Project Code

Automatic Number Plate Recognition

BSc (Hons) Software Development Year 4

Student name: Michael Reid
Student ID: C00112726
Project supervisor: Mr. Nigel Whyte

Table of Contents

1 Android 3

1.1 AndroidManifest.xml 3

1.2 Camera Activity 4

1.3 Camera Preview 9

1.4 EditPasswordActivity.java 12

1.5 EditRegActivity.java 14

1.6 LoginActivity.java 17

1.7 MainMenuActivity.java 20

1.8 ProcessingSettingsActivity.java 21

1.9 SettingsActivity.java 22

1.10 SaveImageTask.java 23

1.11 ProcessImageTask.java 26

1.12 LoginTask.java 28

1.13 DatabaseConnectionTask.java 30

1.14 ChangePasswordTask.java 32

1.15 TaskCallback.java 34

1.16 TaskCallbackJSON.java 34

1.17 BadImageException.java 34

1.18 OpenCvImageProcessor.java 35

1.19 TemplateLoader.java 41

1.20 ImageTemplates.java 43

1.21 ImageTemplate.java 44

1.22 NumberPlate.java 45

1.23 Util.java 46

1.24 activity_camera.xml 47

1.25 activity_edit_ref.xml 49

1.26 activity_login.xml 51

1.27 activity_main_menu.xml 52

1.28 activity_processing_settings.xml 53

1.29 activity_settings.xml 58

1.30 strings.xml 59

2 Web Service 61

2.1 webapp.py 61

2.2 mobile_api.py 65

2.3 emailverifier.py 67

3 Web Page 68

3.1 base.html 68
3.2 Login.html 69
3.3 Logout.html 70
3.4 Main_menu.html 71
3.5 Register_staff_verify.html 72
3.6 Register_vehicle.html 73
3.7 Register_vehicle_verify.html 74
3.8 View_all_vehicles.html 75
3.9 View_all_infractions.html 76
3.10 Stylesheet.css 78

1 Android

1.1 AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="c00112726.itcarlow.ie.finalyearproject">

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

    <!-- Required Features -->
    <uses-feature android:name="android.hardware.camera" />
    <uses-feature android:name="android.hardware.camera.autofocus" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/icon"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.AppCompat.DayNight">
        <activity
            android:name=".activities.LoginActivity"
            android:screenOrientation="portrait"
            android:label="@string/app_name"
            android:noHistory="true" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".activities.MainMenuActivity"
            android:screenOrientation="portrait"/>
        <activity
            android:name=".activities.EditRegActivity"
            android:screenOrientation="sensorLandscape"/>
        <activity
            android:name=".activities.CameraActivity"
            android:screenOrientation="sensorLandscape"
            android:theme="@style/Theme.AppCompat.Light.NoActionBar.FullScreen">
            </activity>
        <activity
            android:name=".activities.EditPasswordActivity"
            android:configChanges="keyboardHidden|orientation"/>
        <activity
            android:name=".activities.SettingsActivity"
        android:screenOrientation="portrait"/>
        <activity android:name=".activities.ProcessingSettingsActivity">
        </activity>
    </application>

</manifest>
```

1.2 Camera Activity

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.hardware.Camera;
import android.hardware.Camera.PictureCallback;
import android.net.Uri;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.FrameLayout;
import android.widget.Toast;

import org.json.JSONException;
import org.json.JSONObject;
import org.opencv.android.BaseLoaderCallback;
import org.opencv.android.OpenCVLoader;

import java.io.File;

import c00112726.itcarlow.ie.finalyearproject.R;
import c00112726.itcarlow.ie.finalyearproject.misc.NumberPlate;
import c00112726.itcarlow.ie.finalyearproject.misc.Util;
import c00112726.itcarlow.ie.finalyearproject.tasks.DatabaseConnectionTask;
import c00112726.itcarlow.ie.finalyearproject.tasks.ProcessImageTask;
import c00112726.itcarlow.ie.finalyearproject.tasks.SaveImageTask;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallback;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallbackJSON;

/**
 * Author: Michael Reid
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 03/02/2016
 */
@SuppressWarnings("deprecation")
public class CameraActivity extends AppCompatActivity implements TaskCallback, TaskCallbackJSON
{

    private static final String TAG = "CameraActivity";
    protected CameraPreview mCameraPreview;
    protected Camera mCamera;
    protected FrameLayout mPreview;
    protected boolean mCanTakePicture;
    private File imageFile;

    /**
     * Callback which is called when the camera is told to take a picture.
     * From here, we spawn an AsyncTask to perform the save.
     */
    protected PictureCallback mPictureCallback = new PictureCallback() {
        @Override
        public void onPictureTaken(byte[] data, Camera camera) {
            mCamera.stopPreview();
            SaveImageTask task = new SaveImageTask(CameraActivity.this, mCameraPreview);
            task.execute(data);
        }
    };

    /**
     * Define what happens when the screen is tapped.
     * In this case, we tell the camera to take a picture
     */
    protected View.OnClickListener mPreviewOnClickListener = new View.OnClickListener() {
```

```
    @Override
    public void onClick(View v) {
        if (mCanTakePicture) {
            mCanTakePicture = false;
            mCamera.takePicture(null, null, mPictureCallback);
        }
    }
};

/**
 * A callback used to link the app with OpenCV manager.
 * In the case it is not installed, the option to install it
 * will be presented.
 */
protected BaseLoaderCallback mOpenCVCallback = new BaseLoaderCallback(this) {
   @Override
   public void onManagerConnected(int status) {
          switch (status) {
                case BaseLoaderCallback.SUCCESS:
                      Log.i(TAG, "OpenCV Loaded");
                      break;
                default:
                      super.onManagerConnected(status);
                      break;
          }
   }
};

protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_camera);

    mCamera = getCameraInstance();
    mCameraPreview = new CameraPreview(this, mCamera);

    mPreview = (FrameLayout) findViewById(R.id.camera_preview);
    mPreview.addView(mCameraPreview);
    mPreview.setOnClickListener(mPreviewOnClickListener);

    mCanTakePicture = true;

    // Attempt to load OpenCV from OpenCV Manager.
    if(!OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION_3_1_0, this, mOpenCVCallback)) {
        Log.e(TAG, "Cannot connect to OpenCV Manager");
    }
    else {
        Log.i(TAG, "OpenCV was initialised");
    }
}

@Override
protected void onPause() {
    super.onPause();



    // When the activity is paused, we have to immediately release the camera.
    // If we don't, if another activity or app tries to access it, it will be locked.
    if (mCamera != null) {
        mCamera.stopPreview();
        mCamera.release();
        mCamera = null;
    }

    // We no longer have a device camera, so we remove the preview display as well.
    if (mCameraPreview != null) {
        mPreview.removeView(mCameraPreview);
```

```
        mCameraPreview = null;
    }
}

@Override
protected void onResume() {
    super.onResume();

    // When the activity resumes, we get a camera instance.
    if (mCamera == null) {
        mCamera = getCameraInstance();
    }

    // We also need somewhere to display the camera preview.
    if (mCameraPreview == null) {
        mCameraPreview = new CameraPreview(this, mCamera);
        mPreview.addView(mCameraPreview);
    }

    mCanTakePicture = true;
    mCamera.startPreview();
}

/**
 * Safe method to open the camera
 * @return the device camera
 */
private Camera getCameraInstance() {
    Camera camera = null;
    try {
        camera = Camera.open();
    }
    catch (Exception e) {
        Log.e(TAG, e.getMessage());
    }
    return camera;
}

private void startCamera() {
    mCanTakePicture = true;
    mCamera.startPreview();
}

/**
 * Callback called by an AsyncTask to inform the Activity that it
 * has finished it's task. Allows the Activity to decide what to do next.
 * @param json The JSON result from the Web API
 */
@Override
public void onTaskComplete(JSONObject json) {
    if(json == null) {
        String message = getString(R.string.connect_failed);
        Util.showToast(this, message, Toast.LENGTH_SHORT);
        startCamera();
        return;
    }
    String KEY_REGISTRATION = "registration";
    String KEY_INFRACTION = "infraction";
    try {
        String infraction = json.getString(KEY_INFRACTION);
        String registration = json.getString(KEY_REGISTRATION);
        if(Boolean.parseBoolean(infraction)) {
            String message = getString(R.string.infraction_occured);
            message +=  "\n" + getString(R.string.registration) + registration;
            Util.showToast(this, message, Toast.LENGTH_LONG);
        }
        else {
```

```
                String message = getString(R.string.no_infraction);
                message += "\n" + getString(R.string.registration) + registration;
                Util.showToast(this, message, Toast.LENGTH_LONG);
            }
        } catch (JSONException e) {
            Log.e(TAG, e.getMessage());
            String message = getString(R.string.bad_json);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
        }
        startCamera();
    }

    /**
     * Callback called by an AsyncTask to inform the Activity that it
     * has finished it's task. Allows the Activity to decide what to do next.
     * @param numberPlate The OCR result of the ImageProcessing
     */
    @Override
    public void onTaskComplete(NumberPlate numberPlate) {
        if(numberPlate == null) {
            String message = getString(R.string.segment_failed);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            numberPlate = new NumberPlate();
        }
        showConfirmationDialog(numberPlate);
    }

    /**
     * Callback called by an AsyncTask to inform the Activity that it
     * has finished it's task. Allows the Activity to decide what to do next.
     * @param file The file where the image was saved
     */
    @Override
    public void onTaskComplete(File file) {
        if(file == null) {
            String message = getString(R.string.save_fail);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return;
        }

        imageFile = file;
        notifyDeviceOfNewFile(file);

        ProcessImageTask pit = new ProcessImageTask(this);
        pit.execute(file);
}

    /**
     * Inform device a new file has been saved to it
     * @param file the new file
     */
    private void notifyDeviceOfNewFile(File file) {
        final Intent mediaScanIntent = new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE);
        final Uri contentUri = Uri.fromFile(file);
        mediaScanIntent.setData(contentUri);
        sendBroadcast(mediaScanIntent);
    }

    /**
     * Display a confirmation dialog to validate the OCR result
     * @param numberPlate The OCR result of the ImageProcessing
     */
    private void showConfirmationDialog(final NumberPlate numberPlate) {

        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle(R.string.result_validate);
        builder.setMessage(getString(R.string.number_plate) +
```

```
                numberPlate.toString() + "\n" +
                getString(R.string.edit_continue));
        builder.setCancelable(false);
        builder.setNeutralButton(getString(R.string.cancel), new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                dialog.dismiss();
                startCamera();
            }
        });
        builder.setNegativeButton(getString(R.string.edit), new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                editClick(numberPlate);
            }
        });
        builder.setPositiveButton(getString(R.string.continue_str),
                new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                okClick(numberPlate);
            }
        });

        builder.show();
    }

    /**
     * Called when the user validates the reg. Sends OCR result to database.
     * @param numberPlate The OCR result of the ImageProcessing
     */
    private void okClick(NumberPlate numberPlate) {
        if(Util.isNetworkAvailable(this)) {
            DatabaseConnectionTask dbt = new DatabaseConnectionTask(this);
            dbt.execute(numberPlate);
        }
        else {
            String message = getString(R.string.no_network_1);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            startCamera();
        }
    }

    /**
     * Called when the user want to edit the reg. Starts new activity
     * @param numberPlate The OCR result of the ImageProcessing
     */
    private void editClick(NumberPlate numberPlate) {
        Intent intent = new Intent(this, EditRegActivity.class);
        intent.putExtra("number plate", numberPlate);
        intent.putExtra("image file", imageFile);
        startActivity(intent);
    }
}
```

1.3 Camera Preview

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.app.Activity;
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Rect;
import android.hardware.Camera;
import android.util.Log;
import android.view.Display;
import android.view.OrientationEventListener;
import android.view.Surface;
import android.view.SurfaceHolder;
import android.view.SurfaceView;

import java.io.IOException;

/**
 * Author: Michael Reid
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 03/02/2016
 */
@SuppressWarnings("deprecation")
public class CameraPreview extends SurfaceView implements SurfaceHolder.Callback {

    private static final String TAG = " CameraPreview";

    protected SurfaceHolder mSurfaceHolder;
    protected Camera mCamera;
    protected Context mContext;

    protected OrientationEventListener mOrientationListener;

    protected Rect mGuideBox;

    public CameraPreview(Context context, Camera camera) {
        super(context);

        mContext = context;
        setCamera(camera);

        mSurfaceHolder = getHolder();
        mSurfaceHolder.addCallback(this);

