COVID-19 CHATBOT

CLOUD COMPUTING

DECEMBER 10, 2021

THE UNIVERSITY OF TEXAS AT ARLINGTON
SIMRAN SINGHI & SUCHARITHA CHEENEPALLI

Final Project

Project Title: Covid19 Chatbot

https://myapp-hbaazayfza-uc.a.run.app/

Application Overview:

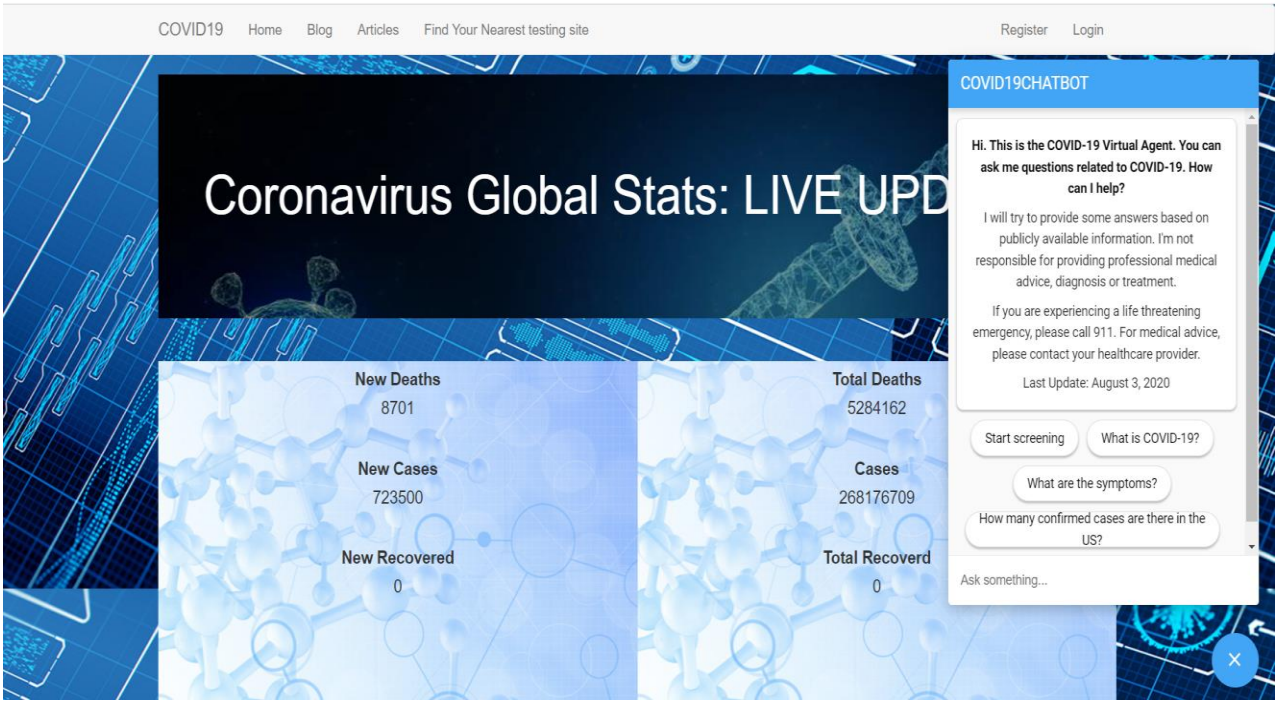
This is a web application built using the Python Flask framework. This project is a place where the users can look at the CDC updated Covid statistics on the dashboard. Users can access Covid articles through login or registration and also interact with the Chatbot which is designed to simulate conversation with human users, especially over the internet and provide Covid 19 related queries as per the CDC regulations integrated into the home page. The chatbot application is a Google cloud model which we have integrated into our application.

The Blog page shows a few interesting articles which can be accessed through login. Once the user logs in, they have a dashboard to view, edit create or update their articles. Altogether this application is user-friendly consisting of a register, login, homepage, Chatbot, Covid stats blogs a list of covid specific articles, and a dashboard containing the user's list of published articles.

User Interactions and functions enabled to by the service:

The user interacts with the application from the landing page where the user can view the latest Covid statistics as per CDC updates.

1. LANDING PAGE



2. REGISTER

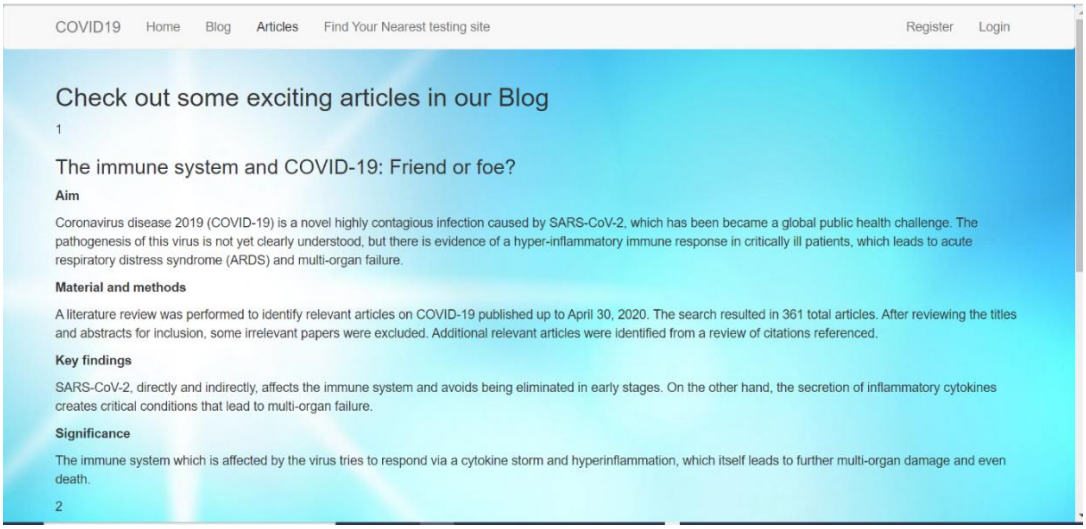
New users can register through the register page, or an existing user can log in going to the login page.

3. LOGIN

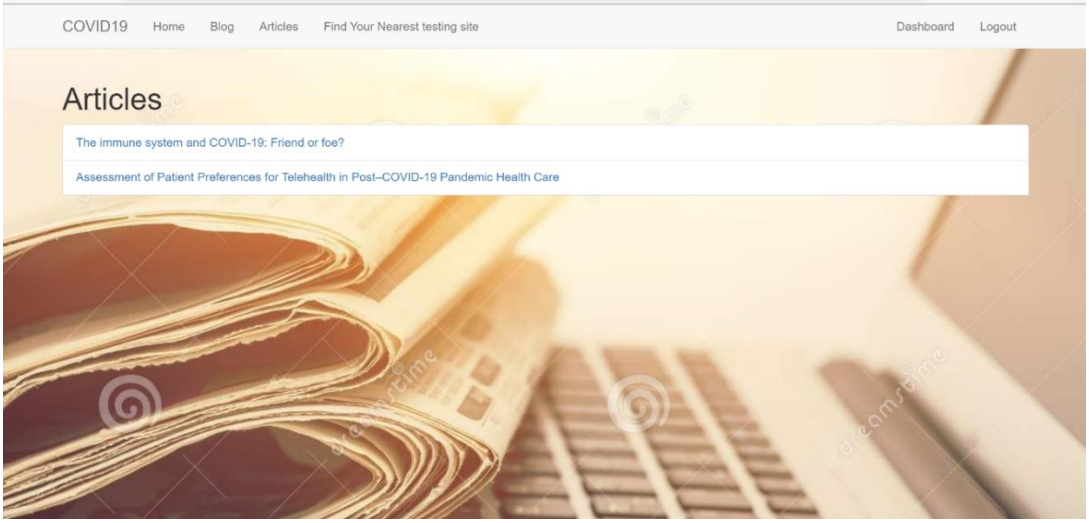
The chatbot gets activated asking for help on the home page. The user can open it and get their questions answered. For example, the user can ask queries related to safe things which can/cannot be done during the Covid 19situation. The agent should understand the query and give back an appropriate response as per the CDC guidelines. Once all questions are answered the chat asked for feedback and quits until the next open is requested.

