CSE 162 Mobile Computing

Face Detection

Hua Huang
Department of Computer Science and Engineering
University of California, Merced, CA

Goal

Familiarize with the Android ML Kit

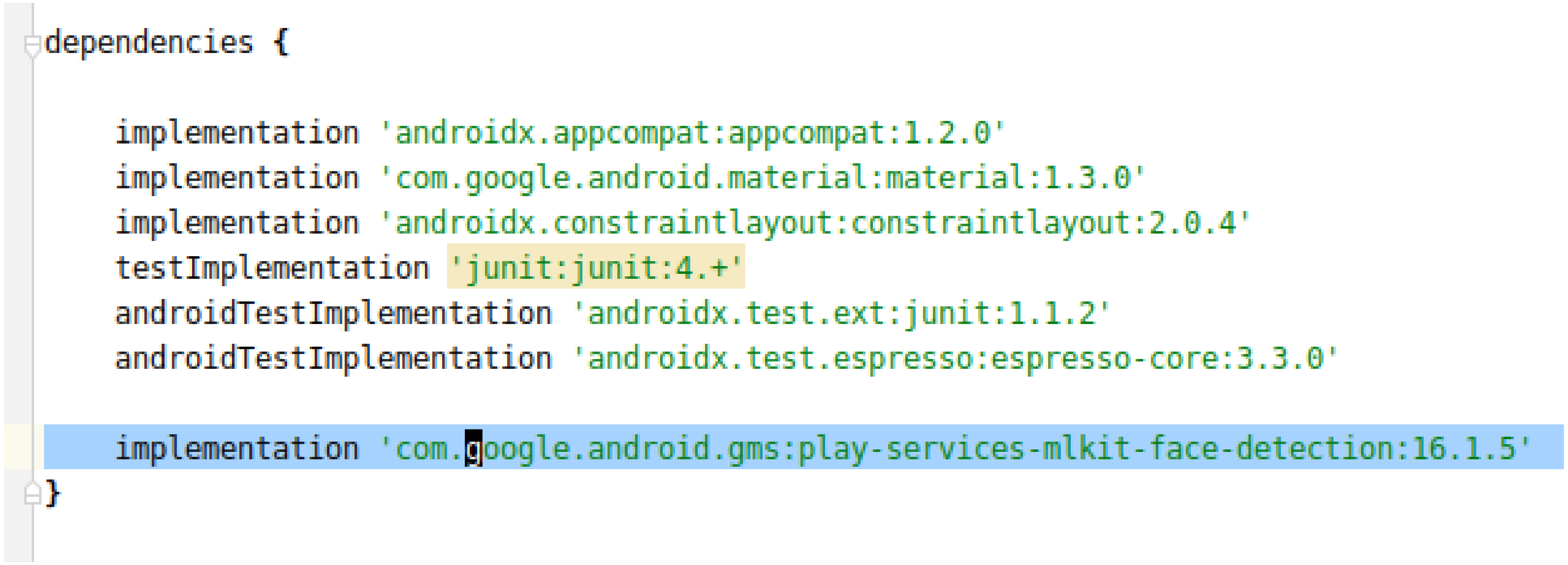
Detect face in images

Feature

- Display an image
- Use ML Kit to find the face
- Highlight the face in the image

Setup the dependency

- In the app/build.gradle
- implementation 'com.google.android.gms:play-services-mlkit-face-detection:16.1.5'



```
dependencies {

    implementation 'androidx.appcompat:appcompat:1.2.0'
    implementation 'com.google.android.material:material:1.3.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.0.4'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.2'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.3.0'

    implementation 'com.google.android.gms:play-services-mlkit-face-detection:16.1.5'
}
```

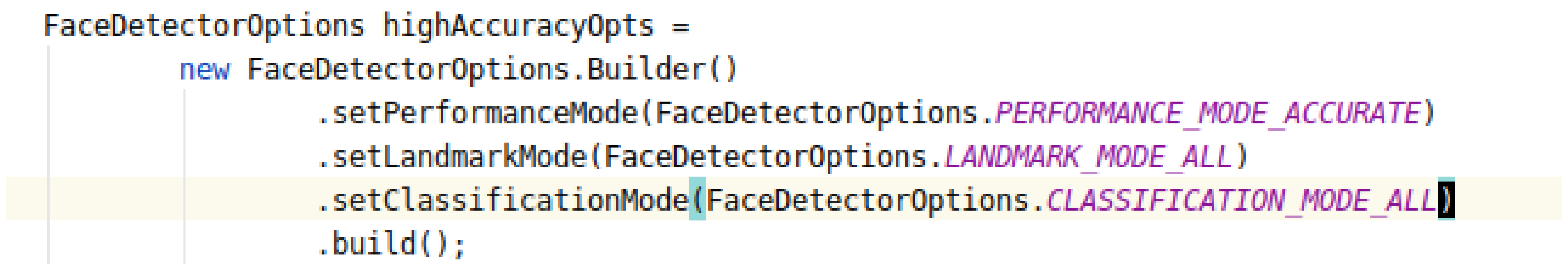
In the manifest.xml

- configure the app to automatically download the model to the device after theapp is installed

```
<application ...>
...
  <meta-data
     android:name="com.google.mlkit.vision.DEPENDENCIES"
     android:value="face" />
  <!-- To use multiple models: android:value="face,model2,model3" -->
  </application>
```

