
id="changeButton">Change Password</a>
        </div>
    </body>
</html>
```

login.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.min.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script type="text/javascript" src="userlogin.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">

    </head>
    <body class="body">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <script type="text/javascript" src="cordova.js"></script>

                <form id="loginForm" method="post">
                <label class="labels" >Username</label>

                <input type="text" name = "userName" required class="input">

                <label class="labels">Password</label>

                <input type="password" name = "userPassword" required
class="input">

                <button class="btn btn-default btn-lg a" id="loginButton"
onclick="userlogFunction();">Login</button>

                <button class="btn btn-default btn-lg a" id="backButton"
onclick=back()>Back</button>

            </form>
        </div>
    </body>
</html>
```

register.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
       <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">
         <script type="text/javascript" src="registerTwo.js"></script>
    </head>
    <body class="body">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <script type="text/javascript" src="cordova.js"></script>
            <form id="registerForm" method="post" >
                <label class="labels">Username</label>

                <input type="text" name = "registerUserName" required class="input"
id="registerUserName">

                <label class="labels">Password</label>

                <input type="password" name = "registerUserPassword" required
class="input" id="registerUserPassword">

                <label class="labels">Confirm Password</label>

                <input type="password" name = "confirmRegisterUserPassword"
required class="input" id="confirmRegisterUserPassword">

                <button href="home.html" class="btn btn-default btn-lg a"
id="registerButton" onclick=register()>Register</button>

                <button href="index.html" class="btn btn-default btn-lg a"
id="backButton" onclick=back()>Back</button>
            </form>
        </div>
    </body>
</html>
```

changePassword.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script type="text/javascript" src="changePassword.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">

    </head>
    <body class="body">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <script type="text/javascript" src="cordova.js"></script>
            <form id="changePasswordForm" method="post">
                <label class="labels">Username</label>

                <input type="text" name = "changeUserName" required class="input"
id="changeUserName">

                <label class="labels"> Current Password</label>

                <input type="password" name = "oldUserPassword" required
class="input" id="oldUserPassword">

                <label class="labels"> New Password</label>

                <input type="password" name = "changeUserPassword" required
class="input" id="changeUserPassword">

                <label class="labels">Confirm New Password</label>

                <input type="password" name = "confirmChangeUserPassword" required
class="input" id="confirmChangeUserPassword">

                <button class="btn btn-default btn-lg a" id="changeButton"
onclick=changePass()>Change Password</button>

                <button class="btn btn-default btn-lg a" id="backButton"
onclick=back()>Back</button>
            </form>
        </div>
    </body>
</html>
```

home.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script type="text/javascript" src="currentLocation.js"></script>
        <script type="text/javascript" src="logout.js"></script>
        <script type="text/javascript" src="animal_Paths.js"></script>
        <script type="text/javascript" src="update_gps.js"></script>
        <script type="text/javascript" src="update_animal.js"></script>
        <script type="text/javascript" src="get_delete_details.js">
        </script>
        <script type="text/javascript" src="populate_details.js">
        </script>
        <script type="text/javascript" src="select_animals.js">
        </script>
         <script type="text/javascript" src="storage.js">
        </script>
        <script type="text/javascript" src="select_animal_boundary.js">
        </script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">


    </head>
    <body class="body" onload="clear_storage();">

        <button class=" a menubutton button1" id="addAnimalProfileButton"
onclick="window.location='addAnimalProfile.html';">Add Animal Profile</button>
        <button class=" a menubutton button2" id="createBoundaryButton"
onclick="get_animal();">Create Boundary</button>
        <button class=" a menubutton button3"
id="currentLocationButton"  onclick="animal_location()">Latest Location</button>
        <button class=" a menubutton button4" id="latestPathButton"
onclick="get_animal_data()">Analyse Paths</button>
        <button class=" a menubutton button5" id="gpsConfigurationButton"
onclick="update_gps()">GPS Configuration</button>
        <button class=" a menubutton button6" id="myProfileButton"
onclick="window.location='cluster.html';">Cluster Locations</button>
        <button class=" a menubutton button6"
id="myProfileButton"  onclick="get_animals()">Update Animal Profile</button>
        <button class=" a menubutton button7"
id="myProfileButton"  onclick="delete_details()">Delete Animal Profile</button>
        <button class=" a menubutton button7"
id="myProfileButton"  onclick="logout_user()">Logout</button>
    </body>
</html>
```

addAnimalProfile.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
       <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">
         <script type="text/javascript" src="animalProfile.js"></script>
    </head>
    <body class="body">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <script type="text/javascript" src="cordova.js"></script>
            <form id="profileForm">
                <label class="labels">Animal ID</label>

                <input type="text" name = "animalID" required class="input"
id="animalID">

                <label class="labels">Type</label>

                <select type="choose" name = "animalType" required class="select"
id="animalType" placeholder ="---Select Animal Type---">

                <option value="cow">Cow</option>
                <option value="dog">Dog</option>
                <option value="horse">Horse</option>
                <option value="cat">Cat</option>
                <option value="sheep">Sheep</option>
                <option value="other">Other</option>
                </select>

                <label class="labels">Breed</label>

                <input type="text" name = "animalBreed" required class="input"
id="animalBreed"  placeholder="Angus">

                <label class="labels">Weight (Kilograms)</label>

                <input type="number" name = "animalWeight" required class="input"
id="animalWeight" placeholder="50">

                <label class="labels">Gender</label>
```

```
                <input type="text" name = "animalGender" required class="input"
id="animalGender" pattern="[mfMF]{1}"  placeholder="M or F">

                <label class="labels">Trackers Sim Card Number)</label>

                <input type="number" name = "trackingNum" required class="input"
id="trackingNum" placeholder="0851234567">

                <button class="btn btn-default btn-lg a" id="registerButton"
onclick="add_animal()">Add Animal</button>

                <button class="btn btn-default btn-lg a" id="backButton"
onclick=back()>Back</button>
            </form>
        </div>
    </body>
</html>
```

create_boundary.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width"/>
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script src="https://code.jquery.com/jquery-migrate-3.0.0.js"></script>

        <script src="http://cdn.leafletjs.com/leaflet/v0.7.7/leaflet.js"></script>

        <script src='https://api.tiles.mapbox.com/mapbox-gl-js/v0.34.0/mapbox-
gl.js'></script>
        <link href='https://api.tiles.mapbox.com/mapbox-gl-js/v0.34.0/mapbox-
gl.css' rel='stylesheet' />

        <script type="text/javascript" src="backButton.js"></script>
        <script type="text/javascript" charset="utf-8" src="cordova.js"></script>
        <script type="text/javascript" src="populate_animal.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>


        <style>
            .map { position:absolute; top:0%; bottom: 40%; width:100%;  }
            .data{position:absolute; top:60%; bottom: 0%;}
            .marker {
                    display: block;
                    border: none;
                    border-radius: 50%;
                    cursor: pointer;
                    padding: 0;
                    }
            .coordinates {
                        background: rgba(0,0,0,0.5);
                        color: #fff;
                        position: absolute;
                        bottom: 10px;
                        left: 10px;
                        padding:5px 10px;
                        margin: 0;
                        font-size: 11px;
                        line-height: 18px;
                        border-radius: 3px;
                        display: none;
                        }
        </style>
</head>
<body class="body" onload="onLoad();">
```

```
    <div class="map" id ="map">

    <script>
        mapboxgl.accessToken =
'pk.eyJ1IjoiZ2FybDk0IiwiYSI6ImNpeTVucXc5YzAwNG0ycW82MTFqeWl5bmQifQ.L3m2Ue3OOucOI3Xr
hwjUIQ';
        // Holds mousedown state for events. if this
        // flag is active, we move the point on `mousemove`.
        var isDragging;

        // Is the cursor over a point? if this
        // flag is active, we listen for a mousedown event.
        var isCursorOverPoint;

        var coordinates = document.getElementById('coordinates');
        var map = new mapboxgl.Map({
            container: 'map',
            style: 'mapbox://styles/garl94/ciya2fzm200692roox15xpp7q',
            center: [-7.6921, 53.1424],
            zoom: 6
        });

        var canvas = map.getCanvasContainer();

        var geojson = {
            "type": "FeatureCollection",
            "features": [{
                "type": "Feature",
                "geometry": {
                    "type": "Point",
                    "coordinates": [-7.6921, 53.1424]
                }
            }]
        };

        function mouseDown() {
            if (!isCursorOverPoint) return;

            isDragging = true;

            // Set a cursor indicator
            canvas.style.cursor = 'grab';

            // Mouse events
            map.on('touchmove', onMove);
            map.once('touchend', onUp);
        }

        function onMove(e) {
            if (!isDragging) return;
            var coords = e.lngLat;
            console.log(coords);

            // Set a UI indicator for dragging.
            canvas.style.cursor = 'grabbing';

            // Update the Point feature in `geojson` coordinates
            // and call setData to the source layer `point` on it.
            geojson.features[0].geometry.coordinates = [coords.lng, coords.lat];
            map.getSource('point').setData(geojson);
        }
```

```
        function onUp(e) {
            if (!isDragging) return;
            var coords = e.lngLat;

            // Print the coordinates of where the point had
            // finished being dragged to on the map.