4. BLOGS:

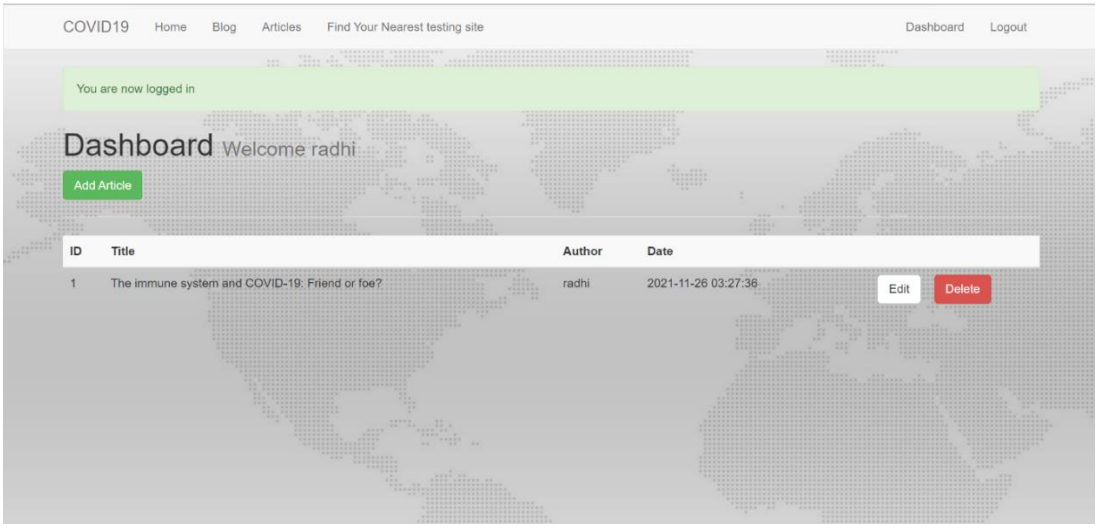
The Blog page shows some articles published by various users over time and if a user is interested, they can further login and publish the article through the dashboard “add Article button”. We have all the update, create, delete operations available in here.



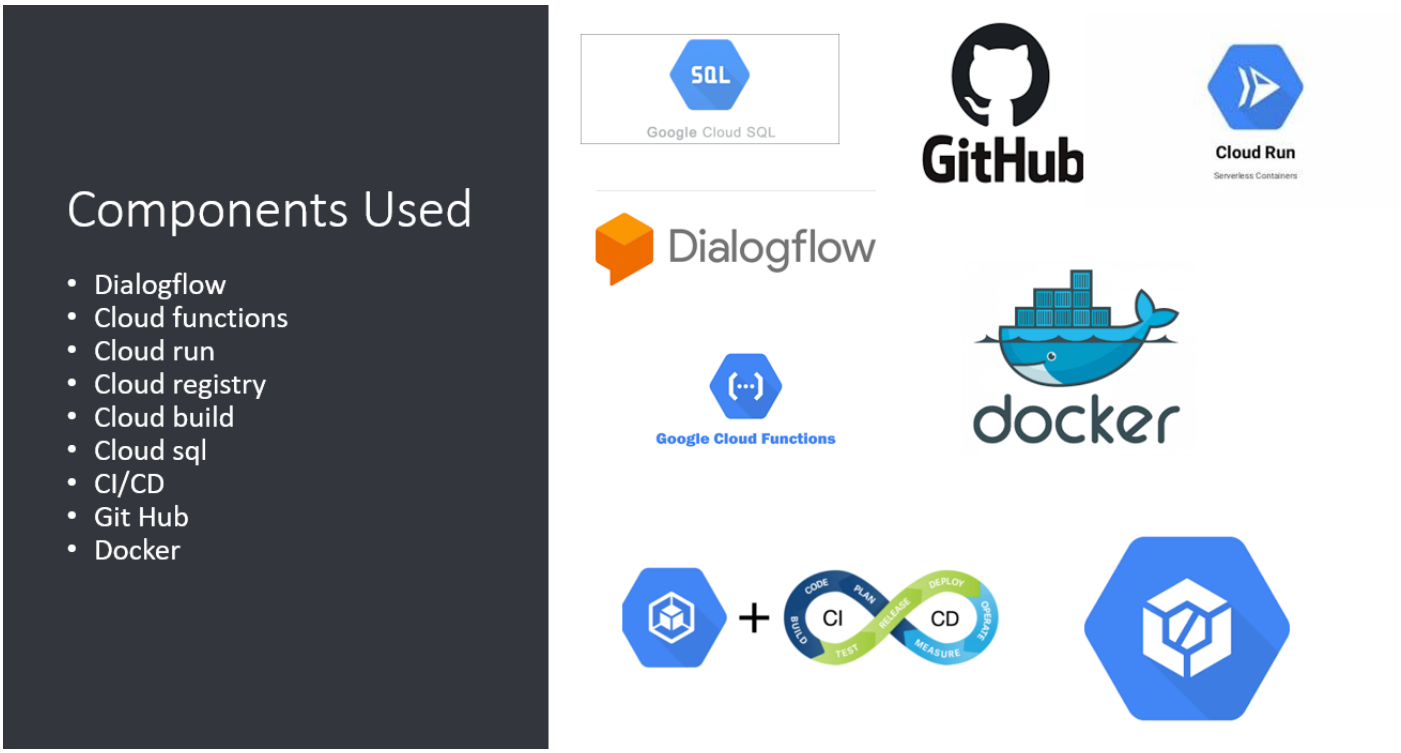
5. Articles published or added by the logged in user can be viewed here!



6. Dashboard: Articles can be added, Deleted or Updated From here.

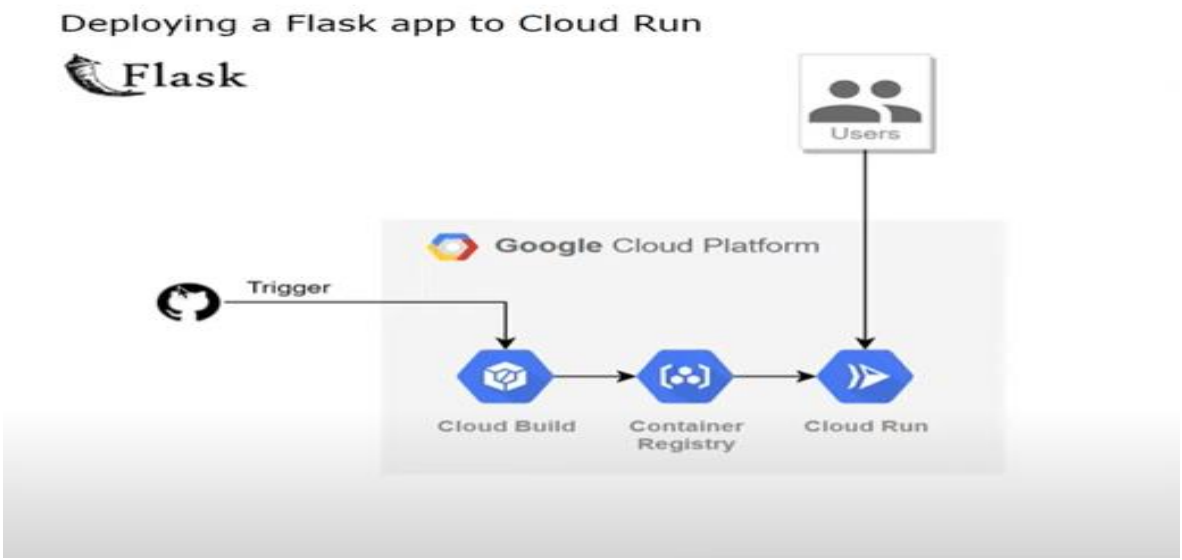


Components used:



1. **Dialogflow** and **Cloud Functions:** Used to create Covid-19 Chatbot.
2. **Cloud Run:** - Compute Platform to run container.
3. **Cloud Registry: -** Used to manage Docker images. Docker images are pushed to cloud registry.
4. **Cloud Build: -** Used to execute builds on google cloud platform and implement CI/CD.
5. **Cloud SQL: -** Used to setup Application database.
6. **GITHUB:** Version control system for code hosting and collaboration.
7. **Docker:** To dockerize the Application.
8. **Technologies used for Creating the Application:**
    a. Python Language
    b. Flask Framework
    c. MySQL
    d. VScode Editor

