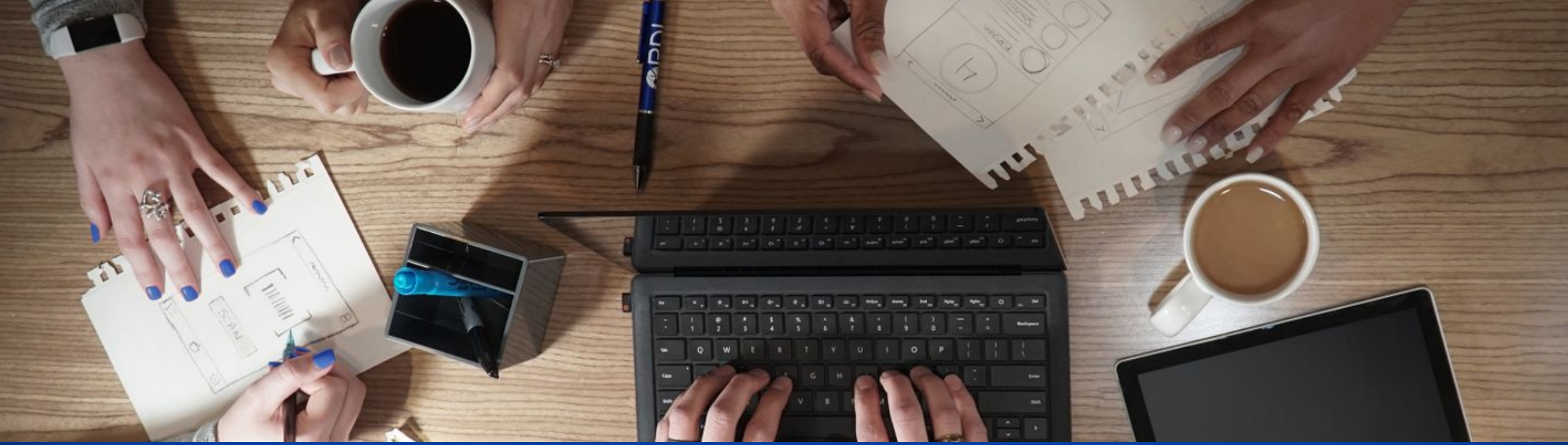
Discover Cloud Vision machine learning APIs for image detection

by PDI



Our People

“Alone we can do so little; together we can do so much.”

Presenting

J R Vitoria

Project Manager / Software Engineer

11 years in

Full Stack

Desktop Applications

Cryptography

Software Reverse Engineering

Paul Hachem

Software Architect

25+ years experience

Full Stack

Systems Engineering

Cloud Mobile

ArDrodus Williams

Software Engineer

3 years in

Desktop Application

Full Stack

Cloud Mobile

Here with us

Chris Berry

Chief Technology Officer

Steve Antonakakis

VP, DevOps

Ernesto Priego

VP, Product Management

Here with us

Emilene Rodley

Manager, HR

Allison Woodward

Manager, Marketing

Dan Gilbert

Manager, DevOps

Cederick Johnson

Director, Marketing

Mitchell Byrd

Engineer, DevOps

Contact Us

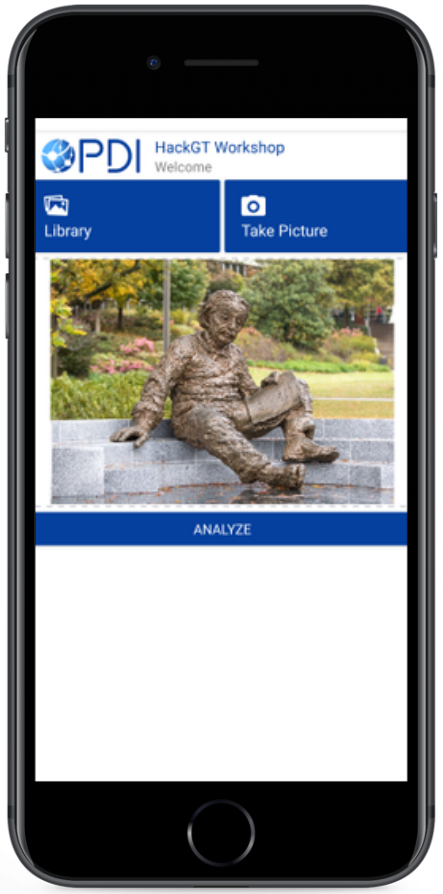
Slack Channel: sponsor-PDI

Technical Questions:
[PDI] Paul Hachem
[PDI] ArDrodus Williams
[PDI] J R Vitoria

The Project

"Expect the best, plan for the worst, and prepare to be surprised."