            $("#longitude").val(coords.lng);
            $("#latitude").val(coords.lat);
            isDragging = false;

            // Unbind mouse events
            map.off('touchmove', onMove);
        }

        map.on('load', function() {
            // Add a single point to the map
            map.addSource('point', {
                "type": "geojson",
                "data": geojson
            });

            map.addLayer({
                "id": "point",
                "type": "circle",
                "source": "point",
                "paint": {
                    "circle-radius": 10,
                    "circle-color": "#3887be"
                }
            });

            // If a feature is found on map movement,
            // set a flag to permit a mousedown events.
            map.on('click', function(e) {
                var features = map.queryRenderedFeatures(e.point, { layers:
['point'] });

                // Change point and cursor style as a UI indicator
                // and set a flag to enable other mouse events.
                if (features.length) {
                    map.setPaintProperty('point', 'circle-color', '#3bb2d0');
                    canvas.style.cursor = 'move';
                    isCursorOverPoint = true;
                    map.dragPan.disable();
                } else {
                    map.setPaintProperty('point', 'circle-color', '#3887be');
                    canvas.style.cursor = '';
                    isCursorOverPoint = false;
                    map.dragPan.enable();
                }
            });

            // Set `true` to dispatch the event before other functions call it.
This
            // is necessary for disabling the default map dragging behaviour.
            map.on('touchstart', mouseDown, true);
        });
    </script>

 </div>
    <div class="body">
```

```
        <div class="data">
            <pre id='coordinates' class='coordinates'></pre>

            <label class="labels">Select An Animal</label>
            <select type="choose" name="animal" id="animal" class="select"
onchange="populate_fields()"></select>

            <script type="text/javascript">
               populate();
            </script>

            <input type="hidden" name="longitude" id="longitude" class="input">

            <input type="hidden" name="latitude" id="latitude" class="input">

            <label class="labels">Boundary Radius(meters)</label>
            <input type="number" name="distance" id="distance" class="input">

            <input type="hidden" name ="trackingNum" required class="input"
id="trackingNum">

            <button  class="btn btn-default btn-lg a button7" id="boundary"
onclick="sendSMS()">Create Boundary</button>
        </div>
    </div>
</body>
</html>
```

currentLocation.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width"/>
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script src="https://code.jquery.com/jquery-migrate-3.0.0.js"></script>
         <meta name='viewport' content='initial-scale=1,maximum-scale=1,user-
scalable=no' />
        <script src='https://api.tiles.mapbox.com/mapbox-gl-js/v0.31.0/mapbox-
gl.js'></script>
        <link href='https://api.tiles.mapbox.com/mapbox-gl-js/v0.31.0/mapbox-
gl.css' rel='stylesheet' />
        <script type="text/javascript" src="backButton.js"></script>
        <script type="text/javascript" charset="utf-8" src="cordova.js"></script>

        <style>
        body { margin:0; padding:0; }
        #map { position:absolute; top:0; bottom:0; width:100%; }
        .mapboxgl-popup
        {
            max-width: 400px;
            font: 12px/20px 'Helvetica Neue', Arial, Helvetica, sans-serif;
        }
    </style>
    </head>
    <body class="body" onload="onLoad()">

        <div class="map" id ="map">
                <script>
                    var gps
=JSON.parse(window.localStorage.getItem("coordinates"));
                    var coordinate_array;

                    mapboxgl.accessToken =
'pk.eyJ1IjoiZ2FybDk0IiwiYSI6ImNpeTVucXc5YzAwNG0ycW82MTFqeWl5bmQifQ.L3m2Ue3OOucOI3Xr
hwjUIQ';
                    var map = new mapboxgl.Map({
                        container: 'map', // container id
                        style: 'mapbox://styles/mapbox/satellite-v9', //stylesheet
location
                        center: [-6.934135900000001,52.8365072], // starting
position longitude followed by latitude
                        zoom: 5 // starting zoom
                    });


                    map.on('load', function () {
                        var i = 0;
                        var outer_loop = 0;
                        var id_value ="";
```

```
                        var details ="";
                        var id_array =[];

                        while(outer_loop < gps.length)
                            {

                                var separated_coordinates =
gps.splice(outer_loop,outer_loop+6);
                                details = "Time: "+ separated_coordinates[i+2] + "
Date: " + separated_coordinates[i+3];

                                            id_value =
separated_coordinates[i+4].toString();
                                            id_array.push(id_value);
                                            map.addLayer({
                                                "id": id_value,
                                                "type": "symbol",
                                                "source": {
                                                    "type": "geojson",
                                                    "data": {
                                                        "type":
"FeatureCollection",

                                                        "features": [{
                                                            "type": "Feature",
                                                            "geometry": {
                                                                "type": "Point",
                                                                "cluster":true,
                                                                "coordinates":
[parseFloat(separated_coordinates[i]),parseFloat(separated_coordinates[i+1])]
                                                            },
                                                            "properties": {
                                                                "description":
details,

                                                                "title":
separated_coordinates[i+4],

                                                                "icon": "dog-park"
                                                            }
                                                        }]
                                                    }
                                                },
                                                "layout": {
                                                    "icon-image": "{icon}-15",
                                                    "text-field": "{title}",
                                                    "text-font": ["Open Sans
Semibold", "Arial Unicode MS Bold"],

                                                    "text-offset": [0, 0.6],
                                                    "text-anchor": "top"
                                                }

                                            });

                        }
                        for(var i =0; i < id_array.length; i++)
                        {
                            map.on('click', function (e)
                                        {
                                            var features =
map.queryRenderedFeatures(e.point, { layers: id_array[i] });

                                            if (!features.length) {
                                                return;
                                            }
```

```
                                        var feature = features[0];
                                        // Populate the popup and set its
coordinates
                                        // based on the feature found.
                                        var popup = new mapboxgl.Popup()
                                            .setLngLat(feature.geometry.coordin
ates)
                                            .setHTML(feature.properties.descrip
tion)
                                            .addTo(map);
                                    });

                            }

                        });

                        window.localStorage.clear();
                    </script>
                </div>
            </body>
        </html>
```

analyseLocation.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width"/>
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script src="https://code.jquery.com/jquery-migrate-3.0.0.js"></script>
         <meta name='viewport' content='initial-scale=1,maximum-scale=1,user-
scalable=no' />
        <script src='https://api.tiles.mapbox.com/mapbox-gl-js/v0.31.0/mapbox-
gl.js'></script>
        <link href='https://api.tiles.mapbox.com/mapbox-gl-js/v0.31.0/mapbox-
gl.css' rel='stylesheet' />
        <script type="text/javascript" src="backButton.js"></script>
        <script type="text/javascript" charset="utf-8" src="cordova.js"></script>

        <style>
        body { margin:0; padding:0; }
        #map { position:absolute; top:0; bottom:0; width:100%; }
    </style>
    </head>
    <body class="body" onload="onLoad();">

        <div class="map" id ="map">
                <script>
                    var the_gps
=JSON.parse(window.localStorage.getItem("analyse_coordinates"));
                    var coordinate_array = Object.values(the_gps);

                    var temp_start = coordinate_array[0];
                    var start_coordinates = Object.values(temp_start);
                    var id_array =[];

                    mapboxgl.accessToken =
'pk.eyJ1IjoiZ2FybDk0IiwiYSI6ImNpeTVucXc5YzAwNG0ycW82MTFqeWl5bmQifQ.L3m2Ue3OOucOI3Xr
hwjUIQ';
                    var map = new mapboxgl.Map({
                        container: 'map', // container id
                        style: 'mapbox://styles/mapbox/satellite-v9', //stylesheet
location
                        center: [start_coordinates[0],start_coordinates[1]], //
starting position longitude followed by latitude
                        zoom: 18 // starting zoom
                    });


                    map.on('load', function () {

                        var i = 0;
                        var ii = 0;
                        var animal_ids=[];
```

```
                        var details ="";
                        var id_value ="";

                        var id =0;

                        //GET UNIQUE ANIMAL ID'S
                        for(i;i<coordinate_array.length;i++)
                            {
                                var temp_object = coordinate_array[i];
                                var temp_array = Object.values(temp_object);

                                if(temp_array.length == 5)
                                    {
                                        if(animal_ids.indexOf(temp_array[4]) == -1
| animal_ids.length == 0)
                                            {
                                                animal_ids.push(temp_array[4]);
                                            }
                                    }

                            }

                        i = 0;//reset value of i
                        var animal_coordintes=[];
                        var animal_data=[];

                        for(i;i<animal_ids.length;i++)
                            {
                                //FOR EACH ANIMAL ID GET THE COORDINATES ASSOCIATED
WITH IT AND STORE IN AN ARRAY OF ARRAYS
                                var animal = animal_ids[i];

                                ii = 0;
                                for(ii;ii<coordinate_array.length;ii++)
                                    {
                                        var temp_object = coordinate_array[ii];
                                        var temp_array =
Object.values(temp_object);
                                        if(temp_array.length == 5 & temp_array[4]
== animal )
                                            {
                                                animal_data.push([temp_array[0],tem
p_array[1],temp_array[2],temp_array[3],temp_array[4]]);
                                                animal_coordintes.push([temp_array[
0],temp_array[1]]);
                                            }

                                    }

                                id = parseInt(Math.floor((Math.random() * 100000) +
1)).toString();

                                                map.addLayer({
                                                        "id": id,
                                                        "type": "line",
                                                        "source": {
                                                            "type": "geojson",
                                                            "data": {
                                                                "type": "Feature",
                                                                "properties": {
                                                                },
                                                                "geometry": {
```

```
                                                        "type":
"LineString",
                                                        "coordinates":a
nimal_coordintes
                                                    }
                                                }
                                            },
                                            "layout": {
                                                "line-join": "round",
                                                "line-cap": "round"
                                            },
                                            "paint": {
                                                "line-color": "#500",
                                                "line-width": 8
                                            }
                                        });


                                id = 0;
                                for(ii=0;ii <animal_data.length; ii++)
                                    {
                                        var separated_coordinates =
Object.values(animal_data[ii])
                                        id_value = separated_coordinates[4];
                                        console.log("id_value is: " + id_value);
                                        id_array.push(id_value);
                                        details = "Time: " + separated_coordinates[2] +
" " + "Date: " + separated_coordinates[3]

                                        id = parseInt(Math.floor((Math.random() *
1000000) + 1)).toString();
                                        map.addLayer({
                                            "id": id,
                                            "type": "symbol",
                                            "source": {
                                                "type": "geojson",
                                                "data": {
                                                    "type":
"FeatureCollection",

                                                    "features": [{
                                                        "type": "Feature",
                                                        "geometry": {
                                                            "type":
"Point",

                                                            "cluster":true,
                                                            "coordinates":
[parseFloat(separated_coordinates[0]),parseFloat(separated_coordinates[1])]
                                                        },
                                                        "properties": {
                                                            "description":
details,

                                                            "title":
id_value,
                                                            "icon": "dog-
park"
                                                        }
                                                    }]
                                                }
                                            },
                                            "layout": {
                                                "icon-image": "{icon}-15",
                                                "text-field": "{title}",
```

```
                                                    "text-font": ["Open Sans
Semibold", "Arial Unicode MS Bold"],
                                                    "text-offset": [0, 0.6],
                                                    "text-anchor": "top"
                                                }
                                            });
                                        id = 0;
                                    }
                                animal_coordintes.length = 0;
                                ii =0;

                            }
                            for(var i =0; i < id_array.length; i++)
                                {
                                    map.on('click', function (e)
                                            {
                                            var features =
map.queryRenderedFeatures(e.point, { layers: id_array[i] });

                                            if (!features.length) {
                                                return;
                                            }

                                            var feature = features[0];
                                            // Populate the popup and set its
coordinates
                                            // based on the feature found.
                                            var popup = new mapboxgl.Popup()
                                                .setLngLat(feature.geometry.coo
rdinates)
                                                .setHTML(feature.properties.des
cription)
                                                .addTo(map);
                                        });

