FRANKFURT UNIVERSITY OF APPLIED SCIENCES

HIGH INTEGRITY SYSTEMS

CLOUD COMPUTING
Summer Semester 2022

NOTIFICATION BASED ON FACE DETECTION

by

Parth Desai 1381870

Hardikkumar Dudhat 1381838

Milan Dabhi 1381854

Under the guidance of

Prof. Dr. Christian Baun

CONTENTS

Abstract

Cloud computing is the supply of computer services over the Internet ("the cloud"), including servers, storage, databases, networking, software, analytics, and intelligence to provide quicker innovation, adaptable resources, and scale economies. The way businesses think about IT resources has significantly changed as a result of cloud computing. Cloud Functions is a serverless platform for building event-based microservices. It is a pay-as-you-go system, Functions-as-a-Service (FaaS) to run your code with zero server management. In this project, the firebase real-time database along with the google cloud function is being used. With the help of Python and OpenCV programming, we create a camera-based real-time face recognition. The script is running in the background accessing the webcam feed and applying a face recognition algorithm to identify a known and unknown person. When the system is unable to identify the person, that will trigger an update event on the cloud, and the user will be notified by sending the email notification via the cloud function.

1. INTRODUCTION

For humans, recognizing faces is a simple task. Studies in [1] have demonstrated that infants as young as one to three days old may recognize the faces of familiar people. Facial recognition can be completed rapidly and accurately using the open-source Intel platform known as OpenCV [2]. The ability to deploy applications essentially on any platform, including the cloud, is one of the most significant benefits of developing applications with the Python programming language. It suggests that Python can be launched on portable devices like computers, tablets, or smartphones as well as on cloud servers [3]. So, we have implemented both technologies in this project.

1.1 Background

The most popular IT innovation right now is cloud computing [4]. The concept of cloud computing is new, and it offers excellent prospects in many different fields. A group of computers and servers used for cloud computing are accessible to everyone online [5]. Using applications without installing them, consumers and companies can access their files from any computer with an internet connection thanks to cloud computing. Numerous internet-based on-demand services, including software, hardware, servers, infrastructure, and data storage, are made available through cloud computing [6].

The advantages of cloud computing:

- Boost throughput - With cloud computing, more work can be completed in less time and with fewer resources [7].
- Lower costs - In cloud computing, users share computer gear, software, and data, eliminating the need for hardware and software purchases [7].

- Increased accessibility - With cloud computing, users may access data and files online at any time and from any location [7].
- Less Training is Needed: With cloud computing, more work can be done with fewer personnel. Therefore, users must have a minimal level of training on hardware and software issues [7].

Computer vision is a method that enables us to comprehend how images and videos are stored, how to change them, and how to extract data from them. The foundation or primary tool utilized in artificial intelligence is computer vision. Self-driving cars, robotics, and photo-editing apps all heavily rely on computer vision [8].

1.2 Objectives

The main objective of this project is as follows:

1. Capture live video footage from a webcam and feed it into an image recognition algorithm.
2. Train the model for known person image recognition and predict the person in the real-time video stream.
3. Update data into firebase real-time database.
4. Trigger an event based on updates in the database via cloud function.
5. On successful trigger, a user should be notified by email notification.

2. METHODOLOGY

This project uses OpenCV to facilitate face recognition, Firebase real-time database for storage, and Firebase cloud function for event trigger and email notification. This section gives a brief review of the above mentioned technologies, approaches, and methods used in this project.

2.1 OpenCV

OpenCV is a substantial open-source library for image processing, machine learning, and computer vision. It now plays a significant part in real-time operation, which is crucial in modern systems. Using it, one may analyze pictures and movies to find faces, objects, and even human handwriting. Python can handle the OpenCV array structure for analysis when it is integrated with different libraries, such as NumPy. We use vector space and apply mathematical operations to these features to identify visual patterns and their various features.

OpenCV's initial release was 1.0. OpenCV is free for academic and commercial use because it is distributed under a BSD license. It is compatible with Windows, Linux, Mac OS, iOS, and Android and offers C++, C, Python, and Java interfaces. Real-time applications for improved processing efficiency were the primary consideration when OpenCV was developed. To take advantage of multi-core processing, everything is written in optimized C/C++ [9].

As there are many real-time use cases of OpenCV, we are using it for face recognition in this project.

2.2 Firebase

Google's Firebase is a tool that makes it simple for developers to create,

maintain, and expand their apps. It makes it easier for developers to create apps more quickly and securely. Because there is no programming required on the firebase side, it is simple to use the features more effectively. It offers services to web, unity, android, and iOS. It offers cloud storage. The database used for data storage is a NoSQL one [10].

2.3 Firebase Cloud Function

A serverless execution environment called Google Cloud Function is used to create and connect cloud services, also Function as a service (FaaS). One definition of serverless is the removal of the responsibility of having servers, setting or upgrading software, and patching operating systems by using cloud functions. Google is in charge of all management of the infrastructure and software. We just have to add the code.

Python and JavaScript are two programming languages that are supported by Google Cloud Function (Node.js). This makes it easier for developers who are proficient in Python or Java to upload functions in rapid implementation [11].

We have deployed the firebase function written in Node.js to send email notifications.

The main characteristics of cloud functions are:

- Serverless
- Pay as you go
- Event-driven
- Connects and extends cloud services
- Scales automatically

3. Installation Guide

The part explains how to put the plan into action to achieve desired goals. Although some precise milestones have been discussed in this section, a hybrid strategy was adopted throughout the development.

To configure the project, try walking through these steps which begin with the required setup steps to integrate the python environment and go through setting up the firebase console.