        // deprecated since 3.0
        mSurfaceHolder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
        mSurfaceHolder.setKeepScreenOn(true);

        mOrientationListener = new OrientationEventListener(context) {
            @Override
            public void onOrientationChanged(int orientation) {
                setCameraDisplayOrientation();
            }
        };

        mGuideBox = createGuideBox();
        setWillNotDraw(false);
    }

    @Override
    public void surfaceCreated(SurfaceHolder holder) {
        try {
            mCamera.setPreviewDisplay(mSurfaceHolder);
            mCamera.startPreview();
```

```
        } catch (Exception e) {
            Log.e(TAG, e.getMessage());
        }
        mOrientationListener.enable();
    }

    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
        mGuideBox = createGuideBox();
    }

    @Override
    public void surfaceDestroyed(SurfaceHolder holder) {
        mOrientationListener.disable();
    }

    @Override
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);

        // Draw the guide box to the screen
        Paint guideBoxPaintDetails = new Paint();
        guideBoxPaintDetails.setStyle(Paint.Style.STROKE);

        guideBoxPaintDetails.setColor(Color.BLUE);
        guideBoxPaintDetails.setStrokeWidth(15.0f);
        canvas.drawRect(mGuideBox, guideBoxPaintDetails);

        guideBoxPaintDetails.setColor(Color.YELLOW);
        guideBoxPaintDetails.setStrokeWidth(7.0f);
        canvas.drawRect(mGuideBox, guideBoxPaintDetails);
    }

    /**
     * Update the camera with the devices orientation
     */
    private void setCameraDisplayOrientation() {
        Camera.CameraInfo info = new Camera.CameraInfo();
        Display display = ((Activity)mContext).getWindowManager().getDefaultDisplay();

        int rotation = display.getRotation();
        int degrees = 0;

        // convert from device orientation to camera orientation
        switch (rotation) {
            case Surface.ROTATION_90:   degrees = 0;    break;
            case Surface.ROTATION_270:  degrees = 180;  break;
        }

        // Adjust the cameras display rotation
        int displayOrientation = (info.orientation - degrees + 360) % 360;
        mCamera.setDisplayOrientation(displayOrientation);

        // Allow camera to auto focus
        Camera.Parameters params = mCamera.getParameters();
        params.setRotation(degrees);
        if
(params.getSupportedFocusModes().contains(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE)) {
            params.setFocusMode(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE);
        }
        //params.setPictureSize(getWidth(), getHeight());
        try {
            mCamera.setParameters(params);
        }
        catch (Exception e) {
            Log.e(TAG, e.getMessage());
        }
```

```
    }

    /**
     * Create a new guide box to draw to the screen
     * @return Guide box represented as a rectangle
     */
    private Rect createGuideBox() {
        // These values are the standard dimension of an Irish license plate in millimeters.
        final float PLATE_WIDTH = 520.0f;
        final float PLATE_HEIGHT = 110.0f;
        // 4.72727272 repeating
        final float RATIO = PLATE_WIDTH / PLATE_HEIGHT;

        // The display dimensions.
        int width = getWidth();
        int height = getHeight();

        int rectW = (int) (width * 0.50f);
        int rectH = (int) (rectW / RATIO);

        return new Rect(
                (width / 2 - rectW / 2),
                (height / 2 - rectH / 2),
                (width - (width / 2 - rectW / 2)),
                (height - (height / 2 - rectH / 2))
        );
    }

    /**
     * Set the camera to display. Links the device camera
     * to the display surface
     * @param camera the device camera
     */
    private void setCamera(Camera camera) {
        if(camera == mCamera) { return; }

        if (mCamera != null) {
            mCamera.stopPreview();
            mCamera.release();
            mCamera = null;
        }

        mCamera = camera;
        try {
            mCamera.setPreviewDisplay(mSurfaceHolder);
            mCamera.startPreview();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }

    public Rect getGuideBox() {
        return mGuideBox == null ? createGuideBox() : mGuideBox;
    }
}
```

1.4 EditPasswordActivity.java

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.content.Context;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import org.json.JSONObject;

import c00112726.itcarlow.ie.finalyearproject.R;
import c00112726.itcarlow.ie.finalyearproject.misc.Util;
import c00112726.itcarlow.ie.finalyearproject.tasks.ChangePasswordTask;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallbackJSON;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 05/04/2016
 */
public class EditPasswordActivity extends AppCompatActivity implements TaskCallbackJSON {

    private static final String TAG = "EditPasswordActivity";

    protected EditText mCurrentPassword;
    protected EditText mNewPassword;
    protected EditText mConfirmNewPassword;

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

        mCurrentPassword = (EditText)findViewById(R.id.input_current_password);
        mNewPassword = (EditText)findViewById(R.id.input_new_password);
        mConfirmNewPassword = (EditText)findViewById(R.id.input_confirm_new_password);

        Button submitBtn = (Button)findViewById(R.id.btn_submit);
        submitBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if(checkUniquePassword()) {
                    updateUserPasswords();
                }
            }
        });

        Button cancelBtn = (Button)findViewById(R.id.btn_cancel);
        cancelBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                EditPasswordActivity.this.finish();
            }
        });
    }

    private boolean checkUniquePassword() {
        String currentPassword = String.valueOf(mCurrentPassword.getText()).trim();
        String newPassword = String.valueOf(mNewPassword.getText()).trim();
        String confirmPassword = String.valueOf(mConfirmNewPassword.getText()).trim();
```

```
        if(currentPassword.equals("") || newPassword.equals("") || confirmPassword.equals("")) {
            String message = getString(R.string.empty_field);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return false;
        }
        else if(currentPassword.length() < 3 || newPassword.length() < 3 ||
confirmPassword.length() < 3) {
            String message = getString(R.string.password_short);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return false;
        }
        else if(!newPassword.equals(confirmPassword)) {
            String message = getString(R.string.password_no_match);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return false;
        }
        else if(newPassword.equals(currentPassword)) {
            String message = getString(R.string.password_same);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return false;
        }
        else {
            return true;
        }
    }
    private void updateUserPasswords() {
        if(Util.isNetworkAvailable(this)) {
            SharedPreferences sharedPreferences = getSharedPreferences("login",
Context.MODE_PRIVATE);
            String username = sharedPreferences.getString("username", "admin");
            String currentPassword = String.valueOf(mCurrentPassword.getText()).trim();
            String newPassword = String.valueOf(mNewPassword.getText()).trim();

            ChangePasswordTask changePasswordTask = new ChangePasswordTask(this);
            changePasswordTask.execute(username, currentPassword, newPassword);
        }
        else {
            String message = getString(R.string.no_network_1);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
        }
    }
    @Override
    public void onTaskComplete(JSONObject json) {
        if(json == null) {
            String message = getString(R.string.connect_failed);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return;
        }
        try {
            String response = json.getString("success");
            if(Boolean.parseBoolean(response)) {
                String message = getString(R.string.password_change_success);
                Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
                finish();
            }
            else {
                String reason = (json.getString("reason")) + "\n";
                String message = getString(R.string.password_change_fail);
                Toast.makeText(this, message + reason, Toast.LENGTH_SHORT).show();
            }
        } catch (Exception e) {
            String message = getString(R.string.bad_json);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            Log.e(TAG, e.getMessage());
        }
    }
}
```

1.5 EditRegActivity.java

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.Toast;

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

import java.io.File;

import c00112726.itcarlow.ie.finalyearproject.R;
import c00112726.itcarlow.ie.finalyearproject.misc.NumberPlate;
import c00112726.itcarlow.ie.finalyearproject.misc.Util;
import c00112726.itcarlow.ie.finalyearproject.tasks.DatabaseConnectionTask;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallbackJSON;

/**
 * Author: Michael Reid
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 26/03/2016
 */
public class EditRegActivity extends AppCompatActivity implements TaskCallbackJSON {

    private static final String TAG = "EditRegActivity";

    private EditText mNumberPlateNumber;
    private EditText mEditYear;
    private EditText mEditCounty;
    private EditText mEditReg;

    private NumberPlate mNumberPlate;

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

        final File file = (File)getIntent().getSerializableExtra("image file");
        BitmapFactory.Options bmOptions = new BitmapFactory.Options();
        Bitmap image = BitmapFactory.decodeFile(file.getAbsolutePath(), bmOptions);

        ImageView imageView = (ImageView) findViewById(R.id.imageView);
        imageView.setImageBitmap(image);

        mNumberPlate = (NumberPlate)getIntent().getSerializableExtra("number plate");

        mNumberPlateNumber = (EditText)findViewById(R.id.regNumber);

        mEditYear = (EditText)findViewById(R.id.year);
        mEditCounty = (EditText)findViewById(R.id.county);
        mEditReg = (EditText)findViewById(R.id.registration);

        mEditYear.addTextChangedListener(new TextWatcher() {
            @Override
```

```
        public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
        @Override
        public void afterTextChanged(Editable s) {}
        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            updateRegDisplay();
        }
    });
    mEditCounty.addTextChangedListener(new TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence s, int start, int count, int after) {}
        @Override
        public void afterTextChanged(Editable s) {}
        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            updateRegDisplay();
        }
    });
    mEditReg.addTextChangedListener(new TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence s, int start, int count, int after) {
        }

        @Override
        public void afterTextChanged(Editable s) {
        }

        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            updateRegDisplay();
        }
    });

    mEditYear.setText(mNumberPlate.getYear());
    mEditCounty.setText(mNumberPlate.getCounty());
    mEditReg.setText(mNumberPlate.getReg());

    Button continueButton = (Button)findViewById(R.id.ok);
    continueButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            uploadToDatabase();
        }
    });
    Button cancelButton = (Button) findViewById(R.id.cancel);
    cancelButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            finish();
        }
    });
}

@Override
public void onTaskComplete(JSONObject json) {
    if(json == null) {
        String message = getString(R.string.connect_failed);
        Util.showToast(this, message, Toast.LENGTH_SHORT);
        return;
    }
    String KEY_REGISTRATION = "registration";
    String KEY_INFRACTION = "infraction";
    try {
        String infraction = json.getString(KEY_INFRACTION);
        String registration = json.getString(KEY_REGISTRATION);
        if(Boolean.parseBoolean(infraction)) {
            String message = getString(R.string.infraction_occured);
```

```
                message += "\nRegistration: " + registration;
                Util.showToast(this, message, Toast.LENGTH_LONG);
            }
            else {
                String message = getString(R.string.no_infraction);
                message += "\nRegistration: " + registration;
                Util.showToast(this, message, Toast.LENGTH_LONG);
            }
        } catch (JSONException e) {
            Log.e(TAG, e.getMessage());
            String message = getString(R.string.bad_json);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
        }
        finish();
    }

    private void updateRegDisplay() {
        String year = String.valueOf(mEditYear.getText());
        String county = String.valueOf(mEditCounty.getText());
        String reg = String.valueOf(mEditReg.getText());
        String newText = year + "-" + county + "-" + reg;
        mNumberPlateNumber.setText(newText);
    }

    private void uploadToDatabase() {
        String year = String.valueOf(mEditYear.getText());
        String county = String.valueOf(mEditCounty.getText());
        String reg = String.valueOf(mEditReg.getText());

        mNumberPlate.setYear(year);
        mNumberPlate.setCounty(county);
        mNumberPlate.setReg(reg);