Prepare for the image detection

- In MainActivity.java onCreate()
- configure the detection options

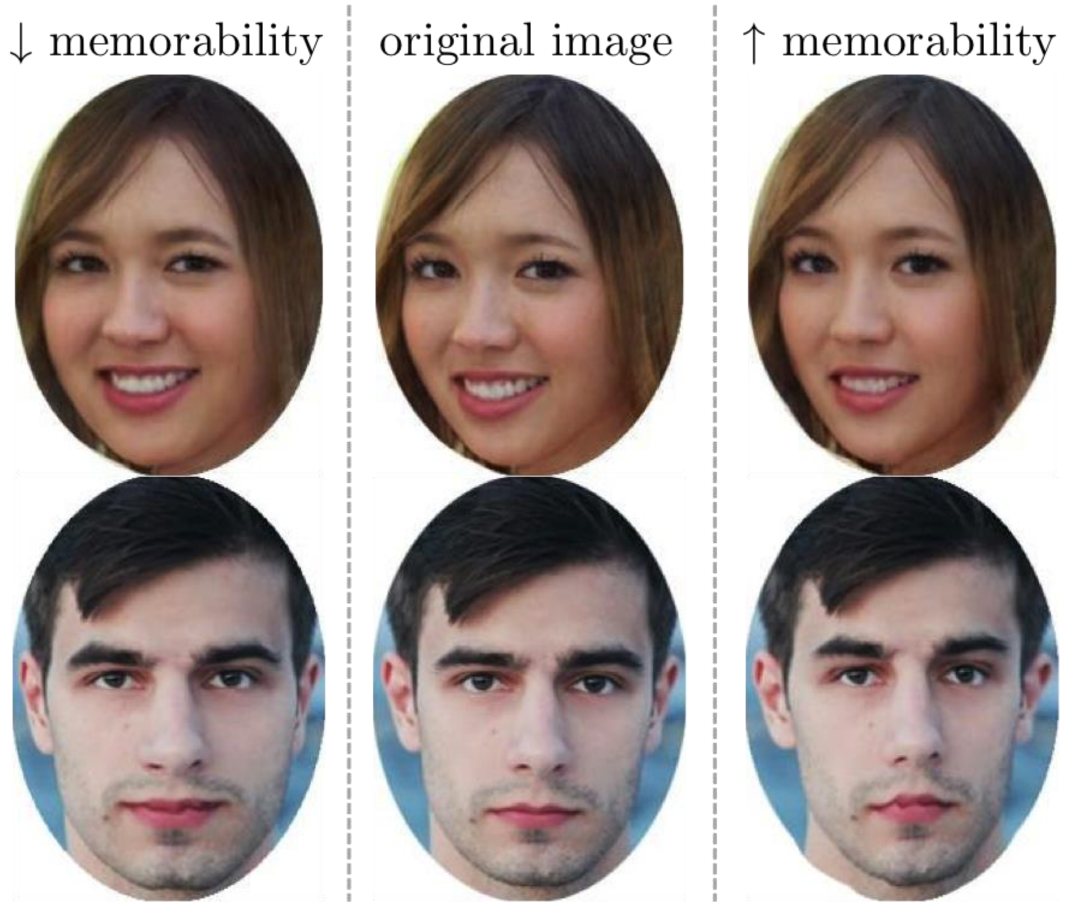


```
FaceDetectorOptions highAccuracyOpts =
        new FaceDetectorOptions.Builder()
                .setPerformanceMode(FaceDetectorOptions.PERFORMANCE_MODE_ACCURATE)
                .setLandmarkMode(FaceDetectorOptions.LANDMARK_MODE_ALL)
                .setClassificationMode(FaceDetectorOptions.CLASSIFICATION_MODE_ALL)
                .build();
```

Prepare the image for processing

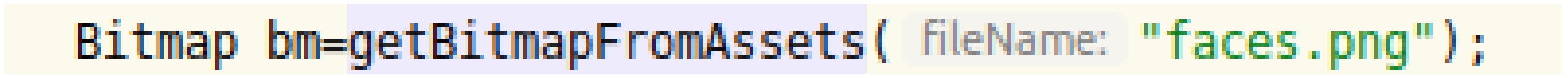
- place the image file into the path: FaceDetector/app/src/main/assets/
- create the folder if needed.
- The app has access to the files in this folder.

sample image

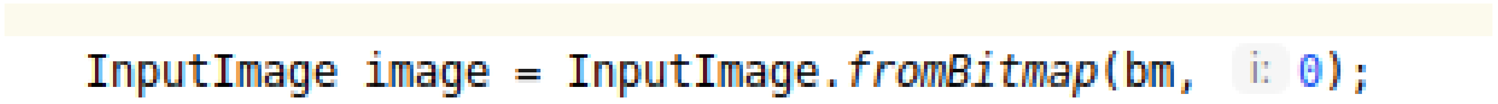


In MainActivity.java

- read the image
- Convert the image to the appropriate format using InputImage object



