20. ML Kit for Firebase

Firebase ML Kit

- Announced by Google at I/O 2018
- Contains prebuilt APIs which provide ML functionalities for Android and iOS apps
- Support for running custom models

Reducing App Development Time

- An app/idea that needs machine learning features
  - Basic knowledge of Python and Tensorflow (for custom models)
  - Basics of App Development

- Features Off-the-shelf
  - Text Recognition
  - Face Detection
  - Barcode Scanning
  - Image Labeling
  - Landmark Recognition
  - Hosting custom Tensorflow models

Types of APIs

- On Device
  - Free
  - Runs without Internet
  - Low Accuracy

- Cloud
  - Paid (free for first 1000 calls per month)
  - Needs Internet connectivity
  - High Accuracy

1. Gradle (app)

```
dependencies {

      …

    implementation 'com.google.firebase:firebase-ml-vision:18.0.1'
    implementation 'com.google.firebase:firebase-ml-vision-image-label-model:17.0.2'
    implementation 'com.google.firebase:firebase-ml-vision-face-model:17.0.2'
    implementation 'com.google.firebase:firebase-ml-model-interpreter:16.2.3'
}
apply plugin: 'com.google.gms.google-services'
```

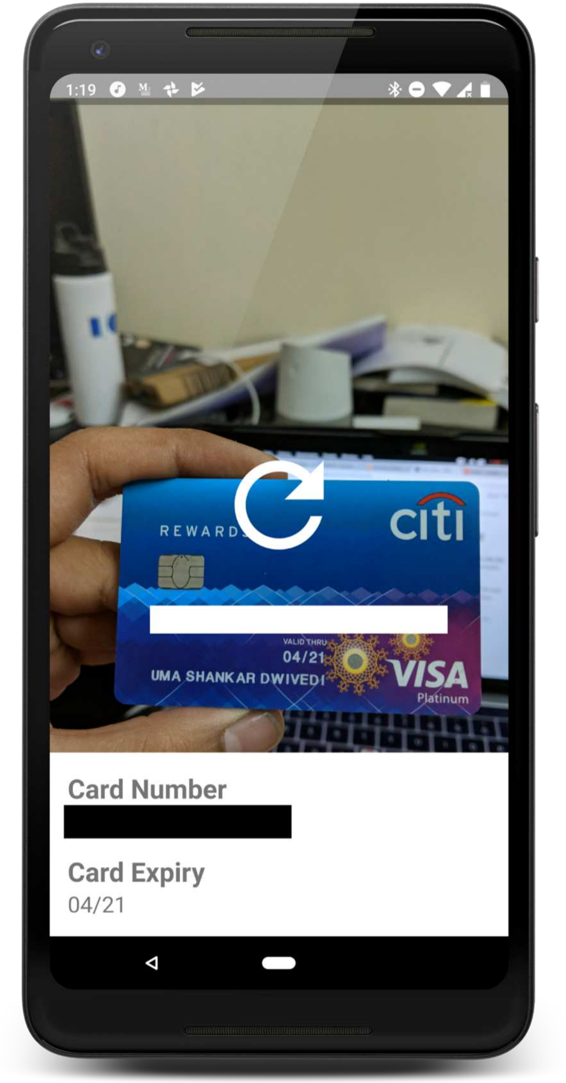
2. Gradle (Project), 3. JSON

```
dependencies {
    …
    classpath 'com.google.gms:google-services:3.2.0' // google-services plugin

}
```

Download google-services.json and place in app directory

Text Recognition

- Extract text from images
- Cloud and On Device APIs available



Card Number

[redacted]

Card Expiry

04/21

Codelab

- Codelab link on Website
- Complete this codelab in parallel with slides

Recognize text in images

```
FirebaseVisionImage image = FirebaseVisionImage.fromBitmap(mSelectedImage);
FirebaseVisionTextRecognizer recognizer = FirebaseVision.getInstance()
        .getOnDeviceTextRecognizer();
recognizer.processImage(image)
        .addOnSuccessListener(
                new OnSuccessListener<FirebaseVisionText>() {
                    @Override
                    public void onSuccess(FirebaseVisionText texts) {
                        mTextButton.setEnabled(true);
                        processTextRecognitionResult(texts);
                    }
                })
        .addOnFailureListener(
                new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {
                        // Task failed with an exception
                    }
                });
```