                                }


                        });
                        window.localStorage.clear();
                        </script>
                </div>
            </body>
        </html>
```

gpsConfiguration.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width"/>
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script src="https://code.jquery.com/jquery-migrate-3.0.0.js"></script>
        <script type="text/javascript" src="backButton.js"></script>
        <script type="text/javascript" charset="utf-8" src="cordova.js"></script>
        <script type="text/javascript" src="change_gps_config.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">

    </head>
    <body class="body" onload="load_data();">
        <script>
            onLoad();//listen for back button event
        </script>
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <div class="mainBody">

        <label class="labels" >Select an animal</label>

                <select type="choose" name = "animal" required class="select"
id="animal">
                </select>
        <br>
        <label class="labels">Times Metric</label>

                <select type="choose" name = "metric" required class="select"
id="metric">
                    <option value="seconds">Seconds</option>
                    <option value="minutes">Minutes</option>
                </select>
        <br>
        <label class="labels">Intermittent Retrieval Times</label>

                <input type="number" name = "time" required class="select"
id="time" placeholder ="5">

        <button class="btn btn-default btn-lg a" id="updateGps"
onclick="change_gps()">Update Gps Configuration</button>
        </div>

    </body>
</html>
```

cluster.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script type="text/javascript" src="cordova.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">
        <script type="text/javascript" src="get_cluster_details.js"></script>
    </head>
    <body class="body">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <form id ="graph_form">

                <label class="labels">Cluster Distances (meters)</label>
                <select type="number" name ="eps" required class="input" id="eps">
                    <option value="5">5</option>
                    <option value="10">10</option>
                    <option value="15">15</option>
                    <option value="20">20</option>
                    <option value="25">25</option>
                    <option value="30">30</option>
                    <option value="35">35</option>
                    <option value="40">40</option>
                    <option value="45">45</option>
                    <option value="50">50</option>
                </select>

                <label class="labels">Number of Locations per Cluster</label>
                <select type="number" name ="num_loc" required class="input"
id="num_loc">
                    <option value="5">5</option>
                    <option value="10">10</option>
                    <option value="15">15</option>
                    <option value="20">20</option>
                    <option value="25">25</option>
                    <option value="30">30</option>
                    <option value="35">35</option>
                    <option value="40">40</option>
                    <option value="45">45</option>
                    <option value="50">50</option>
                </select>

                <label class="labels">Select a Start Date</label>
```

```
                <input type="date" name ="start_date" required class="input"
id="start_date">

                <label class="labels">Select an End Date</label>
                <input type="date" name ="end_date" required class="input"
id="end_date">


                <button class="btn btn-default btn-lg a" id="registerButton"
onclick=get_cluster_data()>Display Cluster Locations</button>

                <button class="btn btn-default btn-lg a" id="backButton"
onclick=back()>Back</button>
            </form>
        </div>
    </body>
</html>
```

movement_graph.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script type="text/javascript" src="cordova.js"></script>
        <script src="http://d3js.org/d3.v3.min.js"></script>
        <script type="text/javascript" src="backButton.js"></script>
        <meta name='viewport' content='initial-scale=1,maximum-scale=1,user-
scalable=no' />
        <script src='https://api.tiles.mapbox.com/mapbox-gl-js/v0.31.0/mapbox-
gl.js'></script>
        <link href='https://api.tiles.mapbox.com/mapbox-gl-js/v0.31.0/mapbox-
gl.css' rel='stylesheet' />
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">
        <style>
            body { margin:0; padding:0; }
            #map { position:absolute; top:0; bottom:0; width:100%; }
            .mapboxgl-popup
            {
                max-width: 400px;
                font: 12px/20px 'Helvetica Neue', Arial, Helvetica, sans-serif;
            }
        </style>



    </head>
    <body class="body" onload="onLoad()">

        <div class="map" id ="map">
                <script>
                    var gps
=JSON.parse(window.localStorage.getItem("graph_Data"));
                    var coordinate_array = Object.values(gps);
                    if(coordinate_array.length <= 1)
                        {
                            bootbox.alert("No data available, try again
later",function()
                                {
                                    window.location.replace("home.html");
                                });
                        }
                    else
                    {
```

```
                    var id_array=[];
                    var start = Object.values(coordinate_array[0][0])


                    mapboxgl.accessToken =
'pk.eyJ1IjoiZ2FybDk0IiwiYSI6ImNpeTVucXc5YzAwNG0ycW82MTFqeWl5bmQifQ.L3m2Ue3OOucOI3Xr
hwjUIQ';
                                var map = new mapboxgl.Map({
                                    container: 'map', // container id
                                    style:
'mapbox://styles/mapbox/satellite-v9', //stylesheet location
                                    center: [start[0],start[1]], //
starting position longitude followed by latitude
                                    zoom: 10 // starting zoom
                                });

                    map.on('load', function ()
                            {
                                for(var i = 0; i< coordinate_array.length;i++)
                                    {

                                    var separated_coordinates =
Object.values(coordinate_array[i]);
                                    if(separated_coordinates != " ")
                                        {
                                            var data =
Object.values(separated_coordinates)
                                            var details =data[1];

                                            var id_value =
Object.values(details);

                                            var animal_description = 'Name:
' + id_value[0] + "\n";
                                            animal_description =
animal_description + 'Animal Type: ' + id_value[1] + "\n";
                                            animal_description =
animal_description + 'Animal Breed: ' + id_value[2] + "\n";
                                            animal_description =
animal_description + 'Animal Weight: ' + id_value[3] + "\n";
                                            animal_description =
animal_description + 'Animal Breed: ' + id_value[4] + "\n";
                                            animal_description =
animal_description + 'Animal Tracking Number: ' + id_value[5] + "\n";


                                            id_array.push(id_value[0])
                                            var id =
parseInt(Math.floor((Math.random() * 1000000) + 1)).toString();
                                            map.addLayer({
                                                "id": id,
                                                "type": "symbol",
                                                "source": {
                                                    "type": "geojson",
                                                    "data": {
                                                        "type":
"FeatureCollection",

                                                        "features": [{
                                                            "type":
"Feature",

                                                            "geometry": {
```

```
                                                                    "type":
"Point",
                                                                    "cluster":t
rue,
                                                                    "coordinate
s": [parseFloat(data[0][0]),parseFloat(data[0][1])]
                                                                },
                                                                "properties":{
                                                                    "descripti
on": animal_description,
                                                                    "title":
id_value[0],
                                                                    "icon":
"triangle"
                                                                }

                                                            }]
                                                        }
                                                    },
                                                    "layout": {
                                                        "icon-image": "{icon}-
15",
                                                        "text-field":
"{title}",
                                                        "text-font": ["Open
Sans Semibold", "Arial Unicode MS Bold"],
                                                        "text-offset": [0,
0.6],
                                                        "text-anchor": "top"
                                                    }

                                                });
                                            }
                                        }

                                for(var i =0; i < id_array.length; i++)
                                {
                                    map.on('click', function (e)
                                    {
                                        var features =
map.queryRenderedFeatures(e.point, { layers: id_array[i] });

                                        if (!features.length) {
                                            return;
                                        }

                                        var feature = features[0];
                                        // Populate the popup and set its
coordinates
                                        // based on the feature found.
                                        var popup = new mapboxgl.Popup()
                                            .setLngLat(feature.geometry.coo
rdinates)
                                            .setHTML(feature.properties.des
cription)
                                            .addTo(map);
                                    });
                                }

                            });

                        }
```

```
                window.localStorage.clear();
                </script>
        </div>
    </body>
</html>
```

select_animal.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
        <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script type="text/javascript" src="cordova.js"></script>
        <script type="text/javascript" src="process_select.js"></script>
        <script type="text/javascript" src="animal_Paths.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">

    </head>
    <body class="body" onload="process_animal()">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <form id ="select_form">


                <label class="labels">Select Animal</label>
                <select type="choose" name = "animal" required class="select"
id="animal" onchange="populate_select()">
                </select>

                <input type="hidden" name ="trackingNum" required class="input"
id="trackingNum">

                <label class="labels">Select Another Animal</label>
                <select type="choose" name = "animalTwo" required class="select"
id="animalTwo" onchange="populate_select()">
                </select>

                 <input type="hidden" name ="trackingNumTwo" required class="input"
id="trackingNumTwo">

                <label class="labels">Select a Start Date</label>
                <input type="date" name = "date" required class="select" id="date">

                <label class="labels">Select an End Date</label>
                <input type="date" name = "enddate" required class="select"
id="enddate">


                <button class="btn btn-default btn-lg a" id="registerButton"
onclick="animal_paths()">Submit </button>
```

```
                <button class="btn btn-default btn-lg a" id="backButton"
onclick=back()>Back</button>
            </form>

        </div>
    </body>
</html>
```

update_animal_profile.html

```
<!DOCTYPE html>
<html class="html">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-
scalable=no, width=device-width">
        <title>AniMap</title>
        <link rel="stylesheet" type="text/css" href="loginform.css">
        <link rel="stylesheet" type="text/css" href="/bootstrap-3.3.7-
dist/css/bootstrap.css">
        <link rel="stylesheet" type="text/css" href="bootstrap.css">
        <link href="https://fonts.googleapis.com/css?family=Rajdhani:300"
rel="stylesheet">
       <script src="https://code.jquery.com/jquery-3.0.0.js"></script>
        <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
crossorigin="anonymous"></script>
        <script src="bootbox.min.js"></script>
        <link rel="stylesheet" type="text/css" href="bootbox.min.js">

         <script type="text/javascript" src="process_animal_update.js"></script>
        <script type="text/javascript" src="update_new_details.js"></script>
    </head>
    <body class="body" onload="process_animal();">
        <h1 class="heading"><img src="Logo.png" class="logo"></h1>
        <br>
        <div class="mainBody">
            <script type="text/javascript" src="cordova.js"></script>
            <form id="updateProfileForm" >
                <label class="labels">Update Animal</label>
                <select type="choose" name = "animal" required class="select"
id="animal" onchange="populate_fields()">
                </select>

                <label class="labels">Animal ID</label>

                <input type="text" name = "animalID" required class="input"
id="animalID">

                <label class="labels">Type</label>

                <select type="choose" name = "animalType" required class="select"
id="animalType" placeholder ="---Select Animal Type---">

                <option value="cow">Cow</option>
                <option value="dog">Dog</option>
                <option value="horse">Horse</option>
                <option value="cat">Cat</option>
                <option value="sheep">Sheep</option>
                <option value="other">Other</option>
                </select>

                <label class="labels">Breed</label>

                <input type="text" name = "animalBreed" required class="input"
id="animalBreed"  placeholder="Angus">

                <label class="labels">Weight (Kilograms)</label>
```

```
                <input type="number" name = "animalWeight" required class="input"
id="animalWeight" placeholder="50">

                <label class="labels">Gender</label>

                <input type="text" name = "animalGender" required class="input"
id="animalGender" pattern="[mfMF]{1}"  placeholder="M or F">