        if(Util.isNetworkAvailable(EditRegActivity.this)) {
            DatabaseConnectionTask dbt = new DatabaseConnectionTask(this);
            dbt.execute(mNumberPlate);
        }
        else {
            String message = getString(R.string.no_network_2);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
        }
    }
}
```

1.6 LoginActivity.java

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.app.AppCompatDelegate;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

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

import c00112726.itcarlow.ie.finalyearproject.R;
import c00112726.itcarlow.ie.finalyearproject.misc.Util;
import c00112726.itcarlow.ie.finalyearproject.tasks.LoginTask;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallbackJSON;



public class LoginActivity extends AppCompatActivity implements TaskCallbackJSON {

    static {
        AppCompatDelegate.setDefaultNightMode(
                AppCompatDelegate.MODE_NIGHT_AUTO);
    }

    private static final String TAG = "LoginActivity";

    private static final int MIN_USERNAME_LENGTH = 3;
    private static final int MIN_PASSWORD_LENGTH = 3;
    private static final int MAX_USERNAME_LENGTH = 10;
    private static final int MAX_PASSWORD_LENGTH = 10;

    private static final String KEY_SUCCESS = "success";

    private EditText mUsernameText;
    private EditText mPasswordText;

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

        mUsernameText =(EditText)findViewById(R.id.input_email);
        mPasswordText =(EditText)findViewById(R.id.input_password);

        Button loginButton = (Button) findViewById(R.id.btn_login);
        loginButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String username = String.valueOf(mUsernameText.getText()).trim();
                String password = String.valueOf(mPasswordText.getText()).trim();
                attemptLogin(username, password);
            }
        });
    }

    /**
     * Check username and password supplied match expected lengths
     * @param username username
     * @param password password
```

```
     * @return whether username and password lengths are valid
     */
    protected boolean credentialValid(String username, String password) {
        final int uLength = username.length();
        final int pLength = password.length();
        return (uLength >= MIN_USERNAME_LENGTH && uLength <= MAX_USERNAME_LENGTH &&
                pLength >= MIN_PASSWORD_LENGTH && pLength <= MAX_PASSWORD_LENGTH);
    }

    /**
     * Attempt to log in using supplied username and password.
     * First a check is performed to ensure credentials are in correct format.
     * Second a check is performed to ensure there is a network connection.
     * Finally if everything is ok, start task.
     * @param username username
     * @param password password
     */
    protected void attemptLogin(String username, String password) {
        if(!credentialValid(username, password)) {
            String message = getString(R.string.request_credentials_1);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return;
        }

        if(!Util.isNetworkAvailable(this)) {
            String message = getString(R.string.no_network_1);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return;
        }

        LoginTask lt = new LoginTask(this);
        lt.execute(username, password);
    }

    /**
     * Called when LoginTask thread completes
     */
    @Override
    public void onTaskComplete(JSONObject json) {
        if(json == null) {
            String message = getString(R.string.connect_failed);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
            return;
        }

        try {
            String response = json.getString(KEY_SUCCESS);
            if(Boolean.parseBoolean(response)) {
                String message = getString(R.string.login_success);
                Util.showToast(this, message, Toast.LENGTH_SHORT);
                onLoginSuccess();
            }
            else {
                String message = getString(R.string.login_fail);
                Util.showToast(this, message, Toast.LENGTH_SHORT);
            }
        } catch (JSONException e) {
            Log.e(TAG, e.getMessage());
            String message = getString(R.string.bad_json);
            Util.showToast(this, message, Toast.LENGTH_SHORT);
        }
    }

    private void onLoginSuccess() {
        SharedPreferences sharedPreferences = getSharedPreferences("login",
Context.MODE_PRIVATE);
        SharedPreferences.Editor editor = sharedPreferences.edit();
```

```
        String username = String.valueOf(mUsernameText.getText()).trim();
        editor.putString("username", username );
        editor.apply(); // .comit();

        Intent intent = new Intent(this, MainMenuActivity.class);
        startActivity(intent);
    }
}
```

1.7 MainMenuActivity.java

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.View;
import android.widget.Button;

import c00112726.itcarlow.ie.finalyearproject.R;

public class MainMenuActivity extends AppCompatActivity {

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

        Button start = (Button)findViewById(R.id.btn_ocr);
        start.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainMenuActivity.this, CameraActivity.class);
                startActivity(intent);
            }
        });

        Button settings = (Button)findViewById(R.id.btn_settings);
        settings.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainMenuActivity.this, SettingsActivity.class);
                startActivity(intent);
            }
        });

        Button logout = (Button)findViewById(R.id.btn_logout);
        logout.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // TODO Auto-generated method stub
        super.onCreateOptionsMenu(menu);
        MenuInflater mymenu = getMenuInflater();
        mymenu.inflate(R.menu.menu_main, menu);
        return true;
    }
}
```

1.8 ProcessingSettingsActivity.java

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.widget.Toast;

import c00112726.itcarlow.ie.finalyearproject.R;
import c00112726.itcarlow.ie.finalyearproject.misc.Util;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 08/04/2016
 */
public class ProcessingSettingsActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_processing_settings);
        setTitle("Pre-processing Settings");
    }

    @Override
    protected void onResume() {
        super.onResume();

        String message = "Feature not fully supported";
        Util.showToast(this, message, Toast.LENGTH_LONG);
    }
}
```

1.9 SettingsActivity.java

```
package c00112726.itcarlow.ie.finalyearproject.activities;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import c00112726.itcarlow.ie.finalyearproject.R;
import c00112726.itcarlow.ie.finalyearproject.misc.Util;

public class SettingsActivity extends AppCompatActivity {

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

        Button advanceSettings = (Button)findViewById(R.id.btn_advanced_settings);
        advanceSettings.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //String message = getString(R.string.not_implemented);
                //Util.showToast(SettingsActivity.this, message, Toast.LENGTH_SHORT);
                Intent intent = new Intent(SettingsActivity.this,
ProcessingSettingsActivity.class);
                startActivity(intent);
            }
        });

        Button changePassword = (Button)findViewById(R.id.btn_change_password);
        changePassword.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(SettingsActivity.this, EditPasswordActivity.class);
                startActivity(intent);
            }
        });
    }
}
```

1.10 SaveImageTask.java

```
package c00112726.itcarlow.ie.finalyearproject.tasks;

import android.app.ProgressDialog;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Matrix;
import android.graphics.Rect;
import android.media.ExifInterface;
import android.os.AsyncTask;
import android.os.Environment;
import android.util.Log;
import android.view.Surface;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

import c00112726.itcarlow.ie.finalyearproject.activities.CameraPreview;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallback;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: C00112726@itcarlow.ie
 * Date: 05/02/2016
 */
public class SaveImageTask extends AsyncTask<byte[], String, File> {

    private static final String IMAGE_DATA_PATH =
            Environment.getExternalStorageDirectory().toString() + "/MyAppFolder/AppImages/";
    private static final String TAG = "SaveImageTask";

    private TaskCallback mTaskCallback;
    private CameraPreview mCameraPreview;

    private ProgressDialog mProgressDialog;

    public SaveImageTask(TaskCallback taskCallback, CameraPreview cameraPreview) {
        mTaskCallback = taskCallback;
        mCameraPreview = cameraPreview;
    }

    @Override
    protected void onPreExecute() {
        mProgressDialog = new ProgressDialog((Context) mTaskCallback);
        mProgressDialog.setMessage("Saving Image...");
        mProgressDialog.setCanceledOnTouchOutside(false);
        mProgressDialog.setCancelable(false);
        mProgressDialog.show();
    }

    @Override
    protected File doInBackground(byte[]... data) {

        File imageFile = createOutputPictureFile();
        if(imageFile == null) {
            Log.e(TAG, "No file created");
            return null;
        }
        try {
            Bitmap image = BitmapFactory.decodeByteArray(data[0], 0, data[0].length);
            Bitmap croppedImage = cropImage(image);
```

```
            FileOutputStream out = new FileOutputStream(imageFile);
            croppedImage.compress(Bitmap.CompressFormat.JPEG, 100, out);
            croppedImage = correctRotation(croppedImage, imageFile.getAbsolutePath());
            croppedImage.compress(Bitmap.CompressFormat.JPEG, 100, out);

            out.flush();
            out.close();
            return imageFile;
        } catch (IOException e) {
            Log.e(TAG, e.getMessage());
            return null;
        }
    }

    @Override
    public void onPostExecute(File file) {
        if(mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
        }

        if(mTaskCallback != null) {
            mTaskCallback.onTaskComplete(file);
        }
    }

    private File createOutputPictureFile() {
        File imageStorageDirectory = new File(IMAGE_DATA_PATH);

        // If the default save directory doesn't exist, try and create it
        if (!imageStorageDirectory.exists()){
            if (!imageStorageDirectory.mkdirs()){
                Log.e(TAG, "Required media storage does not exist");
                return null;
            }
        }

        // Create a timestamp and use it as part of the file name
        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyyMMddHHmmss", Locale.UK);
        String timeStamp = dateFormat.format(new Date());
        String fileName = "img_"+ timeStamp + ".jpg";

        return new File (imageStorageDirectory, fileName);
    }

    private Bitmap cropImage(Bitmap image) {

        Rect guideBox           = mCameraPreview.getGuideBox();

        int cameraPreviewWidth  = mCameraPreview.getWidth();
        int cameraPreviewHeight = mCameraPreview.getHeight();

        int imageWidth          = image.getWidth();
        int imageHeight         = image.getHeight();

        float widthRatio        = (float)imageWidth / (float)cameraPreviewWidth;
        float heightRatio       = (float)imageHeight / (float)cameraPreviewHeight;

        int cropX               = (int)(guideBox.left * widthRatio);
        int cropY               = (int)(guideBox.top * heightRatio);
        int cropWidth           = (int)(guideBox.width() * widthRatio);
        int cropHeight          = (int)(guideBox.height() * heightRatio);

        return Bitmap.createBitmap(image, cropX, cropY, cropWidth, cropHeight);
    }

    private Bitmap correctRotation(Bitmap image, String path) {
        try {
```

```
            ExifInterface exif = new ExifInterface(path);
            int exifOrientation = exif.getAttributeInt(
                    ExifInterface.TAG_ORIENTATION,
                    ExifInterface.ORIENTATION_NORMAL);

            Log.d(TAG, "Orientation: " + exifOrientation);

            int rotate = 0;

            switch (exifOrientation) {
                case ExifInterface.ORIENTATION_ROTATE_90:
                    rotate = 0;
                    break;

                case ExifInterface.ORIENTATION_ROTATE_270:
                    rotate = 180;
                    break;
            }

            Log.v(TAG, "Rotation: " + rotate);

            if (rotate != 0) {

                // Getting width & height of the given image.
                int w = image.getWidth();
                int h = image.getHeight();

                // Setting pre rotate
                Matrix mtx = new Matrix();
                mtx.preRotate(rotate);

                // Rotating Bitmap
                image = Bitmap.createBitmap(image, 0, 0, w, h, mtx, false);
            }

            // Convert to ARGB_8888
            return image.copy(Bitmap.Config.ARGB_8888, true);

        } catch (IOException e) {
            Log.e(TAG, "Couldn't correct orientation: " + e.toString());
        }
        return image;
    }
}
```

1.11 ProcessImageTask.java

```
package c00112726.itcarlow.ie.finalyearproject.tasks;

import android.app.ProgressDialog;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.AsyncTask;
import android.util.Log;

import org.opencv.core.Mat;

import java.io.File;
import java.util.List;
import java.util.Map;

import c00112726.itcarlow.ie.finalyearproject.exceptions.BadImageException;
import c00112726.itcarlow.ie.finalyearproject.misc.NumberPlate;
import c00112726.itcarlow.ie.finalyearproject.processing.OpenCvImageProcessor;
import c00112726.itcarlow.ie.finalyearproject.processing.template.ImageTemplates;
import c00112726.itcarlow.ie.finalyearproject.processing.template.TemplateLoader;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallback;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 06/04/2016
 */
public class ProcessImageTask extends AsyncTask<File, String, NumberPlate> {

    private static final String TAG = "ProcessImageTask";

    private TaskCallback mTaskCallback;
    private ProgressDialog mProgressDialog;

    public ProcessImageTask(TaskCallback taskCallback) {
        mTaskCallback = taskCallback;
    }

    @Override
    protected void onPreExecute() {
        mProgressDialog = new ProgressDialog((Context) mTaskCallback);
        mProgressDialog.setMessage("Processing...");
        mProgressDialog.setCanceledOnTouchOutside(false);
        mProgressDialog.setCancelable(false);
        mProgressDialog.show();
    }

    @Override
    protected NumberPlate doInBackground(File... data) {
        try {
            File imageFile = data[0];
            String path = imageFile.getAbsolutePath();
            Bitmap image = BitmapFactory.decodeFile(path);

            publishProgress("Processing Image");
            // process image
            Mat imageProcessed = OpenCvImageProcessor.process(image);

            publishProgress("Segmenting Image");
            // segment image
            Map<String, List<Mat>> imageSegmented;
            // There may be nothing to segment, e.g. a black or white screen
            try {
                imageSegmented = OpenCvImageProcessor.segmentImage(imageProcessed);
            }
```

```
            catch (BadImageException e) {
                Log.d(TAG, e.getMessage());
                return null;
            }

            publishProgress("Loading Image Templates");
            ImageTemplates templates = TemplateLoader.loadTemplates((Context) mTaskCallback,
"templates");

            publishProgress("Performing OCR");
            NumberPlate numberPlate = OpenCvImageProcessor.performOCR(imageSegmented,
templates);
            Log.d(TAG, "Recognised text: " + numberPlate.toString());

            return numberPlate;
        }
        catch (Exception e) {
            Log.e(TAG, e.getMessage());
            return null;
        }
    }

    @Override
    public void onPostExecute(NumberPlate numberPlate) {
        if(mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
            mProgressDialog = null;
        }

        mTaskCallback.onTaskComplete(numberPlate);
    }

    @Override
    protected void onProgressUpdate(String... status) {
        mProgressDialog.setMessage(status[0]);
    }
}
```