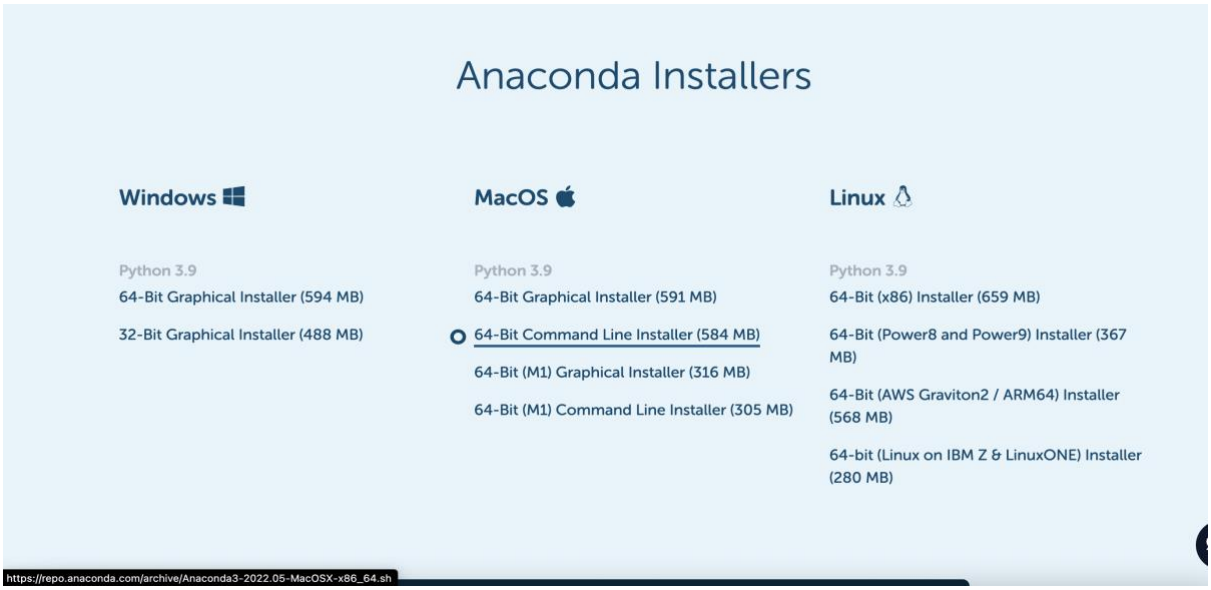
3.1 Setting up Python Environment

The open-source Anaconda distribution is the easiest way to perform Python data science and machine learning on Linux, Windows, and Mac operating systems (OS). The Anaconda distribution has pre-installed most popular libraries such as NumPy, Pandas, Conda, SciPy, etc. So, users do not have to deal with installing these libraries.

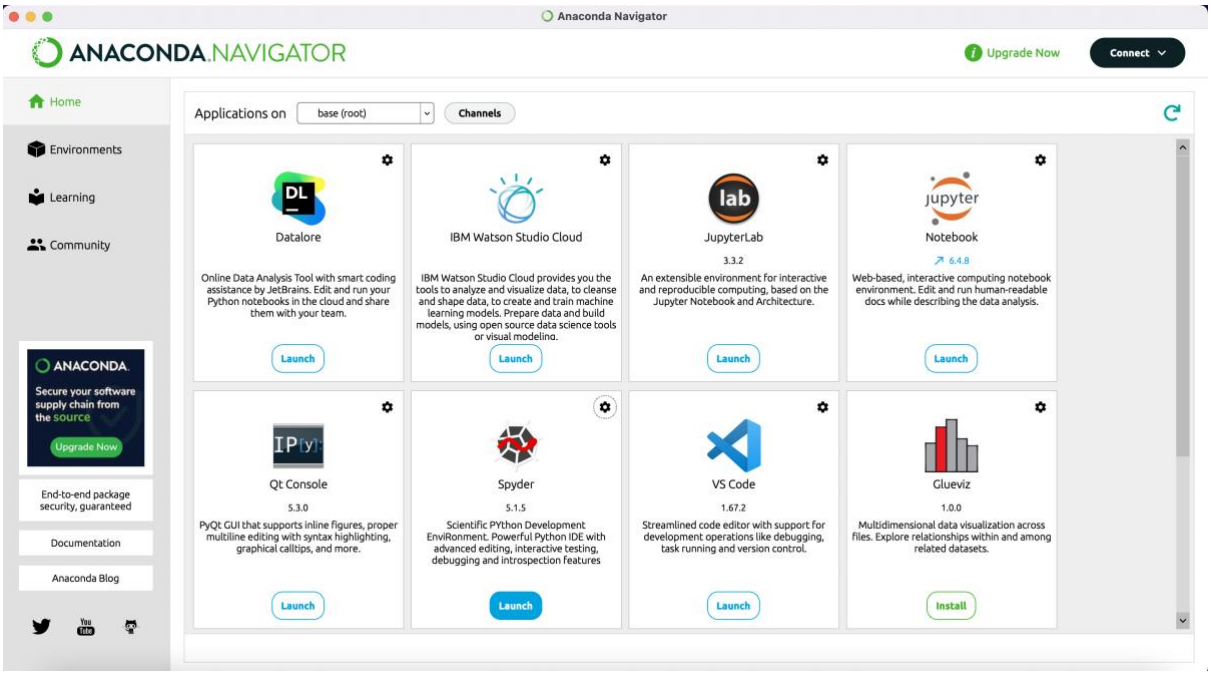
To download Anaconda distribution for different OS, go to the link:
https://www.anaconda.com/products/distribution#Downloads

For more information about Anaconda, go to the link
https://www.anaconda.com/

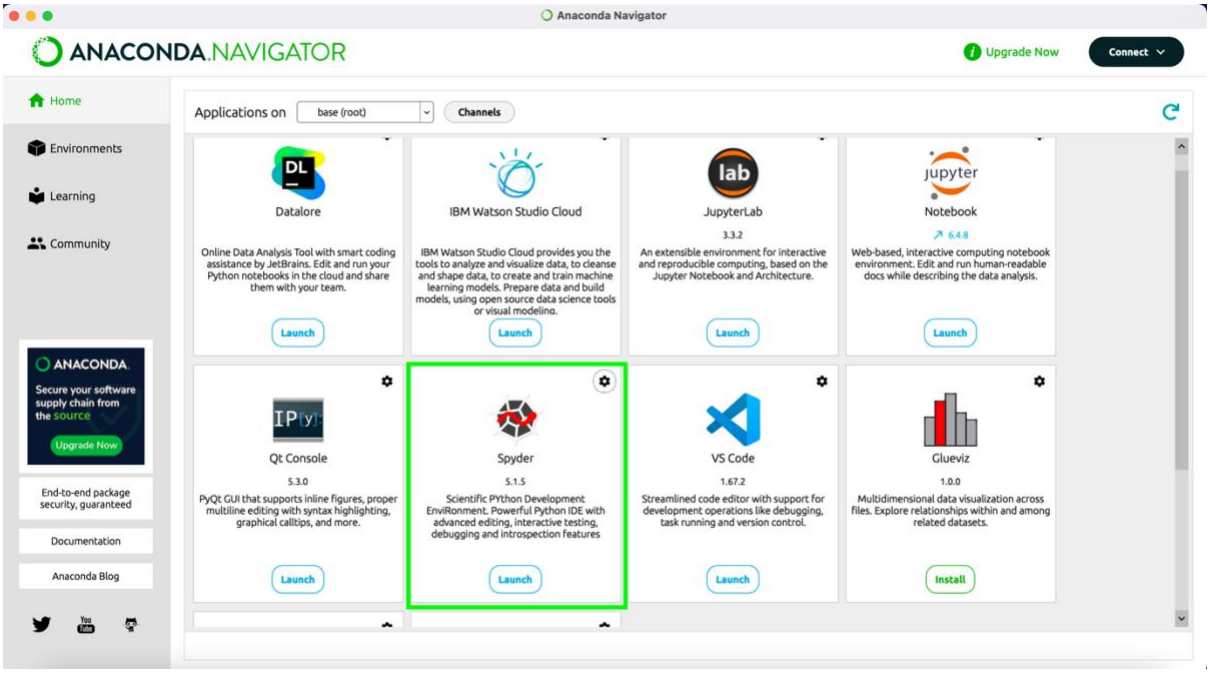
The page should look like this.



Choose the supported version and download it. We are using a *64-Bit (M1) Graphical Installer* in this setup. Open the downloaded installer.



Open any IDE which supports the python environment. Spyder is Integrated Development Environment (IDE) that is used in this project. Here we can run python files.