Extract text from blocks

```
String resultText = result.getText();
for (FirebaseVisionText.TextBlock block: result.getTextBlocks()) {
  String blockText = block.getText();
  Float blockConfidence = block.getConfidence();
  List<RecognizedLanguage> blockLanguages = block.getRecognizedLanguages();
  Point[] blockCornerPoints = block.getCornerPoints();
  Rect blockFrame = block.getBoundingBox();
  for (FirebaseVisionText.Line line: block.getLines()) {
    String lineText = line.getText();
    Float lineConfidence = line.getConfidence();
    List<RecognizedLanguage> lineLanguages = line.getRecognizedLanguages();
    Point[] lineCornerPoints = line.getCornerPoints();
    Rect lineFrame = line.getBoundingBox();
    for (FirebaseVisionText.Element element: line.getElements()) {
      String elementText = element.getText();
      Float elementConfidence = element.getConfidence();
      List<RecognizedLanguage> elementLanguages = element.getRecognizedLanguages();
      Point[] elementCornerPoints = element.getCornerPoints();
      Rect elementFrame = element.getBoundingBox();
    } } }
```

FirebaseVisionText.TextBlock

- A block of text (think of it as a paragraph) as deemed by the OCR engine.

Rect	getBoundingBox()Returns the axis-aligned bounding rectangle of the detected text.
Float	getConfidence()The confidence of the recognized text.
Point[]	getCornerPoints()Gets the four corner points in clockwise direction starting with top-left.
synchronized List<FirebaseVisionText.Line>	getLines()Gets an unmodifiable list of FirebaseVisionText.Lines that make up this text block.
List<RecognizedLanguage>	getRecognizedLanguages()Gets a list of recognized languages together with confidence.
String	getText()Gets the recognized text as a string.

FirebaseVisionText.Line

- Represents a line of text.

Rect	getBoundingBox()Returns the axis-aligned bounding rectangle of the detected text.
Float	getConfidence()The confidence of the recognized text.
Point[]	getCornerPoints()Gets the four corner points in clockwise direction starting with top-left.
synchronized List<FirebaseVisionText.Element>	getElements()Gets a unmodifiable list of FirebaseVisionText.Elements that make up this text line.
List<RecognizedLanguage>	getRecognizedLanguages()Gets a list of recognized languages together with confidence.
String	getText()Gets the recognized text as a string.

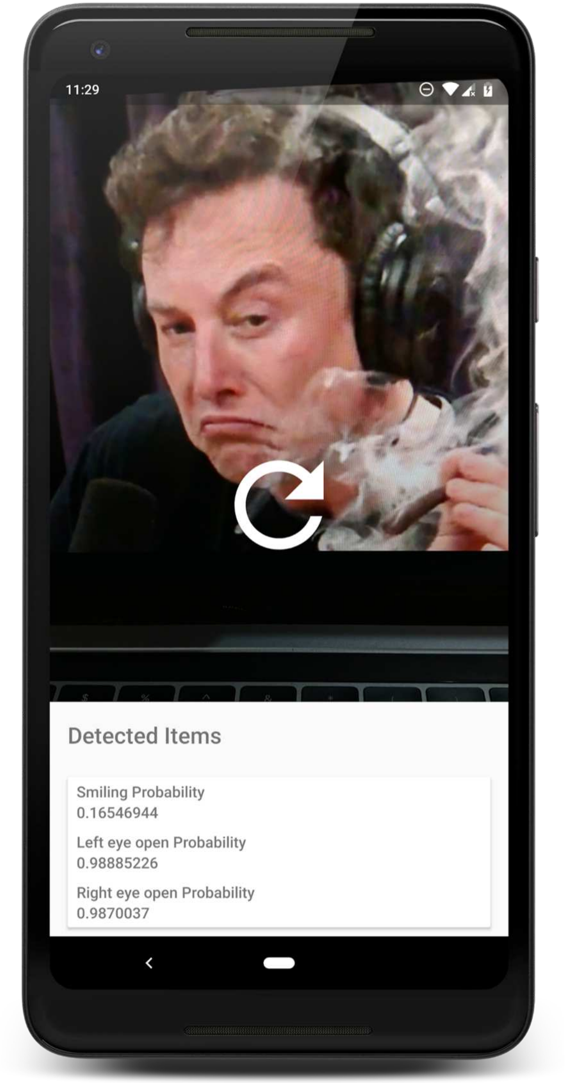
FirebaseVisionText.Element

- Roughly equivalent to a space-separated "word" in most Latin languages, or a character in others.