                <label class="labels">Trackers Sim Card Number)</label>

                <input type="number" name = "trackingNum" required class="input"
id="trackingNum" placeholder="0851234567">

                <input type="hidden" name ="oldTrackingId" required class="input"
id="oldTrackingId">

                 <input type="hidden" name ="oldanimalIdentifier" required
class="input" id="oldanimalIdentifier">

                <button class="btn btn-default btn-lg a" id="registerButton"
onclick=update_details()>Update Animal</button>

                <button class="btn btn-default btn-lg a" id="backButton"
onclick=back()>Back</button>
            </form>
        </div>
    </body>
</html>
```

JavaScript Code

animal_Paths.js

```
function animal_paths()
{
    event.preventDefault();
        $.ajax({
            url: "http://gProject.pythonanywhere.com/analyse_paths",
            type: 'POST',
            data: $('#select_form').serialize(),
            async: true})
            .done(function(response) {
                var result = JSON.parse(response);
                console.log(result);
                if(result['status'])
                    {
                        if (result['status'] == 'No Animal Selected')
                            {
                                bootbox.alert("Please Select An Animal", function()
                                    {
                                        window.location.replace("select_animal.html
");
                                    })
                            }
                        if (result['status'] == 'Your session has timed out, please
log in again')
                            {
                                bootbox.alert("Your session has timed out, please
log in again", function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if (result['status'] == 'No Date Selected')
                            {
                                bootbox.alert("No Date Selected. Please select a
date.",function()
                                    {
                                        window.location.replace("select_animal.html
");
                                    })
                            }
                        if (result['status'] == 'Date order')
                            {
                                bootbox.alert("The end date cannot be before the
start date",function()
                                    {
                                        window.location.replace("select_animal.html
");
                                    })
                            }
                        if (result['status'] == 'Same Dates')
                            {
                                bootbox.alert("The dates cannot be the
same",function()
                                    {
                                        window.location.replace("select_animal.html
");
```

```
                                    })
                                }

                            }
                        else
                        {
                            process_path_data(response);
                            window.location.replace ("analyseLocation.html");
                        }

                })
};
function back()
{
    window.location.replace ("home.html");
};
function process_path_data(location_paths)
{
    window.localStorage.setItem("analyse_coordinates", location_paths);
}
```

animalProfile.js

```
function add_animal()
{
    event.preventDefault()
        $.ajax({
            url: "http://gProject.pythonanywhere.com/animalProfile",
            data: $('#profileForm').serialize(),
            type: 'POST',
            async: false})
            .done(function(response) {
                var result = JSON.parse(response);
                if (result["status"] == "ok")
                    {
                        bootbox.alert("Animal profile added
successfully",function()
                            {
                                window.location.replace("home.html");
                            })
                    }
                if(result["status"] == "Animal id already exists")
                    {
                        bootbox.alert("The animal id already exists please try
again",function()
                            {
                                window.location.replace ("addAnimalProfile.html");
                            })
                    }
                if(result["status"] == "Wrong gender")
                    {
                        bootbox.alert("The animal gender can only be M or
F",function()
                            {
                                window.location.replace ("addAnimalProfile.html");
                            })
                    }
                if(result["status"] == "Empty fields")
                    {
                        bootbox.alert("Please fill in all the fields",function()
                            {
                                window.location.replace ("addAnimalProfile.html");
                            })
                    }
                if(result["status"] == "logged out")
                    {
                        bootbox.alert("You need to login",function()
                            {
                                window.location.replace ("index.html");
                            })
                    }
                if(result["status"] == "Session timed out, please log in again")
                    {
                        bootbox.alert("You need to login",function()
                            {
                                window.location.replace ("index.html");
                            })
                    }
        })
}
```

```
function back()
{
    window.location.replace ("home.html");
};
```

backButton.js

```
//Following code got from
http://docs.phonegap.com/en/4.0.0/cordova_events_events.md.html#backbutton
function onLoad() {
        document.addEventListener("deviceready", onDeviceReady, false);
    }

    // device APIs are available
    function onDeviceReady() {
        // Register the event listener
        document.addEventListener("backbutton", onBackKeyDown, false);
    }

    // Handle the back button
    function onBackKeyDown()
    {

        window.location.replace("home.html");
    }
```

change_gps_config.js

```
function load_data()
{
    var animal_data =
JSON.parse(window.localStorage.getItem("associated_animals"));

    var animal_array = Object.values(animal_data);
    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);

            if(temp_array.length == 2)
                {
                    if(parsed_array.indexOf(temp_array[0] == -1 |
parsed_array.length ==0))
                        {
                            parsed_array.push(temp_array[0]);
                        }

                }
        }
    //populate drop down list with animal names
    var select_tag = document.getElementById("animal");
    var option;
    option = document.createElement("option");
    option.text = "--Select an Animal--";
    select_tag.add(option);

    for(var i =0; i < parsed_array.length;i++)
        {
            option = document.createElement("option");
            option.text = parsed_array[i];
            select_tag.add(option);
        }


}
function change_gps()
{
    //this function takes user input the sends a text to the NanoTracker
    var animal = document.getElementById("animal").value;
    if(animal == "--Select an Animal--")
        {
            bootbox.alert("Please select an animal",function()
                {
                    window.location.replace("gpsConfiguration.html");
                })
        }
    var metric = document.getElementById("metric").value;
    var time = parseInt(document.getElementById("time").value);
    var phone_number;

    if(typeof(time) != "number" & Number.isInteger(time) == false)
        {
            bootbox.alert("You must enter an integer value",function()
                {
                    window.location.replace("gpsConfiguration.html");
                })
        }
```

```
    if(metric != "seconds")
        {
            time = time * 60;
        }


    var animal_data =
JSON.parse(window.localStorage.getItem("associated_animals"));

    var animal_array = Object.values(animal_data);
    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);

            if(temp_array.length == 2)
                {
                    if(temp_array[0] == animal)
                        {
                            phone_number = temp_array[1];
                        }

                }
        }
    var message ="interval:"
    message = message + time + ','

    //https://github.com/cordova-sms/cordova-sms-plugin

    var success = function () { alert('Message sent successfully'); };

    var error = function (e) { alert('Message Failed:' + e); };
    SMS.sendSMS(phone_number, message, success, error);
    window.location.replace("home.html");
}
```

changePassword.js

```
function changePass()
{
    event.preventDefault();
    console.log("In changePass");
        $.ajax({
            url: "http://gProject.pythonanywhere.com/changePassword",
            data: $('#changePasswordForm').serialize(),
            type: 'POST',
            async: false})
            .done(function(response) {
                console.log(response);

                var result = JSON.parse(response);
                var updated = "updated";
                var noExistance = "Username does not exist";
                var empty = "Empty fields";
                var oldPassCorrect = "Old password is incorrect";
                var newPassMatch = "The new passwords do not match";


                if (result["status"] == updated)
                {
                    bootbox.alert("Your details have been updated",function()
                        {
                            window.location.replace("home.html");
                        })

                }
                else if(result["status"] == noExistance)
                {
                    bootbox.alert("The user name does not exist",function()
                        {
                            window.location.replace ("changePassword.html");
                        })
                }
                else if(result["status"] == empty)
                {
                    bootbox.alert("Please fill in all the fields",function()
                        {
                            window.location.replace ("changePassword.html");
                        })
                }
                else if(result["status"] == oldPassCorrect)
                {
                    bootbox.alert("The current password is incorrect",function()
                        {
                            window.location.replace ("changePassword.html");
                        })
                }
                else if(result["status"] == newPassMatch)
                {
                    bootbox.alert("The new passwords do not match",function()
                        {
                            window.location.replace ("changePassword.html");
                        })
                }
        })

};
```

```
function back()
{
    window.location.replace ("index.html");
};
```

currentLocation.js

```
function animal_location()
{
    event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/currentLocation",
            type: 'POST',
            data: $('#select_form').serialize(),
            async: true})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if (result["status"] == "Not logged In")
                    {
                        bootbox.alert("Session timeout. Please Login.",function()
                            {
                                window.location.replace("index.html");
                            })
                    }
                else
                    {
                        process_data(result)
                        window.location.replace ("currentLocation.html");
                    }

        })
};
function back()
{
    window.location.replace ("home.html");
};
function process_data(location_data)
{
    window.localStorage.setItem("coordinates", JSON.stringify(location_data));
}
```

delete_animal.js

```
function process_animal_delete()
{
    var animal_data = JSON.parse(window.localStorage.getItem("delete_animals"));

    var animal_array = Object.values(animal_data);
    var parsed_array =[];
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(parsed_array.indexOf(temp_array[0] == -1 |
parsed_array.length ==0))
                        {
                            parsed_array.push(temp_array[0]);
                        }

                }
        }
    //populate drop down list with animal names
    var select_tag = document.getElementById("animal");
    var option;
    option = document.createElement("option");
    option.text = "--Select an Animal--";
    select_tag.add(option);

    for(var i =0; i < parsed_array.length;i++)
        {
            option = document.createElement("option");
            option.text = parsed_array[i];
            select_tag.add(option);
        }

}
function populate_fields()
{
    //populate hidden input fields with the selected animals tracking id
    var animal_data = JSON.parse(window.localStorage.getItem("delete_animals"));
    var animal_array = Object.values(animal_data);
    var choice = document.getElementById("animal").value;

    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(temp_array[0] == choice)
                        {
                            $("#trackingNum").val(temp_array[5]);
                        }
                }
        }
    window.localStorage.clear();
}
function back()
{
    window.location.replace("home.html");
}
```

get_cluster_details.js

```
function get_cluster_data()
{
    event.preventDefault();

    $.ajax({
            url: "http://gProject.pythonanywhere.com/get_cluster_data",
            data: $('#graph_form').serialize(),
            type: 'POST',
            async: false})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if(result['status'])
                    {

                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }

                        if (result["status"] == "Empty Fields")
                            {
                                bootbox.alert("Please select an animal",function()
                                    {
                                        window.location.replace("cluster.html");
                                    })
                            }
                        if (result['status'] == 'Date order')
                            {
                                bootbox.alert("The end date cannot be before the
start date",function()
                                    {
                                        window.location.replace("cluster.html");
                                    })
                            }
                        if (result['status'] == 'Same Dates')
                            {
                                bootbox.alert("The dates cannot be the
same",function()
                                    {
                                        window.location.replace("cluster.html");
                                    })
                            }

                    }
                else
                {
                    window.localStorage.setItem('graph_Data',JSON.stringify(result)
);
                    window.location.replace("movement_graph.html");
                }

        })
}
```

```
function back()
{
    event.preventDefault();
    window.location.replace("home.html");
}
```

get_delete_details.js