For the next step, install the following python libraries that are required to do face-recognition; cmake, dib, face_recognition, etc.

Install **CMake** in Anaconda. CMake is an open-source, cross-platform family of tools designed to build, test, and package software. It is used to control the software compilation process using simple platform and compiler-independent configuration files.

```
(base) parth_desai@Parths-Air ~ % pip install cmake
Requirement already satisfied: cmake in ./opt/anaconda3/lib/python3.9/site-packa
ges (3.22.5)
```

Install **OpenCV** in Anaconda. OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

```
(base) parth_desai@Parths-Air ~ % pip install opencv-python
Requirement already satisfied: opencv-python in ./opt/anaconda3/lib/python3.9/site-packages (4.6.0.66)
Requirement already satisfied: numpy>=1.17.3 in ./opt/anaconda3/lib/python3.9/site-packages (from opencv-python) (1.22.4)
(base) parth_desai@Parths-Air ~ %
```

Install **Dlib** in Anaconda. Dlib is a C++ toolkit containing machine learning algorithms and tools. Here Dlib is used for face detection and facial landmark detection. The frontal face detector in Dlib is simple, fast, and powerful. It is written in C++, that is the reason behind installing Cmake beforehand.

```
(base) parth_desai@Parths-Air ~ % pip install dlib
Requirement already satisfied: dlib in ./opt/anaconda3/lib/python3.9/site-packages (19.24.0)
(base) parth_desai@Parths-Air ~ %
```

Install **face_recognition** in Anaconda. This is the base library that is being using for face recognition and is very important for this project. It is built using Dlib's face recognition feature.

```
(base) parth_desai@Parths-Air ~ % pip install face_recognition
Requirement already satisfied: face_recognition in ./opt/anaconda3/lib/python3.9/site-packages (1.3.0)
Requirement already satisfied: face-recognition-models>=0.3.0 in ./opt/anaconda3/lib/python3.9/site-packages (from face_recognition) (0.3.0)
Requirement already satisfied: Click>=6.0 in ./opt/anaconda3/lib/python3.9/site-packages (from face_recognition) (8.1.3)
Requirement already satisfied: Pillow in ./opt/anaconda3/lib/python3.9/site-packages (from face_recognition) (9.1.1)
Requirement already satisfied: numpy in ./opt/anaconda3/lib/python3.9/site-packages (from face_recognition) (1.22.4)
Requirement already satisfied: dlib>=19.7 in ./opt/anaconda3/lib/python3.9/site-packages (from face_recognition) (19.24.0)
(base) parth_desai@Parths-Air ~ %
```

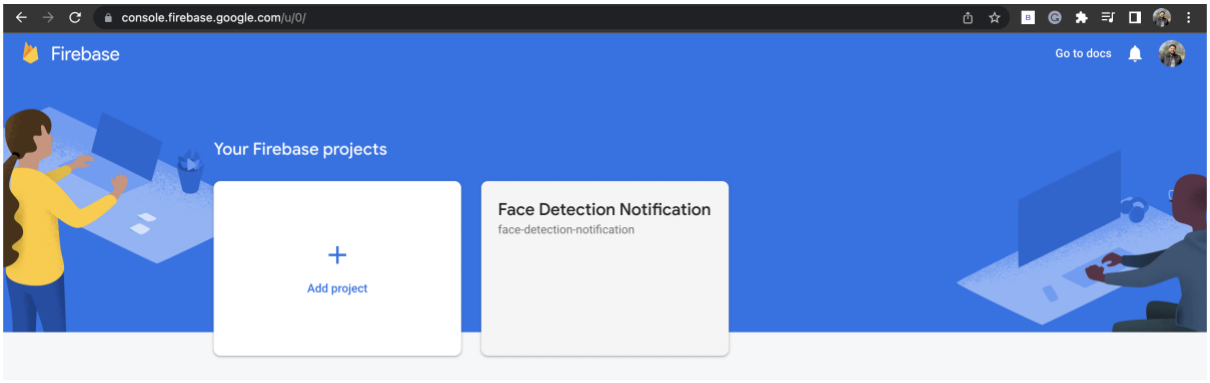
Install **firebase_admin.** This project uses a firebase real-time database and to connect with it, this library is required. The official documentation is liked here (https://firebase.google.com/docs/database/admin/start).

```
(base) parth_desai@Parths-Air ~ % pip install firebase_admin
Requirement already satisfied: firebase_admin in ./opt/anaconda3/lib/python3.9/s
ite-packages (5.2.0)
Requirement already satisfied: google-api-python-client>=1.7.8 in ./opt/anaconda
3/lib/python3.9/site-packages (from firebase_admin) (2.39.0)
Requirement already satisfied: cachecontrol>=0.12.6 in ./opt/anaconda3/lib/pytho
n3.9/site-packages (from firebase_admin) (0.12.11)
Requirement already satisfied: google-cloud-storage>=1.37.1 in ./opt/anaconda3/l
ib/python3.9/site-packages (from firebase_admin) (1.41.1)
```

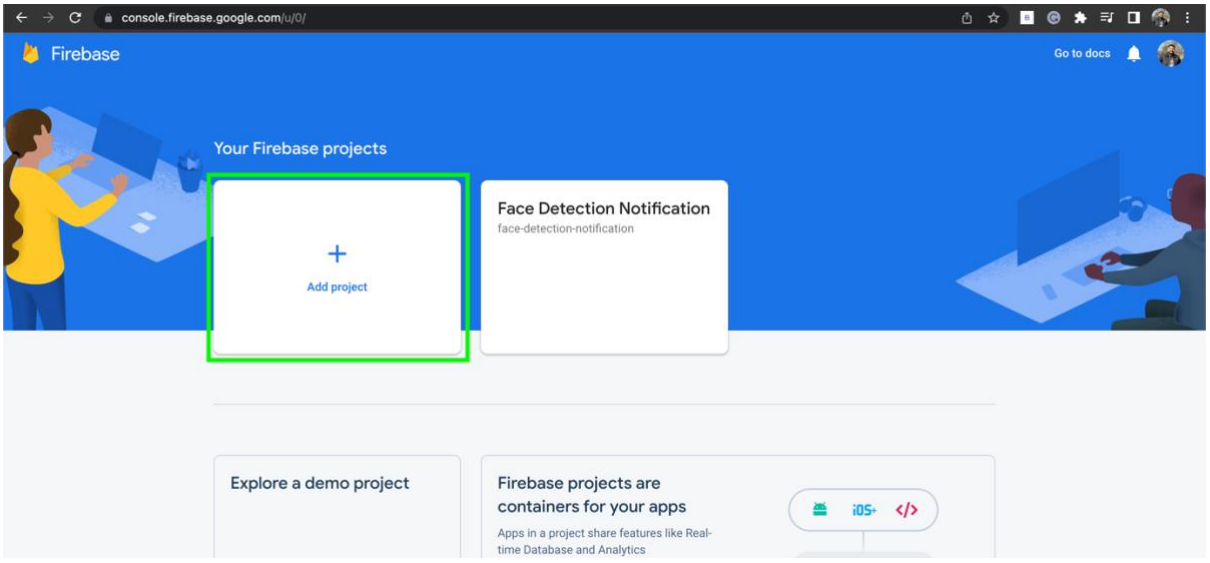
3.2 Setting up Firebase

Login into the Firebase console with a google account.