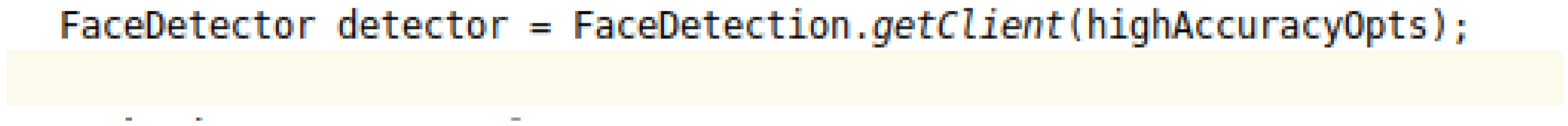
```
Bitmap bm=getBitmapFromAssets( fileName: "faces.png");
```



```
InputImage image = InputImage.fromBitmap(bm, i: 0);
```

```
private Bitmap getBitmapFromAssets(String fileName){

    AssetManager am = getAssets();
    InputStream is = null;
    try{

        is = am.open(fileName);
    }catch(IOException e){
        e.printStackTrace();
    }
    Bitmap bitmap = BitmapFactory.decodeStream(is);
    return bitmap;
}
```

Get an instance of FaceDetector



```
FaceDetector detector = FaceDetection.getClient(highAccuracyOpts);
```

Process the image

```
Task<List<Face>> result =
        detector.process(image)
                .addOnSuccessListener(
                        new OnSuccessListener<List<Face>>() {
                            @Override
                            public void onSuccess(List<Face> faces) {
                                // Task completed successfully
                                // ...
                            }
                        })
                .addOnFailureListener(
                        new OnFailureListener() {
                            @Override
                            public void onFailure(@NonNull Exception e) {
                                // Task failed with an exception
                                // ...
                            }
                        });
```

Get the detection result

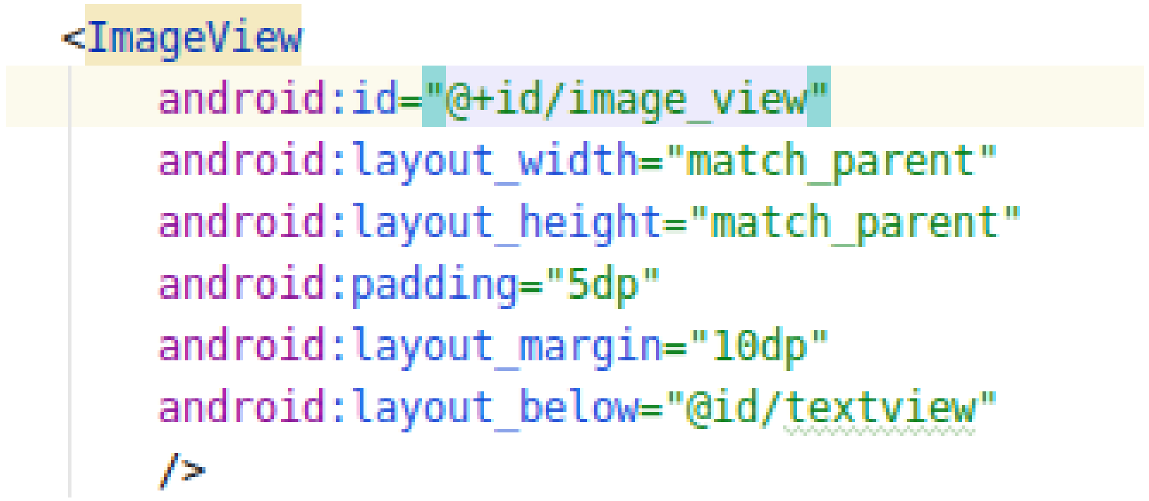
- If the face detection operation succeeds, a list of Face objects are passed to the success listener.
- Each Face object represents a face that was detected in the image.
- For each face, you can get its bounding coordinates in the input image, as well as any other information you configured the face detector to find.
- In this lab, we want to plot a rectangle for each face.

Many facial features can be detected. We focus on foundingbox of the face

```
Rect bounds = face.getBoundingBox();
float rotY = face.getHeadEulerAngleY();  // Head is rotated to the right rotY degrees
float rotZ = face.getHeadEulerAngleZ();  // Head is tilted sideways rotZ degrees
// If landmark detection was enabled (mouth, ears, eyes, cheeks, and
// nose available):
FaceLandmark leftEar = face.getLandmark(FaceLandmark.LEFT_EAR);
if (leftEar != null) {
    PointF leftEarPos = leftEar.getPosition();
}
// If contour detection was enabled:
List<PointF> leftEyeContour =
        face.getContour(FaceContour.LEFT_EYE).getPoints();
List<PointF> upperLipBottomContour =
        face.getContour(FaceContour.UPPER_LIP_BOTTOM).getPoints();
```

Display the results

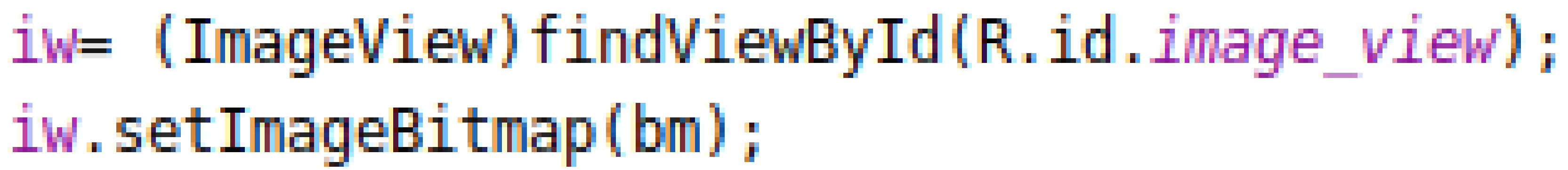
- In activity_main.xml, add imageview



```xml
<ImageView
    android:id="@+id/image_view"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="5dp"
    android:layout_margin="10dp"
    android:layout_below="@id/textview"
    />
```