1.12 LoginTask.java

```
package c00112726.itcarlow.ie.finalyearproject.tasks;

import android.app.ProgressDialog;
import android.content.Context;
import android.os.AsyncTask;
import android.util.Log;

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

import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallbackJSON;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 31/03/2016
 */
public class LoginTask extends AsyncTask<String, Void, JSONObject> {

    private static final String TAG = "LoginTask";
    private static final String LOGIN_URL = "http://mikereid73.pythonanywhere.com/login";

    protected TaskCallbackJSON mTaskCallback;
    protected ProgressDialog mProgressDialog;

    public LoginTask(TaskCallbackJSON TaskCallback) {
        mTaskCallback = TaskCallback;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        mProgressDialog = new ProgressDialog((Context) mTaskCallback);
        mProgressDialog.setTitle("Please wait");
        mProgressDialog.setMessage("Authenticating...");
        mProgressDialog.setIndeterminate(true);
        mProgressDialog.setCancelable(false);
        mProgressDialog.setCanceledOnTouchOutside(false);
        mProgressDialog.show();
    }

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

        String username = params[0];
        String password = params[1];

        JSONObject json = new JSONObject();
        try {
            json.put("username", username);
            json.put("password", password);
        } catch (JSONException e) {
            Log.e(TAG, e.getMessage());
        }

        try {
```

```
            URL url = new URL(LOGIN_URL);
            HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
            urlConnection.setReadTimeout(10000); /* milliseconds */
            urlConnection.setConnectTimeout(15000); /* milliseconds */
            urlConnection.setRequestMethod("POST");
            urlConnection.setDoInput(true);
            urlConnection.setDoOutput(true);
            urlConnection.setRequestProperty("Content-Type", "application/json;charset=utf-8");
            urlConnection.connect();

            // Post the json to the server
            DataOutputStream wr = new DataOutputStream(urlConnection.getOutputStream());
            wr.writeBytes(json.toString());
            wr.flush();
            wr.close();

            // Receive the response from the server
            InputStream in = new BufferedInputStream(urlConnection.getInputStream());
            BufferedReader reader = new BufferedReader(new InputStreamReader(in));
            StringBuilder sb = new StringBuilder();
            String line;
            while ((line = reader.readLine()) != null) {
                sb.append(line);
            }
            urlConnection.disconnect();
            reader.close();
            in.close();

            return new JSONObject(sb.toString());

        }catch (IOException e) {
            Log.e(TAG, e.getMessage());
        }
        catch (JSONException e) {
            Log.i(TAG, e.getMessage());
        }

        return null;
    }

    @Override
    protected void onPostExecute(JSONObject result) {
        super.onPostExecute(result);
        if(mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
            mProgressDialog = null;
        }

        mTaskCallback.onTaskComplete(result);
    }
}
```

1.13 DatabaseConnectionTask.java

```
package c00112726.itcarlow.ie.finalyearproject.tasks;

import android.app.ProgressDialog;
import android.content.Context;
import android.content.SharedPreferences;
import android.os.AsyncTask;
import android.util.Log;

import org.json.JSONObject;

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

import c00112726.itcarlow.ie.finalyearproject.misc.NumberPlate;
import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallbackJSON;

/**
 * Author: Michael Reid
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 26/03/2016
 */
public class DatabaseConnectionTask extends AsyncTask<NumberPlate, Void, JSONObject> {

    private static final String TAG = "DatabaseConnectionTask";
    private static final String LOGIN_URL = "http://mikereid73.pythonanywhere.com/post";
    private TaskCallbackJSON mTaskCallback;
    private ProgressDialog mProgressDialog;

    public DatabaseConnectionTask(TaskCallbackJSON taskCallback) {
        mTaskCallback = taskCallback;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();

        mProgressDialog = new ProgressDialog((Context) mTaskCallback);
        mProgressDialog.setMessage("Sending to database...");
        mProgressDialog.setCanceledOnTouchOutside(false);
        mProgressDialog.setCancelable(false);
        mProgressDialog.show();
    }

    @Override
    protected JSONObject doInBackground(NumberPlate... params) {

        try {
            NumberPlate reg = params[0];
            SharedPreferences sharedPreferences =
((Context)mTaskCallback).getSharedPreferences("login", Context.MODE_PRIVATE);
            String username = sharedPreferences.getString("username", "admin");
            JSONObject jsonObject = new JSONObject();
            jsonObject.put("guess", reg.getWrongGuess());
            jsonObject.put("actual", reg.toString());
            jsonObject.put("username", username);

            URL url = new URL(LOGIN_URL);
            HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
            urlConnection.setReadTimeout(10000); /* milliseconds */
            urlConnection.setConnectTimeout(15000); /* milliseconds */
```

```
            urlConnection.setRequestMethod("POST");
            urlConnection.setDoInput(true);
            urlConnection.setDoOutput(true);

            //make some HTTP header nicety
            urlConnection.setRequestProperty("Content-Type", "application/json;charset=utf-8");
            //urlConnection.setRequestProperty("X-Requested-With", "XMLHttpRequest");

            //open
            urlConnection.connect();

            // Post the json to the server
            DataOutputStream wr = new DataOutputStream(urlConnection.getOutputStream());
            wr.writeBytes(jsonObject.toString());
            wr.flush();
            wr.close();

            // Receive the response from the server
            InputStream in = new BufferedInputStream(urlConnection.getInputStream());
            BufferedReader reader = new BufferedReader(new InputStreamReader(in));
            StringBuilder sb = new StringBuilder();
            String line;
            while ((line = reader.readLine()) != null) {
                sb.append(line);
            }
            urlConnection.disconnect();
            reader.close();
            in.close();

            return new JSONObject(sb.toString());
        }
        catch (Exception e) {
            Log.e(TAG, e.getMessage());
        }

        return null;
    }

    @Override
    protected void onPostExecute(JSONObject json) {
        if(mTaskCallback == null) { return; }
        if(mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
            mProgressDialog = null;
        }

        mTaskCallback.onTaskComplete(json);

    }
}
```

1.14 ChangePasswordTask.java

```
package c00112726.itcarlow.ie.finalyearproject.tasks;

import android.app.ProgressDialog;
import android.content.Context;
import android.os.AsyncTask;
import android.util.Log;

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

import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.DataOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

import c00112726.itcarlow.ie.finalyearproject.tasks.callbacks.TaskCallbackJSON;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 05/04/2016
 */
public class ChangePasswordTask extends AsyncTask<String, Void, JSONObject> {

    private static final String TAG = "ChangePasswordTask";
    private static final String LOGIN_URL =
"http://mikereid73.pythonanywhere.com/change_password";

    private TaskCallbackJSON mTaskCallback;
    private ProgressDialog mProgressDialog;

    public ChangePasswordTask(TaskCallbackJSON taskCallback) {
        mTaskCallback = taskCallback;
    }

    @Override
    protected void onPreExecute() {
        super.onPreExecute();

        mProgressDialog = new ProgressDialog((Context) mTaskCallback);
        mProgressDialog.setMessage("Applying changes...");
        mProgressDialog.setCanceledOnTouchOutside(false);
        mProgressDialog.setCancelable(false);
        mProgressDialog.show();
    }

    @Override
    protected JSONObject doInBackground(String... params) {
        String username = params[0];
        String oldPassword = params[1];
        String newPassword = params[2];

        JSONObject json = new JSONObject();
        try {
            json.put("username", username);
            json.put("old_password", oldPassword);
            json.put("new_password", newPassword);
        } catch (JSONException e) {
            Log.e(TAG, e.getMessage());
        }
```

```
        try {
            URL url = new URL(LOGIN_URL);
            HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
            urlConnection.setReadTimeout(10000); /* milliseconds */
            urlConnection.setConnectTimeout(15000); /* milliseconds */
            urlConnection.setRequestMethod("POST");
            urlConnection.setDoInput(true);
            urlConnection.setDoOutput(true);
            urlConnection.setRequestProperty("Content-Type", "application/json;charset=utf-8");
            urlConnection.connect();

            // Post the json to the server
            DataOutputStream wr = new DataOutputStream(urlConnection.getOutputStream());
            wr.writeBytes(json.toString());
            wr.flush();
            wr.close();

            // Receive the response from the server
            InputStream in = new BufferedInputStream(urlConnection.getInputStream());
            BufferedReader reader = new BufferedReader(new InputStreamReader(in));
            StringBuilder sb = new StringBuilder();
            String line;
            while ((line = reader.readLine()) != null) {
                sb.append(line);
            }
            urlConnection.disconnect();
            reader.close();
            in.close();

            return new JSONObject(sb.toString());

        }catch (IOException e) {
            Log.e(TAG, e.getMessage());
        }
        catch (JSONException e) {
            Log.i(TAG, e.getMessage());
        }

        return null;
    }

    @Override
    protected void onPostExecute(JSONObject json) {
        super.onPostExecute(json);
        if(mProgressDialog != null && mProgressDialog.isShowing()) {
            mProgressDialog.dismiss();
            mProgressDialog = null;
        }

        mTaskCallback.onTaskComplete(json);
    }
}
```

1.15 TaskCallback.java

```
package c00112726.itcarlow.ie.finalyearproject.tasks.callbacks;

import java.io.File;

import c00112726.itcarlow.ie.finalyearproject.misc.NumberPlate;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 01/04/2016
 */
public interface TaskCallback {
    void onTaskComplete(NumberPlate numberPlate);
    void onTaskComplete(File file);
}
```

1.16 TaskCallbackJSON.java

```
package c00112726.itcarlow.ie.finalyearproject.tasks.callbacks;

import org.json.JSONObject;

import c00112726.itcarlow.ie.finalyearproject.misc.NumberPlate;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 06/04/2016
 */
public interface TaskCallbackJSON {
    void onTaskComplete(JSONObject json);
}
```

1.17 BadImageException.java

```
package c00112726.itcarlow.ie.finalyearproject.exceptions;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 07/04/2016
 */
public class BadImageException extends Exception {
    public BadImageException(String message) {
        super(message);
    }
}
```