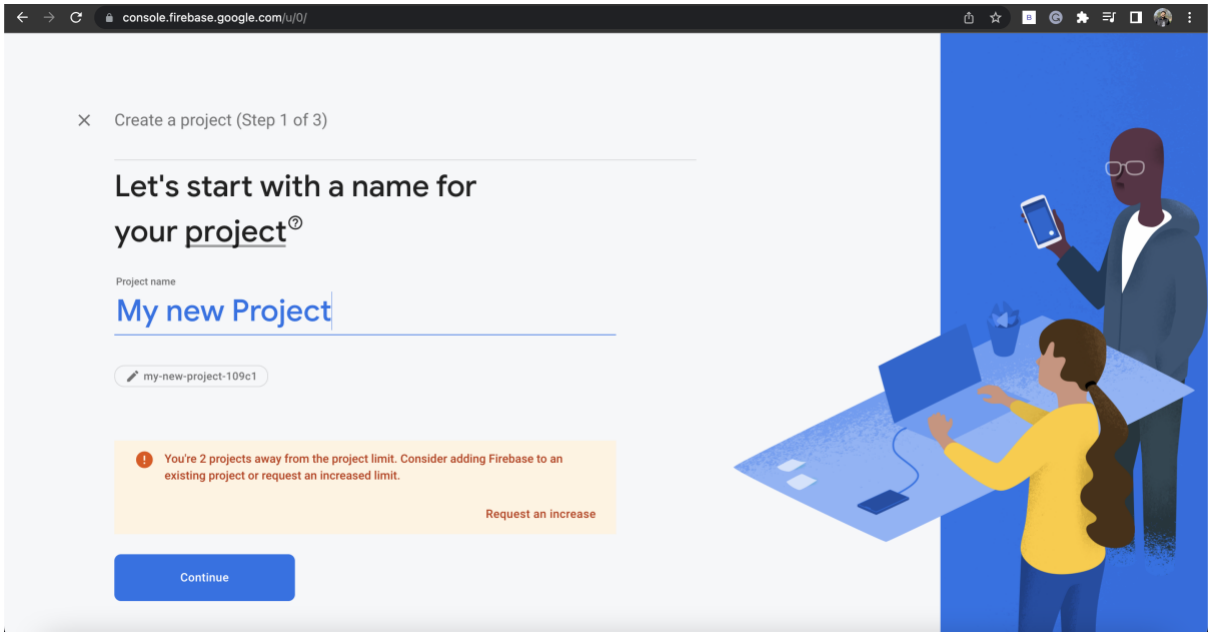
https://console.firebase.google.com/



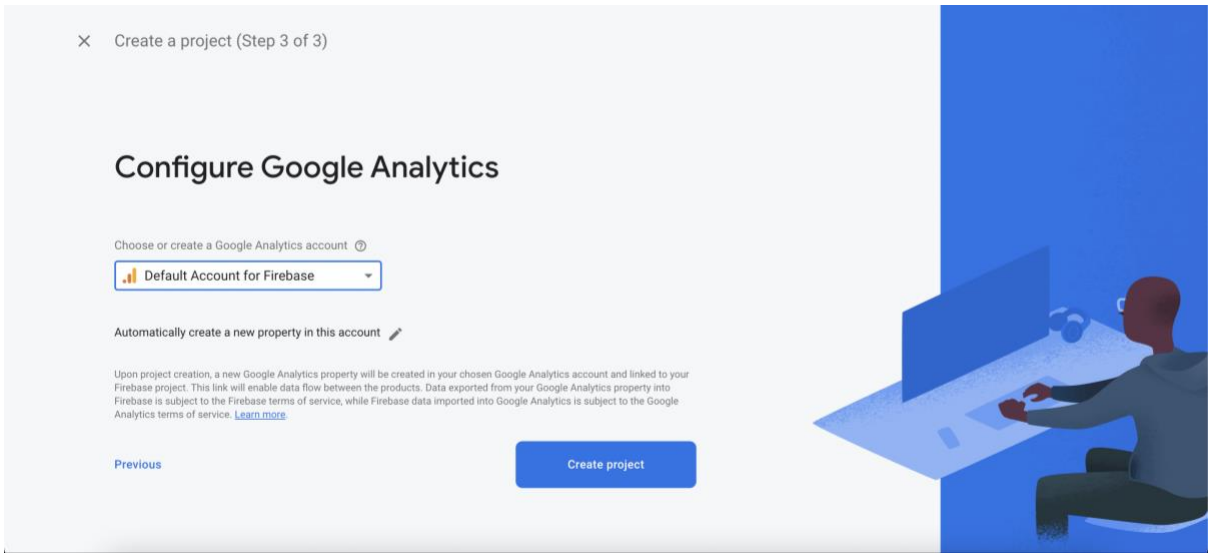
Create a new project, by clicking the “add project” button.



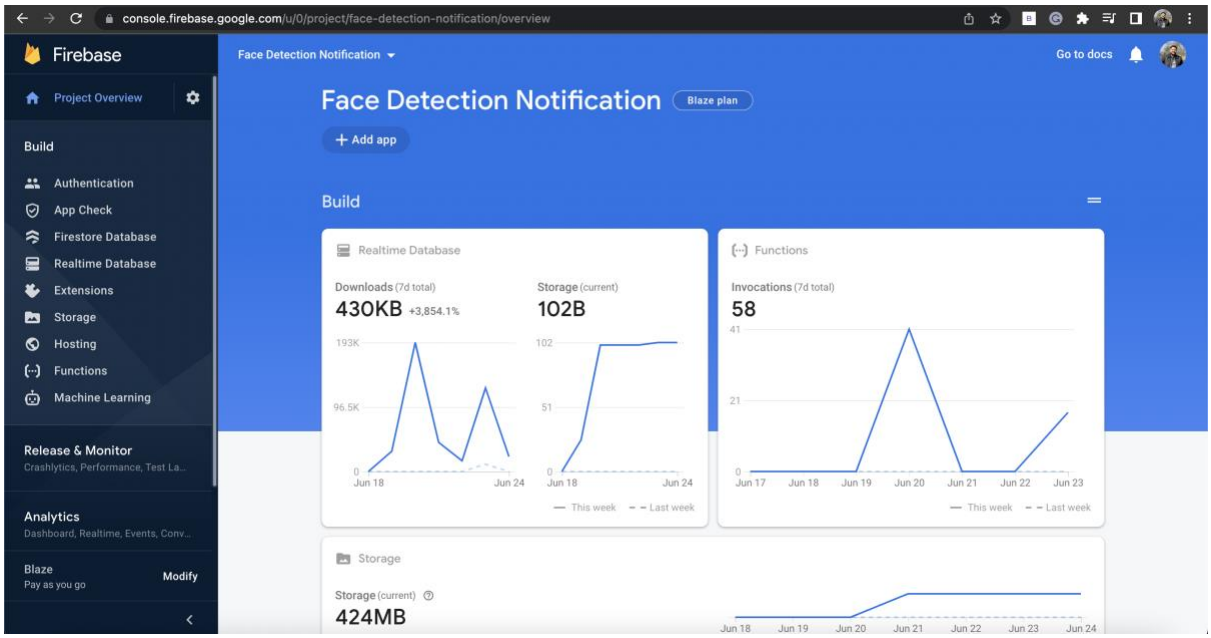
Create a Project. Enter a name for the project.



