# ABSTRACT

An AI chatbot using a machine learning model based on intents. The chatbot is designed to understand the intent behind user input and provide appropriate responses. The article covers the process of developing and training the machine learning model on a dataset of conversational data, and the integration of the model into the chatbot's backend. The chatbot was tested for accuracy and performance using various techniques, including human evaluation and automated testing. It also covers the challenges and limitations of building an effective chatbot, including handling user input errors and maintaining context across conversations. It provides insights into the potential of using machine learning models to build intelligent chatbots that can provide personalized and efficient support to users in various industries.

# ACKNOWLEDGEMENT

The successful completion of this mini project report would not have been possible without the support and assistance of many individuals and organizations. I feel immensely blessed to have gotten this during the course of my program. I would like to take this opportunity to offer my earnest admiration to each and every one of them.

I am indebted and thankful to my learned and revered Dr. S. N. Zaware, Head of Department of Computer science and Engineering, for her encouraging support and providing a meticulous platform to learn.

I owe my deepest gratitude to our Guide Prof. Mrs. Prajwal Gaikwad for her upbeat personality, kindness, encouraging support and willingness to help, have tangibly and greatly improve the quality of my internship report and brought up to its present status. Thanks classmates who helped me directly or indirectly to accomplish my work. Finally, I thank all my teachers, who were the people, who prepared us for this endeavour. I owe you all my success.

## INTRODUCTION:

Machine learning models are growing in popularity for a variety of applications, including speech processing, natural language processing, and image recognition. However, deploying and scaling these models can pose challenges, particularly for those without a strong background in infrastructure management. Fortunately, cloud providers like Google Cloud Platform have developed serverless platforms to address this challenge. In this article, we will explore how to deploy a machine learning model as a serverless application using Google App Engine. Google App Engine is a fully managed serverless platform that enables users to deploy web applications in various programming languages such as Python, Java, and Go. By providing an environment for running applications, it eliminates the need to worry about infrastructure management, server scaling, and deployment.

## Cloud Computing

Cloud computing is a term that refers to the delivery of computing resources and services over the internet, including storage, software, and processing power. It provides users with a flexible and scalable infrastructure without the need to invest in expensive hardware and software.

Cloud computing offers several benefits, including:

1. Scalability: Cloud computing enables users to scale their infrastructure up or down based on demand. Additional resources can be allocated dynamically as the workload increases, ensuring that applications can handle sudden spikes in traffic without manual intervention.
2. Cost-effectiveness: Cloud computing can be more cost-effective than traditional IT infrastructure, as users only pay for the resources they use. Cloud providers offer a range of pricing models, including pay-as-you-go, subscription-based, and reserved instances.
3. Accessibility: Cloud computing services can be accessed from anywhere with an internet connection, enabling users to work from anywhere in the world. This makes it easy for distributed teams to collaborate and share resources.
4. Security: Cloud providers invest heavily in security and offer a range of security features, including data encryption, firewalls, and identity and access management. They also comply with industry standards and regulations, ensuring that data is secure and compliant.

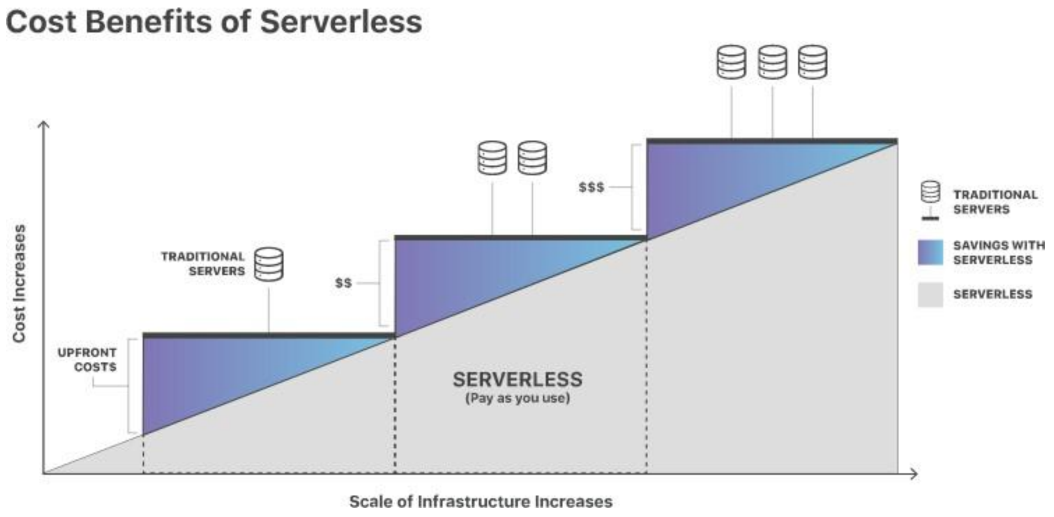
Cloud computing involves the use of remote servers on the internet to store, manage, and process data. Cloud computing providers such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform offer various services, including virtual machines, storage, databases, and serverless computing.