In onCreate()

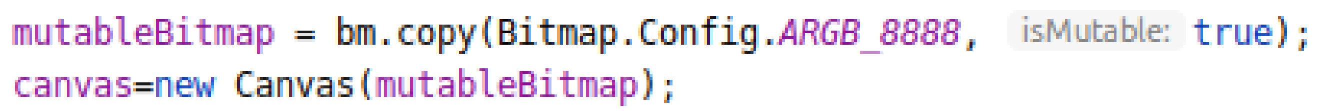
- display the image



```
iw= (ImageView)findViewById(R.id.image_view);
iw.setImageBitmap(bm);
```

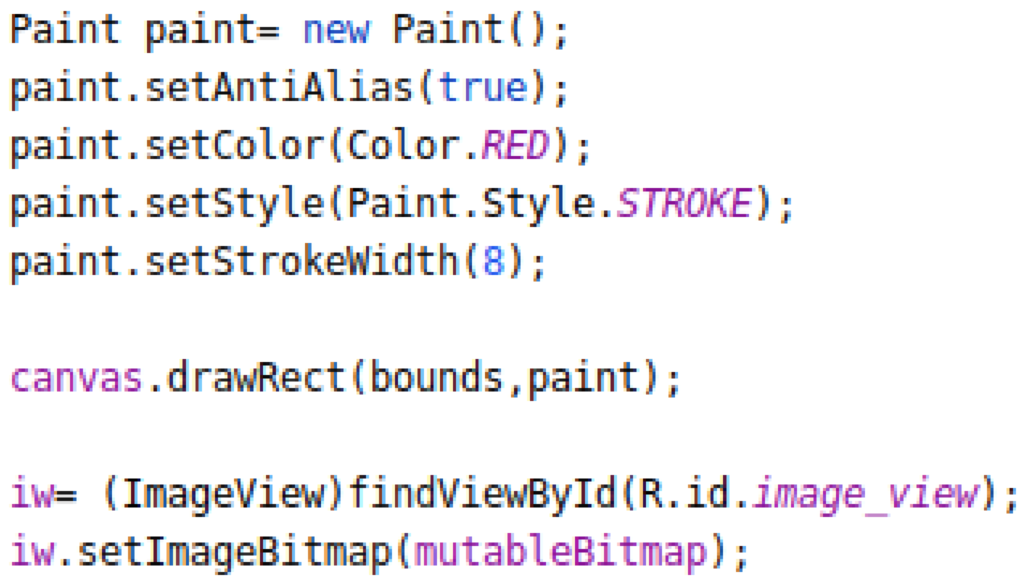
Use Canvas to draw the detection box

- in onCreate(), create a copy of the face image to draw upon



```
mutableBitmap = bm.copy(Bitmap.Config.ARGB_8888,  isMutable: true);
canvas=new Canvas(mutableBitmap);
```