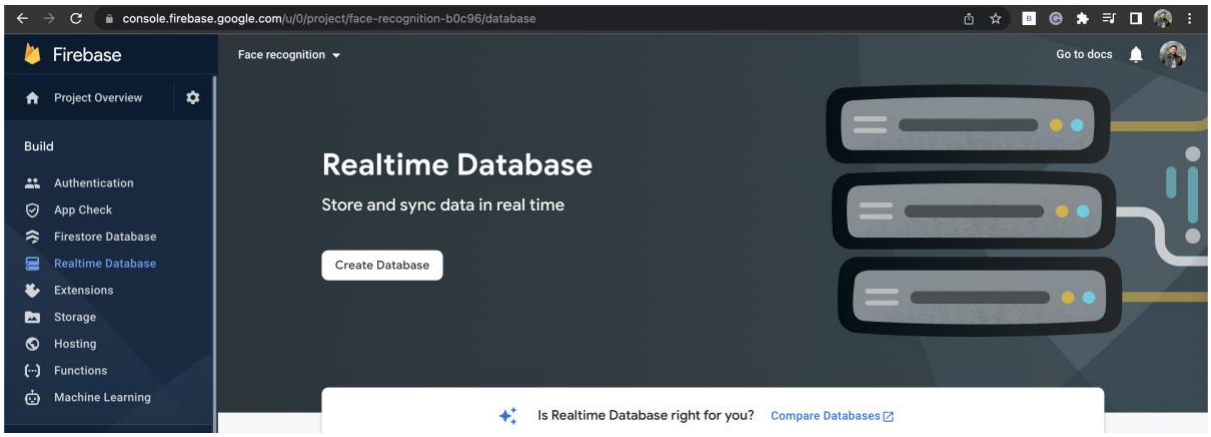
Continue all the way through and create a project. We are using the firebase function for the project that requires entering the billing details. Firebase provides $300 credits to the new users for 30 days to use it which would be sufficient for such small-scale projects. For Billing, usage can be checked through the firebase dashboard and can be disabled at any moment.



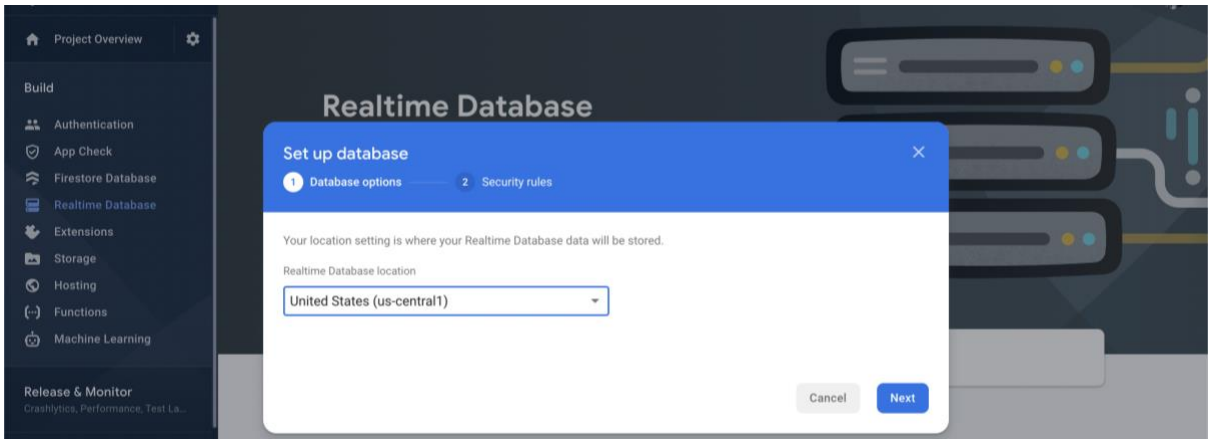
After the successful setup of the project. Open the project. The dashboard of the project should look like this.



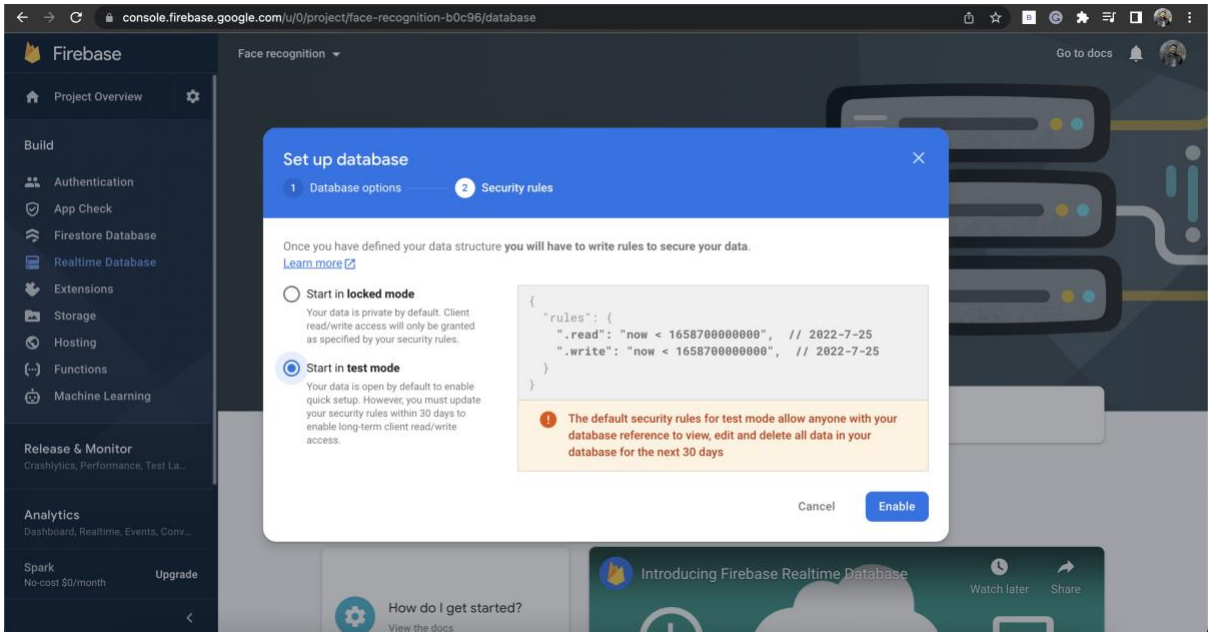
The Firebase real-time database is used for this project. Follow the below steps to set up the real-time database. Go to the Realtime Database tab and click on create a database.



Select Realtime Database location, in this case, the United States (us-central-1) server is used. That should not make difference for such a small-scale project.