Architecture



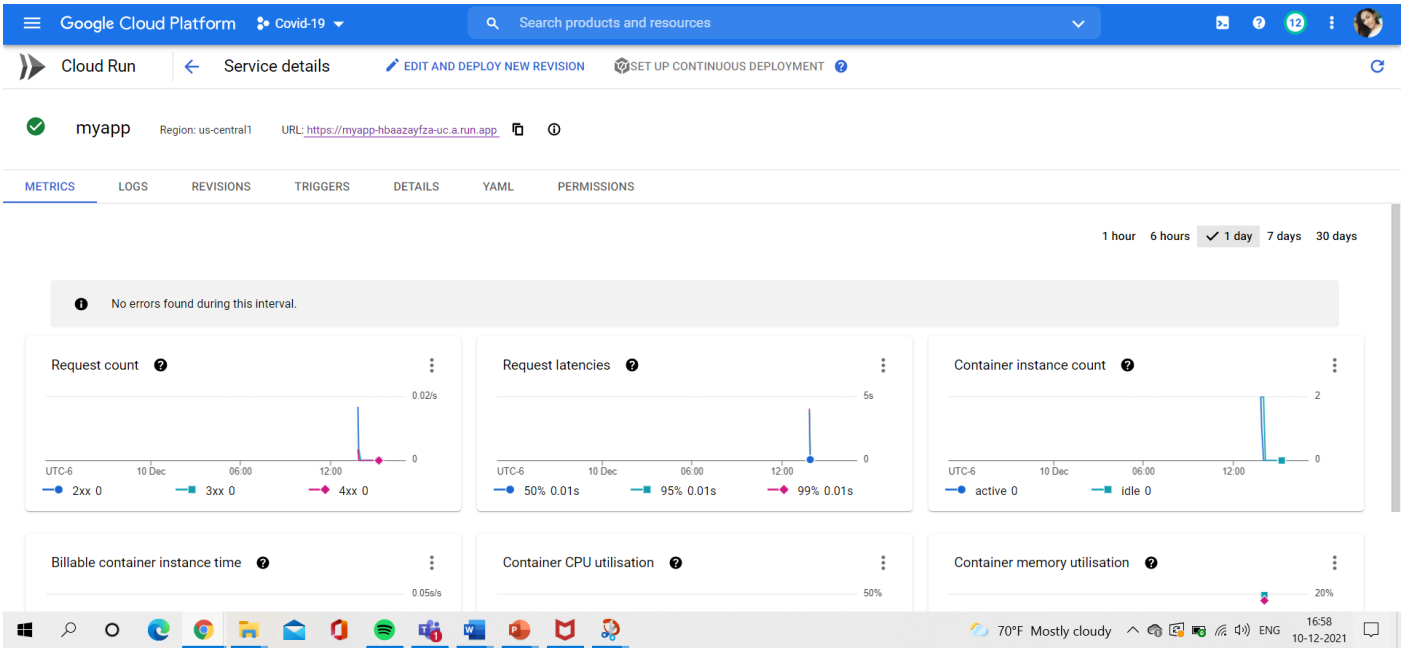
Compute service: Cloud Run

Cloud Run is a managed compute platform that enables you to run containers that are invocable via requests or events. Cloud Run is serverless: it abstracts away all infrastructure management, so you can focus on what matters most — building great applications. The pipeline was configured to be triggered every time new code was pushed into the master branch of the Cloud Source Repository. Once that happens, the pipeline:

- Builds the Docker image;
- Pushes the built Docker image into Container Registry;

- Executes the build using cloud Build;

The Cloud run instance



https://myapp-hbaazayfza-uc.a.run.app

The application is built and deployed and can access it through the endpoint generated by

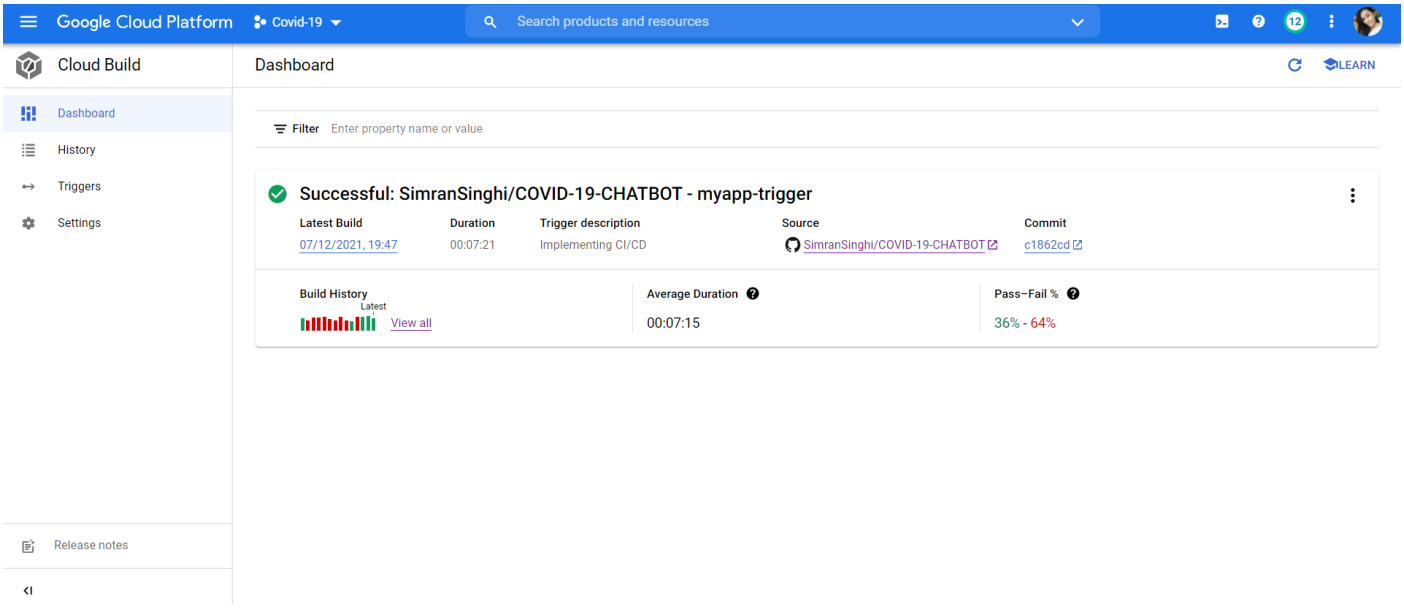
Cloud Run.

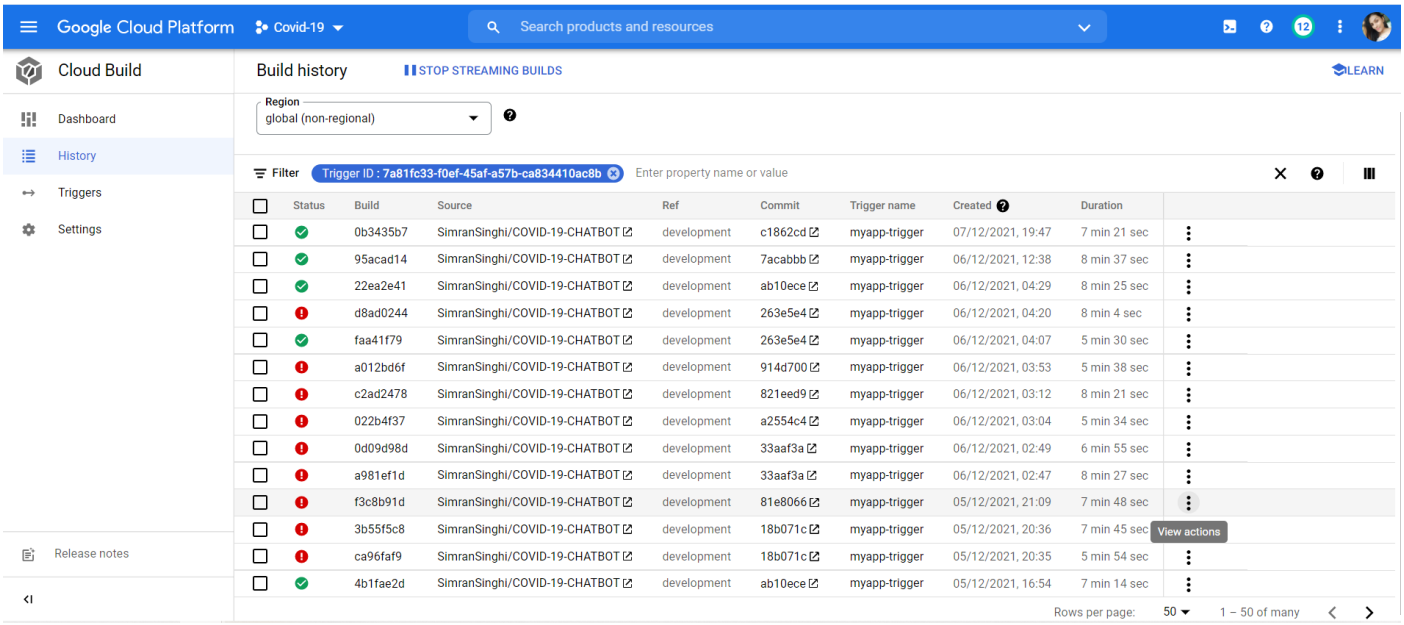
Cloud Source Repository

Google Cloud Source Repositories are fully-featured, private Git repositories hosted on Google Cloud Platform. Collaborate easily and securely manage your code on a fully featured, scalable, private Git repository.