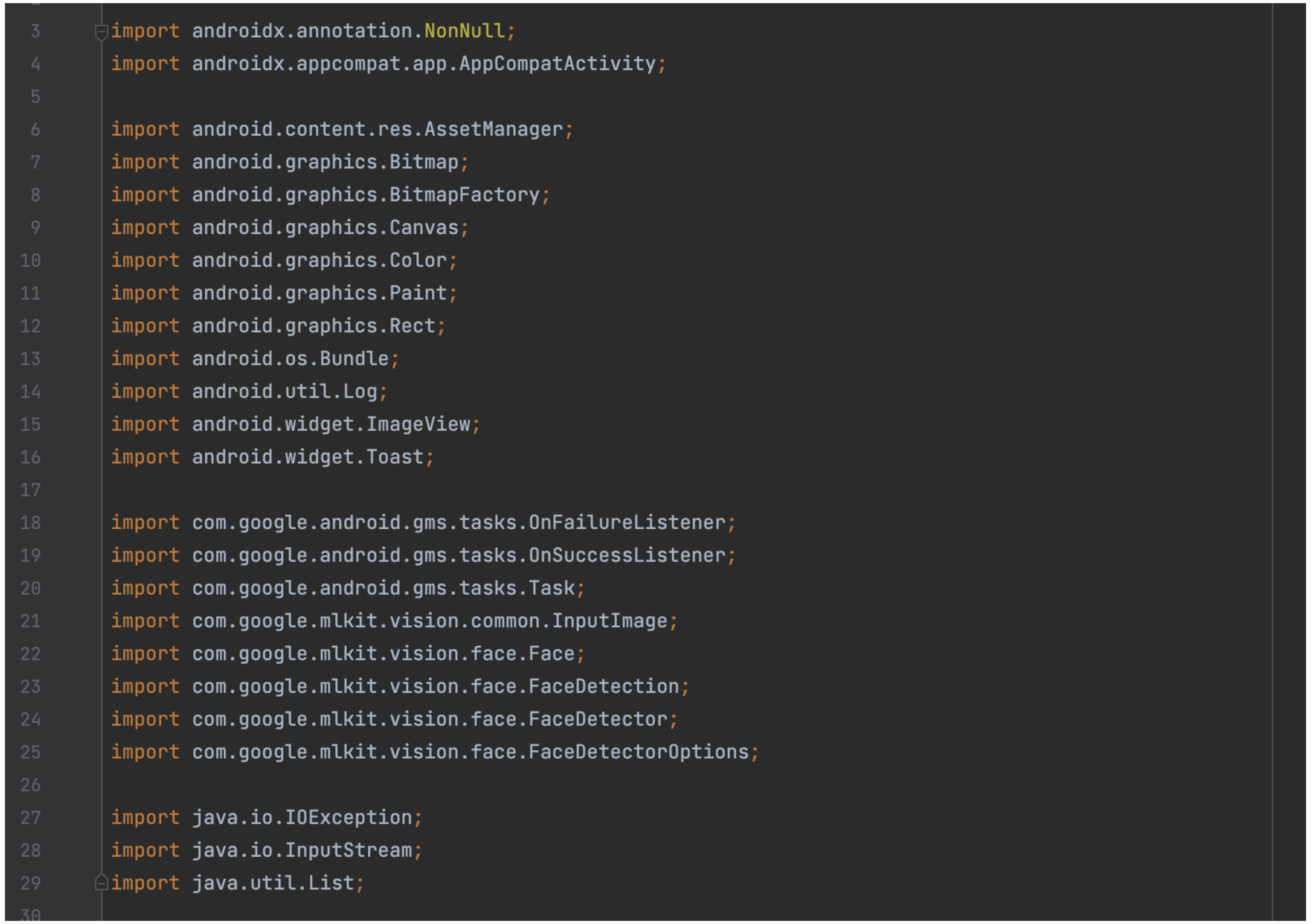
- When the face detection is successful, use the canvas to draw the detection boxes.



```
Paint paint= new Paint();
paint.setAntiAlias(true);
paint.setColor(Color.RED);
paint.setStyle(Paint.Style.STROKE);
paint.setStrokeWidth(8);

canvas.drawRect(bounds,paint);

iw= (ImageView)findViewById(R.id.image_view);
iw.setImageBitmap(mutableBitmap);
```