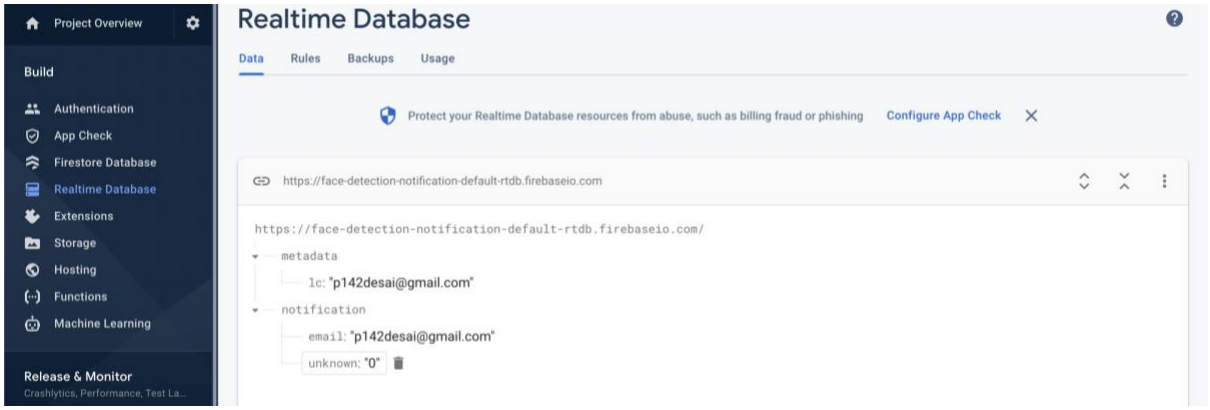


Here in this step, there is two options “start in locked mode” or “start in test mode”. The project is in the initial state so the 2nd option is enabled.

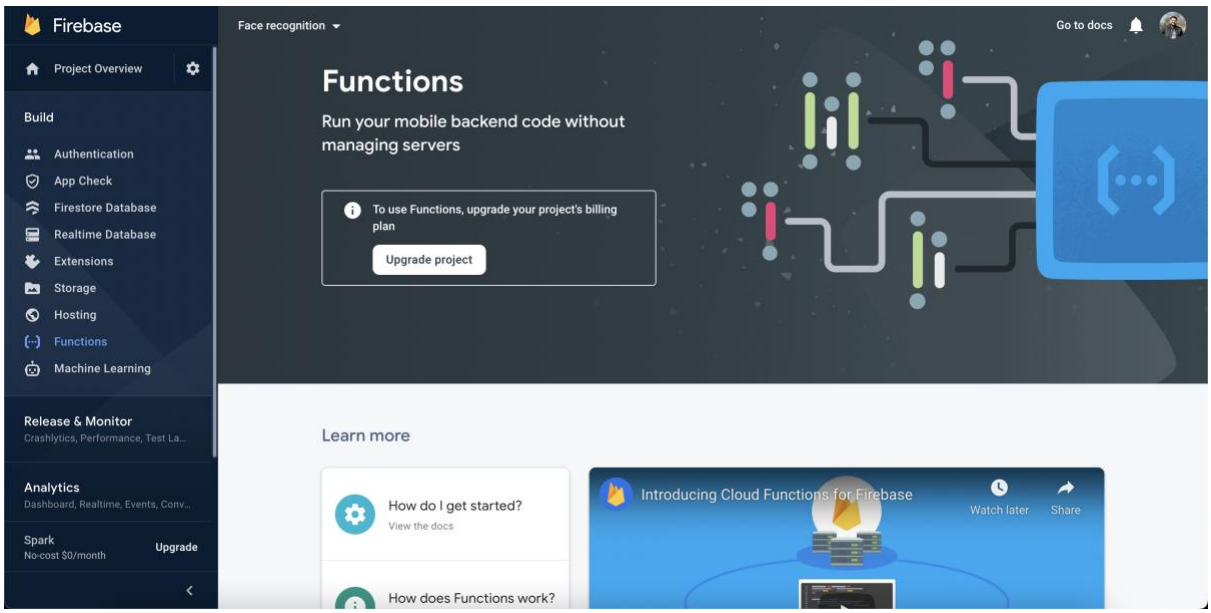


After a successful setup, the real-time database should look like this.

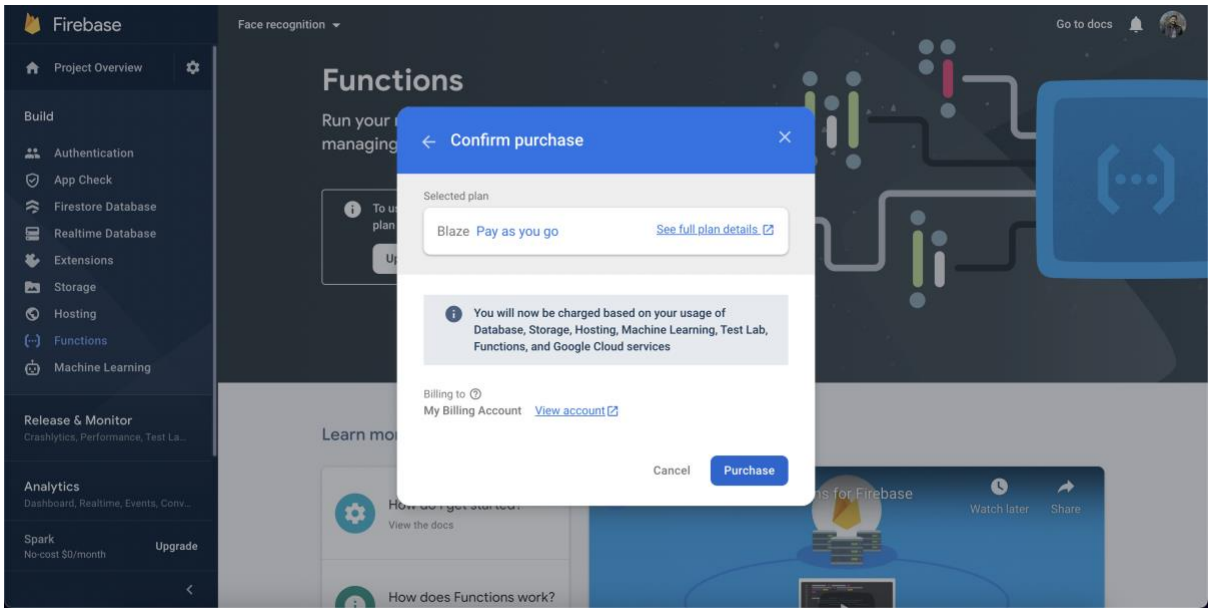
Replicate the below-shown tree structure to complete the default setup of the project.