1.18 OpenCvImageProcessor.java

```
package c00112726.itcarlow.ie.finalyearproject.processing;

import android.graphics.Bitmap;
import android.util.Log;

import org.opencv.android.Utils;
import org.opencv.core.Core;
import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.core.MatOfPoint;
import org.opencv.core.Rect;
import org.opencv.core.Size;
import org.opencv.imgproc.Imgproc;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import c00112726.itcarlow.ie.finalyearproject.exceptions.BadImageException;
import c00112726.itcarlow.ie.finalyearproject.misc.NumberPlate;
import c00112726.itcarlow.ie.finalyearproject.processing.template.ImageTemplate;
import c00112726.itcarlow.ie.finalyearproject.processing.template.ImageTemplates;

/**
 * Author: Michael Reid
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 03/02/2016
 */
public class OpenCvImageProcessor {

    /* The lowest value of RGB */
    private static final int MIN_RGB_BOUND = 0;
    /* The highest value of RGB */
    private static final int MAX_RGB_BOUND = 255;
    /* The average width of a plate character */
    private static final float TARGET_CHAR_WIDTH = 36.0f;
    /* The average height of a plate character */
    private static final float TARGET_CHAR_HEIGHT = 70.0f;
    /* The average aspect ratio of a character */
    private static final float TARGET_ASPECT_RATIO = TARGET_CHAR_WIDTH / TARGET_CHAR_HEIGHT;
    /* A generous 55% error tolerance */
    private static final float ERROR_ALLOWED = 0.55f;
    /* Minimum height of a character */
    private static final float MIN_CHAR_HEIGHT = 50.0f;
    /* Maximum height of a character*/
    private static final float MAX_CHAR_HEIGHT = 100f;
    /* Minimum character aspect ratio of a character. Allows things like the number 1 */
    private static final float MIN_CHAR_ASPECT = 0.05f;
    /* Maximum character aspect ratio, taking error allowance into account */
    private static final float MAX_CHAR_ASPECT = TARGET_ASPECT_RATIO + TARGET_ASPECT_RATIO *
ERROR_ALLOWED;
    /* Width image is resized to. Conforms to legal Irish plate width */
    private static final float RESIZE_WIDTH = 520;
    /* Height image is resized to. Conforms to legal Irish plate height */
    private static final float RESIZE_HEIGHT = 110;
    /* The width of the kernel used when performing image opening */
    private static final float MORPH_OPEN_WIDTH = 7;
    /* The height of the kernel used when performing image opening */
    private static final float MORPH_OPEN_HEIGHT = 5;

    /* No instance allowed */
    private OpenCvImageProcessor() {
```

```
}

/**
 * Utility method to convert an Android Bitmap to an OpenCV Mat
 * @param input image to convert
 * @param type type Mat type
 * @return new Mat
 */
public static Mat bitmapToMat(Bitmap input, int type) {
    int width = input.getWidth();
    int height = input.getHeight();
    Mat output = new Mat(height, width, type);

    Utils.bitmapToMat(input, output);

    return output;
}

/**
 * Pre-process the image for character segmentation.
 * Calls a number of OpenCV methods.
 * @param image image to process
 * @return Mat of processed image
 */
public static Mat process(Bitmap image) {
    // Convert to Mat, OpenCv only works with Mat
    Mat input = bitmapToMat(image, CvType.CV_8UC3);
    Mat output = new Mat();

    // Resize image. Allows code to work regardless of image quality and size
    Imgproc.resize(input, output, new Size(RESIZE_WIDTH, RESIZE_HEIGHT));
    // Convert to greyscale. Working with colour images is not an option.
    Imgproc.cvtColor(output, output, Imgproc.COLOR_BGR2GRAY);

    // Calculate the Otsu value to use as a lower threshold value.
    double ostuValue = Imgproc.threshold(
            output,
            new Mat(), // we don't care about changing any Mats, we want the return value
            MIN_RGB_BOUND,
            MAX_RGB_BOUND,
            Imgproc.THRESH_BINARY | Imgproc.THRESH_OTSU
    );

    // Threshold the image, using the Otsu value calculated.
    // Invert black and white for later steps.
    Imgproc.threshold(
            output,
            output,
            ostuValue,
            MAX_RGB_BOUND,
            Imgproc.THRESH_BINARY_INV
    );

    // Create a structuring element to be used when opening the image.
    Mat element = Imgproc.getStructuringElement(
            Imgproc.MORPH_RECT,
            new Size(MORPH_OPEN_WIDTH, MORPH_OPEN_HEIGHT)
    );

    // Perform image opening.
    // Erode followed by a Dilate
    Imgproc.morphologyEx(
            output,
            output,
            Imgproc.MORPH_OPEN,
            element
```

```
        );
        return output;
    }

    /**
     * Takes in a list of found contours in an image and creates a bounding box for them.
     * The boxes are filtered to remove any which don't have the required dimension.
     * @param contours List of found contours
     * @return List of rectangles representing bounding boxes
     */
    private static List<Rect> getBoundingBoxes(List<MatOfPoint> contours) {
        final List<Rect> boundingBoxes = new ArrayList<>();

        //For each contour found
        for (int i = 0; i < contours.size(); i++) {
            MatOfPoint currentContour = contours.get(i);
            Rect box = Imgproc.boundingRect(currentContour);
            float charAspect = (float)box.width / (float)box.height;

            if (    charAspect >= MIN_CHAR_ASPECT &&
                    charAspect <= MAX_CHAR_ASPECT &&
                    box.height >= MIN_CHAR_HEIGHT &&
                    box.height <= MAX_CHAR_HEIGHT) {
                boundingBoxes.add(box);
            } else {
                contours.remove(i--);
            }
        }
        sort(boundingBoxes);
        return boundingBoxes;
    }

    /**
     * Sort bounding boxes based on their x coordinate.
     * Ensures the segmented images are in order, left to right
     * @param boxes bounding boxes to order
     */
    public static void sort(List<Rect> boxes) {
        Collections.sort(boxes, new Comparator<Rect>() {
            @Override
            public int compare(Rect r1, Rect r2) {
                return r1.x > (r2.x) ? 1 : -1;
            }
        });
    }

    /**
     * Segments the individual characters from the processed image.
     * @param processedImage The pre processed image
     * @return A map containing the characters of the image, separated as year, county, reg
     * @throws BadImageException Thrown if no characters are found
     */
    public static Map<String, List<Mat>> segmentImage(Mat processedImage) throws
BadImageException {
        // Find all contours in the image and store them in a list.
        List<MatOfPoint> contours = new ArrayList<>();
        Imgproc.findContours(
                processedImage.clone(),
                contours,
                new Mat(),                    // don't care about hierarchy
                Imgproc.RETR_EXTERNAL,        // ignore contours within contours
                Imgproc.CHAIN_APPROX_NONE
        );
        // Throw exception if no contours are found
        if(contours.size() <= 0) {
            throw new BadImageException("No contours were detected");
        }
```

```
        // Get bounding boxes for the found contours
        List<Rect> boundingBoxes = getBoundingBoxes(contours);

        // Throw exception if no bounding boxes are returned
        if(boundingBoxes.size() <= 0) {
            throw new BadImageException("No bounding boxes were detected");
        }

        // Using the bounding boxes, segment those areas from main image
        List<Mat> segmentedImages = new ArrayList<>();
        for (int i = 0; i < boundingBoxes.size(); i++) {
            Rect currentBoundingBox = boundingBoxes.get(i);
            Mat currentMat = processedImage.submat(currentBoundingBox);
            segmentedImages.add(currentMat);
        }

        // Split the reg and return characters
        return splitIrishReg(segmentedImages, boundingBoxes);
    }

    /**
     * Split the bounding boxes into 3 parts; year, county, and reg
     * This is useful when performing OCR, to avoid mismatching numbers and letters.
     * Find the two biggest gaps between bounding boxes, which represent the hyphen.
     * @param mats List of segmented characters
     * @param boundingBoxes List of bounding boxes of segmented characters
     * @return Map of the segmented characters, split up into year, county, reg
     */
    public static Map<String, List<Mat>> splitIrishReg(List<Mat> mats, List<Rect> boundingBoxes)
{
        int biggestGap = 0;
        int secondBiggestGap = 0;
        int biggestGapIndex = -1;
        int secondBiggestGapIndex = -1;

        for(int i = 1; i < boundingBoxes.size(); i++) {
            Rect current = boundingBoxes.get(i);
            Rect previous = boundingBoxes.get(i - 1);
            int distance = current.x - (previous.x + previous.width);

            if (distance > biggestGap) {
                // Biggest is now second biggest
                secondBiggestGap = biggestGap;
                secondBiggestGapIndex = biggestGapIndex;

                // New Biggest
                biggestGap = distance;
                biggestGapIndex = i;
            }
            else if (distance > secondBiggestGap && distance <= biggestGap) {
                // distance <= biggestGap : gaps might be same size
                secondBiggestGap = distance;
                secondBiggestGapIndex = i;
            }
        }
        // Swap values to ensure subList doesn't try to map backwards.
        // i.e.  0-4, 5-2, etc
        if(secondBiggestGapIndex < biggestGapIndex) {
            int temp = biggestGapIndex;
            biggestGapIndex = secondBiggestGapIndex;
            secondBiggestGapIndex = temp;
        }

        try {
            List<Mat> year = mats.subList(0, biggestGapIndex);
            List<Mat> county = mats.subList(biggestGapIndex, secondBiggestGapIndex);
```

```
            List<Mat> reg = mats.subList(secondBiggestGapIndex, mats.size());

            // Stick results in a map and return it
            Map<String, List<Mat>> fullRegSplit = new HashMap<>();
            fullRegSplit.put("year", year);
            fullRegSplit.put("county", county);
            fullRegSplit.put("reg", reg);
            return fullRegSplit;
        }
        catch (Exception e) {
            Log.e("Reg Split Failed", e.getMessage());
            return null;
        }
    }

    /**
     * Perform character recognition on the 3 groups of images
     * @param images year, county, reg
     * @param imageTemplates templates for available characters
     * @return Image represented as a NumberPlate object
     */
    public static NumberPlate performOCR(Map<String, List<Mat>> images,
                                         ImageTemplates imageTemplates) {

        List<Mat> yearImages = images.get("year");
        List<Mat> countyImages = images.get("county");
        List<Mat> regImages = images.get("reg");

        String year = matchNumbers(yearImages, imageTemplates);
        String county = matchLetters(countyImages, imageTemplates);
        String reg = matchNumbers(regImages, imageTemplates);

        // Store result in a simple number plate class
        return new NumberPlate(year, county, reg);
    }

    /**
     * Perform template matching on images considered to be letters
     * @param segmentedImages list of character images
     * @param imageTemplates character templates
     * @return segmentedImages represented as a String
     */
    public static String matchLetters(List<Mat> segmentedImages, ImageTemplates imageTemplates)
{
        final List<ImageTemplate> letterTemplates = imageTemplates.getLetterTemplates();
        final StringBuilder sb = new StringBuilder();

        for(int x = 0; x < segmentedImages.size(); x++) {
            Mat current = segmentedImages.get(x).clone(); // clone to avoid changing original
size
            Imgproc.resize(current, current, new Size(TARGET_CHAR_WIDTH, TARGET_CHAR_HEIGHT));

            double bestMatchAccuracy = 0;
            String bestMatchName = "";

            // match letters
            for (ImageTemplate currentLetter: letterTemplates) {
                List<Mat> templates = currentLetter.getImages();
                for(int i = 0; i < templates.size(); i++) {
                    Mat template = templates.get(i);

                    // / Do the matching and normalise
                    Mat result = new Mat();
                    Imgproc.matchTemplate(current, template, result, Imgproc.TM_CCOEFF);

                    // / Localising the best match with minMaxLoc
                    Core.MinMaxLocResult minMaxResult = Core.minMaxLoc(result);
```

```
                    if(minMaxResult.maxVal > bestMatchAccuracy) {
                        bestMatchAccuracy = minMaxResult.maxVal;
                        bestMatchName = currentLetter.toString();
                    }
                }
            }
            sb.append(bestMatchName);
        }
        if(segmentedImages.size() == 1 && sb.toString().equals("O")) {
            sb.deleteCharAt(0);
            sb.append("D");
        }
        return sb.toString();
    }

    /**
     * Perform template matching on images considered to be numbers
     * @param segmentedImages list of number images
     * @param imageTemplates character templates
     * @return segmentedImages represented as a String
     */
    public static String matchNumbers(List<Mat> segmentedImages, ImageTemplates imageTemplates)
{
        final List<ImageTemplate> numberTemplates = imageTemplates.getNumberTemplates();
        final StringBuilder sb = new StringBuilder();

        for(int x = 0; x < segmentedImages.size(); x++) {
            Mat current = segmentedImages.get(x).clone(); // clone to avoid changing original
size
            Imgproc.resize(current, current, new Size(TARGET_CHAR_WIDTH, TARGET_CHAR_HEIGHT));

            double bestMatchAccuracy = 0;
            String bestMatchName = "";

            // match numbers
            for (ImageTemplate currentNumber: numberTemplates) {
                List<Mat> templates = currentNumber.getImages();
                for(int i = 0; i < templates.size(); i++) {
                    Mat template = templates.get(i);

                    // / Do the matching and normalise
                    Mat result = new Mat();
                    Imgproc.matchTemplate(current, template, result, Imgproc.TM_CCOEFF);

                    // / Localising the best match with minMaxLoc
                    Core.MinMaxLocResult mmr = Core.minMaxLoc(result);
                    if(mmr.maxVal > bestMatchAccuracy) {
                        bestMatchAccuracy = mmr.maxVal;
                        bestMatchName = currentNumber.toString();
                    }
                }
            }
            sb.append(bestMatchName);
        }
        return sb.toString();
    }
}
```

1.19 TemplateLoader.java

```
package c00112726.itcarlow.ie.finalyearproject.processing.template;

import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;

import org.opencv.core.CvType;
import org.opencv.core.Mat;
import org.opencv.core.Size;
import org.opencv.imgproc.Imgproc;

import java.io.IOException;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.List;

import c00112726.itcarlow.ie.finalyearproject.processing.OpenCvImageProcessor;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 02/04/2016
 */
public class TemplateLoader {

    private final static String[] dirs = {
            "/image_set_1/",
            "/image_set_2/",
            "/image_set_3/",
    };

    private final static String[] letters = {
            "C", "D", "E", "G", "H",
            "K", "L", "M", "N", "O", "R",
            "S", "T", "W", "X", "Y"
    };

    private final static String[] numbers = {
            "0", "1", "2", "3", "4",
            "5", "6", "7", "8", "9"
    };

    private TemplateLoader() {}

    public static ImageTemplates loadTemplates(Context context, String directory) throws
IOException {
        List<ImageTemplate> letterTemplates = new ArrayList<>();
        List<ImageTemplate> numberTemplates = new ArrayList<>();

        for (String letter : letters) {
            letterTemplates.add(loadImageTemplates(context, directory, letter));
        }

        for (String number : numbers) {
            numberTemplates.add(loadImageTemplates(context, directory, number));
        }

        return new ImageTemplates(letterTemplates, numberTemplates);
    }

    private static ImageTemplate loadImageTemplates(Context context, String directory, String
templateName) throws IOException {
        final List<Mat> images = new ArrayList<>();

        for (String dir : dirs) {
```

```
            try {
                String filename = directory + dir + templateName + ".jpg";
                InputStream is = context.getAssets().open(filename);
                Bitmap bitmap = BitmapFactory.decodeStream(is);
                Mat mat = OpenCvImageProcessor.bitmapToMat(bitmap, CvType.CV_8UC3);
                Imgproc.cvtColor(mat, mat, Imgproc.COLOR_BGR2GRAY);
                Imgproc.resize(mat, mat, new Size(36.0f, 70.0f));
                images.add(mat);
            }
            catch(Exception e) {
                e.printStackTrace();
            }
        }

        return new ImageTemplate(templateName, images);
    }
}
```