### Serverless Cloud Computing:

Serverless computing is a method of providing backend services on an as-used basis. A serverless provider allows users to write and deploy code without the hassle of worrying about the underlying infrastructure. A company that gets backend services from a serverless vendor is charged based on their computation and do not have to reserve and pay for a fixed amount of bandwidth or number of servers, as the service is auto-scaling.

In the early days of the web, anyone who wanted to build a web application had to own the physical hardware required to run a server, which is a cumbersome and expensive undertaking.

Then came cloud computing, where fixed numbers of servers or amounts of server space could be rented remotely. Developers and companies who rent these fixed units of server space generally over-purchase to ensure that a spike in traffic or activity will not exceed their monthly limits and break their applications. This means that much of the server space that gets paid for can go to waste. Cloud vendors have introduced auto-scaling models to address the issue, but even with auto-scaling an unwanted spike in activity, such as a DDoS Attack, could end up being very expensive.



Serverless computing allows developers to purchase backend services on a flexible 'pay-as-you-go' basis, meaning that developers only have to pay for the services they use. This is like switching from a cell phone data plan with a monthly fixed limit, to one that only charges for each byte of data that actually gets used.

The term 'serverless' is somewhat misleading, as there are still servers providing these backend services, but all of the server space and infrastructure concerns are handled by the vendor. Serverless means that the developers can do their work without having to worry about servers at all.

### What kind of backend services can serverless computing provide?

Most serverless providers offer database and storage services to their customers, and many also have Function-as-a-Service (FaaS) platforms, like Cloudflare Workers. FaaS allows developers to execute small pieces of code on the network edge. With FaaS, developers can build a modular architecture, making a codebase that is more scalable without having to spend resources on maintaining the underlying backend.

### What are the advantages of serverless computing?

- **Lower costs** - Serverless computing is generally very cost-effective, as traditional cloud providers of backend services (server allocation) often result in the user paying for unused space or idle CPU time.
- **Simplified scalability** - Developers using serverless architecture don't have to worry about policies to scale up their code. The serverless vendor handles all of the scalings on demand.

- **Simplified backend code** - With FaaS, developers can create simple functions that independently perform a single purpose, like making an API call.
- **Quicker turnaround** - Serverless architecture can significantly cut time to market. Instead of needing a complicated deployment process to roll out bug fixes and new features, developers can add and modify code on a piecemeal basis.