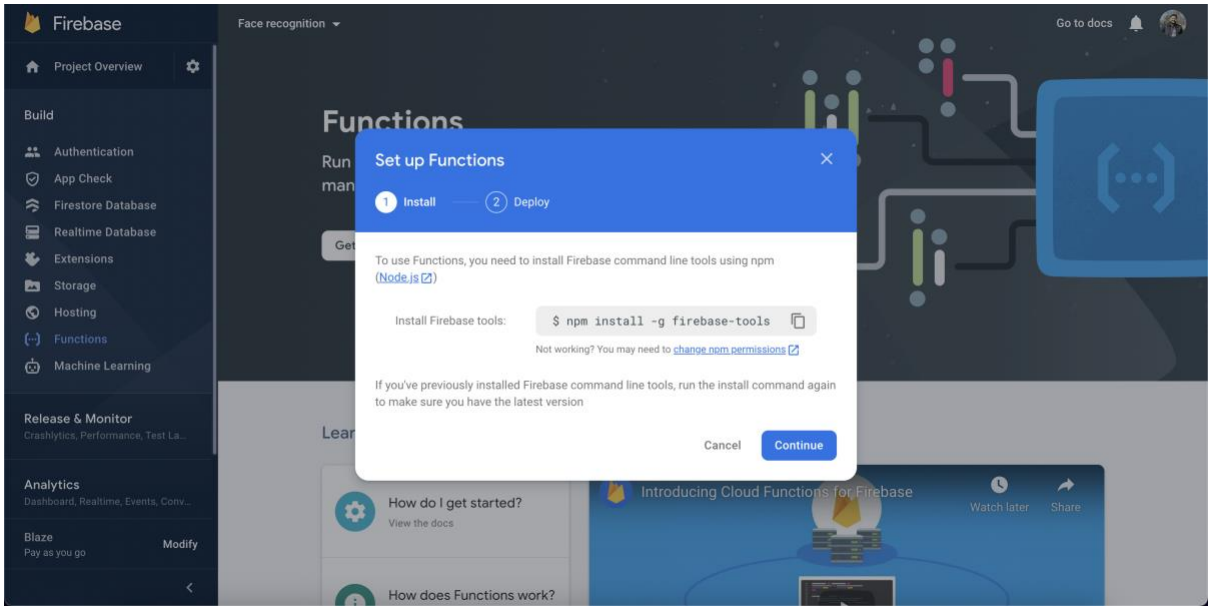
Now deploy the firebase function to send an email as the database update event. Go to the Functions tab from the side navigation bar.



To deploy the functions, the project must have billing details. Click on the Upgrade project button and add billing details.



The next step is to set up the firebase function. Firebase-tools is a pre-requisite for that and install it using npm.

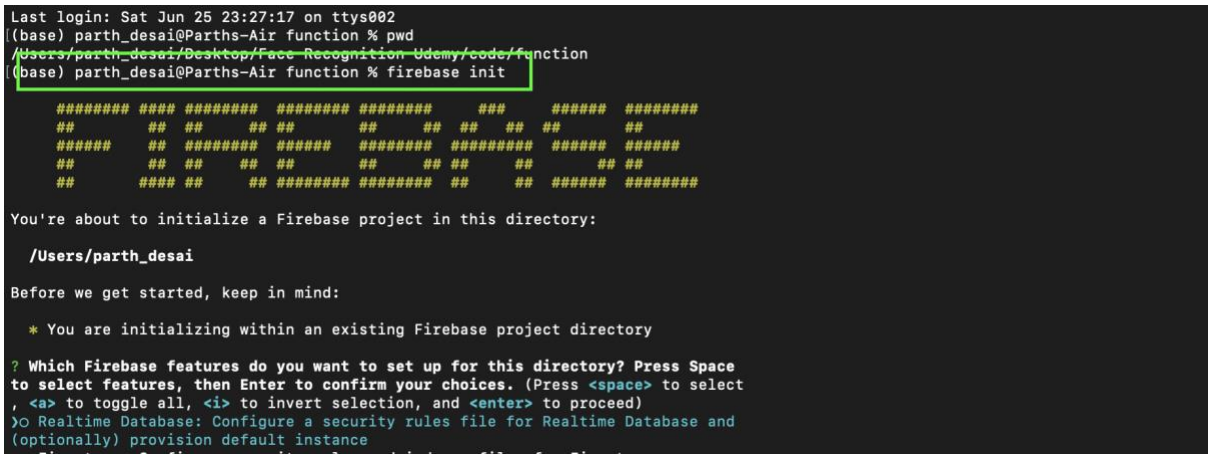


Now create the function folder in the code directory and inside that run the following

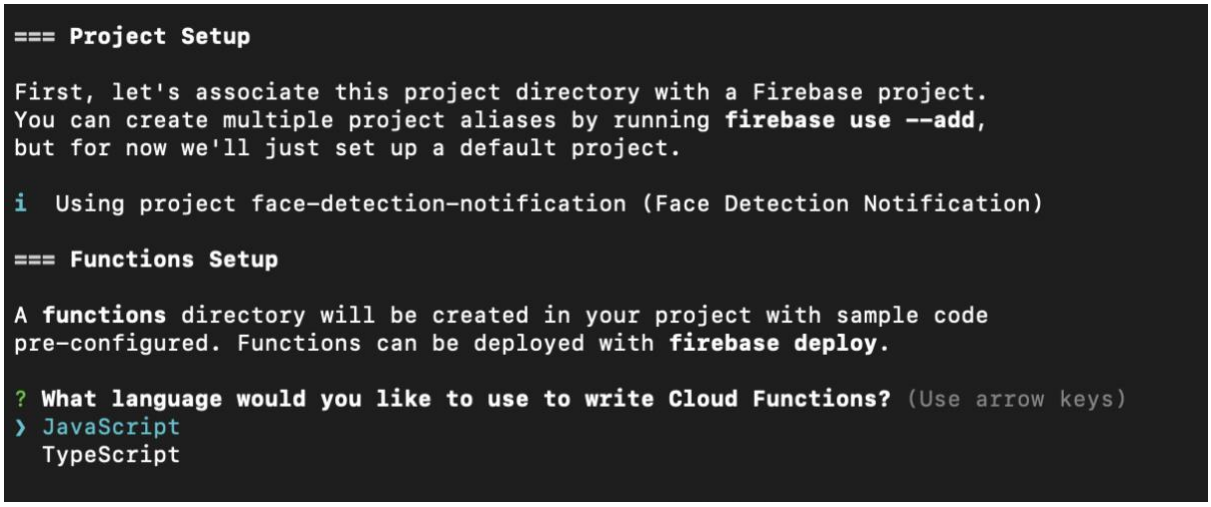
command to set up the firebase function.

Command: *firebase init*

Now there are multiple options available for setup. Navigate via arrow keys and select “Functions” by using the space key and then press enter to proceed.



The cloud function is written in Node.js, so select JavaScript.



Select “yes” for the following step. Use Command: *firebase deploy*, to deploy the function into the cloud.