The project



- Create a mobile app based on Google's Vision API to parse prices from fuel signs
  - Cross platform mobile app
  - Uses Google Vision API
  - Supports gallery and camera modes
  - Easily exports results

Rewards

- 2 Holy Stone HS270 GPS 4K Drones
- 2 Holy Stone HS700D FPV 2K Drones
- Awarded to single team
  - Must meet all project requirements
  - Has best over all design
  - Has the most additional features

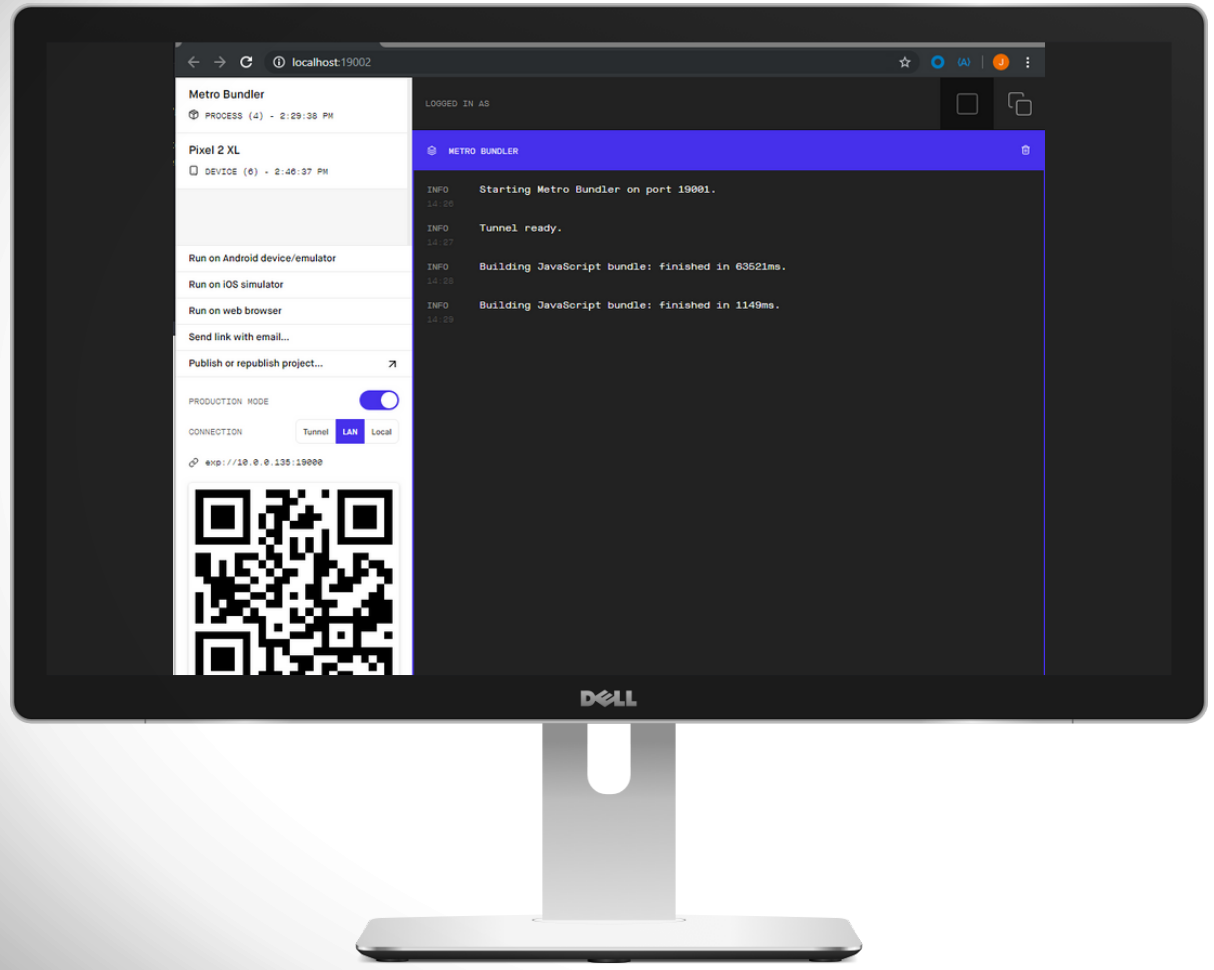
Demo

"Because sharing is caring"