1.20 ImageTemplates.java

```
package c00112726.itcarlow.ie.finalyearproject.processing.template;

import java.util.List;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 02/04/2016
 */
public class ImageTemplates {

    private List<ImageTemplate> letterTemplates;
    private List<ImageTemplate> numberTemplates;

    public ImageTemplates(List<ImageTemplate> letterTemplates, List<ImageTemplate>
numberTemplates) {
        this.letterTemplates = letterTemplates;
        this.numberTemplates = numberTemplates;
    }

    public List<ImageTemplate> getLetterTemplates() {
        return letterTemplates;
    }

    public List<ImageTemplate> getNumberTemplates() {
        return numberTemplates;
    }
}
```

1.21 ImageTemplate.java

```
package c00112726.itcarlow.ie.finalyearproject.processing.template;

import org.opencv.core.Mat;

import java.util.List;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 02/04/2016
 */
public class ImageTemplate {

    private String templateName;
    private List<Mat> images;

    public ImageTemplate(String templateName, List<Mat> images) {
        this.templateName = templateName;
        this.images = images;
    }

    public List<Mat> getImages() {
        return images;
    }

    @Override
    public String toString() {
        return templateName;
    }
}
```

1.22 NumberPlate.java

```
package c00112726.itcarlow.ie.finalyearproject.misc;

import java.io.Serializable;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 01/04/2016
 */
public class NumberPlate implements Serializable {

    private String year;
    private String county;
    private String reg;
    private String wrongGuess;

    public NumberPlate() {
        this("","","");
    }

    public NumberPlate(String year, String county, String reg) {
        this.year = year;
        this.county = county;
        this.reg = reg;

        wrongGuess = year + county + reg;
    }

    public String getWrongGuess() {
        return wrongGuess;
    }

    public String getCounty() {
        return county;
    }

    public void setCounty(String county) {
        this.county = county;
    }

    public String getYear() {
        return year;
    }

    public void setYear(String year) {
        this.year = year;
    }

    public String getReg() {
        return reg;
    }

    public void setReg(String reg) {
        this.reg = reg;
    }

    @Override
    public String toString() {
        return year + county + reg;
    }
}
```

1.23 Util.java

```
package c00112726.itcarlow.ie.finalyearproject.misc;

import android.content.Context;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.widget.Toast;

/**
 * Author: Michael Reid.
 * ID: C00112726
 * Email: c00112726@itcarlow.ie
 * Date: 05/04/2016
 */
public class Util {

    private Util() {}

    public static boolean isNetworkAvailable(Context context) {
        ConnectivityManager connMgr = (ConnectivityManager)
                context.getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
        return (networkInfo != null && networkInfo.isConnected()) ;
    }

    public static void showToast(Context context, String message, int duratAion) {
        Toast.makeText(context, message, duration).show();
    }
}
```

1.24 activity_camera.xml

```
<?xml version="1.0" encoding="utf-8"?>

<FrameLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <FrameLayout
        android:id="@+id/camera_preview"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>

</FrameLayout>
```

activity_edit_password.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:fitsSystemWindows="true">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="56dp"
        android:paddingLeft="24dp"
        android:paddingRight="24dp">

        <!-- Email Label -->
        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp">
            <EditText android:id="@+id/input_current_password"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="textPassword"
                android:hint="@string/current_password" />
        </android.support.design.widget.TextInputLayout>

        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp">
            <EditText android:id="@+id/input_new_password"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="textPassword"
                android:hint="@string/new_password"/>
        </android.support.design.widget.TextInputLayout>

        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp">
            <EditText android:id="@+id/input_confirm_new_password"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="textPassword"
                android:hint="@string/confirm_password"/>
        </android.support.design.widget.TextInputLayout>
```

```
        <Button
            android:id="@+id/btn_submit"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:layout_marginBottom="24dp"
            android:padding="12dp"
            android:text="@string/change_password"/>

        <Button
            android:id="@+id/btn_cancel"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp"
            android:text="@string/cancel"/>

    </LinearLayout>
</ScrollView>
```

1.25 activity_edit_ref.xml

```
<?xml version="1.0" encoding="utf-8"?>

<ScrollView
xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:fitsSystemWindows="true">

    <RelativeLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingLeft="16dp"
        android:paddingRight="16dp">

        <LinearLayout
            android:id="@+id/editArea"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_centerHorizontal="true"
            android:orientation="vertical">

            <ImageView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:id="@+id/imageView"
                android:background="#000000"
                android:padding="1dp"/>

            <EditText
                android:id="@+id/regNumber"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:editable="false"
                android:textIsSelectable="false"
                android:focusable="false"/>

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Year"
                android:gravity="center"/>

            <EditText
                android:id="@+id/year"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:inputType="number"
                android:maxLines="1"/>

            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="County"
                android:gravity="center"/>

            <EditText
                android:id="@+id/county"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:inputType="textCapCharacters"
                android:maxLines="1"/>
```

```
            <TextView
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:text="Registration"
                android:gravity="center"/>

            <EditText
                android:id="@+id/registration"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:inputType="number"
                android:maxLines="1"/>

        </LinearLayout>

        <Button
            android:id="@+id/ok"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Continue"
            android:layout_gravity="end"
            android:layout_alignTop="@+id/cancel"
            android:layout_alignParentEnd="true"/>

        <Button
            android:id="@+id/cancel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Cancel"
            android:layout_below="@+id/editArea"
            android:layout_alignParentStart="true"/>

    </RelativeLayout>
</ScrollView>
```

1.26 activity_login.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:fitsSystemWindows="true">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="56dp"
        android:paddingLeft="24dp"
        android:paddingRight="24dp">

        <ImageView android:src="@drawable/logo"
            android:layout_width="wrap_content"
            android:layout_height="72dp"
            android:layout_marginBottom="24dp"
            android:layout_gravity="center_horizontal"
            android:contentDescription="@null" />

        <!-- Email Label -->
        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp">
            <EditText android:id="@+id/input_email"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="textEmailAddress"
                android:hint="@string/email_text" />
        </android.support.design.widget.TextInputLayout>

        <android.support.design.widget.TextInputLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="8dp"
            android:layout_marginBottom="8dp">
            <EditText android:id="@+id/input_password"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="textPassword"
                android:hint="@string/password_text"/>
        </android.support.design.widget.TextInputLayout>

        <Button
            android:id="@+id/btn_login"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:layout_marginBottom="24dp"
            android:padding="12dp"
            android:text="@string/login_text" />

    </LinearLayout>
</ScrollView>
```

1.27 activity_main_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:fitsSystemWindows="true">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="56dp"
        android:paddingLeft="24dp"
        android:paddingRight="24dp">

        <Button
            android:id="@+id/btn_ocr"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:layout_marginBottom="24dp"
            android:padding="12dp"
            android:text="@string/start_ocr"/>

        <Button
            android:id="@+id/btn_settings"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:layout_marginBottom="24dp"
            android:padding="12dp"
            android:text="@string/settings"/>

        <Button
            android:id="@+id/btn_logout"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:layout_marginBottom="24dp"
            android:padding="12dp"
            android:text="@string/logout"/>

    </LinearLayout>
</ScrollView>
```

1.28 activity_processing_settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:fitsSystemWindows="true">

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

        <TextView
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="@string/blur_settings"
            android:paddingTop="12dp"/>

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

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:text="@string/kernel_width"/>

            <EditText
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="number"/>

        </LinearLayout>

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

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:text="@string/kernel_height"/>

            <EditText
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="number"/>

        </LinearLayout>

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingBottom="12dp">

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:text="@string/distribution"/>
```

```
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number"/>

</LinearLayout>

<View
    android:layout_width="match_parent"
    android:layout_height="2dp"
    android:background="?android:attr/listDivider"/>

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:text="@string/grey_settings"
    android:paddingTop="12dp"/>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:paddingBottom="12dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/grey_mode"/>

    <ListView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="number"/>

</LinearLayout>

<View
    android:layout_width="match_parent"
    android:layout_height="2dp"
    android:background="?android:attr/listDivider"/>

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:text="@string/thresh_settings"
    android:paddingTop="12dp"/>

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/lower_bound"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number"/>
```

```
</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:paddingBottom="12dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/upper_bound"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number"/>

</LinearLayout>

<View
    android:layout_width="match_parent"
    android:layout_height="2dp"
    android:background="?android:attr/listDivider"/>

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:text="@string/morph_settings"
    android:paddingTop="12dp"/>

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/open_width"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number"/>

</LinearLayout>

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

    android:paddingBottom="12dp">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/open_height"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
        android:inputType="number"/>

</LinearLayout>

<View
    android:layout_width="match_parent"
    android:layout_height="2dp"
    android:background="?android:attr/listDivider"/>

<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center"
    android:text="@string/contour_settings"
    android:paddingTop="12dp"/>

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/min_aspect"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number"/>

</LinearLayout>

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/max_aspect"
        android:layout_gravity="center"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="number"/>

</LinearLayout>

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="@string/min_width"/>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```
            android:inputType="number"/>

        </LinearLayout>

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

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
                android:text="@string/min_height"/>

            <EditText
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:inputType="number"/>

        </LinearLayout>

        <Button
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Apply"/>

    </LinearLayout>
</ScrollView>
```

1.29 activity_settings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:fitsSystemWindows="true">

    <LinearLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="56dp"
        android:paddingLeft="24dp"
        android:paddingRight="24dp">

        <Button
            android:id="@+id/btn_advanced_settings"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:layout_marginBottom="24dp"
            android:padding="12dp"
            android:text="@string/advanced_settings"/>

        <Button
            android:id="@+id/btn_change_password"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="24dp"
            android:layout_marginBottom="24dp"
            android:padding="12dp"
            android:text="@string/change_password"/>

        </LinearLayout>

</ScrollView>
```

1.30 strings.xml

```
<resources>
    <string name="app_name">Plate Check</string>

    <!-- Strings related to login -->
    <string name="prompt_email">Email</string>
    <string name="prompt_password">Password</string>
    <string name="action_sign_in">Sign in</string>
    <string name="error_invalid_email">This email address is invalid</string>
    <string name="error_invalid_password">This password is too short</string>
    <string name="error_incorrect_password">This password is incorrect</string>
    <string name="error_field_required">This field is required</string>
    <string name="login_text">Login</string>
    <string name="email_text">Email</string>
    <string name="password_text">Password</string>
    <string name="login_success">Login Success</string>
    <string name="login_fail">Login Failed</string>

    <!-- Strings related to main menu -->
    <string name="start_ocr">Start Session</string>
    <string name="settings">Settings</string>
    <string name="logout">Log Out</string>

    <!-- Strings related to edit password -->
    <string name="current_password">Current Password</string>
    <string name="new_password">New Password</string>
    <string name="confirm_password">Confirm New Password</string>
    <string name="change_password">Change Password</string>
    <string name="cancel">Cancel</string>
    <string name="password_change_success">Password was successfully changed</string>
    <string name="password_change_fail">Password change failed</string>

    <!-- Strings related to CameraActivity -->
    <string name="no_infraction">No infraction occurred</string>
    <string name="infraction_occured">Infraction occurred. Details have been logged.</string>

    <!-- Strings related to settings -->
    <string name="advanced_settings">Advance Settings</string>

    <string name="no_network_1">No network connection detected</string>
    <string name="no_network_2">No network connection detected. Please check device
settings</string>
    <string name="request_credentials_1">Please enter a username and password</string>


    <!-- Debug stuff -->
    <string name="not_implemented">Feature currently unavailable</string>
    <string name="bad_json">Bad JSON format. Contact an admin.</string>
    <string name="connect_failed">Failed to connect to server</string>
    <string name="segment_failed">Segmentation failed</string>

    <!-- Settings -->
    <string name="process_settings"><b><b><b>Pre-processing Settings</b></b></b></string>

    <string name="blur_settings"><b><i>Blur Settings</i></b></string>
    <string name="kernel_width">Kernel width</string>
    <string name="kernel_height">Kernel height</string>
    <string name="distribution">Distribution</string>

    <string name="grey_settings"><b><i>Grey Settings</i></b></string>
    <string name="grey_mode">Grey Mode</string>

    <string name="thresh_settings"><b><i>Threshold Settings</i></b></string>
    <string name="lower_bound">Lower bound</string>
    <string name="upper_bound">Upper bound</string>

    <string name="morph_settings"><b><i>Image Morphology Settings</i></b></string>
```

```
    <string name="open_width">Open width</string>
    <string name="open_height">Open height</string>