Cloud Build

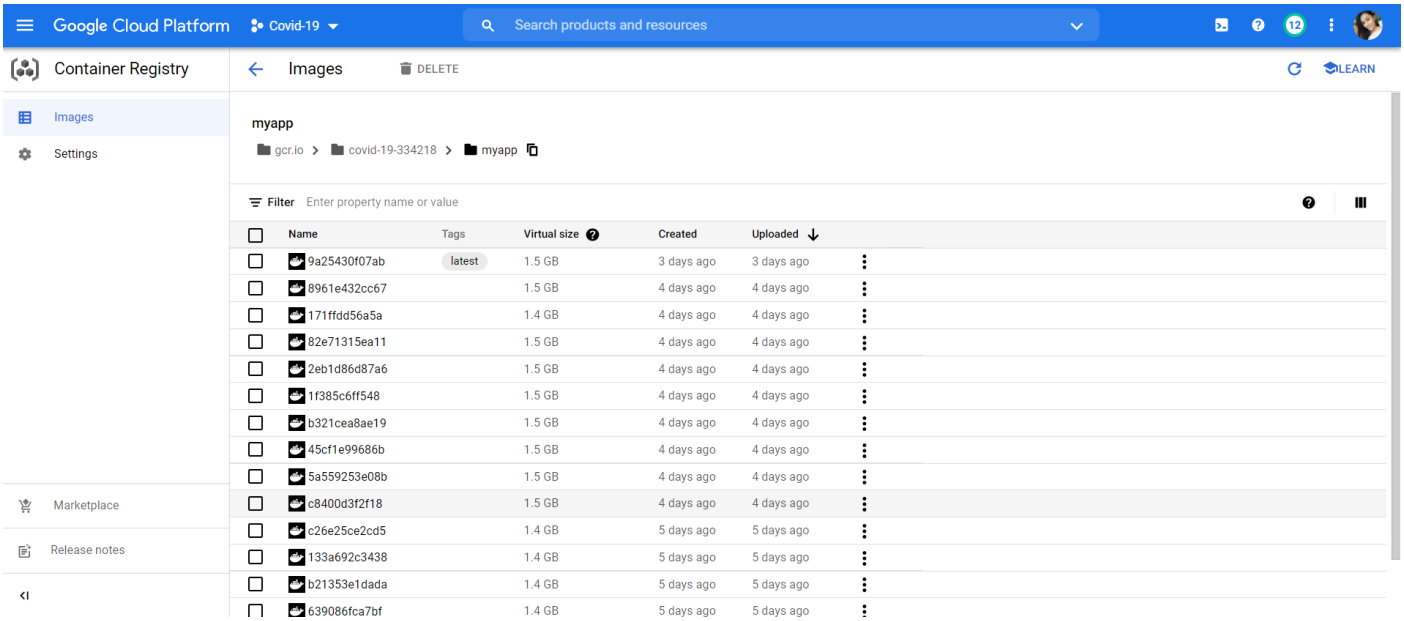
Cloud Build is a service that executes your builds on Google Cloud Platform infrastructure. Cloud Build can import source code from Google Cloud Storage, Cloud Source Repositories, GitHub, or Bitbucket, execute a build to your specifications, and produce artifacts such as Docker containers or Java archives.





Container Registry

Container Registry is a private container image registry that runs on Google Cloud.

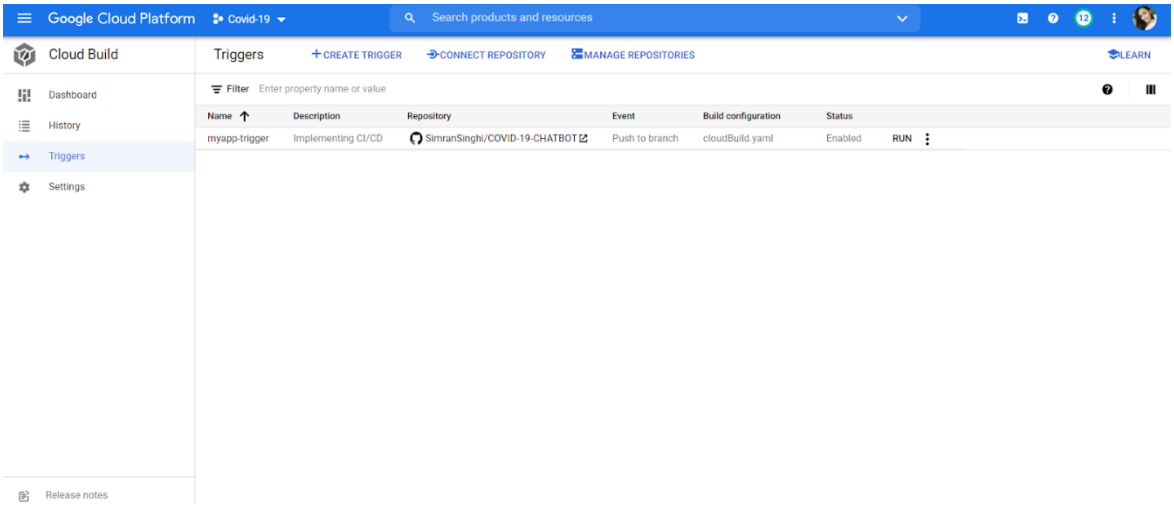


Configuring Cloud Build Pipeline

The steps to build pipeline defined in YML file called [cloudBuild.yaml](). The pipeline is composed of three steps.

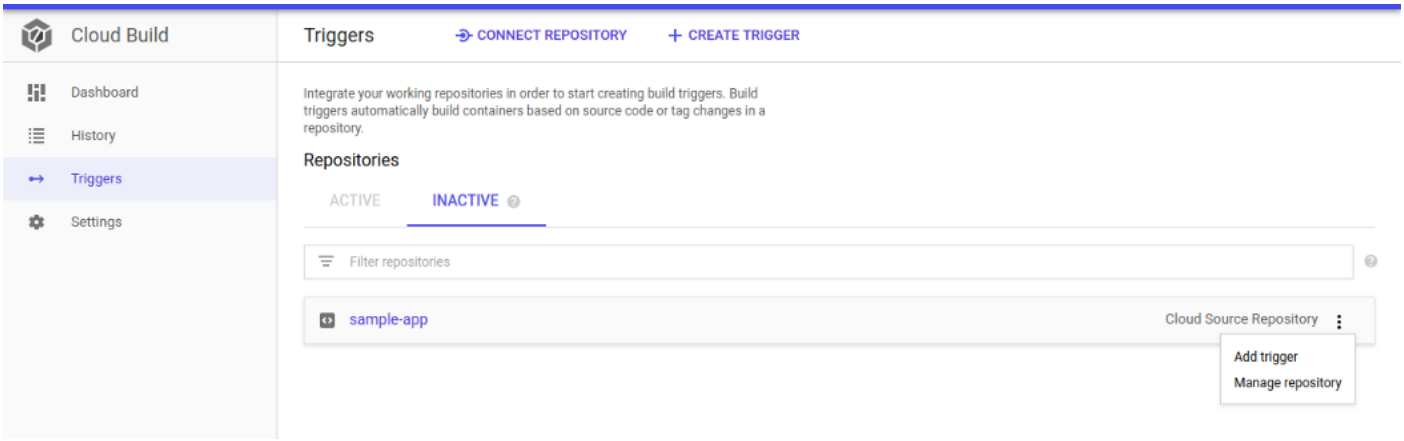
1. The first step is responsible for building and tagging the Docker image of our application.
2. The second step is responsible for pushing the Docker image built on the step one to Container Registry.
3. The third step is responsible for deploying the Docker image to Cloud Run.

The dashboard showing the details of all the triggers:



Set up the Cloud Build Trigger:

From the menu, select **Cloud Build**, select **Trigger**, Select the Repository we created in CSR, click on: and select **Add trigger**