Agenda

- Get ready
- Overview
- Internals

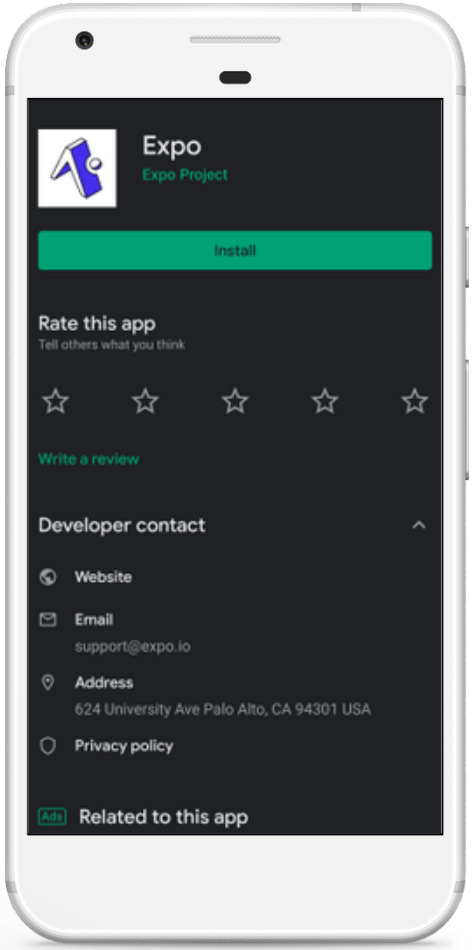
Overview



- Stage your Expo application

  Terminal > Expo Start

Overview



- Download the Expo App

  or use an emulator on your local system

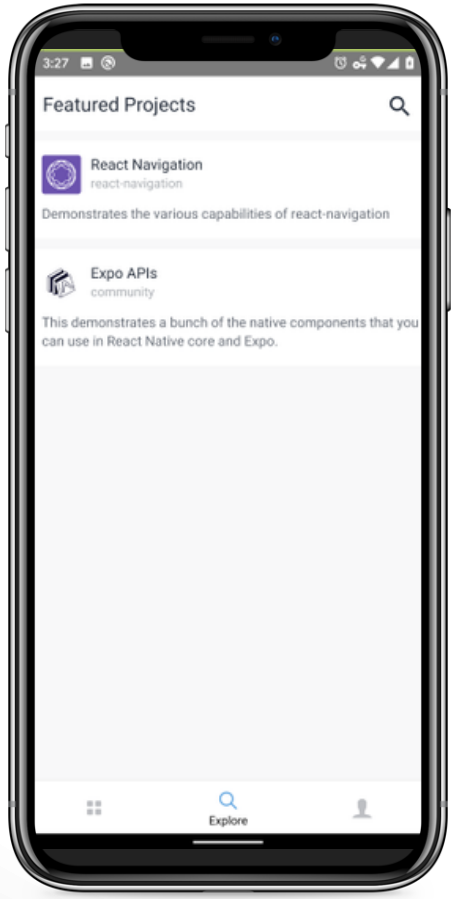
- Android

  https://play.google.com/store/apps/details?id=host.exp.exponent&hl=en_US

- iOS

  apps.apple.com/us/app/expo-client/id982107779

Overview



- Expo App
- Select the Projects button
- Scan the QR code
- Give your app any needed permissions

Overview

- Features
- Make it look good

  Consumers choose apps with their eyes
- Handle Gallery and Camera images
- Let them see what is about to be analyzed
- Visualize the results

Overview

- For the Geeks

  Give them all the data. They can handle the truth.