Rect	getBoundingBox()Returns the axis-aligned bounding rectangle of the detected text.
Float	getConfidence()The confidence of the recognized text.
Point[]	getCornerPoints()Gets the four corner points in clockwise direction starting with top-left.
List <RecognizedLanguage>	getRecognizedLanguages()Gets a list of recognized languages together with confidence.
String	getText()Gets the recognized text as a string.

Face Detection

- Detect faces and facial landmarks
- Only on Device API available



Detected Items

Smiling Probability
0.16546944

Left eye open Probability
0.98885226

Right eye open Probability
0.9870037

Configure the face detector

```
// High-accuracy landmark detection and face classification
    FirebaseVisionFaceDetectorOptions highAccuracyOpts =
    new FirebaseVisionFaceDetectorOptions.Builder()
    .setPerformanceMode(FirebaseVisionFaceDetectorOptions.ACCURATE)
    .setLandmarkMode(FirebaseVisionFaceDetectorOptions.ALL_LANDMARKS)
    .setClassificationMode(FirebaseVisionFaceDetectorOptions.ALL_CLASSIFICATIONS)
    .build();

// Real-time contour detection of multiple faces
    FirebaseVisionFaceDetectorOptions realTimeOpts =
    new FirebaseVisionFaceDetectorOptions.Builder()
    .setContourMode(FirebaseVisionFaceDetectorOptions.ALL_CONTOURS)
    .build();
```

Detect Image

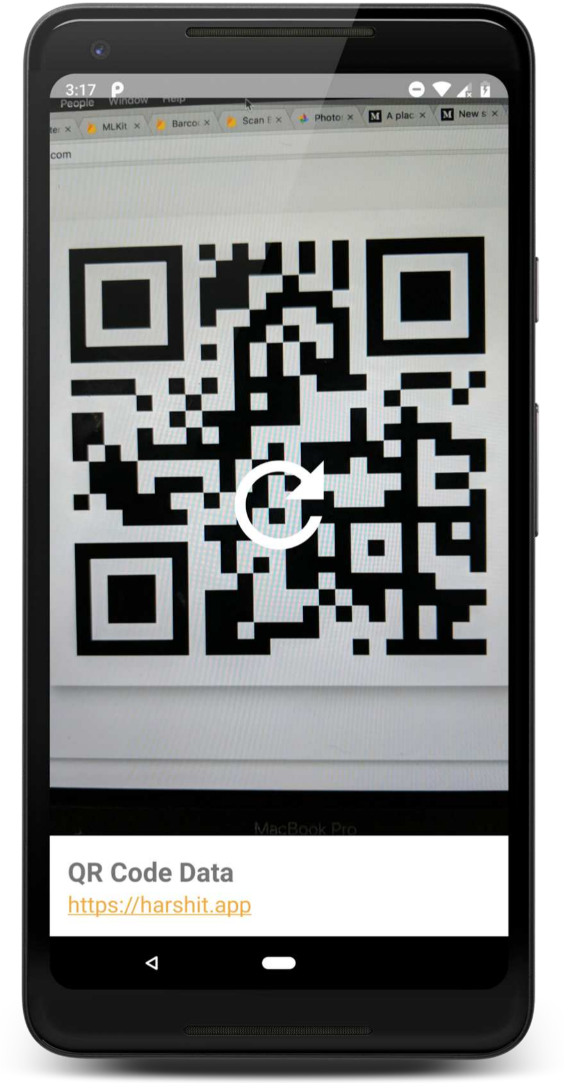
```
FirebaseVisionImage image = FirebaseVisionImage.fromBitmap(bitmap);

FirebaseVisionFaceDetector detector = FirebaseVision.getInstance()
    .getVisionFaceDetector(options);

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

Contour detection response