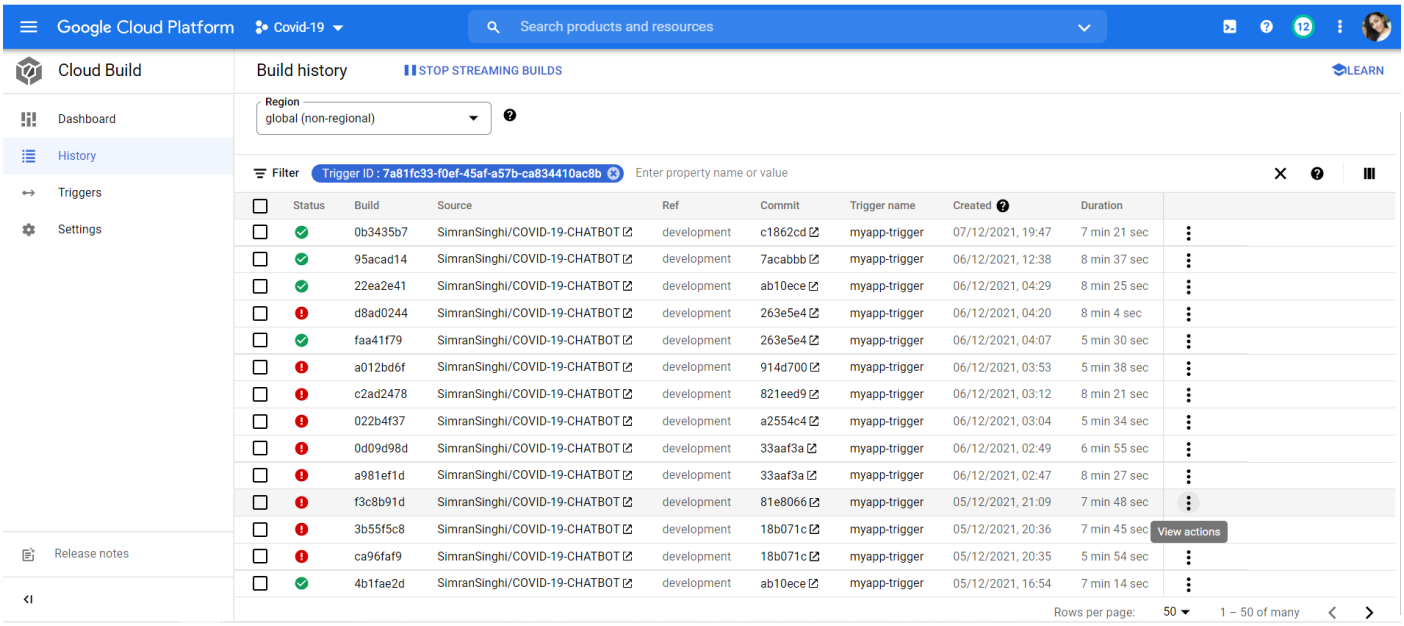


Here, we specify:

- The **Name** and **Description** of the trigger;
- That the build should be triggered whenever stuff is pushed into the **master** branch of the repository.
- That the build configuration is provided by the cloudbuild.yaml file from our repository;

- That the **email** variable from our [cloudbuild.yaml] should be replaced with the **service account** value. As described before, this variable is used for managing our generated Docker image, as well as to set the name of the deployed Cloud Run service.

The Application Revisions deployed using the Trigger.

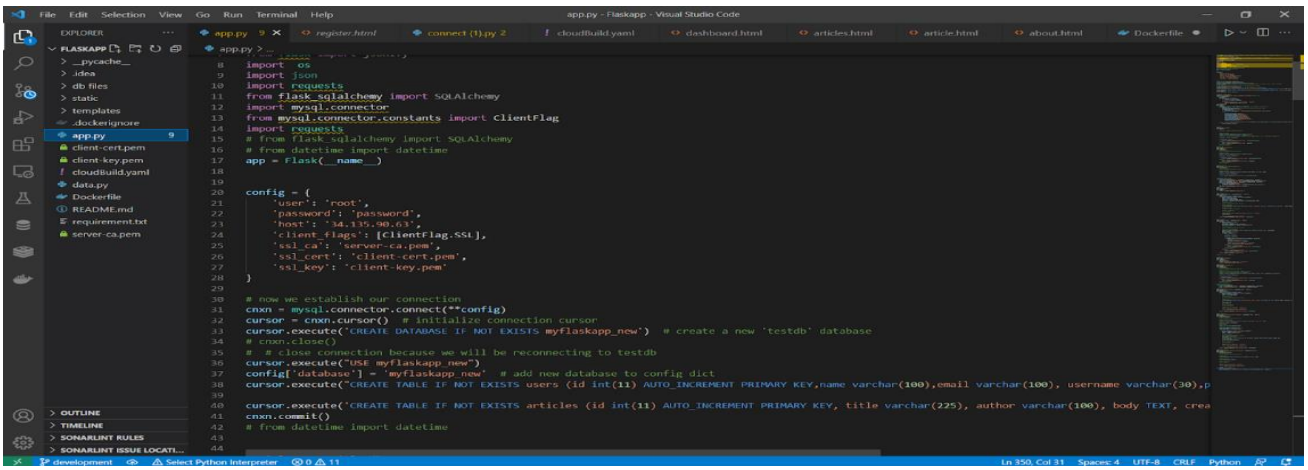


Cloud SQL

We choose Cloud SQL because it is fully managed relational database service for MySQL, PostgreSQL, and SQL Server. The relational databases have their rich extension collections, configuration flags and developer ecosystem, without the hassle of self-management.

Cloud SQL instance: The instance can be created by selecting all the preferred configurations and later we connected the instance using the MySQL client in Cloud Shell.

Connection Code



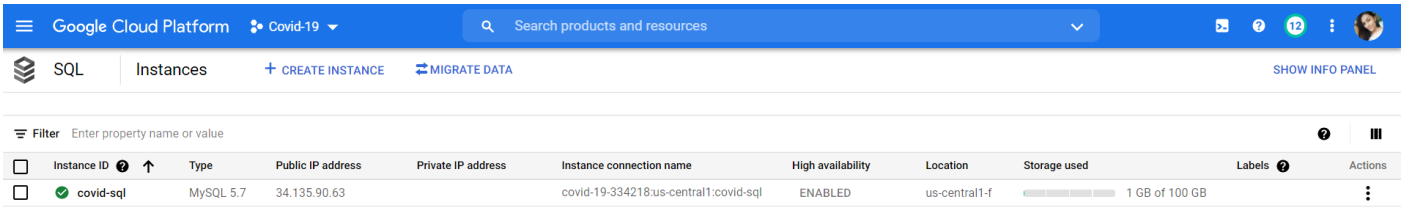
Creating a database and adding tables to our database.

For our application, we require two tables namely Users: for storing all the registration and login credentials and Articles table to maintain and perform CRUD operations on articles.

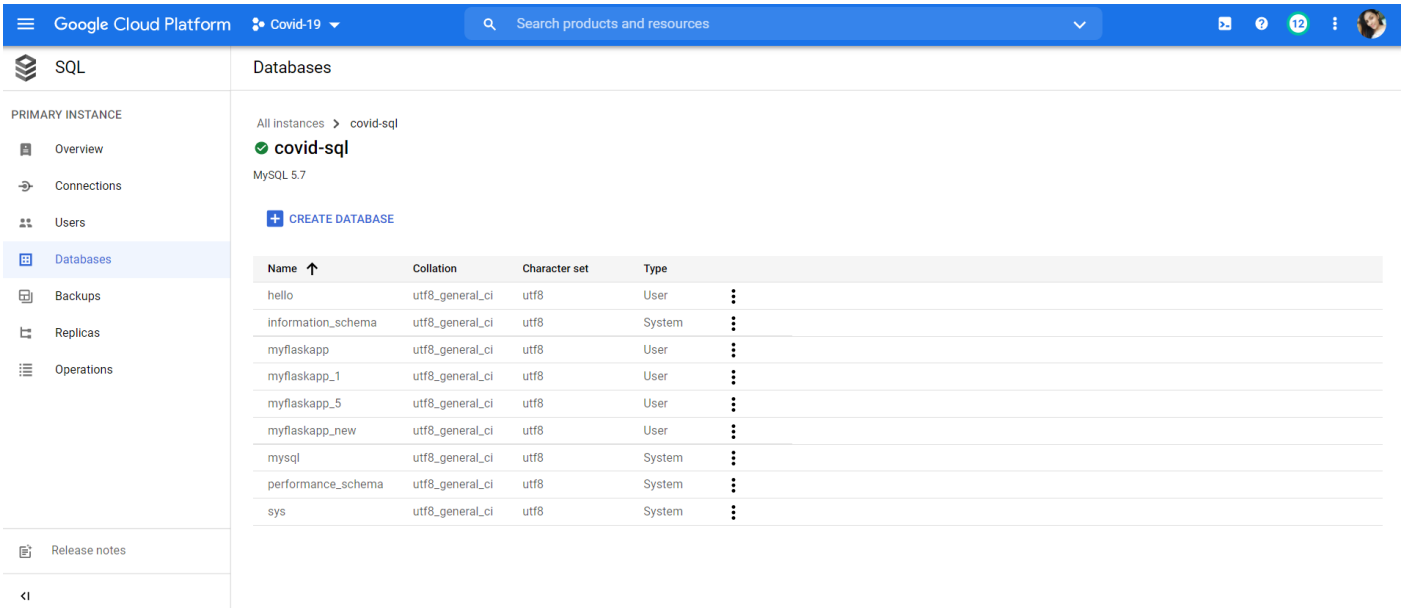
For the creation of them into a database, we created a database and added all the variables required in each table appropriately**.**

The database is created and tables required for our application have been added.