Setup local

- Node.js LTS

  https://nodejs.org/en/download/

- Visual Studio Code

  code.visualstudio.com/Download

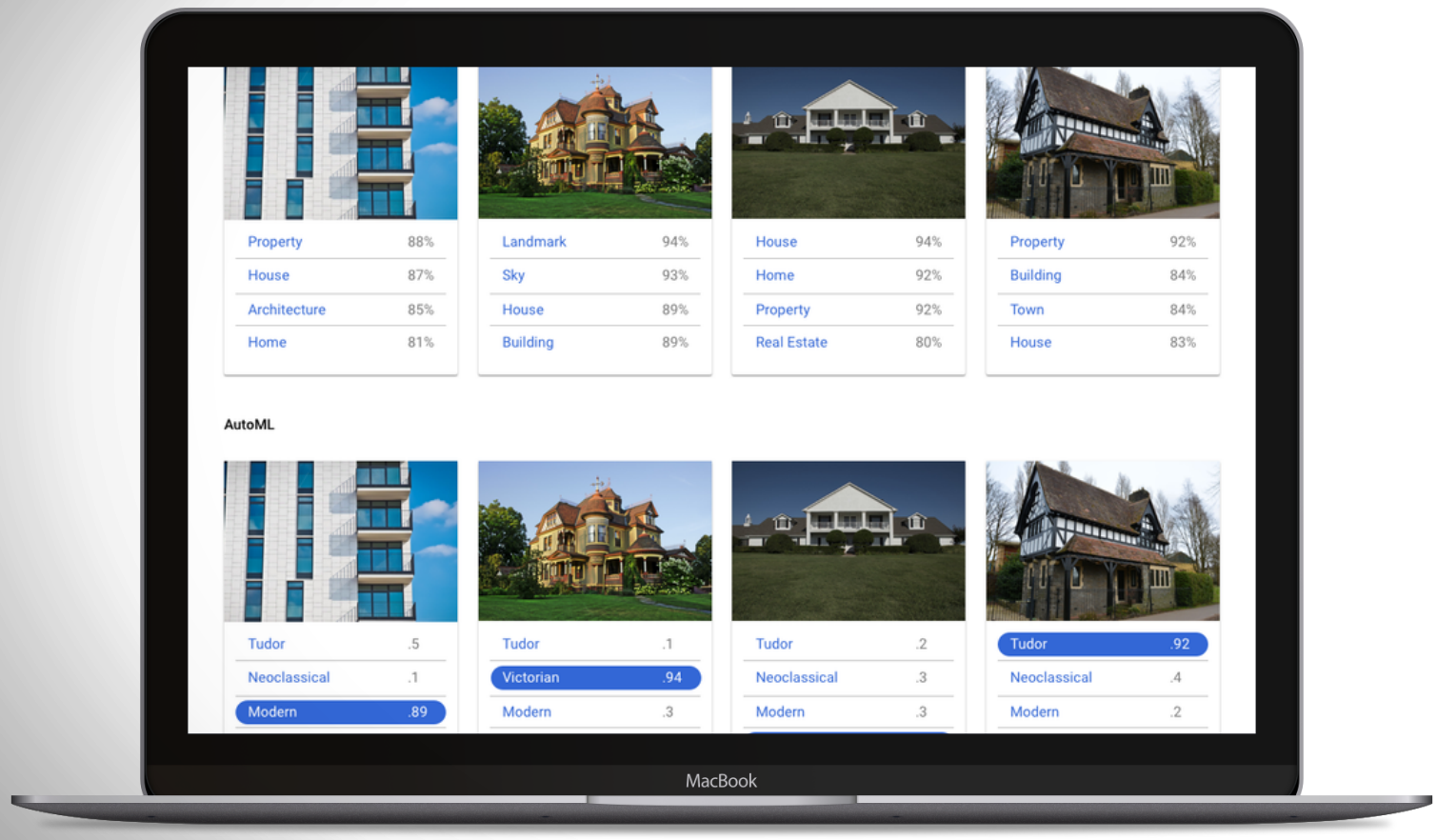
- React Native

  npm install -g react-native-cli

- Expo

  npm install expo-cli --global

Setup cloud services



- Firebase

  console.firebase.google.com/

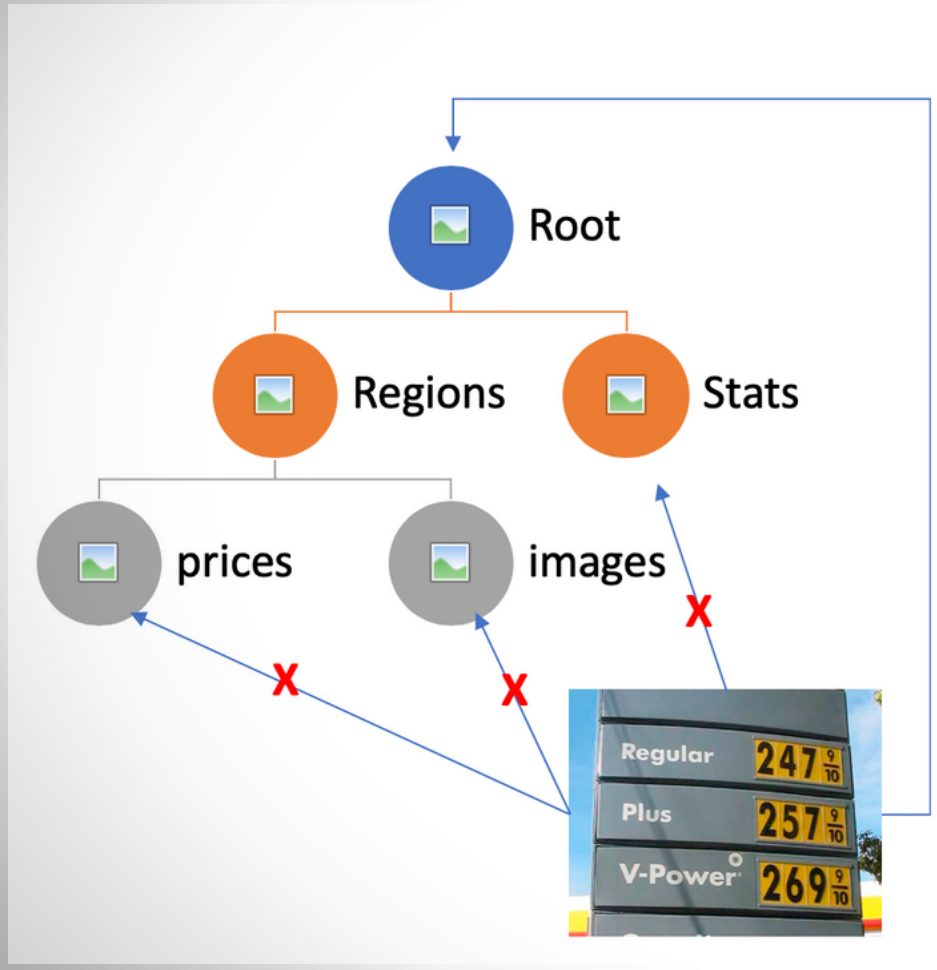
  npm install -S firebase

- Google Cloud Platform

  console.cloud.google.com

  Vision API

React Native



- One way data flow:
  Any time data changes, RN starts at the top of the component and works its way down to change only what's needed.
  You never interact with the UI directly to tell a screen to refresh
- DOM diffing to change only the parts that are required
- Hot Reloading
- Native components

Code it

environment.js ● | firebase.js

config ▸ environment.js ▸ ...

```
1   //environment.js
2   var environments = {
3     staging: {
4       FIREBASE_API_KEY: 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX',
5       FIREBASE_AUTH_DOMAIN: 'gt-hackathon-workshop.firebaseapp.com',
6       FIREBASE_DATABASE_URL: 'https://gt-hackathon-workshop.firebaseio.com/'
7       FIREBASE_PROJECT_ID: 'gt-hackathon-workshop',
8       FIREBASE_STORAGE_BUCKET: 'gs://gt-hackathon-workshop.appspot.com/',
9       FIREBASE_MESSAGING_SENDER_ID: '920491288477',
10      GOOGLE_CLOUD_VISION_API_KEY: 'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX'
11    },
12    production: {
13      // Warning: This file still gets included in your native binary and is
14    }
15  };
16
17  function getReleaseChannel() {
18    let releaseChannel = Expo.Constants.manifest.releaseChannel;
19    if (releaseChannel === undefined) {
20      return 'staging';
21    } else if (releaseChannel === 'staging') {
22      return 'staging';
23    } else {
24      return 'staging';
25    }
26  }
```

- Set your credentials
  - Firebase API Key
  - Google Cloud Vision API Key

React Native - Creating a Component

LibCard.js

```
'use strict';
import React, {Component}  from 'react';
import { Icon, Card, CardItem, Body, Text } from 'native-base';

export class LibCard extends Component {
  constructor(props) {
    super(props);
  }

  render() {
    return (
      <Card  style={{flex:1}}>
        <CardItem style={{backgroundColor:'#06409e'}} button
          <Body>
            <Icon name='images' style={{color:'white'}} />
            <Text style={{color:'white'}}>
              Library
            </Text>
          </Body>
        </CardItem>
      </Card>
    );
  }
}
```