    <string name="contour_settings"><b><i>Contour Settings</i></b></string>
    <string name="min_width">Minimum width</string>
    <string name="min_height">Minimum height</string>
    <string name="max_width">Maximum width</string>
    <string name="max_height">Maximum height</string>
    <string name="min_aspect">Minimum aspect</string>
    <string name="max_aspect">Maximum aspect</string>

    <string name="registration">"Registration: "</string>
    <string name="save_fail">Save Failed.</string>
    <string name="result_validate">Result Validation</string>
    <string name="number_plate">"Number Plate: "</string>
    <string name="edit_continue">Edit or Continue?</string>
    <string name="edit">Edit</string>
    <string name="continue_str">Continue</string>
    <string name="empty_field">One or more fields are empty</string>
    <string name="password_short">Password too short</string>
    <string name="password_no_match">New passwords don\'t match</string>
    <string name="password_same">New password cannot be the same as the current
password</string>


</resources>
```

2 Web Service

2.1 webapp.py

```
from flask import Flask, render_template, request, redirect, session
from flask import jsonify
from flask_restful import Resource, Api, reqparse
from functools import wraps
from werkzeug import check_password_hash, generate_password_hash

from mobile_api import LogRecord, AttemptLogin, EditPassword

import databasecontrol

""" Define the connection values for the database. """
DBconfig = {
                'HOST' : 'mikereid73.mysql.pythonanywhere-services.com',
                'USER' : 'mikereid73',
                'PASSWORD' : 'alongpassword',
                'DATABASE' : 'mikereid73$ProjectDB'
            }

app = Flask(__name__)
app.secret_key = '101010'
api = Api(app)

api.add_resource(LogRecord, '/post')
api.add_resource(AttemptLogin, '/login')
api.add_resource(EditPassword, '/change_password')

def check_is_logged_in(func):
    @wraps(func)
    def wrapped_function(*args, **kwargs):
        if session.get('logged_in') == True:
            return func(*args, **kwargs)
        return 'Please log in to view this page.'
    return wrapped_function

@app.route('/login')
def display_login_page() -> 'html':
    file = 'login.html'
    return render_template(file, the_title = 'Log In')

def authenticate_user(username, password):
    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL = "SELECT password FROM AdminStaff WHERE username=%s"
        cursor.execute(_SQL, (username,))
        data = cursor.fetchone()
        cursor.close()
        if data:
            encrypted_password = data[0]
            return check_password_hash(encrypted_password, password)
    return False

""" Attempts to log the user in. """
@app.route('/attempt_login', methods=['POST'])
def process_login() -> 'html':
    username = request.form['username']
    password = request.form['password']

    # Check if username/password supplied match those in DB.
    credential_match = authenticate_user(username, password)

    if credential_match:
        session['logged_in'] = True
        return redirect('/main_menu')
    return 'Username or password is incorrect.'
```

```
""" Direct the user to the logout page. """
""" Can't go here if not logged in. """
@app.route('/logout')
@check_is_logged_in
def display_logout_page() -> 'html':
    file = 'logout.html'
    return render_template(file, the_title='Log Out')

""" Attempts to log the user out. """
@app.route('/attempt_logout', methods=['POST'])
def process_logout() -> 'html':
    # Clear stored information from session
    session.clear()
    return redirect('/login')

@app.route('/')
def default_page():
    if session.get('logged_in') == True:
        return redirect('/main_menu')
    return redirect('/login')

@app.route('/main_menu')
@check_is_logged_in
def main_page():
    file = 'main_menu.html'
    return render_template(file, the_title = 'Dashboard')

@app.route('/register_new_vehicle')
@check_is_logged_in
def display_register_vehicle_page() -> 'html':
    file = 'register_vehicle.html'
    return render_template(file, the_title = 'Register New Vehicle')

@app.route('/attempt_register_vehicle', methods=['POST'])
@check_is_logged_in
def process_register_vehicle() -> 'html':
    first_name = request.form['first_name']
    last_name = request.form['last_name']
    vehicle_registration = request.form['vehicle_registration']
    email = request.form['email']

    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL =  "INSERT INTO MainCarpark "
        _SQL += "(last_name,first_name,vehicle_registration,email) "
        _SQL += "VALUES (%s,%s,%s,%s)"
        cursor.execute(_SQL, (last_name,first_name,vehicle_registration,email,))
        cursor.close()

    return render_template('register_vehicle_verify.html',
                           first_name=first_name,
                           last_name=last_name,
                           vehicle_registration=vehicle_registration,
                           email=email,
                           the_title='Register Vehicle Success')

@app.route('/view_all_vehicles', methods=['GET'])
@check_is_logged_in
def display_all_registered_vehicles() -> 'html':
    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL =  "Select first_name,last_name,vehicle_registration,email "
        _SQL += "FROM MainCarpark "
        _SQL += "ORDER by last_name"
        cursor.execute(_SQL,)
        all_vehicles = cursor.fetchall()
        cursor.close()
```

```
    return render_template('view_all_vehicles.html',
                           all_vehicles=all_vehicles,
                           the_title='View All Registered Vehicles')

@app.route('/register_new_staff')
@check_is_logged_in
def display_register_page() -> 'html':
    file = 'register_staff.html'
    return render_template(file, the_title = 'Register')

@app.route('/attempt_register', methods=['POST'])
@check_is_logged_in
def process_register() -> 'html':
    username = request.form['username']
    password = request.form['password']
    email = request.form['email']

    encrypted_password = generate_password_hash(password)
    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL =  "INSERT INTO Staff "
        _SQL += "(username, password,email) "
        _SQL += "VALUES (%s,%s,%s)"
        cursor.execute(_SQL, (username, encrypted_password,email,))
        cursor.close()

    return render_template('register_staff_verify.html',
                           username=username,
                           email=email,
                           the_title='Register Staff Success')

@app.route('/view_all_infractions', methods=['GET'])
@check_is_logged_in
def getAllInfractions():
    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL =  "SELECT
Infraction.vehicle_registration,last_name,first_name,email,time,logged_by "
        _SQL += "FROM Infraction "
        _SQL += "INNER JOIN MainCarpark "
        _SQL += "ON Infraction.vehicle_registration = MainCarpark.vehicle_registration "
        _SQL += "ORDER by Infraction.vehicle_registration, time"
        cursor.execute(_SQL,)
        all_data = cursor.fetchall()
        cursor.close()

    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL =  "SELECT
Infraction.vehicle_registration,last_name,first_name,email,time,logged_by "
        _SQL += "FROM Infraction "
        _SQL += "INNER JOIN MainCarpark "
        _SQL += "ON Infraction.vehicle_registration = MainCarpark.vehicle_registration "
        _SQL += "WHERE time >= CURDATE() "
        _SQL += "ORDER by vehicle_registration, time"
        cursor.execute(_SQL,)
        today_data = cursor.fetchall()
        cursor.close()

    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL =  "SELECT
Infraction.vehicle_registration,last_name,first_name,email,time,logged_by "
        _SQL += "FROM Infraction "
        _SQL += "INNER JOIN MainCarpark "
        _SQL += "ON Infraction.vehicle_registration = MainCarpark.vehicle_registration "
        _SQL += "WHERE time >= DATE_SUB(CURDATE(), INTERVAL DAYOFMONTH(CURDATE())-1 DAY) "
        _SQL += "ORDER by vehicle_registration, time"
        cursor.execute(_SQL,)
        month_data = cursor.fetchall()
        cursor.close()
```

```
    with databasecontrol.usedatabase(DBconfig) as cursor:
        _SQL =  "SELECT
Infraction.vehicle_registration,last_name,first_name,email,COUNT(Infraction.vehicle_registration
) "
        _SQL += "FROM Infraction,MainCarpark "
        _SQL += "WHERE Infraction.vehicle_registration=MainCarpark.vehicle_registration "
        _SQL += "GROUP by Infraction.vehicle_registration "
        _SQL += "HAVING COUNT(Infraction.vehicle_registration) > 1 "
        _SQL += "ORDER by COUNT(Infraction.vehicle_registration)"

        cursor.execute(_SQL,)
        repeat_data = cursor.fetchall()
        cursor.close()

    return render_template('view_infractions.html',
                           all_data=all_data,
                           today_data=today_data,
                           month_data=month_data,
                           repeat_data=repeat_data,
                           the_title='All Logged Infractions')

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

2.2 mobile_api.py

```
from flask import jsonify
from flask_restful import Resource, reqparse
from werkzeug import check_password_hash, generate_password_hash

import databasecontrol
from emailverifier import send_verification_email

""" Define the connection values for the database. """
DBconfig = {
                'HOST' : 'mikereid73.mysql.pythonanywhere-services.com',
                'USER' : 'mikereid73',
                'PASSWORD' : 'alongpassword',
                'DATABASE' : 'mikereid73$ProjectDB'
            }

class LogRecord(Resource):
    def post(self):
        parser = reqparse.RequestParser()
        parser.add_argument("guess")
        parser.add_argument("actual")
        parser.add_argument("username")

        args = parser.parse_args()
        guess = args["guess"]
        actual = args["actual"]
        logged_by = args["username"]

        with databasecontrol.usedatabase(DBconfig) as cursor:
            _SQL =  "INSERT INTO Log "
            _SQL += "(guess,actual,logged_by) "
            _SQL += "VALUES (%s,%s,%s)"
            cursor.execute(_SQL, (guess,actual,logged_by,))
            cursor.close()

        with databasecontrol.usedatabase(DBconfig) as cursor:
            _SQL =  "SELECT first_name,vehicle_registration,email "
            _SQL += "FROM MainCarpark "
            _SQL += "WHERE vehicle_registration=%s"
            cursor.execute(_SQL, (actual,))
            data = cursor.fetchone();
            cursor.close()

            if data:
                with databasecontrol.usedatabase(DBconfig) as cursor:
                    _SQL =  "INSERT INTO Infraction "
                    _SQL += "(vehicle_registration,logged_by) "
                    _SQL += "VALUES (%s,%s)"
                    cursor.execute(_SQL, (actual, logged_by,))
                    cursor.close()

                # send email: name, email, reg
                try:
                    name = data[0]
                    reg = data[1]
                    email = data[2]
                    send_verification_email(name, email, reg)
                except Exception:
                    return jsonify(registration=actual, infraction=True)

                return jsonify(registration=actual, infraction=True)
            return jsonify(registration=actual, infraction=False)



class AttemptLogin(Resource):
        def post(self):
```

```
            parser = reqparse.RequestParser()
            parser.add_argument("username")
            parser.add_argument("password")

            args = parser.parse_args()
            username = args["username"]
            password = args["password"]

            with databasecontrol.usedatabase(DBconfig) as cursor:
                _SQL =  "SELECT password "
                _SQL += "FROM Staff "
                _SQL += "WHERE username=%s"
                cursor.execute(_SQL, (username,))
                data = cursor.fetchone()
                cursor.close()

                if data and check_password_hash(data[0], password):
                    return jsonify(username=username, success=True)

                return jsonify(username=username, success=False)

class EditPassword(Resource):
    def post(self):
        parser = reqparse.RequestParser()
        parser.add_argument('username')
        parser.add_argument('old_password')
        parser.add_argument('new_password')

        args = parser.parse_args()
        username = args['username']
        old_password = args["old_password"]
        new_password = args["new_password"]

        with databasecontrol.usedatabase(DBconfig) as cursor:
            _SQL =  "SELECT password "
            _SQL += "FROM Staff "
            _SQL += "WHERE username=%s"
            cursor.execute(_SQL, (username,))
            data = cursor.fetchone();
            cursor.close()

            if data and not check_password_hash(data[0], old_password):
                reason = "Username or passsword do not match"
                return jsonify(username=username, success=False, reason=reason)

            if data and check_password_hash(data[0], new_password):
                reason = "New password is the same as the old password"
                return jsonify(username=username, success=False, reason=reason)

        encrypted_password = generate_password_hash(new_password)
        with databasecontrol.usedatabase(DBconfig) as cursor:
            _SQL =  "UPDATE Staff "
            _SQL += "SET password=%s "
            _SQL += "WHERE username=%s"
            cursor.execute(_SQL, (encrypted_password,username,))
            cursor.close()
            return jsonify(username=username, success=True)
```