CLOUD SQL INSTANCE

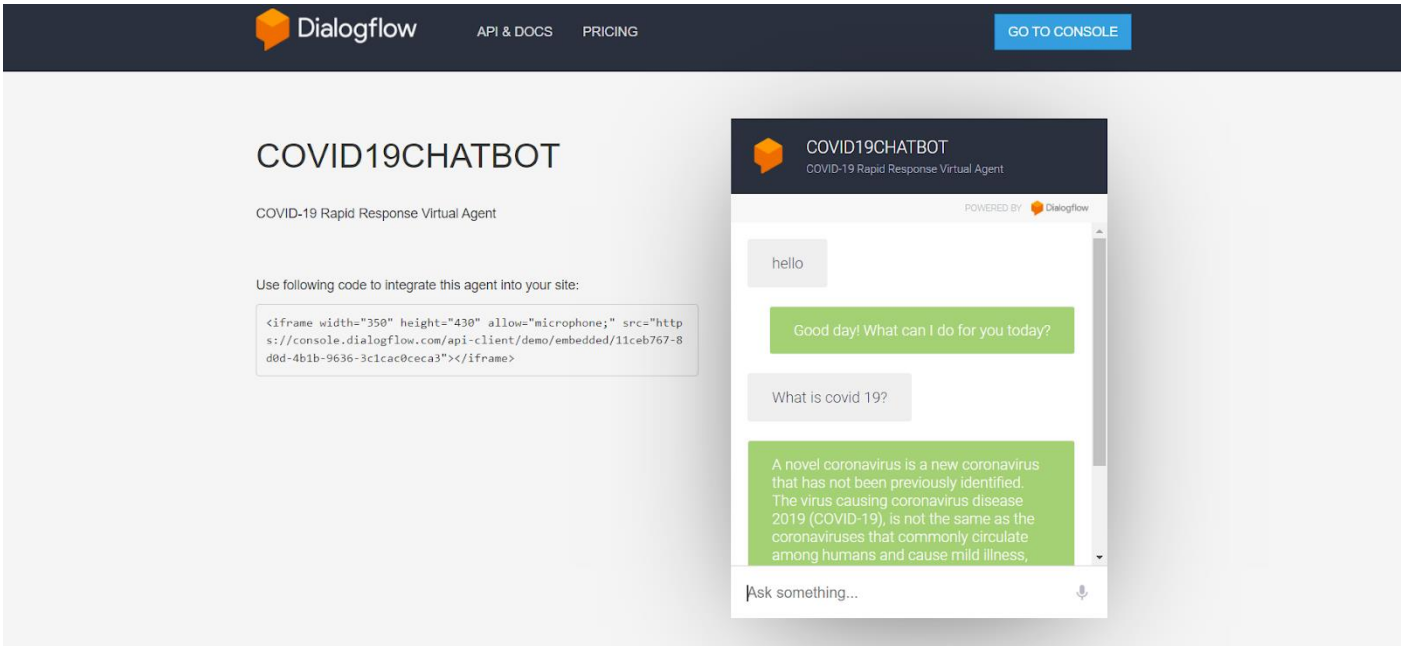


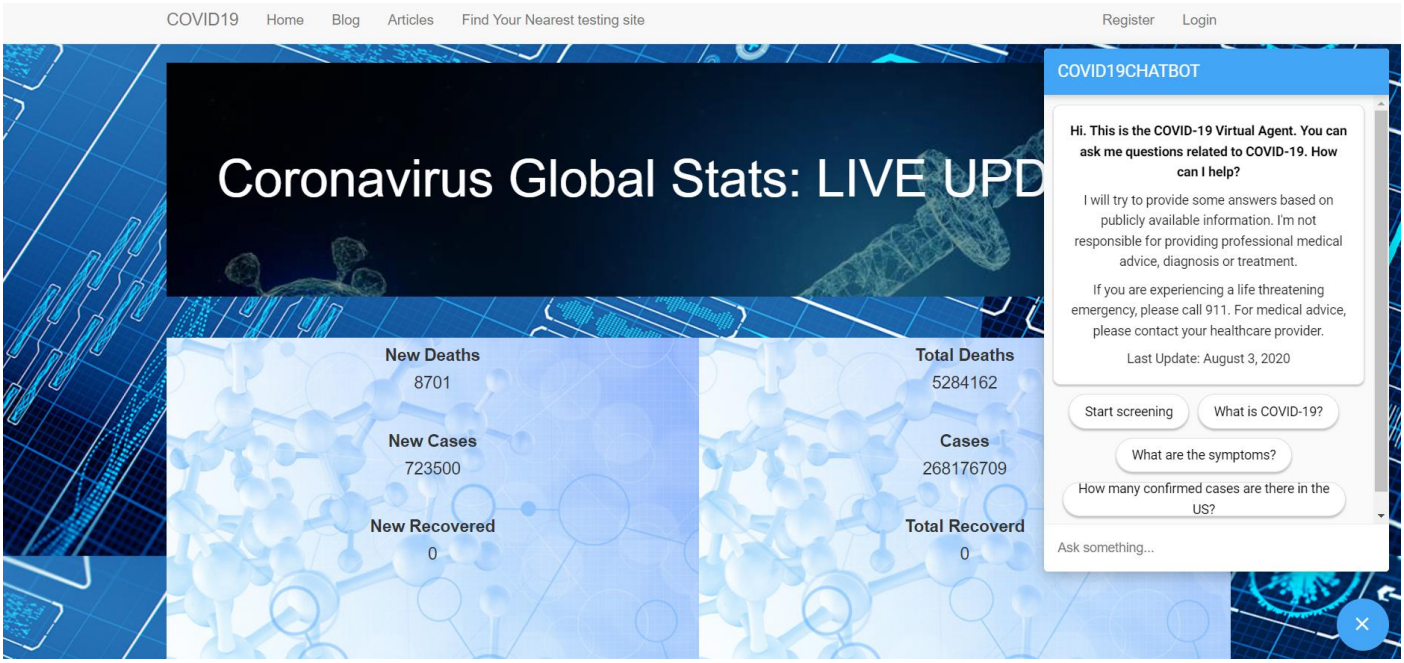
DATABASES



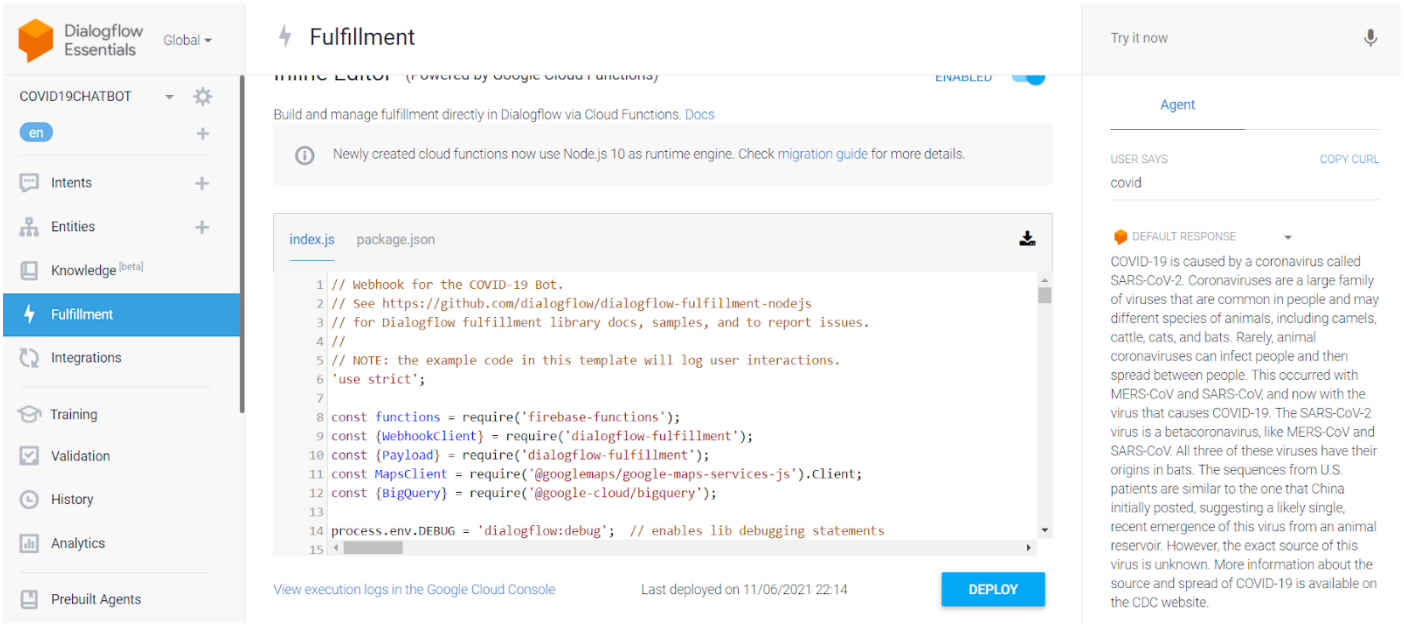
Dialog Flow: Virtual Agent:

WEB DEMO

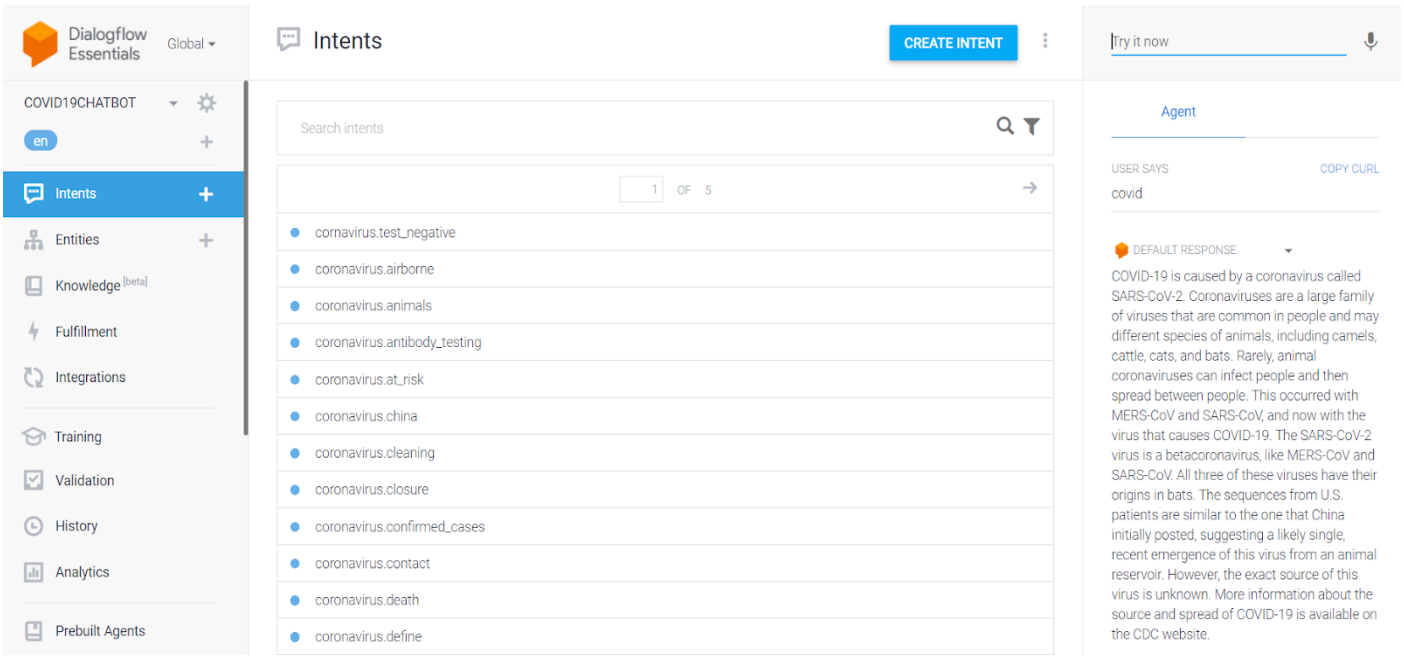




1. Created a new agent
2. Uploaded the code
3. Enabled fulfillment via cloud functions



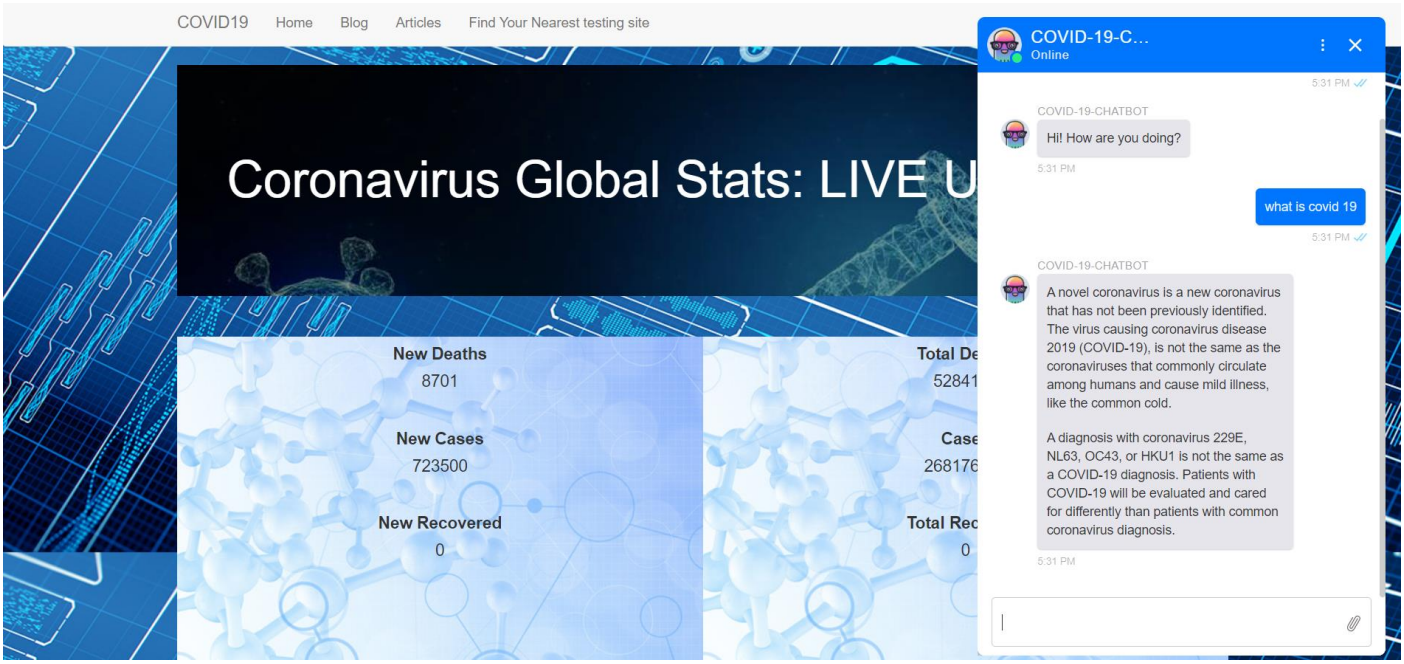
4. Train the model for the intents enable webhook to enable API response.
5. Test the Model using Simulator



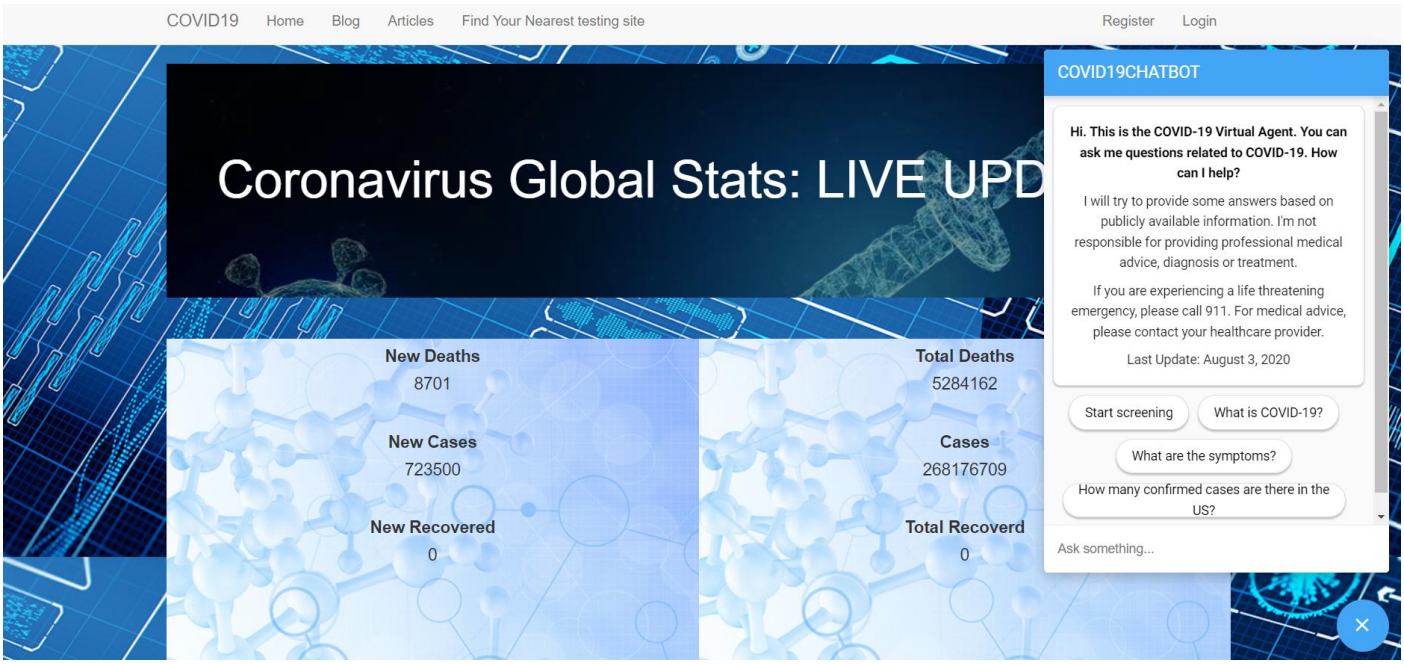
6. Integration of DialogFlow chatbot. We have tried two ways. One is through web demo and other is

using dialogflow messenger. The deployed Application contains DialogFlow Messenger.