- Create new app
  - expo init <app_name>
  - cd <app_name>
  - expo start
- Create new Component
  - Import
  - Extend a Component (Elements, JSX)
  - render()
  - Export

Data handling

State data

```
state = {
  image: null,
  uploading: false,
  filename: null };
```

Update state data

```
this.setState({ uploading: true });
```

Read state data

```
let { image, filename } = this.state;
```

Passing parameters (props)

```
<LibCard onPress={this._pickImage}/>
```

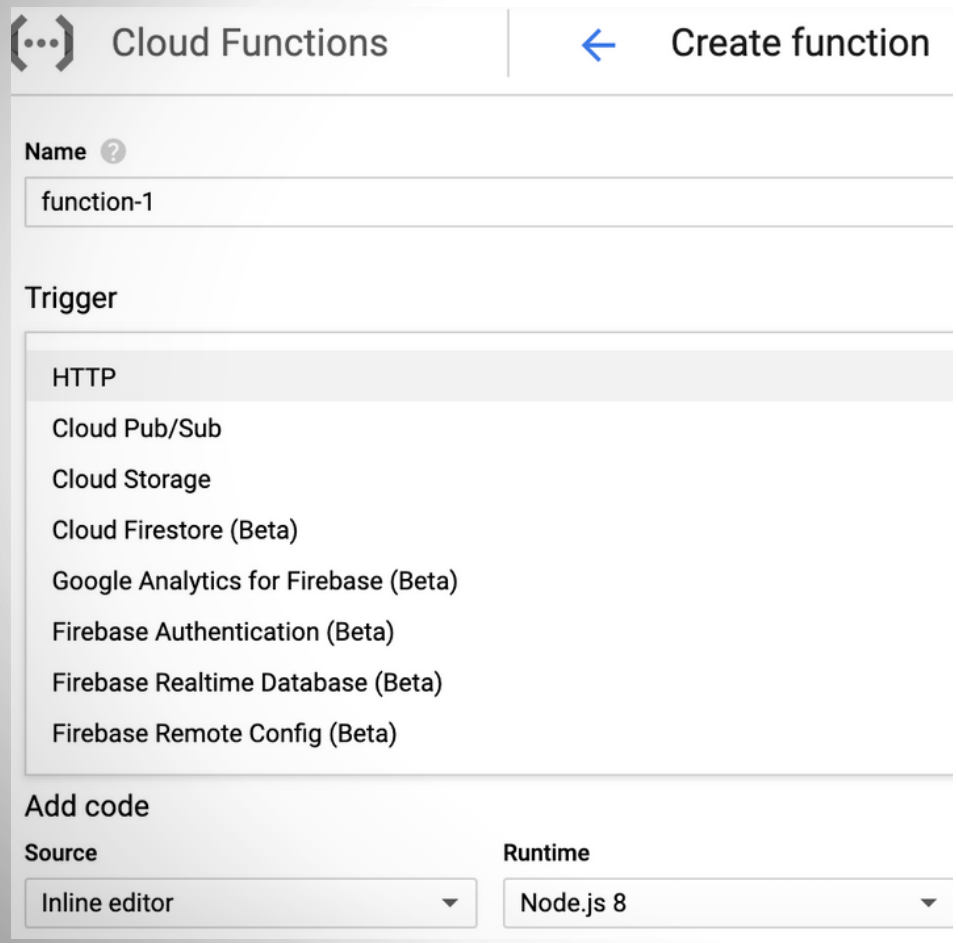
Reading parameters (props)

```
<CardItem style={{backgroundColor:'#06409e'}} button 
  onPress={this.props.onPress}>
```

Google Vision API

```
let body = JSON.stringify({
 requests: [
      { features: [ { type: 'LABEL_DETECTION', maxResults: 8 }],
       image: {source: {imageUri: image}}}]});
let response = await
      fetch('https://vision.googleapis.com/v1/images:annotate?key='
      +Environment['GOOGLE_CLOUD_VISION_API_KEY'],
       {headers: {Accept: 'application/json','Content-Type':
       'application/json'},method: 'POST', body: body});
let responseJson = await response.json();
```

Google Cloud Functions



- Create Cloud Function

  Select Trigger

  Select language (Node.js, Go, Python)

- use case: Send Slack message + pub/sub whenever an uploaded image has inappropriate content