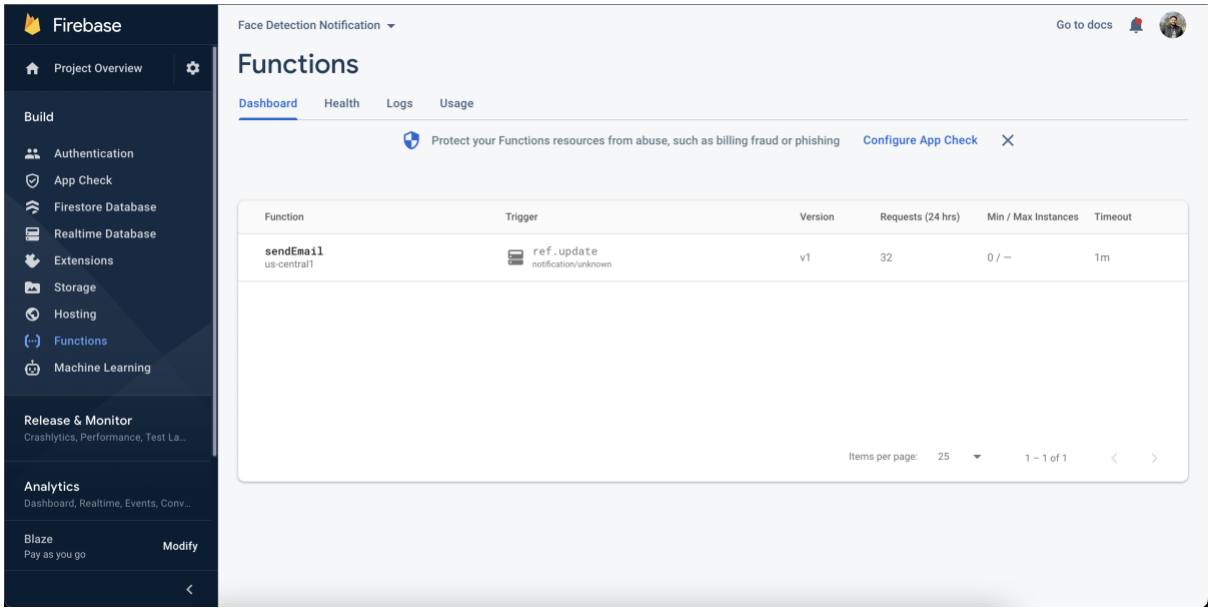
```
[? File functions/package.json already exists. Overwrite? Yes
 ✔  Wrote functions/package.json
[? File functions/.eslintrc.js already exists. Overwrite? Yes
 ? File functions/index.js already exists. Overwrite? Yes
 ✔  Wrote functions/.eslintrc.js
 ✔  Wrote functions/index.js
[? File functions/.gitignore already exists. Overwrite? Yes
 ✔  Wrote functions/.gitignore
[? Do you want to install dependencies with npm now? Yes
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: undefined,
npm WARN EBADENGINE   required: { node: '16' },
npm WARN EBADENGINE   current: { node: 'v18.4.0', npm: '8.12.1' }
npm WARN EBADENGINE }

up to date, audited 299 packages in 538ms

27 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

i  Writing configuration info to firebase.json...
i  Writing project information to .firebaserc...

✔  Firebase initialization complete!
```

After using the command, it will take a while to deploy the function and most probably give an error if billing details are not correct.

Check logs if you are facing unwanted errors.

```
Last login: Sat Jun 25 23:27:30 on ttys000
(base) parth_desai@Parths-Air functions %  firebase deploy

=== Deploying to 'face-detection-notification'...

i  deploying functions
Running command: npm --prefix "$RESOURCE_DIR" run lint

> lint
> eslint .

✔  functions: Finished running predeploy script.
i  functions: ensuring required API cloudfunctions.googleapis.com is enabled...
i  functions: ensuring required API cloudbuild.googleapis.com is enabled...
✔  functions: required API cloudbuild.googleapis.com is enabled
✔  functions: required API cloudfunctions.googleapis.com is enabled
i  functions: preparing codebase default for deployment
```

After successful deployment, the Functions tab should look like this.


Firebase
Project Overview
Build
Authentication
App Check
Firestore Database
Realtime Database
Extensions
Storage
Hosting
Functions
Machine Learning
Release & Monitor
Crashlytics, Performance, Test La...
Analytics
Dashboard, Realtime, Events, Conv...
Blaze
Pay as you go
Modify
Face Detection Notification
Go to docs
Functions
Dashboard
Health
Logs
Usage
Protect your Functions resources from abuse, such as billing fraud or phishing
Configure App Check
Function
Trigger
Version
Requests (24 hrs)
Min / Max Instances
Timeout
sendEmail
us-central1
ref.update
notification/unknown
v1
32
0 / –
1m
Items per page:
25
1 – 1 of 1

4. CONCLUSION

Cloud computing services are focused on sharing. Many services, including IaaS, SaaS, PaaS, and FaaS, are offered via cloud computing. FaaS and python scripts are used for real-time face detection and firebase as a database for the project. Any authenticated user can update the cloud function and real-time database from the google cloud console.

This report begins with a quick explanation of cloud computing and then lists the sources used in the project. Then explanation of the code for the various functions and how we implemented them. We were finally able to launch and integrate the project. The results were fantastic because the system sends emails each time an unknown person is detected by the algorithm.

No system in the world is perfect neither is this, many things can improve such as the facial recognition algorithm being pretty basic because the main aim of the project was to integrate with the cloud and that part is working seamlessly. Apart from that, there can be another cloud function that could be deployed for face data storage.

REFERENCES

[1] Chiara Turati, Viola Macchi Cassia, F. S., and Leo, I. Newborns face recognition: Role of inner and outer facial features. Child Development 77, (2006), 297–311.

[2] Bradski G, Kaehler A. Learning OpenCV: Computer vision with the OpenCV library. " O'Reilly Media, Inc."; 2008 Sep 24.

[3] Zhao, W., Chellappa, R., Phillips, P., and Rosenfeld, A. Face recognition: A literature survey. Acm Computing Surveys (CSUR) 35, 4 (2003), 399–458.

[4] Rajesh Piplode and Umesh Kumar Singh, "An Overview and Study of Security Issues & Challenges in Cloud Computing," International Journal of Advanced Research in Computer Science and Software Engineering, ISSN: 2277 128X, Volume-2, Issue-9, September 2012.

[5] P. Senthil, N. Boopal and R.Vanathi, "Improving the Security of Cloud Computing using Trusted Computing Technology," International Journal of Modern Engineering Research (IJMER), ISSN: 2249-6645, Volume-2, Issue-1, Jan-Feb 2012, pp-320-325.

[6] Ganesh V. Gujar, Shubhangi Sapkal and Mahesh V. Korade, "STEP-2 User Authentication for Cloud Computing," International Journal of Engineering and Innovative Technology (IJEIT), ISSN:2277-3754, Volume-2, Issue-10, April 2013.

[7] Pawle AA, Pawar VP. Face recognition system (FRS) on cloud computing for user authentication. International Journal of Soft Computing and Engineering (IJSCE). 2013 Sep;3(4):189-92.

[8] Voulodimos, Athanasios, et al. "Deep learning for computer vision: A brief review." Computational intelligence and neuroscience 2018 (2018).

[9] Bradski, Gary, and Adrian Kaehler. "OpenCV." Dr. Dobb's journal of software tools 3 (2000): 120.

[10] Moroney, L. (2017). The firebase real-time database. In The Definitive Guide to Firebase (pp. 51-71). Apress, Berkeley, CA.

[11] Moroney, Laurence. "Cloud functions for firebase." The Definitive Guide to Firebase. Apress, Berkeley, CA, 2017. 139-161