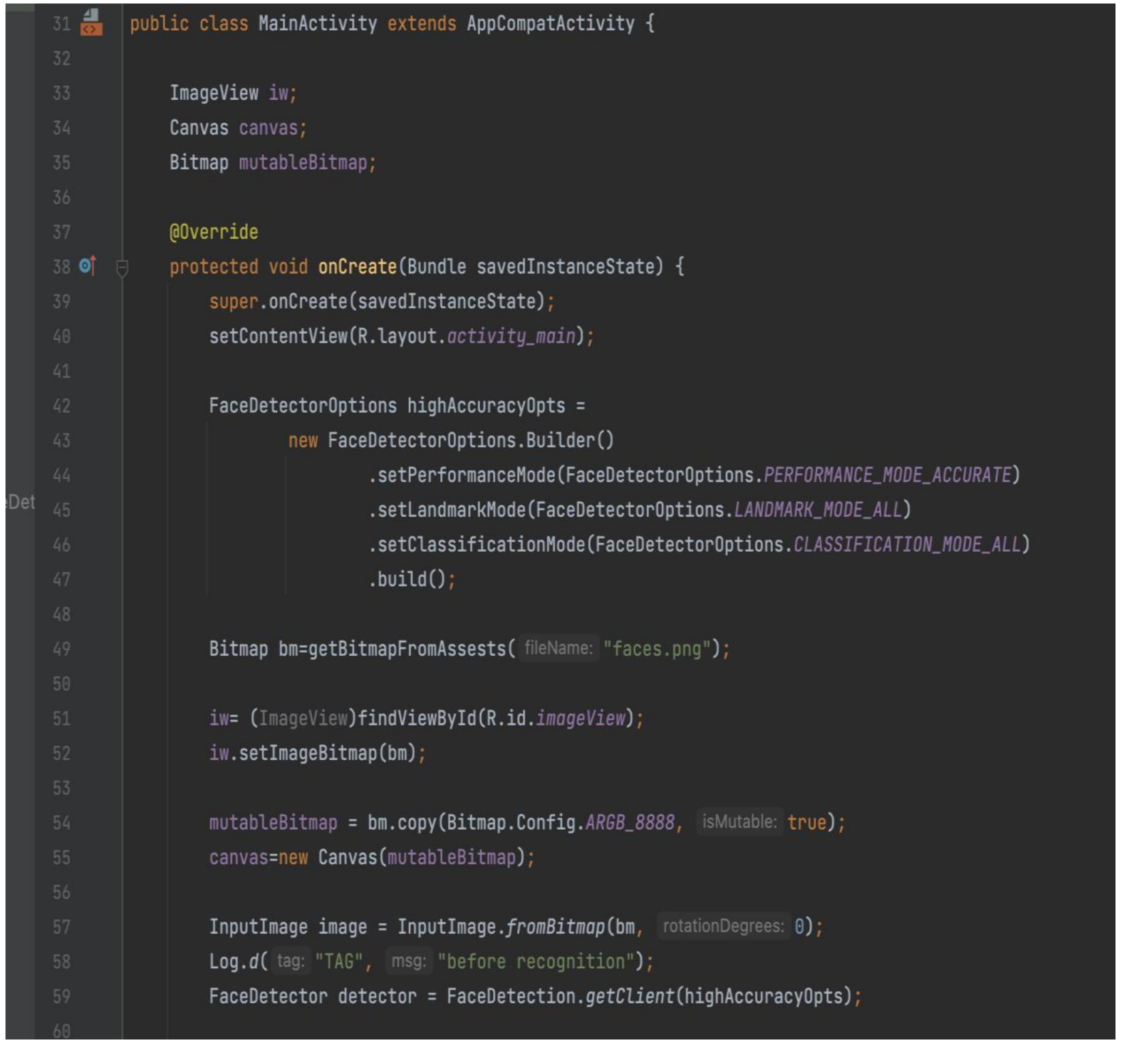
MainActivity. java

- Import section



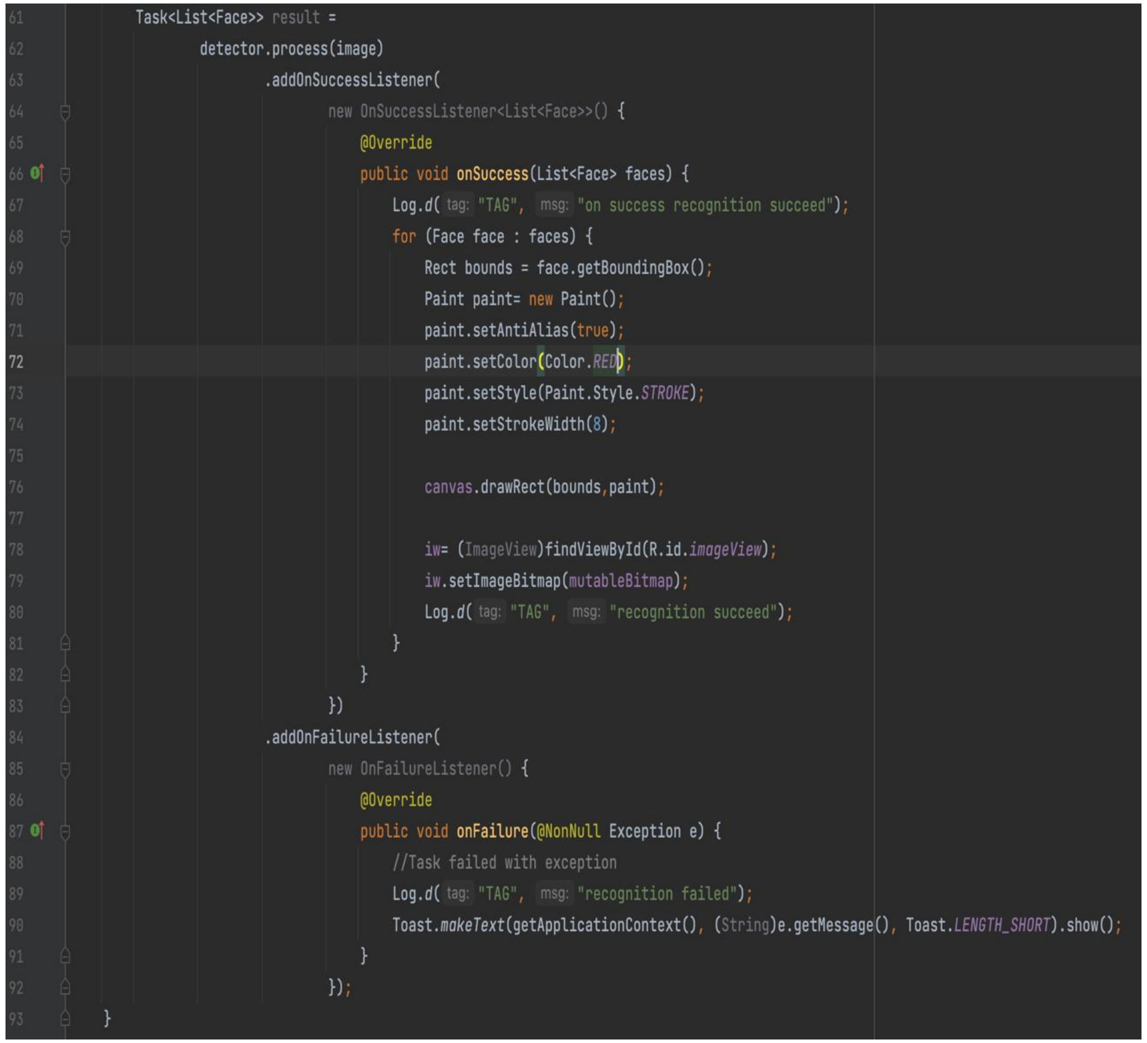
```java
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.res.AssetManager;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Rect;
import android.os.Bundle;
import android.util.Log;
import android.widget.ImageView;
import android.widget.Toast;

import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import com.google.mlkit.vision.common.InputImage;
import com.google.mlkit.vision.face.Face;
import com.google.mlkit.vision.face.FaceDetection;
import com.google.mlkit.vision.face.FaceDetector;
import com.google.mlkit.vision.face.FaceDetectorOptions;

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

MainActivity. java



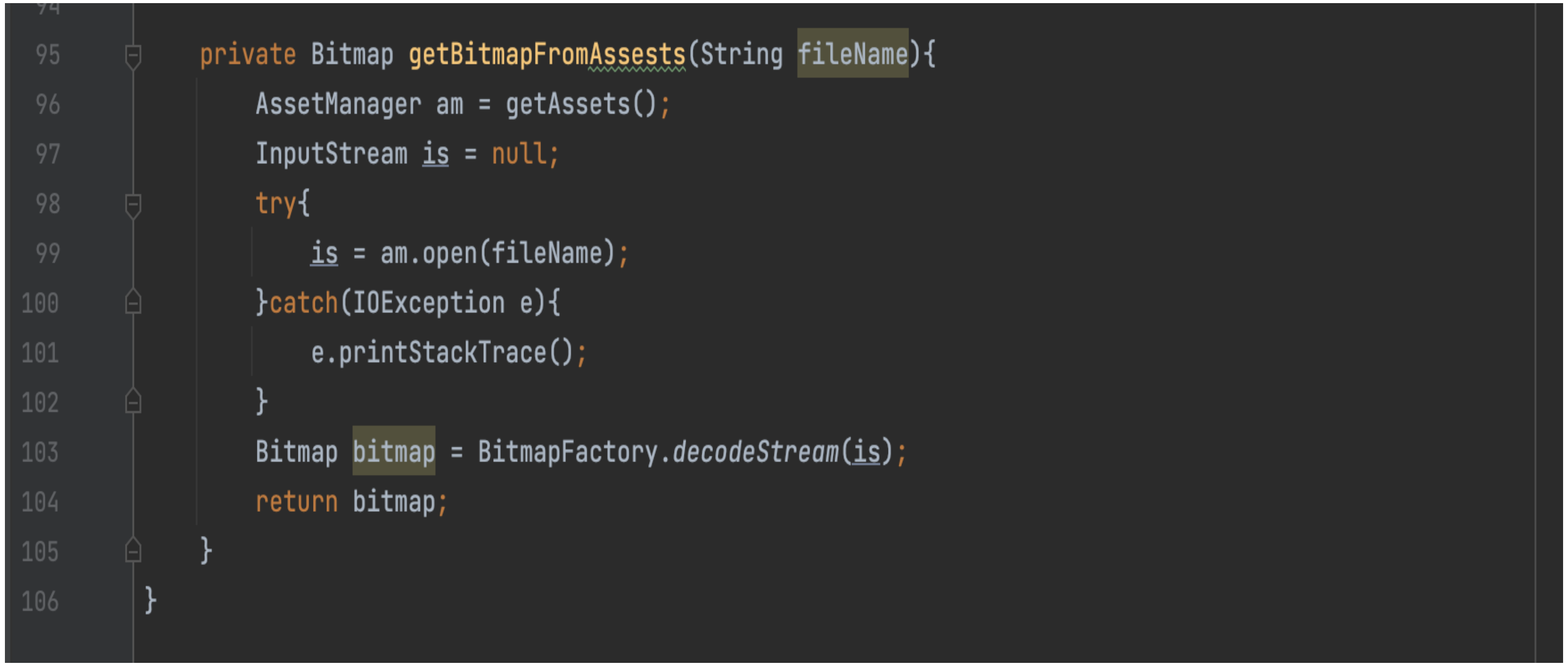
```
public class MainActivity extends AppCompatActivity {