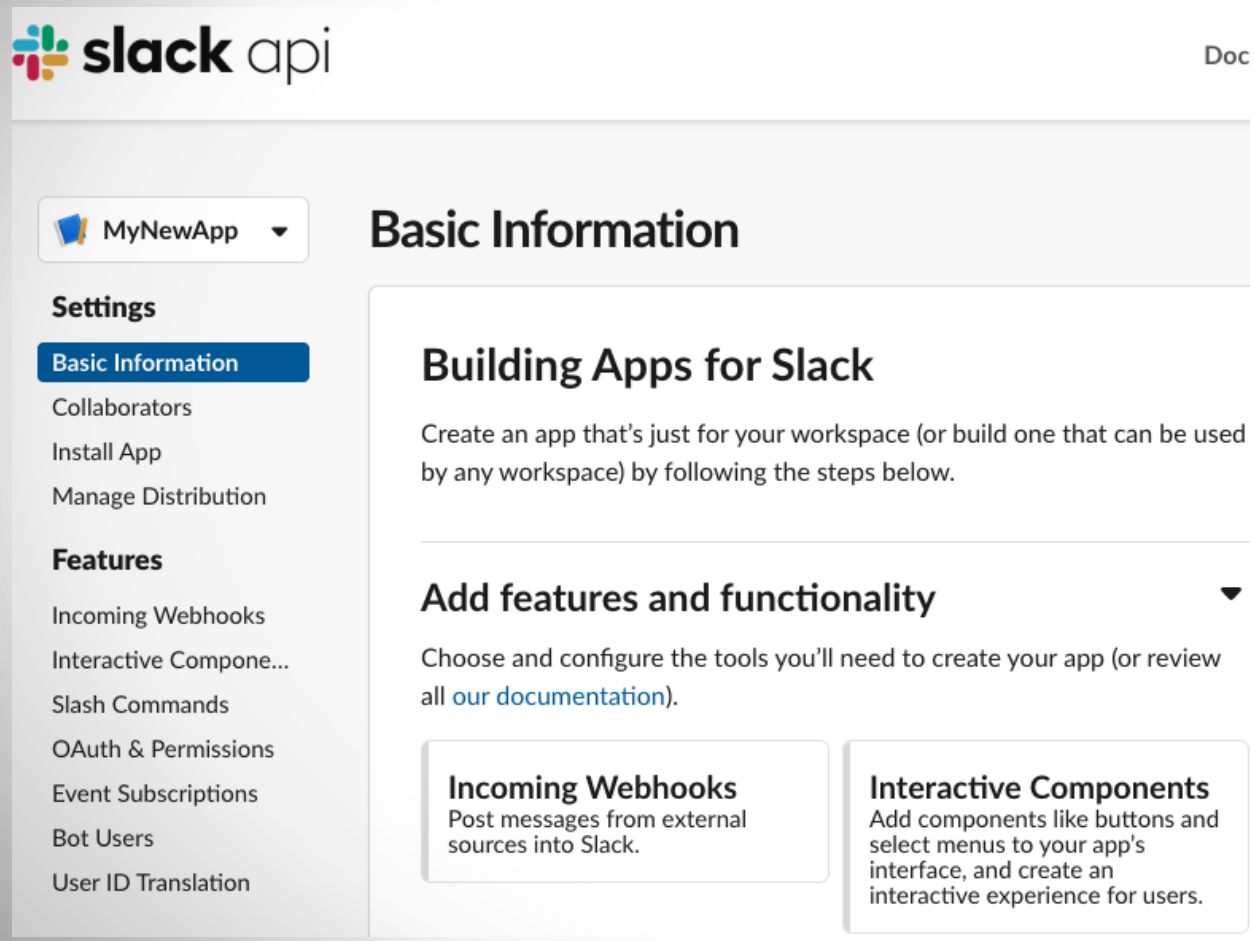
Google Cloud Functions

```
async function detect(bucketName, filename) {
  const [result] = await vision.safeSearchDetection(`gs://${bucketName}/${filename}`)
  const detections = result.safeSearchAnnotation;
  let topicName = "image-alert";
  if (!detections.violence.includes('UNLIKELY')) {
    console.log(`Violence: ${detections.violence}`);
    const message= createMessage("Image Violence:"+detections.violence);
    publishResult(topicName, message);
    sendToSlack(message);
  }
  function publishResult(topicName, data) {
    const dataBuffer = Buffer.from(JSON.stringify(data));
    return pubsub
      .topic(topicName) .publish(dataBuffer);
  }
  function sendToSlack(data) {
    webhook.send(data);
  };
  function createMessage(data) {
    const message = {
      text: data
    };
    return message;
  }
```

```
// Get a reference to the Pub/Sub component
const {PubSub} = require('@google-cloud/pubsub');
const pubsub = new PubSub();

// Get a reference to the Cloud Vision API component
const Vision = require('@google-cloud/vision');
const vision = new Vision.ImageAnnotatorClient();

const { IncomingWebhook } = require('@slack/webhook');
const url = "https://hooks.slack.com/services/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";
const webhook = new IncomingWebhook(url);
```