```
function delete_details()
{
    event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/update_animals",
            type: 'GET',
            async: false})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if(result['status'])
                    {
                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if(result["status"] == "No animals")
                            {
                                bootbox.alert("No animals associated with your
account",function()
                                    {
                                        window.location.replace("addAnimalProfile.h
tml");
                                    })
                            }
                    }
                else
                    {
                        window.localStorage.setItem("delete_animals",
JSON.stringify(result));
                        window.location.replace("delete_animal.html");
                    }
        })
}
```

logout.js

```
function logout_user()
{
        $.ajax({
            url: "http://gProject.pythonanywhere.com/logout",
            type: 'GET',
            async: false})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if (result["status"] == "You have logged out successfully")
                {
                    bootbox.alert("You have logged out successfully",function()
                        {
                            window.location.replace("index.html");
                        })

                }
                else if(result["status"] == "Logout failed")
                {
                    bootbox.alert("Logout failed please try again",function()
                        {
                            window.location.replace ("home.html");
                        })
                }
        })
};
```

populate_animal.js

```
function populate()
{
    var animal_data = JSON.parse(window.localStorage.getItem("boundary_animals"));
    var animal_array = Object.values(animal_data);
    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(parsed_array.indexOf(temp_array[0] == -1 |
parsed_array.length ==0))
                        {
                            parsed_array.push(temp_array[0]);
                        }

                }
        }

    var select_tag = document.getElementById("animal");
    var option = document.createElement("option");
    option.text = "--Select an Animal--";
    select_tag.add(option);

    for(var i =0; i < parsed_array.length;i++)
        {
            option = document.createElement("option");
            option.text = parsed_array[i];
            select_tag.add(option);
        }

}
function populate_fields()
{
    var animal_data =
JSON.parse(window.localStorage.getItem("boundary_animals"));
    var animal_array = Object.values(animal_data);
    var choice = document.getElementById("animal").value;

    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(temp_array[0] == choice)
                        {

                            $("#trackingNum").val(temp_array[5]);
                            console.log(temp_array[5]);

                        }
                }
        }

}
function sendSMS()
{
    var animal = document.getElementById("animal").value;
```

```
    if(animal == "--Select an Animal--")
        {
            bootbox.alert("Please select an animal",function()
                {
                    window.location.replace("create_boundary.html");
                })
        }
    var distance = document.getElementById("distance").value;
    var phone_number = document.getElementById("trackingNum").value;
    if(distance == "")
        {
            bootbox.alert("Please select a distance",function()
                {
                    window.location.replace("create_boundary.html");
                })
        }

    var longitude = document.getElementById("longitude").value;
    var latitude = document.getElementById("latitude").value;

    if(longitude =="" || latitude == "")
        {
            bootbox.alert("Please select a point on the map",function()
                {
                    window.location.replace("create_boundary.html");
                })
        }

    var message ="distance:";
    message = message + distance.toString() + " " + longitude.toString() + " " +
latitude.toString() + ",";

    https://github.com/cordova-sms/cordova-sms-plugin

    var success = function () { alert('Message sent successfully'); };
    var error = function (e) { alert('Message Failed:' + e); };
    SMS.sendSMS(phone_number, message, success, error);
    window.localStorage.clear();
    window.location.replace("home.html");
}

function back()
{
    window.location.replace("home.html");
}
```

populate_details.js

```
function populate_animals()
{
    event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/get_animal_data",
            type: 'GET',
            async: false})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if(result['status'])
                    {
                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if(result["status"] == "No animals")
                            {
                                bootbox.alert("No animals associated with your
account",function()
                                    {
                                        window.location.replace("addAnimalProfile.h
tml");
                                    })
                            }
                    }
                else
                    {
                        window.localStorage.setItem("animals",
JSON.stringify(result));
                        window.location.replace("myprofile.html");
                    }
        })
}
```

process_animal_update.js

```
function process_animal()
{
    var animal_data =
JSON.parse(window.localStorage.getItem("associated_animals_update"));
    var animal_array = Object.values(animal_data);
    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(parsed_array.indexOf(temp_array[0] == -1 |
parsed_array.length ==0))
                        {
                            parsed_array.push(temp_array[0]);
                        }

                }
        }

    var select_tag = document.getElementById("animal");
    var option;
    option = document.createElement("option");
    option.text = "--Select an Animal--";
    select_tag.add(option);

    for(var i =0; i < parsed_array.length;i++)
        {
            option = document.createElement("option");
            option.text = parsed_array[i];
            select_tag.add(option);
        }

}
function populate_fields()
{
    var animal_data =
JSON.parse(window.localStorage.getItem("associated_animals_update"));
    var animal_array = Object.values(animal_data);
    var choice = document.getElementById("animal").value;

    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(temp_array[0] == choice)
                        {
                            $("#animalID").val(temp_array[0]);
                            $("#animalType").val(temp_array[1]);
                            $("#animalBreed").val(temp_array[2]);
                            $("#animalWeight").val(temp_array[3]);
                            $("#animalGender").val(temp_array[4]);
                            $("#trackingNum").val(temp_array[5]);
                            $("#oldTrackingId").val(temp_array[5]);
                            $("#oldanimalIdentifier").val(temp_array[0]);
                        }
```

```
                }
            }

}
function back()
{
    window.location.replace("home.html");
}
```

process_animals.js

```
function process_animal()
{
    var animal_data = JSON.parse(window.localStorage.getItem("animals"));

    var animal_array = Object.values(animal_data);
    var parsed_array =[];
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 8)
                {
                    if(parsed_array.indexOf(temp_array[1] == -1 |
parsed_array.length ==0))
                        {
                            parsed_array.push(temp_array[1]);
                        }

                }
        }
    var select_tag = document.getElementById("animal");
    var option;
    option = document.createElement("option");
    option.text = "--Select an Animal--";
    select_tag.add(option);

    for(var i =0; i < parsed_array.length;i++)
        {
            option = document.createElement("option");
            option.text = parsed_array[i];
            select_tag.add(option);
        }

}
function populate_select()
{
    var animal_data = JSON.parse(window.localStorage.getItem("animals"));
    var animal_array = Object.values(animal_data);
    var choice = document.getElementById("animal").value;

    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);

            if(temp_array.length == 8)
                {
                    if(temp_array[1] == choice)
                        {
                            $("#trackingNum").val(temp_array[7]);
                        }
                }
        }
}
function back()
{
    window.location.replace("home.html");
}
```

process_delete.js

```
function remove_animal()
{
    event.preventDefault();

    $.ajax({
            url: "http://gProject.pythonanywhere.com/delete_details",
            data: $('#delete_form').serialize(),
            type: 'POST',
            async: false})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if(result['status'])
                    {

                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if (result["status"] == "Animal profile deleted
successfully")
                            {
                                bootbox.alert("Animal profile deleted
successfully",function()
                                    {
                                        window.location.replace("home.html");
                                    })
                            }
                        if (result["status"] == "Empty fields")
                            {
                                bootbox.alert("Please select an animal",function()
                                    {
                                        window.location.replace("delete_animal.html
");
                                    })
                            }

                    }
        })
}
```

process_select.js

```
function process_animal()
{
    var animal_data = JSON.parse(window.localStorage.getItem("select_animals"));

    var animal_array = Object.values(animal_data);
    var parsed_array =[];
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(parsed_array.indexOf(temp_array[0] == -1 |
parsed_array.length ==0))
                        {
                            parsed_array.push(temp_array[0]);
                        }

                }
        }

    var select_tag = document.getElementById("animal");
    var option;
    option = document.createElement("option");
    option.text = "--Select an Animal--";
    select_tag.add(option);

    for(var i =0; i < parsed_array.length;i++)
        {
            option = document.createElement("option");
            option.text = parsed_array[i];
            select_tag.add(option);
        }


    var select_tag = document.getElementById("animalTwo");
    var option;
    option = document.createElement("option");
    option.text = "--Select an Animal--";
    select_tag.add(option);

    for(var i =0; i < parsed_array.length;i++)
        {
            option = document.createElement("option");
            option.text = parsed_array[i];
            select_tag.add(option);
        }

}
function populate_select()
{
    var animal_data = JSON.parse(window.localStorage.getItem("select_animals"));
    var animal_array = Object.values(animal_data);
    var choice = document.getElementById("animal").value;

    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
```

```
                {
                    if(temp_array[0] == choice)
                        {
                            $("#trackingNum").val(temp_array[5]);
                        }

                }
        }


    var choice = document.getElementById("animalTwo").value;

    var parsed_array =[]
    for(var i =0; i < animal_array.length;i++)
        {
            var temp_array = Object.values(animal_array[i]);
            if(temp_array.length == 6)
                {
                    if(temp_array[0] == choice)
                        {
                            $("#trackingNumTwo").val(temp_array[5]);
                        }

                }
        }


}
function back()
{
    window.location.replace("home.html");
}
```

registerTwo.js

```
function register()
{
    event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/register",
            data: $('#registerForm').serialize(),
            type: 'POST',
            async: false})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if (result["status"] == "ok")
                {
                    bootbox.alert("Registration successful",function()
                        {
                            window.location.replace("home.html");
                        })
                }
                else if(result["status"] == "Passwords incorrect")
                    {
                        bootbox.alert("The passwords do not match please try
again",function()
                            {
                                window.location.replace ("register.html");
                            })
                    }
                else if(result["status"] == "Username already exists")
                {
                    bootbox.alert("The user name already exists please try
again",function()
                            {
                                window.location.replace ("register.html");
                            })
                }
                else if(result["status"] == "Empty fields")
                {
                    bootbox.alert("Please fill in all the fields",function()
                        {
                            window.location.replace ("register.html");
                        })
            }
            else if(result["status"] == "You are already logged in as another
user")
                {
                    bootbox.alert("You are already logged in as another
user",function()
                            {
                                window.location.replace ("index.html");
                            })
            }
        })

};
function back()
{
    window.location.replace ("index.html");
};
```

select_animal_boundary.js

```
function get_animal()
{
    event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/update_animals",
            type: 'GET',
            async: false})
            .done(function(response) {
                var result = JSON.parse(response);
                if(result['status'])
                    {
                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if(result["status"] == "No animals")
                            {
                                bootbox.alert("No animals associated with your
account",function()
                                    {
                                        window.location.replace("addAnimalProfile.h
tml");
                                    })
                            }
                    }
                else
                    {
                        window.localStorage.setItem("boundary_animals",
JSON.stringify(result));
                        window.location.replace("create_boundary.html");
                    }
        })
}
```

select_animals.js