    ImageView iw;
    Canvas canvas;
    Bitmap mutableBitmap;

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

        FaceDetectorOptions highAccuracyOpts =
                new FaceDetectorOptions.Builder()
                        .setPerformanceMode(FaceDetectorOptions.PERFORMANCE_MODE_ACCURATE)
                        .setLandmarkMode(FaceDetectorOptions.LANDMARK_MODE_ALL)
                        .setClassificationMode(FaceDetectorOptions.CLASSIFICATION_MODE_ALL)
                        .build();

        Bitmap bm=getBitmapFromAssests( fileName: "faces.png");

        iw= (ImageView)findViewById(R.id.imageView);
        iw.setImageBitmap(bm);

        mutableBitmap = bm.copy(Bitmap.Config.ARGB_8888,  isMutable: true);
        canvas=new Canvas(mutableBitmap);

        InputImage image = InputImage.fromBitmap(bm,  rotationDegrees: 0);
        Log.d( tag: "TAG",  msg: "before recognition");
        FaceDetector detector = FaceDetection.getClient(highAccuracyOpts);

```

Main activity 1



```
        Task<List<Face>> result =
                detector.process(image)
                        .addOnSuccessListener(
                                new OnSuccessListener<List<Face>>() {
                                    @Override
                                    public void onSuccess(List<Face> faces) {
                                        Log.d( tag: "TAG",  msg: "on success recognition succeed");
                                        for (Face face : faces) {
                                            Rect bounds = face.getBoundingBox();
                                            Paint paint= new Paint();
                                            paint.setAntiAlias(true);
                                            paint.setColor(Color.RED);
                                            paint.setStyle(Paint.Style.STROKE);
                                            paint.setStrokeWidth(8);

                                            canvas.drawRect(bounds,paint);