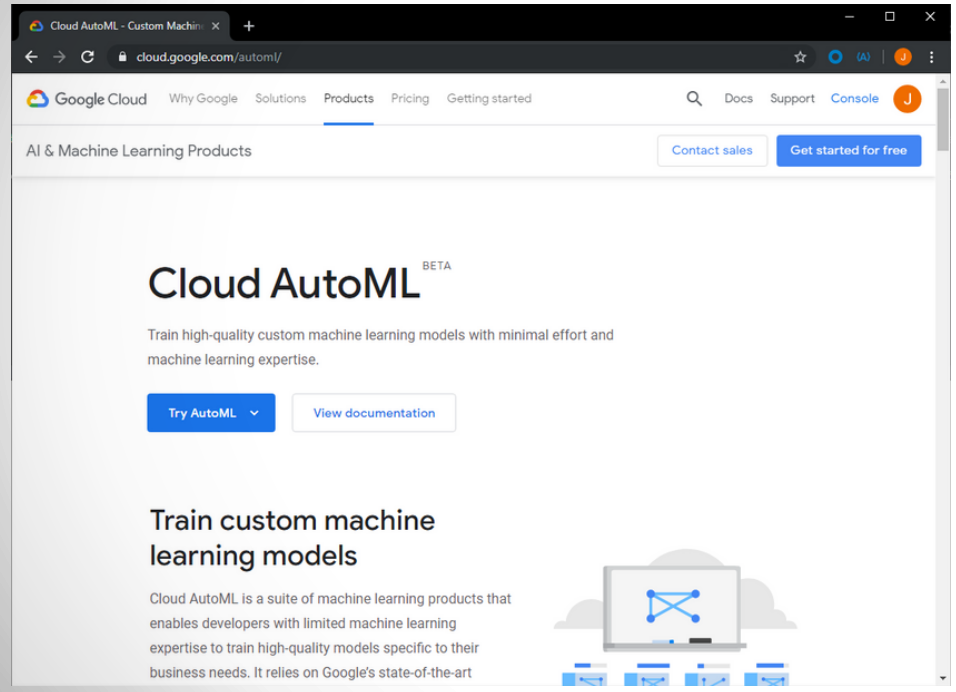
```
exports.helloGCS = (event, context) => {
  const gcsEvent = event;
  console.log(`Processing file: ${gcsEvent.name}`);

  const file = event.data || event;
  detect(file.bucket, file.name);
}
```

Slack api webhook



- Create Slack App (api.slack.com)
- Enable Incoming Webhooks

Bonus Material



- Auto ML

Auto ML Vision

Organize photos by Custom Labels, and upload to train (80/10/10)

Property	92%
Building	84%
Town	84%
House	83%

Vision API

Tudor	.92
Neoclassical	.4
Modern	.2
Ranch	.2

Auto ML

https://cloud.google.com/vision/automl/docs/beginners-guide

Auto ML Model Quality

Confusion Matrix

- Precision:

  over-predicting positives? (too many FP?)

- Recall

  missed predictions

  (too many FN?)

	Actual Positive	**Actual Negative**	
Predicted Positive	**TP**	FP	Precision=TP/(TP+FP)
Predicted Negative	FN	**TN**	
	Recall=TP/(TP+FN)		

References and Resources

- Mike LaPeter

  https://medium.com/@mlapeter/using-google-cloud-vision-with-expo-and-react-native-7d18991da1dd

- Aman Mittal

  https://blog.jscrambler.com/create-a-react-native-image-recognition-app-with-google-vision-api/

- React Native

  https://facebook.github.io/react-native/docs/getting-started

- Nativebase (UI)

  https://nativebase.io

- Google AutoML Vision Beginner's Guide

  https://cloud.google.com/vision/automl/docs/beginners-guide

Questions