```
function get_animal_data()
{
    event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/update_animals",
            type: 'GET',
            async: false})
            .done(function(response) {
                var result = JSON.parse(response);
                if(result['status'])
                    {
                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if(result["status"] == "No animals")
                            {
                                bootbox.alert("No animals associated with your
account",function()
                                    {
                                        window.location.replace("addAnimalProfile.h
tml");
                                    })
                            }
                    }
                else
                    {
                        window.localStorage.setItem("select_animals",
JSON.stringify(result));
                        window.location.replace("select_animal.html");
                    }
        })
}
```

storage.js

```
function clear_storage()
{
    window.localStorage.clear();
}
```

update_animal.js

```
function get_animals()
{
    event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/update_animals",
            type: 'GET',
            async: false})
            .done(function(response) {
                var result = JSON.parse(response);
                if(result['status'])
                    {
                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if(result["status"] == "No animals")
                            {
                                bootbox.alert("You need to add an animal to your
account to configure the gps",function()
                                    {
                                        window.location.replace("addAnimalProfile.h
tml");
                                    })
                            }
                    }
                else
                    {
                        window.localStorage.setItem("associated_animals_update",
JSON.stringify(result));
                        window.location.replace("update_animal_profile.html");
                    }

        })
}
```

update_gps.js

```
function update_gps()
{
        event.preventDefault();

        $.ajax({
            url: "http://gProject.pythonanywhere.com/update_gps",
            type: 'GET',
            async: false})
            .done(function(response) {
                console.log(response);
                var result = JSON.parse(response);
                if(result['status'])
                    {
                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if(result["status"] == "No animals")
                            {
                                bootbox.alert("You need to add an animal to your
account to configure the gps",function()
                                    {
                                        window.location.replace("addAnimalProfile.h
tml");
                                    })
                            }
                    }
                else
                    {
                        window.localStorage.setItem("associated_animals",
JSON.stringify(result));
                        window.location.replace("gpsConfiguration.html");
                    }



        })
}
```

update_new_details.js

```
function update_details()
{
        $.ajax({
            url: "http://gProject.pythonanywhere.com/update_animals_details",
            data: $('#updateProfileForm').serialize(),
            type: 'POST',
            async: false})
            .done(function(response) {
                console.log(response);

                var result = JSON.parse(response);

                if(result['status'])
                    {
                        if (result["status"] == "Your session has timed out, please
log in again")
                            {
                                bootbox.alert("Your session has timed out, please
log in again",function()
                                    {
                                        window.location.replace("index.html");
                                    })
                            }
                        if (result["status"] == "Updated Successfully")
                            {
                                bootbox.alert("Animal details updated
successfully",function()
                                    {
                                        window.location.replace("home.html");
                                    })
                            }
                        if (result["status"] == "Wrong gender")
                            {
                                bootbox.alert("Animal gender must be M or
F",function()
                                    {
                                        window.location.replace("update_animal_prof
ile.html");
                                    })
                            }
                        if (result["status"] == "Empty fields")
                            {
                                bootbox.alert("Please fill in all the
fields",function()
                                    {
                                        window.location.replace("update_animal_prof
ile.html");
                                    })
                            }
                        if (result["status"] == "Tracking number already in use")
                            {
                                bootbox.alert("Tracking number already in
use.",function()
                                    {
                                        window.location.replace("update_animal_prof
ile.html");
                                    })
                            }
```

```
                    if (result["status"] == "Animal ID associated with another
animal")
                        {
                            bootbox.alert("Animal ID associated with another
animal",function()
                                {
                                    window.location.replace("update_animal_prof
ile.html");
                                })
                        }
                }
        })
}
```

userlogin.js

```
function userlogFunction()
{
    event.preventDefault();
        $.ajax({
            url: "http://gProject.pythonanywhere.com/userlogin",
            data: $('#loginForm').serialize(),
            type: 'POST',
            async: false})
            .done(function(response) {
                console.log(response);

                var result = JSON.parse(response);
                var loggedIn = "successful";
                var wrongPass = "Incorrect Password";
                var empty = "Empty fields";
                var wrongUserName = "Incorrect user name";
                var newPassMatch = "The new passwords do not match";


                if (result["status"] == loggedIn)
                {
                    bootbox.alert("Login successful", function()
                        {
                            window.location.replace("home.html");
                        })

                }
                else if(result["status"] == wrongPass)
                {
                    bootbox.alert("The password is incorrect", function()
                        {
                            window.location.replace ("Login.html");
                        })

                }
                else if(result["status"] == empty)
                {

                    bootbox.alert("Please fill in all the fields", function()
                        {
                            window.location.replace ("Login.html");
                        })

                }
                else if(result["status"] == wrongUserName)
                {
                    bootbox.alert("The user name is incorrect", function()
                        {
                            window.location.replace ("Login.html");
                        })
                }
        })

};

function back()
{
    window.location.replace ("index.html");
};
```

Python Code

analyse_paths.py

```
from flask import json
from datetime import datetime
from connector import create_connection


def analyse(animal_identifier, tracking_number, second_animal_identifier,
second_tracking_number, start_date, end_date):
    if animal_identifier == '--select an animal--' or second_animal_identifier ==
'--select an animal--':
        return json.dumps({'status': 'No Animal Selected'})

    if start_date == '' or end_date == '':
        return json.dumps({'status': 'No Date Selected'})

    if start_date == end_date:
        return json.dumps({'status': 'Same Dates'})

    if start_date > end_date:
        return json.dumps({'status': 'Date order'})

    locations = get_current_location(tracking_number, second_tracking_number,
start_date,end_date)
    coordinates= []
    temp_list = []
    for items in locations:
        for elements in items:
            temp_list.append(str(elements))
        coordinates.append(temp_list)
        coordinates.append(' ')
        temp_list = []

    return json.dumps(coordinates)


def get_current_location(tracking_number, second_tracking_number,
start_date,end_date):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select
currentCoordinates.longitude,currentCoordinates.latitude,currentCoordinates.time,cu
rrentCoordinates.date,Animal.animalIdentifier from currentCoordinates INNER JOIN
Animal ON currentCoordinates.trackingID=Animal.trackingID where
currentCoordinates.trackingID = %s and currentCoordinates.trackingID = %s  and
currentCoordinates.date >= %s and currentCoordinates.date < %s")
    cursor.execute(query, (tracking_number, second_tracking_number,
str(start_date),str(end_date)))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

animalProfile.py

```
from flask import json
from connector import create_connection


def add_animal(animal_identifier, animal_type, animal_breed, animal_weight,
animal_gender, tracking_number, owner):
    if animal_identifier == '' or animal_type == '' or animal_breed == '' or
animal_weight == '' or animal_gender == '' or tracking_number == '':
            Result = json.dumps({"status": "Empty fields"})
            return Result
    else:
        if animal_gender != 'M' and animal_gender != 'F' and animal_gender != 'm'
and animal_gender != 'f':
            overall_result = json.dumps({"status": "Wrong gender"})
            return overall_result

        exists = check_animal_existance(animal_identifier)
        if len(exists) <= 1:
            ids = get_current_animal_id()
            update_animal_profile(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, owner, tracking_number)
            overall_result = json.dumps({"status": "ok"})
            return overall_result
        else:
            overall_result = json.dumps({"status": "Animal id already exists"})
            return overall_result


def get_current_animal_id():
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("SELECT MAX(id) from Animal")
    cursor.execute(query)
    the_id = cursor.fetchone()
    result = the_id[0]
    cursor.close()
    cnx2.close()
    return result


def update_animal_profile(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, owner, tracking_number):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Insert into Animal (animalIdentifier , typeAnimal, breedAnimal,
weightAnimal,genderAnimal,ownerID,trackingID) values(BINARY %s, BINARY %s, BINARY
%s, %s, BINARY %s, BINARY %s, %s)")
    cursor.execute(query, (animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, owner, tracking_number))
    cnx2.commit()
    cursor.close()
    cnx2.close()


def check_animal_existance(animal_identifier):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select * from Animal where animalIdentifier = BINARY %s")
```

```
cursor.execute(query, (animal_identifier, ))
result = cursor.fetchall()
cursor.close()
cnx2.close()
return result
```

changePassword.py

```
from flask import json
from connector import create_connection


def change_password(user, password, confirmed_password, old_password):
    exists = check_existance(user)
    if len(exists) >= 1:
        temp = exists[0]
        data = temp[2]
        if old_password == data:
            user_id = get_user_id(user)
        else:
            overall_result = json.dumps({"status": "Old password is incorrect"})
            return overall_result
        if password == confirmed_password:
            update_user(str(user), str(password), user_id)

            overall_result = json.dumps({"status": "updated"})
            return overall_result
        else:
            overall_result = json.dumps({"status": "The new passwords do not
match"})
            return overall_result
    else:
        overall_result = json.dumps({"status": "Username does not exist"})
        return overall_result


def get_user_id(user):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("SELECT id from Register where registerUserName = BINARY %s")
    cursor.execute(query, (user,))
    the_id = cursor.fetchone()
    result = the_id[0]
    cursor.close()
    cnx2.close()
    return result


def update_user(user, password, user_id):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Update Register set registerUserName = BINARY %s ,
registerUserPassword = BINARY %s where id = %s")
    cursor.execute(query, (user, password, user_id))
    cnx2.commit()
    cursor.close()
    cnx2.close()


def check_existance(user):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select * from Register where registerUserName = BINARY %s")
    cursor.execute(query, (user,))
    result = cursor.fetchall()
    cursor.close()
```

```
cnx2.close()
return result
```