7. WEBDEMO:



8. Dialoflow Messenger



APIs used to fetch Covid – 19 Statistics

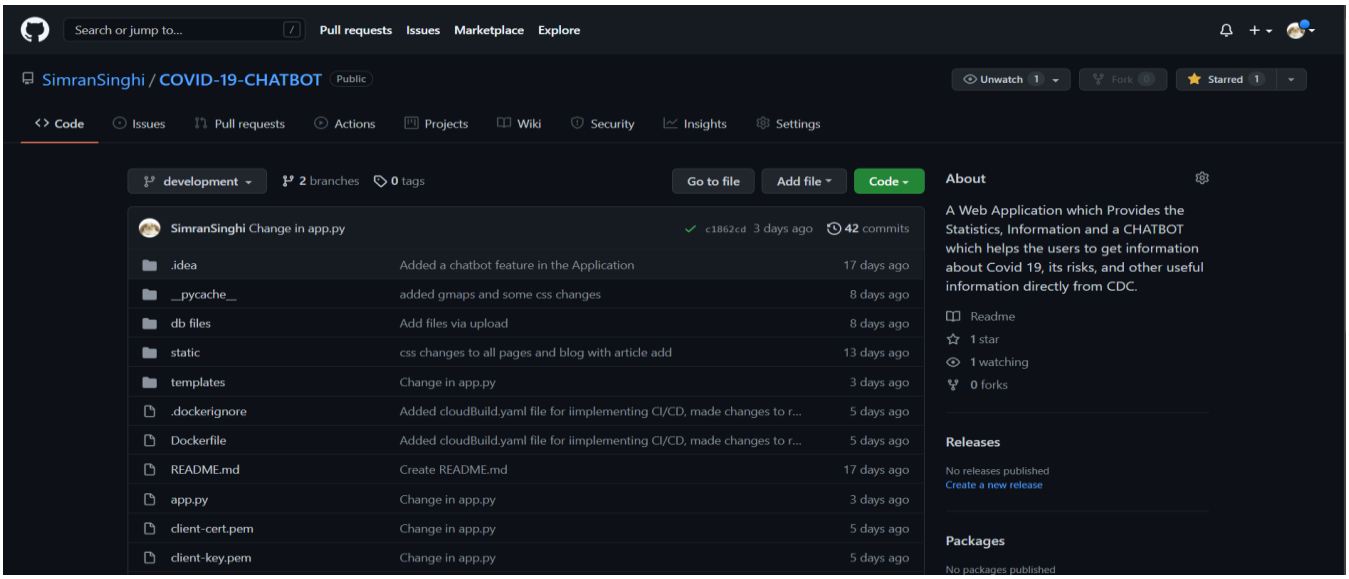
Major code processes/workflows controlling the functionalities in the Application

API: We have used an external API to show the real-time stats of Covid 19. The API it pulls the data containing a number of recovers, totally recovered, newly infected, total deaths, and more. All this information is in the API link below:

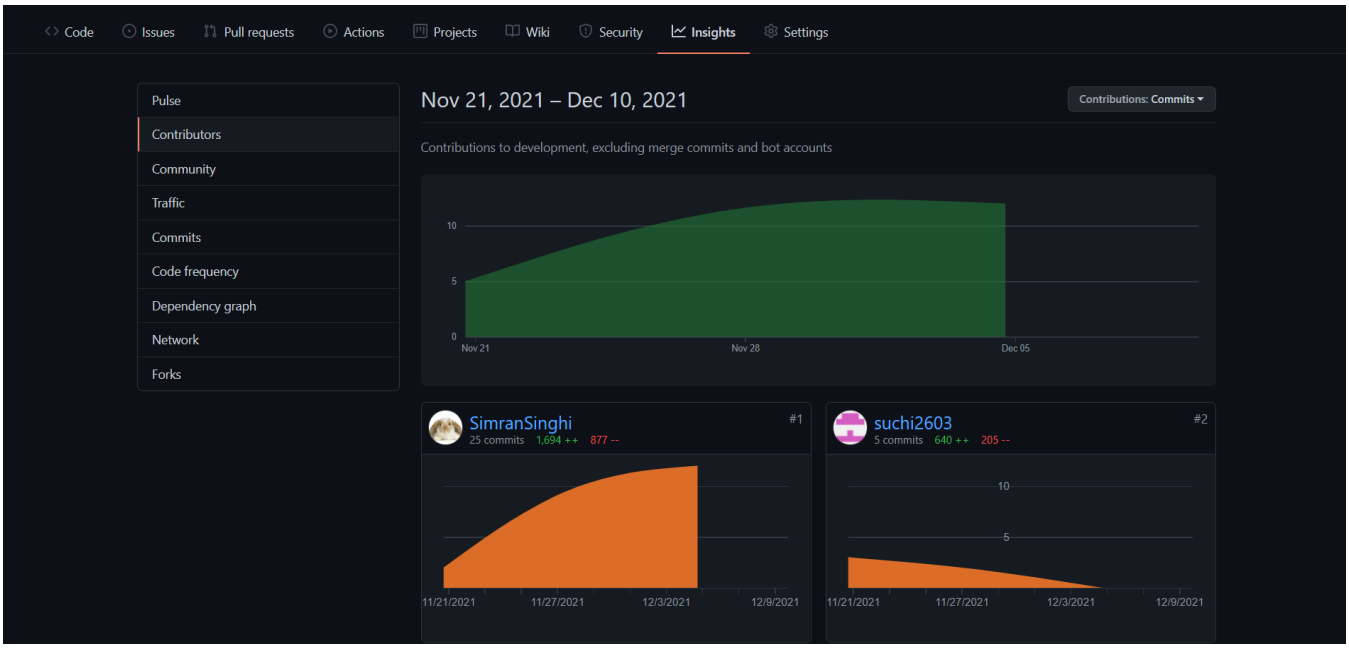
https://api.covid19api.com/summary

Version Control System – GITHUB

https://github.com/SimranSinghi/COVID-19-CHATBOT



Contributions:



Implementation Plan

Who is responsible for which parts of implementation?

To date, we have researched various project ideas and came up with this idea. We have built an application that suit the project requirements.

Responsibilities and roles:

Simran Singhi

Project plan, Requirements Gathering, Dialog flow chatbot development and integration of chatbot to the Application, setting up Github repository, Deployment to Google Cloud Platform using Cloud Run, Cloud Registry and Cloud Build. Implementing CI/CD. Database connection to cloud sql and setting up cloud sql for the project, Bug fixes and Testing of Application, Some contribution to the report and Presentation.

Sucharitha:

Project Planning, Requirements Gathering, Application development, Login, Register, Home, Dashboard, Articles, and Blog page design, Cloud DB Connection assistance, Testing and Bug fixes and report ppt preparation.

Planned milestones and accomplished work:

We managed to complete the application that meets all the functionalities discussed in spite of facing many roadblocks.

Date	**List of Functionalities to be completed**	**Status** Complete(C), Pending (P) and InProgress(IP)
October 2-14, 2021	➢ Project kick off discussion meeting ➢ Project requirements gathering and specification	**Complete**

October 14, 2021	Project Proposal	
October 15-28, 2021	➢ Project flow understanding ➢ Setting up Dialog flow and getting started with the Intent. ➢ Code implementation start ➢ Check for roadblocks or problems ➢ Clear all issues	**Completed**
October 28, 2021	Clearing Roadblocks	
October 29-November 21, 2021	➢ Integrating the APIs ➢ Code Implementation for web app using python and flask ➢ Integrating the chatbot in the web App ➢ Testing the integrated web App	**Completed**
November 22	➢ Start research for deploying the application	**Completed**
November 23-December 2, 2021	➢ Work on the deployment work	**Completed**
December 2, 2021	Code Completion and deployment completion	
December 2-6, 2021	Prepare for the project final presentation	**Completed**
December 6	Project Presentation	

The Application was tested locally by both the members after each modifications and then finally deployed to the Google Cloud Platform. Later on, the deployed version is tested thoroughly. During the testing, we will have tried asking questions in the regions we have trained the chatbot. The application Register, Login, Logout, Article CRUD operations are all working fine as expected.

Signoff: The project proposal, the tasks, and the timelines are agreed upon by Sucharitha Cheenepalli and Simran Singhi.