2.3 emailverifier.py

```
import smtplib

from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText

def send_verification_email(name, email, reg):
    message = MIMEMultipart('alternative')
    html = """  <html>
                    <head>
                    <b><u>Important!</u></b><br><br>
                    </head>
                    <body>
                        """ + name + """,<br>

                        Your vehicle with registration """ + reg + """ has
                        been recorded as parking in the visitors carpark.

                        Please move to the main carpark as soon as possible.<br>

                        <br>Thanks,<br>
                        <b>Carlow IT Admin<br>
                    </body>
                </html>   """

    content = MIMEText(html, 'html')
    message.attach(content)
    message['Subject'] = 'Vehicle Reg: ' + reg

    mail = smtplib.SMTP('smtp.gmail.com',587)
    mail.ehlo()
    mail.starttls()
    mail.login('webappc00112726@gmail.com','random333')
    mail.sendmail('administrator.webappc00112726@gmail.com', email, message.as_string())
    mail.close()

    if __name__ == '__main__':
        send_verification_email()
databasecontrol.py
import mysql.connector

class usedatabase:
       def __init__(self, configuration:dict):
              self.host = configuration['HOST']
              self.user = configuration['USER']
              self.password = configuration['PASSWORD']
              self.database = configuration['DATABASE']

       def __enter__(self):
              self.conn = mysql.connector.connect( host = self.host,
                                                   user = self.user,
                                                                        password =
self.password,
                                                                        database =
self.database)
              self.cursor = self.conn.cursor()
              return self.cursor

       def __exit__(self, exc_type, exc_value,  exc_traceback):
              self.cursor.close()
              self.conn.commit()
              self.conn.close()
```

3 Web Page

3.1 base.html

```
<!DOCTYPE html>
<html>
    <head>
              <link type="text/css" rel="stylesheet" href="/static/css/stylesheet.css"/>
        <title>Plate Check</title>
        <link rel="shortcut icon" href="/static/image/favicon.ico" type="image/x-icon">
        <link rel="icon" href="/static/image/favicon.ico" type="image/x-icon">
      </head>
      <body>
            <div class="navbar">
                  <ul>
                        <li><a href="/view_all_vehicles">View All Registered Vehicles</a></li>
                        <li><a href="/register_new_vehicle">Register New Vehicle</a></li>
                        <li><a href="/register_new_staff">Add Staff</a></li>
                        <li><a href="/view_all_infractions">View Infractions</a></li>
                        <li><a href="/logout">Log Out</a></li>
                  </ul>
        </div>

        <h1>{{ the_title }}</h1>

        {% block body %}

        {% endblock %}

            <div id="footer">
                <div id="copy">
                   <br>
                   Copyright &copy; 2016 by Michael Reid
                   </div>
            </div>
    </body>
</html>
```

3.2 Login.html

```
{% extends "base.html" %}

{% block body %}

<div class="text">
    Please sign in using an admin username and password.
</div><br>
<body>
<form action="/attempt_login" method="POST" name="login" onsubmit="return validateForm()">
<div class="form">
<table>
    <tr>
        <td id="right">
            Username
        </td>
        <td>
            <input type="TEXT" name="username">
        </td>
    </tr>
    <tr>
        <td id="right">
            Password
        </td>
        <td>
            <input type="PASSWORD" name="password">
        </td>
    </tr>
</table>
</div><br>


<script type="text/javascript">
    function validateForm() {
        var username=document.forms["login"]["username"].value;
        var password=document.forms["login"]["password"].value;
        var uLength=username.length;
        var pLength=password.length;
        if (uLength < 3 || pLength < 3) {
            alert("Please Fill All Required Field");
            return false;
        }
    }
</script>
<div class="button">
    <input type="SUBMIT" value="Log In">
</div>
</form>
</body>

{% endblock %}
```

3.3 Logout.html

```
{% extends "base.html" %}

{% block body %}

<form action="/attempt_logout" method="POST">
<div class="text">
    You are now attempting to log out.
    To continue, please click the button below.
</div><br>
<div class="button">
    <input type="Submit" value = "Log Out">
</div>
</form>

{% endblock %}
```

3.4 Main_menu.html

```
{% extends "base.html" %}

{% block body %}

<h2><img src="/static/image/itcarlow.png"></h2>


{% endblock %}
```

Register_staff.html

```
{% extends "base.html" %}

{% block body %}
<body>
<div class="text">
    Please register using a unique username and password.
</div><br>
<div class="form">
<form action="/attempt_register" method="POST">
<table>
    <tr>
        <td  id="right">Username</td>
        <td><input type="TEXT" name="username"></td>
    </tr>
    <tr>
        <td  id="right">Password</td>
        <td><input type="PASSWORD" name="password"></td>
    </tr>
    <tr>
        <td  id="right">Email</td>
        <td><input type="TEXT" name="email"></td>
    </tr>
</table>

<br>
<div class="button">
    <input type="SUBMIT" value="Register">
</div>
</form>
</div>

{% endblock %}
</body>
```

3.5 Register_staff_verify.html

```
{% extends "base.html" %}

{% block body %}
<div class="form">
    <div class="text">
You are now registered.
    </div><br>
<table>
    <tr>
        <td id="right">
          Username:
        </td>
        <td id="left">
            {{ username }}
        </td>
    </tr>
    <tr>
        <td id="right">
            Email:
        </td>
        <td id="left">
            {{ email }}
        </td>

    </tr>
</table>

{% endblock %}
```

3.6 Register_vehicle.html

```
{% extends "base.html" %}

{% block body %}

<div class="text">
    Please enter details of the person and vehicle.
</div><br>
<div class="form">
<form action="/attempt_register_vehicle" method="POST">
<table>
    <tr>
        <td id="right">First Name</td>
        <td><input type="TEXT" name="first_name"></td>
    </tr>
    <tr>
        <td id="right">Surname</td>
        <td><input type="TEXT" name="last_name"></td>
    </tr>
    <tr>
        <td id="right">Vehicle Registration</td>
        <td><input type="TEXT" name="vehicle_registration"></td>
    </tr>
    <tr>
        <td id="right">Email</td>
        <td><input type="TEXT" name="email"></td>
    </tr>
</table>
<br>

<div class="button">
    <input type="SUBMIT" value="Register Vehicle">
</div>
</form>
</div>

{% endblock %}
```

3.7 Register_vehicle_verify.html

```
{% extends "base.html" %}

{% block body %}
<div class="text">
    Vehicle has been successfully registered.
</div><br>
<div class="form">
    <table>
        <tr>
            <td id="right">
               Owner Name:
            </td>
            <td id="left">
                {{ first_name + " " + last_name }}
            </td>
        </tr>
        <tr>
            <td id="right">
                Owner Email:
            </td>
            <td id="left">
                {{ email }}
            </td>
        </tr>
                <tr>
            <td id="right">
                Vehicle Rehistration:
            </td>
            <td id="left">
                {{ vehicle_registration }}
            </td>
        </tr>
    </table>

</div>

{% endblock %}
```

3.8 View_all_vehicles.html

```
<head>
    <title>
        {{ the_title }}
    </title>
</head>

{% extends "base.html" %}

{% block body %}

<div class="text">
    You are viewing all registered vehicles.</p>
</div>

<div class="infra_table">
<table>
    <thead>
        <th>First Name</th>
        <th>Last Name</th>
        <th>Vehicle Registration</th>
        <th>Email</th>
    </thead>
    {% for row in all_vehicles %}
    <tr>
        <td>{{row[0]}}</td>
        <td>{{row[1]}}</td>
        <td>{{row[2]}}</td>
        <td>{{row[3]}}</td>
    </tr>
    {% endfor %}
</table>
</div>


{% endblock %}
```

3.9 View_all_infractions.html

```
{% extends "base.html" %}

{% block body %}

<!-- No change things -->

<body>
<div class="infra_table">
    <title>{{ the_title }}</title>
       <div class="infra">
              <ul>
             <li><a href="#all_offenders" class="box">All Infractions</a></li>
                    <li><a href="#today_offenders" class="box">Today's Infractions</a></li>
                    <li><a href="#month_offenders" class="box">This Month's Infractions</a></li>
                    <li><a href="#repeat_offenders" class="box">Repeat Offenders</a></li>
              </ul>
       </div>

<h2><a name="all_offenders" id="infra_titles"><b>All Infractions</b><br></a></h2>

<table border = 0>
    <thead>
        <th>Vehicle Registration</th>
        <th>Owner</th>
        <th>Email</th>
        <th>Time</th>
        <th>Logged By</th>
    </thead>

    {% for row in all_data %}
    <tr>
        <td>{{row[0]}}</td>
        <td>{{row[2]}} {{row[1]}}</td>
        <td>{{row[3]}}</td>
        <td>{{row[4]}}</td>
        <td>{{row[5]}}</td>
    </tr>
    {% endfor %}
</table><br>


<h2><a name="today_offenders" class="infra_titles"><b>Today's Infractions</b></a></h2>
<table border = 0>
    <thead>
        <th>Vehicle Registration</td>
        <th>Owner</th>
        <th>Email</th>
        <th>Time</th>
        <th>Logged By</th>
    </thead>

    {% for row in today_data %}
    <tr>
        <td>{{row[0]}}</td>
        <td>{{row[2]}} {{row[1]}}</td>
        <td>{{row[3]}}</td>
        <td>{{row[4]}}</td>
        <td>{{row[5]}}</td>
    </tr>
    {% endfor %}
</table><br>

<h2><a name="month_offenders" class="infra_titles"><b>This month's Infractions</b></a></h2>
<table border = 0>
    <thead>
        <th>Vehicle Registration</td>
```

```
        <th>Owner</th>
        <th>Email</th>
        <th>Time</th>
        <th>Logged By</th>
    </thead>

    {% for row in month_data %}
    <tr>
        <td>{{row[0]}}</td>
        <td>{{row[2]}} {{row[1]}}</td>
        <td>{{row[3]}}</td>
        <td>{{row[4]}}</td>
        <td>{{row[5]}}</td>
    </tr>
    {% endfor %}
</table><br>

<h2><a name="repeat_offenders" class="infra_titles"><b>Repeat Offenders</b></a></h2>
<table border = 0>
    <thead>
        <th>Vehicle Registration</th>
        <th>Owner</th>
        <th>Email</th>
        <th>Number of Infractions</th>
    </thead>

    {% for row in repeat_data %}
    <tr>
        <td>{{row[0]}}</td>
        <td>{{row[2]}} {{row[1]}}</td>
        <td>{{row[3]}}</td>
        <td>{{row[4]}}</td>
    </tr>
    {% endfor %}
</table>
</div>
</body>

{% endblock %}
```

3.10 Stylesheet.css

```
body {
       background-color:#fff;
       color:#2B3856;
       min-width:1200;
}

.navbar {
       position: relative;
       top: -20px;
       height: 75px;
       background-color: #2B3856;
}

.navbar ul {
      margin: 0;
      padding: 0;
      text-align: center;

}

.navbar li {
       display: inline;
    list-style: none;
    padding: 0px 30px 0px 30px;

}

.navbar a {
       display:inline-block;
       text-decoration:none;
       text-transform:uppercase;
    font-family:Futura, Tahoma, sans-serif;
       color:#fff;
       text-align:center;
       margin-top:30px;
}

h1 {
       font-family: Verdana;
       font-weight: bold;
       text-align: center;
       padding-top: 25px;
       padding-bottom: 25px;
}

#footer{
       clear:both;
       position: relative;
       bottom: -30px;
       height: 75px;
       background-color: #2B3856;
}

#copy {
       text-align:center;
       font-family: Futura, Tahoma, sans-serif;
       color:#fff;
}



.infra {
       height: 150px;
    position:static;
}
```

```
.infra ul {
      text-align: center;
}

.infra li {
       display: inline;
    list-style: none;
    padding: 0px 20px 0px 20px;

}

.infra a {
       display:inline-block;
       text-decoration:none;
       text-transform:uppercase;
    font-family:Futura, Tahoma, sans-serif;
       color:#000;
       text-align:center;
       margin-top:15px;
}

.box {
  padding: 2em;
  height: 25px;
  border-radius:25px;
  margin: -1px 0 0 -1px;
  color: black;
  background: linear-gradient(
     #dfe4ef, #dfe4ef 50%, #2B3856 50%, #2B3856
  );
  background-size: 100% 202%;
  transition: all 0.4s ease;
}

.box:hover {
  background-position: 100% 100%;
  color:#fff;
}


h2{
    text-transform:uppercase;
    font-size:22px;
    font-family:Verdana;
    color:#2B3856;
    text-align:center;
}

.infra_table table {
    position:relative;
    overflow:auto;
    margin:0 auto;
    text-align:center;
    width:800px;
}

.infra_table thead {
    background-color:#2B3856;
    color:#fff;
}

.infra_table tr:nth-child(even) {
       background-color: #dfe4ef;
       }

.form table {
```

```
    position:relative;
    overflow:auto;
    margin:0 auto;
    text-align:center;
    width:250px;
}

#right {
    text-align:right;
    font-family:Futura, Tahoma, sans-serif;
}
#left {
    text-align:left;
    font-family:Futura, Tahoma, sans-serif;
}

.text {
    font-family:Futura, Tahoma, sans-serif;
    text-align:center;
}

.button {
    text-align:center;
}
```