```
// Task completed successfully
if (faces.size() == 0) {
    showToast("No face found");
    return;
}
mGraphicOverlay.clear();
for (int i = 0; i < faces.size(); ++i) {
    FirebaseVisionFace face = faces.get(i);
    FaceContourGraphic faceGraphic = new FaceContourGraphic(mGraphicOverlay);
    mGraphicOverlay.add(faceGraphic);
    faceGraphic.updateFace(face);
}
```

Barcode Scanning

- Scan and process barcodes
- On Device only API available



QR Code Data

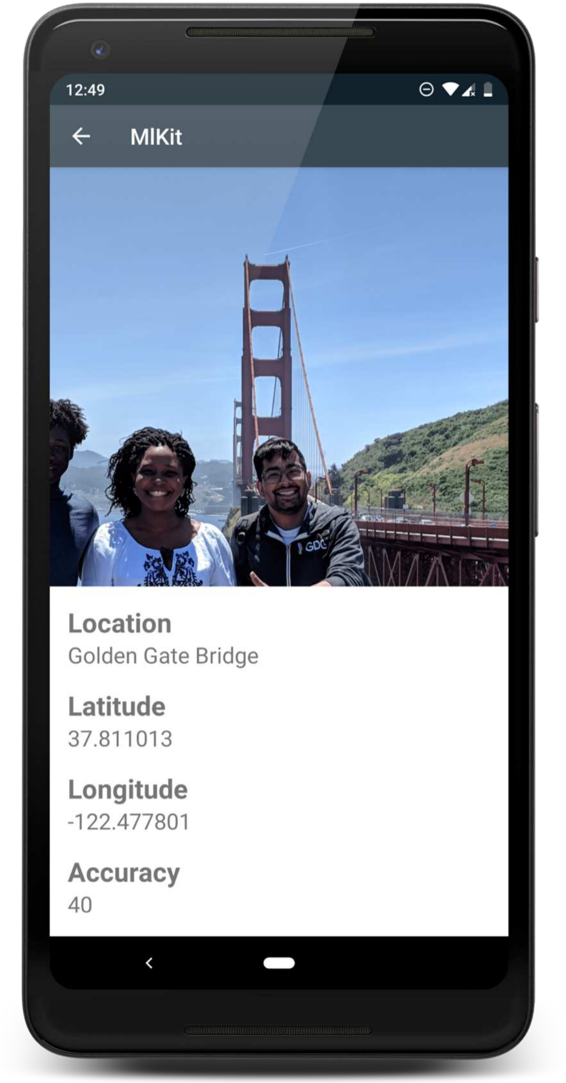
https://harshit.app

ML Kit Barcoder scanning

- Link: https://firebase.google.com/docs/ml-kit/android/read-barcodes
- Similar procedure to text recognition

Landmark Detection

- Identify popular landmarks in an image
- Cloud only API available



Location
Golden Gate Bridge

Latitude
37.811013

Longitude
-122.477801

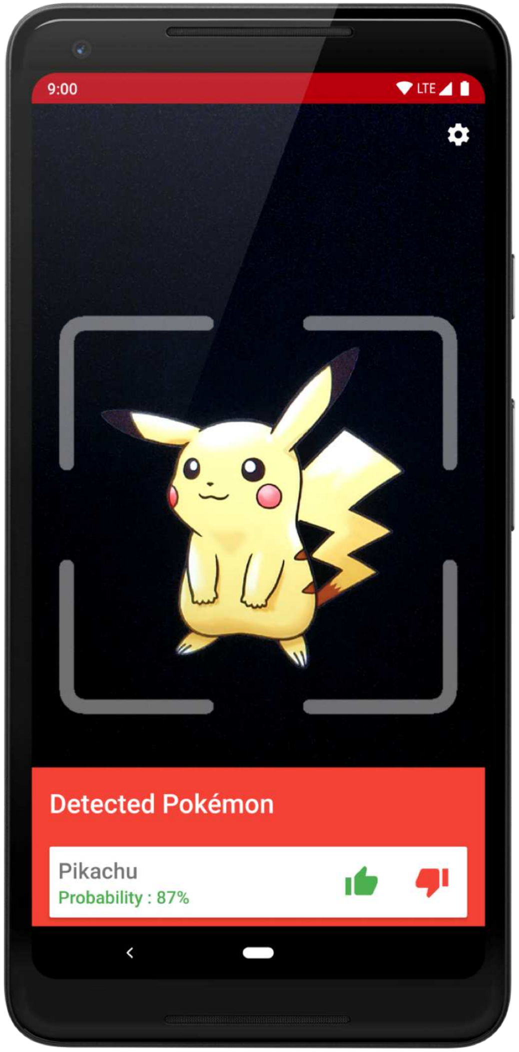
Accuracy
40

Landmark Detection

- Link: https://firebase.google.com/docs/ml-kit/android/recognize-landmarks
- Similar procedure to text recognition

Custom Model / Case Study

- Host your TensorFlow Lite models using Firebase
- ML Kit SDK automatically uses the best-available version of your custom model



Detected Pokémon

Pikachu
Probability : 87%

Training a TensorFlow Model

- See TensorFlow codelab on Website
- Complete Parts 1 and 2