                                            iw= (ImageView)findViewById(R.id.imageView);
                                            iw.setImageBitmap(mutableBitmap);
                                            Log.d( tag: "TAG",  msg: "recognition succeed");
                                        }
                                    }
                                })
                        .addOnFailureListener(
                                new OnFailureListener() {
                                    @Override
                                    public void onFailure(@NonNull Exception e) {
                                        //Task failed with exception
                                        Log.d( tag: "TAG",  msg: "recognition failed");
                                        Toast.makeText(getApplicationContext(), (String)e.getMessage(), Toast.LENGTH_SHORT).show();
                                    }
                                });
    }
```

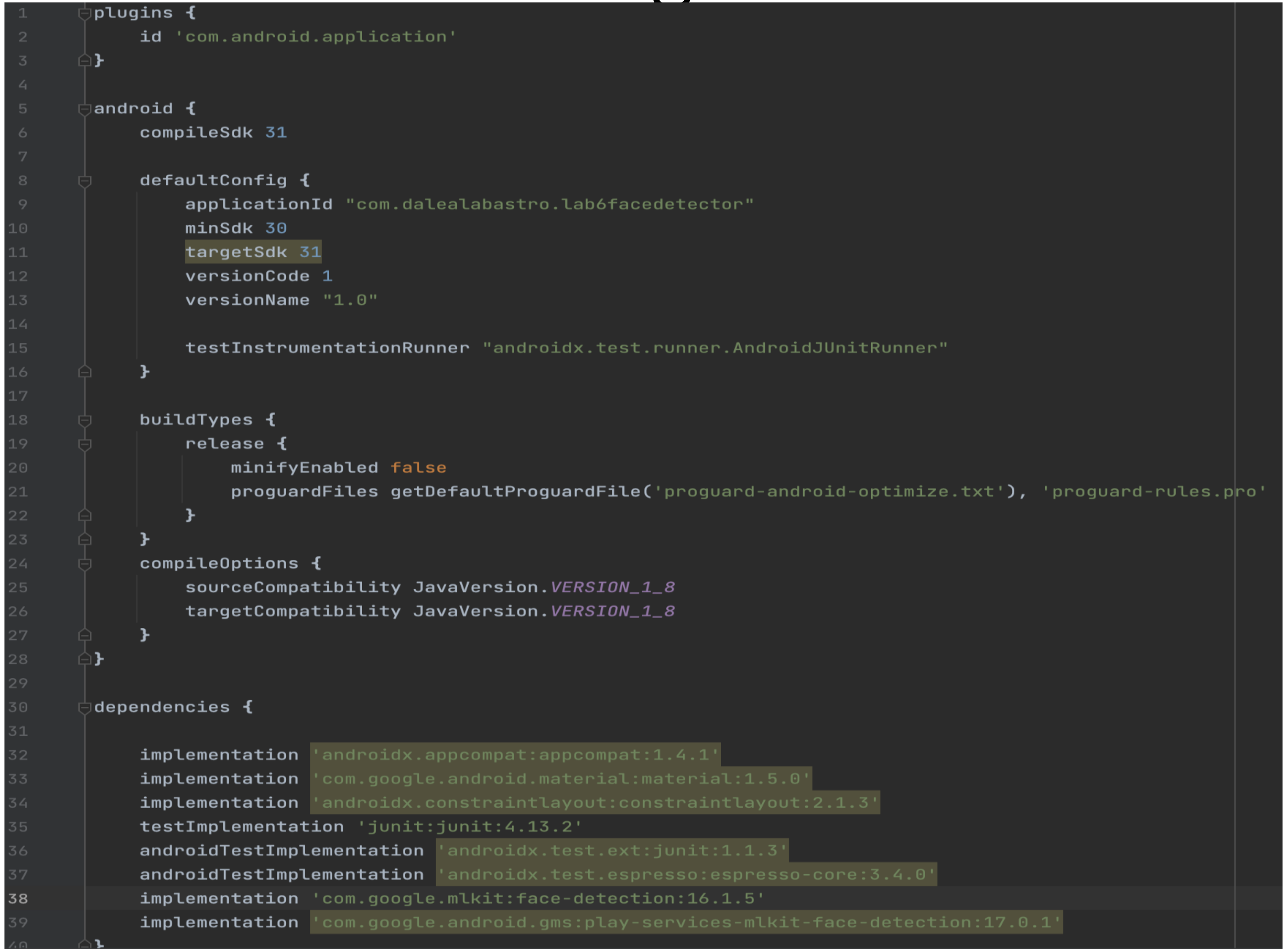
Main activity 2



```
    private Bitmap getBitmapFromAssests(String fileName){
        AssetManager am = getAssets();
        InputStream is = null;
        try{
            is = am.open(fileName);
        }catch(IOException e){
            e.printStackTrace();
        }
        Bitmap bitmap = BitmapFactory.decodeStream(is);
        return bitmap;
    }
}
```

Main Activity 3

Build gradle



```
plugins {
    id 'com.android.application'
}

android {
    compileSdk 31

    defaultConfig {
        applicationId "com.dalealabastro.lab6facedetector"
        minSdk 30
        targetSdk 31
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.4.1'
    implementation 'com.google.android.material:material:1.5.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.3'
    testImplementation 'junit:junit:4.13.2'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
    implementation 'com.google.mlkit:face-detection:16.1.5'
    implementation 'com.google.android.gms:play-services-mlkit-face-detection:17.0.1'
}
```