cluster_data.py

```
from flask import json
from connector import create_connection
import numpy as np
from sklearn.cluster import DBSCAN
from collections import defaultdict


def cluster(distance, num_locations, date, end_date, user):
    coordinate_data = []
    cluster_data = []
    animal_ids = get_associated_animals(user)
    for animal in animal_ids:

        #http://geoffboeing.com/2014/08/clustering-to-reduce-spatial-data-set-
size/
        if distance == '' or num_locations == '' or date == '' or end_date == '':
            overall_result = {"status": "Empty Fields"}
            return overall_result

        if animal[0] != '':
            data = get_coordinates(animal[0], date,end_date)

        if date == end_date:
            return {'status': 'Same Dates'}

        if date > end_date:
            return {'status': 'Date order'}

        if len(data) <= 1:
            overall_result = {"status": "No Data"}
            return overall_result

        kms_per_radian = 6371.0088
        value = float(distance)/1000
        eps = value/kms_per_radian
        coordinates = []
        temp_data = []
        for elements in data:
            for items in elements:
                temp_data.append(float(items))
            coordinates.append(temp_data)
            temp_data = []

        coords = np.matrix(coordinates)
        #min_samples =1 means every point get assigned to a cluster or becomes a
cluster itself
        #ball_tree algorithm hyper spheres multidimensional each point belongs to
one sphere only
        #ball_tree partition data into nested spheres, more costly than kd tree
more efficient multidimensional
        db = DBSCAN(eps, min_samples=int(num_locations), algorithm='ball_tree',
metric='haversine').fit(np.radians(coords))
        #kmeans requires you to specify the number of clusters in advance which
dbscan doesnt it uses eps and min_samples

        cluster_labels = db.labels_
```

```
        #core samples:  A point is a core point if it has more than a specified
number of points (MinPts) within Eps—These are points that are at the interior of a
cluster.

        temp = []
        for i, element in enumerate(cluster_labels):
            coordinate = data[i]
            temp.append((str(element), [coordinate[0], coordinate[1]]))

        d = defaultdict(list)
        for k, v in temp:
            d[k].append(v)

        animal_info = get_animal_info(user)
        animal_data = []
        for the_animal in animal_info:
            for info in the_animal:
                animal_data.append(info)

        for k in d.keys():
            if k != '-1':
                coordinate_data.append(d[k][0])
                coordinate_data.append(animal_data)
                cluster_data.append(coordinate_data)
                cluster_data.append(' ')
                coordinate_data = []

    return cluster_data


def get_coordinates(tracking_id, date,end_date):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select longitude,latitude from currentCoordinates where trackingID =
%s and date >= %s and date < %s")
    cursor.execute(query, (tracking_id, date,end_date))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result


def get_associated_animals(user):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select trackingID from Animal where ownerID = BINARY %s")
    cursor.execute(query, (user,))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result

def get_animal_info(user):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select animalIdentifier, typeAnimal , breedAnimal , weightAnimal ,
genderAnimal , trackingID  from Animal where ownerID = BINARY %s")
    cursor.execute(query, (user,))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

connector.py

```
import mysql.connector


def create_connection():
    cnx2 = mysql.connector.connect(host='gProject.mysql.pythonanywhere-
services.com',
                                   user='gProject', password= '_mf698t_',
                                   database='gProject$Lamina')
    return cnx2
```

currentLocation.py

```
from connector import create_connection


def location(user):
    locations = get_current_location(user)#get all locations associated with the
current user
    track_id = set()
    animals= []
    coordinates= []
    for items in locations:
        track_id.add(items[1])

    for ids in track_id:
        animals.append(get_associated_animals(ids))

    for animal in animals:
        for elements in animal:
            max_id = get_max_id(elements[7])
            currentlocation = get_latest_location(max_id)
            for items in currentlocation:
                for elements in items:
                    coordinates.append(str(elements))
            coordinates.append(' ')

    return coordinates


def get_max_id(tracking_id):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("SELECT MAX(id) from currentCoordinates where trackingID = %s")
    cursor.execute(query, (tracking_id, ))
    the_id = cursor.fetchone()
    result = the_id[0]
    cursor.close()
    cnx2.close()
    return result


def get_latest_location(max_id):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("SELECT
currentCoordinates.longitude,currentCoordinates.latitude,currentCoordinates.time,cu
rrentCoordinates.date, Animal.animalIdentifier from currentCoordinates INNER JOIN
Animal ON currentCoordinates.trackingID=Animal.trackingID where
currentCoordinates.id = %s")
    cursor.execute(query, (max_id, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result


def get_current_location(user_id):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select * from currentCoordinates where username = BINARY %s")
```

```
    cursor.execute(query, (user_id, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result


def get_associated_animals(track_id):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select * from Animal where trackingID = %s")
    cursor.execute(query, (track_id, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

delete_animal.py

```
from flask import json
from connector import create_connection


def remove_profiles(animal_identifier, tracking_number):
    if animal_identifier == '' or tracking_number == '':
        overall_result = json.dumps({"status": "Empty fields"})
        return overall_result
    else:
        delete_animal(tracking_number)
        delete_animal_coordinates(tracking_number)
        overall_result = json.dumps({"status": "Animal profile deleted
successfully"})
        return overall_result


def delete_animal(tracking_number):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Delete from Animal where trackingID = %s")
    cursor.execute(query, (tracking_number, ))
    cnx2.commit()
    cursor.close()
    cnx2.close()


def delete_animal_coordinates(tracking_number):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Delete from currentCoordinates where trackingID = %s")
    cursor.execute(query, (tracking_number, ))
    cnx2.commit()
    cursor.close()
    cnx2.close()
```

graph_movement.py

```
from connector import create_connection


def get_graph_details(owner):
    result = graph_data(owner)
    return result


def graph_data(owner):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select * from Animal where ownerID = BINARY %s")
    cursor.execute(query, (owner, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

insert_coordinates.py

```
from connector import create_connection


def insert_coord(data):
    values = data.split(' ')
    tracking_id = values[0]
    names = get_names(tracking_id)
    insert_data(tracking_id, values[3], values[4], values[2], values[1],
names[0][0])


def get_names(tracking_id):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select ownerID from Animal where trackingID = %s")
    cursor.execute(query, (tracking_id, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result


def insert_data(tracking_id, longitude, latitude, time, date, owner):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Insert into currentCoordinates
(trackingID,longitude,latitude,time,date, username) values(%s, %s, %s,%s, %s, %s)")
    cursor.execute(query, (tracking_id, longitude, latitude, time, date,
str(owner)))
    cnx2.commit()
    cursor.close()
    cnx2.close()
```

lamina.py

```
from flask import Flask, request, json, session
from currentLocation import location
from register import register
from changePassword import change_password
from login import login
from animalProfile import add_animal
from analyse_paths import analyse
from update_gps_config import associated_animals
from update_animal_profile import get_animal_profiles
from update_details import update_animal_details
from delete_animal import remove_profiles
from graph_movement import get_graph_details
from cluster_data import cluster
from insert_coordinates import insert_coord
from set_login_flag import check_flag, is_logged_in, is_logged_out
import random
import hashlib


global session_user
app = Flask(__name__)
app.secret_key = str(random.getrandbits(256))


@app.route('/register', methods=['POST', 'GET'])
def register_user():
    global session_user

    temp_user = request.form['registerUserName']
    if temp_user == '' or request.form['registerUserPassword'] == '' or
request.form['confirmRegisterUserPassword'] == '':
        Result = json.dumps({"status": "Empty fields"})
        return Result
    temp_password =
hashlib.sha256(request.form['registerUserPassword'].encode('utf-8')).hexdigest()
    temp_confirmed_password =
hashlib.sha256(request.form['confirmRegisterUserPassword'].encode('utf-
8')).hexdigest()
    result = register(temp_user, temp_password, temp_confirmed_password)
    for k, v in json.loads(result).items():
        if v == 'ok':
            session_user = temp_user
            is_logged_in(temp_user)
            return result
        else:
            is_logged_out(temp_user)
            return result


@app.route('/changePassword', methods=['POST', 'GET'])
def change_pass():
    global session_user
    user = request.form['changeUserName']
    if user == '' or request.form['oldUserPassword'] == '' or
request.form['changeUserPassword'] == '' or
request.form['confirmChangeUserPassword'] == '':
        Result = json.dumps({"status": "Empty fields"})
        return Result
```

```
    old_password = hashlib.sha256(request.form['oldUserPassword'].encode('utf-
8')).hexdigest()
    password = hashlib.sha256(request.form['changeUserPassword'].encode('utf-
8')).hexdigest()
    confirmed_password =
hashlib.sha256(request.form['confirmChangeUserPassword'].encode('utf-
8')).hexdigest()

    result = change_password(user, password, confirmed_password, old_password)
    for k, v in json.loads(result).items():
        if v == 'updated':
            session_user = user
            is_logged_in(user)
            return result
        else:
            is_logged_out(user)
            return result


@app.route('/userlogin', methods=['POST', 'GET'])
def user_login():
    global session_user

    username = request.form['userName']
    password = hashlib.sha256(request.form['userPassword'].encode('utf-
8')).hexdigest()

    result = login(username, password)
    for k, v in json.loads(result).items():
        if v == 'successful':
            session_user = username
            is_logged_in(username)
            return result
        else:
            return result


@app.route('/animalProfile', methods=['POST', 'GET'])
def add_animals():
    global session_user
    animal_identifier = request.form['animalID']
    animal_type = request.form['animalType']
    animal_breed = request.form['animalBreed']
    animal_weight = request.form['animalWeight']
    animal_gender = request.form['animalGender']
    tracking_number = request.form['trackingNum']
    owner = session_user

    if check_flag(owner) == 1:
        result = add_animal(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, tracking_number, owner)
        return result
    else:
        overall_result = json.dumps({"status": "Session timed out, please log in
again"})
        return overall_result


@app.route('/currentLocation', methods=['POST', 'GET'])
def current_Location():
    global session_user
```

```
    if check_flag(session_user) == 1:
        result = location(session_user)
        return json.dumps(result)#mysql datetime objects not json serializeable
    else:
        overall_result = json.dumps({"status": "Not logged In"})
        return overall_result



@app.route('/analyse_paths', methods=['POST', 'GET'])
def analyse_path():
    global session_user
    if check_flag(session_user) == 1:
        animal_identifier = request.form['animal']
        tracking_number = request.form['trackingNum']

        second_animal_identifier = request.form['animalTwo']
        second_tracking_number = request.form['trackingNumTwo']
        start_date = request.form['date']
        end_date = request.form['enddate']
        result = analyse(animal_identifier, tracking_number,
second_animal_identifier, second_tracking_number, start_date,end_date)
        return result

    else:
        overall_result = json.dumps({"status": "Your session has timed out, please
log in again"})
        return overall_result


@app.route('/logout', methods=['POST', 'GET'])
def logout_user():
    global session_user
    if check_flag(session_user) == 1:
        is_logged_out(session_user)
        overall_result = json.dumps({"status": "You have logged out successfully"})
        return overall_result
    else:
        overall_result = json.dumps({"status": "Logout failed"})
        return overall_result


@app.route('/update_gps', methods=['POST', 'GET'])
def update_gps():

    global session_user
    if check_flag(session_user) == 1:
        result = associated_animals(session_user)
        if result != [[]]:
            return json.dumps(result)
        else:
            overall_result = json.dumps({"status": "No animals"})
            return overall_result
    else:
        overall_result = json.dumps({"status": "Your session has timed out, please
log in again"})
        return overall_result


@app.route('/update_animals', methods=['POST', 'GET'])
def update_animals():
    global session_user
```

```
    if check_flag(session_user) == 1:
        result = get_animal_profiles(session_user)
        if len(result) != 0:
            return json.dumps(result)
        else:
            overall_result = json.dumps({"status": "No animals"})
            return overall_result
    else:
        overall_result = json.dumps({"status": "Your session has timed out, please
log in again"})
        return overall_result


@app.route('/update_animals_details', methods=['POST', 'GET'])
def update_animal():
    global session_user
    animal_identifier = request.form['animalID']
    animal_type = request.form['animalType']
    animal_breed = request.form['animalBreed']
    animal_weight = request.form['animalWeight']
    animal_gender = request.form['animalGender']
    tracking_number = request.form['trackingNum']
    oldtracking_number = request.form['oldTrackingId']
    oldanimal_identifier = request.form['oldanimalIdentifier']
    if check_flag(session_user) == 1:
        result = update_animal_details(animal_identifier, animal_type,
animal_breed, animal_weight, animal_gender, tracking_number, oldtracking_number,
oldanimal_identifier, session_user)
        return result
    else:
        overall_result = json.dumps({"status": "Your session has timed out, please
log in again"})
        return overall_result


@app.route('/delete_details', methods=['POST', 'GET'])
def delete_animals():

    global session_user
    animal_identifier = request.form['animal']
    tracking_number = request.form['trackingNum']

    if check_flag(session_user) == 1:
        result = remove_profiles(animal_identifier, tracking_number)
        return result
    else:
        overall_result = json.dumps({"status": "Your session has timed out, please
log in again"})
        return overall_result


@app.route('/get_animal_data', methods=['POST', 'GET'])
def animal_data():
    global session_user
    if check_flag(session_user) == 1:
        result = get_graph_details(session_user)
        print(result)
        if len(result) != 0:
            return json.dumps(result)
        else:
            overall_result = json.dumps({"status": "No animals"})
            return overall_result
```

```
    else:
        overall_result = json.dumps({"status": "Your session has timed out, please
log in again"})
        return overall_result


@app.route('/get_cluster_data', methods=['POST', 'GET'])
def cluster_data():

    global session_user
    eps = request.form['eps']
    num_loc = request.form['num_loc']
    date = request.form['start_date']
    end_date = request.form['end_date']

    if check_flag(session_user) == 1:
        result = cluster(int(eps), int(num_loc), date, end_date, session_user)
        print(result)
        return json.dumps(result)
    else:
        overall_result = json.dumps({"status": "Your session has timed out, please
log in again"})
        return overall_result


@app.route('/insert', methods=['POST', 'GET'])
def insert_coordinates():
    payload = request.get_data()
    insert_coord(payload.decode())
    return ''


if __name__ == "__main__":
    app.run(debug=True)
```

login.py

```
from flask import json
from connector import create_connection


def login(username,password):
    if username == '' or password == '':
        result = json.dumps({"status": "Empty fields"})
        return result
    else:
        exists = check_existance(username)
        if len(exists) == 1:
            temp = exists[0]
            data = temp[2]
            if password == data:
                overall_result = json.dumps({"status": "successful"})
                return overall_result
            else:
                overall_result = json.dumps({"status": "Incorrect Password"})
                return overall_result
        else:
            overall_result = json.dumps({"status": "Incorrect user name"})
            return overall_result


def check_existance(username):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select * from Register where registerUserName = BINARY %s")
    cursor.execute(query, (username, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

register.py

```
from flask import json
from connector import create_connection


def register(username, password, confirmedPassword):
    exists = check_existance(username)
    if len(exists) < 1:
        if password != confirmedPassword:
            overall_result = json.dumps({"status": "Passwords incorrect"})
            return overall_result

        ids = get_current_id()
        add_user(str(username), str(password))
        overall_result = json.dumps({"status": "ok"})
        return overall_result
    else:
        overall_result = json.dumps({"status": "Username already exists"})
        return overall_result


def get_current_id():
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("SELECT MAX(id) from Register")
    cursor.execute(query)
    theID = cursor.fetchone()
    result = theID[0]
    cursor.close()
    cnx2.close()
    return result


def add_user(username, password):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Insert into Register (registerUserName , registerUserPassword)
values(BINARY %s, BINARY %s)")
    cursor.execute(query, (username, password))
    cnx2.commit()
    cursor.close()
    cnx2.close()


def check_existance(username):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select * from Register where registerUserName = BINARY %s")
    cursor.execute(query, (username, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

set_login_flag.py

```
from connector import create_connection


def check_flag(username):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select isLoggedIn from Register where registerUserName = BINARY %s")
    cursor.execute(query, (username, ))
    result = cursor.fetchone()
    cursor.close()
    cnx2.close()
    return result[0]


def is_logged_in(username):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Update Register set isLoggedIn = 1  where registerUserName = BINARY
%s")
    cursor.execute(query, (username, ))
    cnx2.commit()
    cursor.close()
    cnx2.close()


def is_logged_out(username):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Update Register set isLoggedIn = 0  where registerUserName = BINARY
%s")
    cursor.execute(query, (username, ))
    cnx2.commit()
    cursor.close()
    cnx2.close()
```

update_animal_profile.py

```
from connector import create_connection


def get_animal_profiles(owner):
    animal_data =[]
    animals = associated_animal_info(owner)
    for items in animals:
        animal_data.append(items)
        animal_data.append(' ')
    return animal_data


def associated_animal_info(owner:str):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select
animalIdentifier,typeAnimal,breedAnimal,weightAnimal,genderAnimal,trackingID from
Animal where ownerID = BINARY %s")
    cursor.execute(query, (owner, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

update_details.py

```
from flask import json
from connector import create_connection


def update_animal_details(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, tracking_number, oldtracking_number,
oldanimal_identifier, owner):
    if animal_identifier == '' or animal_type == '' or animal_breed == '' or
animal_weight =='' or animal_gender =='' or tracking_number ==='':
        Result = json.dumps({"status": "Empty fields"})
        return Result
    if animal_gender != 'M' and animal_gender != 'F' and animal_gender != 'm' and
animal_gender != 'f':
            overall_result = json.dumps({"status": "Wrong gender"})
            return overall_result

    if animal_identifier == oldanimal_identifier  and tracking_number ==
oldtracking_number:
        update_animal_table(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, tracking_number, oldtracking_number)
        update_coordinates_table(tracking_number, oldtracking_number)
        Result = json.dumps({"status": "Updated Successfully"})
        return Result

    if animal_identifier == oldanimal_identifier and tracking_number !=
oldtracking_number:
        associated_owners = check_id_existance(tracking_number)
        if len(associated_owners) != 0:
            for item in associated_owners:
                if item[0] == owner:
                    update_animal_table(animal_identifier, animal_type,
animal_breed, animal_weight, animal_gender, tracking_number, oldtracking_number)
                    update_coordinates_table(tracking_number, oldtracking_number)
                    Result = json.dumps({"status": "Updated Successfully"})
                    return Result
                else:
                    Result = json.dumps({"status": "Tracking number already in
use"})
                    return Result
        else:
            update_animal_table(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, tracking_number, oldtracking_number)
            update_coordinates_table(tracking_number, oldtracking_number)
            Result = json.dumps({"status": "Updated Successfully"})
            return Result

    if tracking_number == oldtracking_number and animal_identifier !=
oldanimal_identifier:
        associated_owners = check_name_existance(animal_identifier)
        if len(associated_owners) != 0:
            for ids in associated_owners:
                if ids[0] == tracking_number:
                    update_animal_table(animal_identifier, animal_type,
animal_breed, animal_weight, animal_gender, tracking_number, oldtracking_number)
                    update_coordinates_table(tracking_number, oldtracking_number)
                    Result = json.dumps({"status": "Updated Successfully"})
                    return Result
                else:
```

```
                    Result = json.dumps({"status": "Animal ID associated with
another animal"})
                    return Result
        else:
            update_animal_table(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, tracking_number, oldtracking_number)
            update_coordinates_table(tracking_number, oldtracking_number)
            Result = json.dumps({"status": "Updated Successfully"})
            return Result

    if tracking_number != oldtracking_number and animal_identifier !=
oldanimal_identifier:
        associated_animal_names = check_name_existance(animal_identifier)
        associated_owners_ids = check_id_existance(tracking_number)
        if len(associated_animal_names) != 0 and len(associated_owners_ids) != 0:
            for ids in associated_animal_names:#each trackingID
                for item in associated_owners_ids:#each name
                    if item[0] == owner and ids[0] == tracking_number:
                        update_animal_table(animal_identifier, animal_type,
animal_breed, animal_weight, animal_gender, tracking_number, oldtracking_number)
                        update_coordinates_table(tracking_number,
oldtracking_number)
                        Result = json.dumps({"status": "Updated Successfully"})
                        return Result
                    else:
                        if item[0] != owner:
                            Result = json.dumps({"status": "Tracking number already
in use"})
                            return Result
                        if ids[0] != tracking_number:
                            Result = json.dumps({"status": "Animal ID associated
with another animal"})
                            return Result

        else:
            update_animal_table(animal_identifier, animal_type, animal_breed,
animal_weight, animal_gender, tracking_number, oldtracking_number)
            update_coordinates_table(tracking_number, oldtracking_number)
            Result = json.dumps({"status": "Updated Successfully"})
            return Result


def update_animal_table(animalIdentifier, animal_type, animal_breed, animal_weight,
animal_gender, tracking_number, oldtracking_number):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Update Animal set animalIdentifier = BINARY %s,typeAnimal = BINARY
%s,breedAnimal = BINARY %s,weightAnimal = %s,genderAnimal = BINARY %s,trackingID =
%s where trackingID = %s")
    cursor.execute(query, (animalIdentifier, animal_type, animal_breed,
animal_weight, animal_gender, tracking_number, oldtracking_number))
    cnx2.commit()
    cursor.close()
    cnx2.close()


def update_coordinates_table(tracking_number, oldtracking_number):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Update currentCoordinates set trackingID = %s where trackingID = %s")
    cursor.execute(query, (tracking_number, oldtracking_number))
    cnx2.commit()
```

```
    cursor.close()
    cnx2.close()


def check_name_existance(animalIdentifier):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select trackingID from Animal where animalIdentifier = BINARY %s")
    cursor.execute(query, (animalIdentifier,))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result


def check_id_existance(tracking_number):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select ownerID from Animal where trackingID = %s")
    cursor.execute(query,(tracking_number,))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```

update_gps_config.py

```
from connector import create_connection


def associated_animals(owner):
    animal_data =[]
    animals = associated_animal_info(owner)
    for items in animals:
        animal_data.append(items)
        animal_data.append(' ')
    return animal_data


def associated_animal_info(owner:str):
    cnx2 = create_connection()
    cursor = cnx2.cursor()
    query = ("Select animalIdentifier, trackingID from Animal where ownerID =
BINARY %s")
    cursor.execute(query,(owner, ))
    result = cursor.fetchall()
    cursor.close()
    cnx2.close()
    return result
```