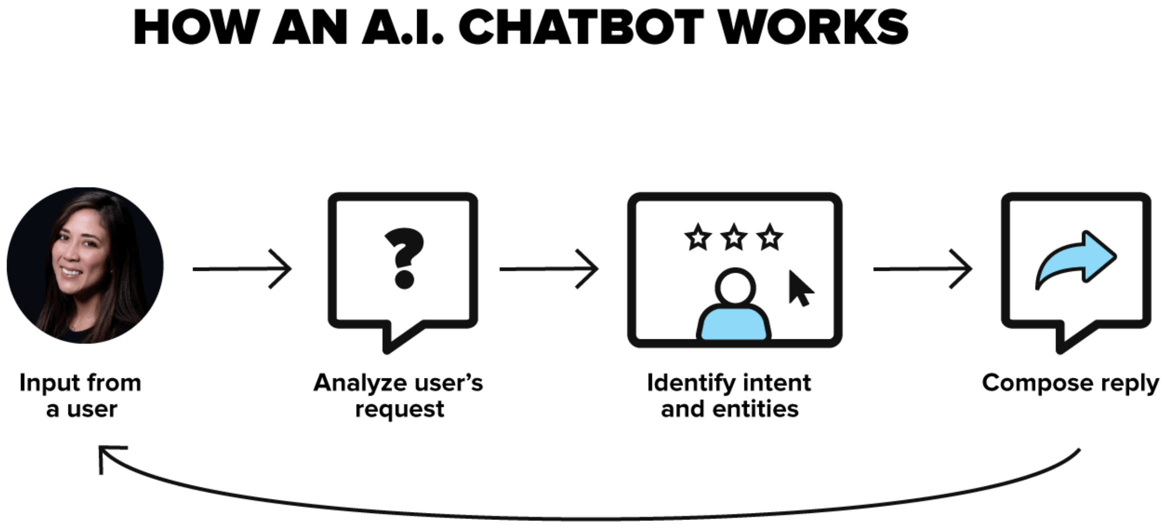
### What is next for serverless?

Serverless computing continues to evolve as serverless providers come up with solutions to overcome some of its drawbacks. One of these drawbacks is cold starts. Typically when a particular serverless function has not been called in a while, the provider shuts down the function to save energy and avoid over-provisioning. The next time a user runs an application that calls that function, the serverless provider will have to spin it up fresh and start hosting that function again. This startup time adds significant latency, which is known as a 'cold start'. Once the function is up and running it will be served much more rapidly on subsequent requests (warm starts), but if the function is not requested again for a while, the function will once again go dormant. This means the next user to request that function will experience a cold start. Up until fairly recently, cold starts were considered a necessary trade-off of using serverless functions.

### Chat Bot Model

For the purposes of this article, I have generated a machine-learning model for AI Chatbot, a machine learning model using intents can be generated. The model is designed to identify the underlying purpose or meaning behind a user's input, which is known as the intent. By recognizing the user's intent, the chatbot can provide an appropriate response.

For instance, a user might input "What's the weather like today?" which is an intent to get information about the weather. By analyzing the user's input, the model can identify the intent and respond with the appropriate information about the weather.



## Google App Engine

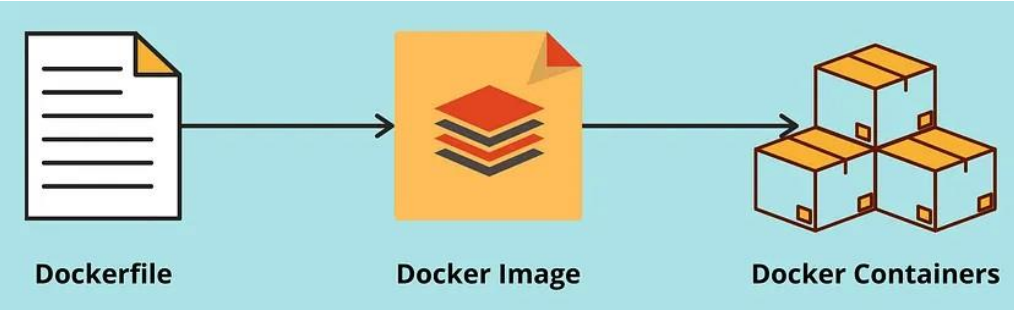
App Engine is a fully managed serverless application platform product from Google, which allows you to easily deploy your web app. Deploying an app is as simple as running the command gcloud app deploy. Since it's serverless, you don't have to worry about managing servers or any infrastructure. It scales automatically in response to the number of requests the app receives. Therefore, when there is no traffic, it scales down to zero. And you only pay for what you use!
In this case, we are going to use this serverless product offering from Google Cloud Platform (GCP) to serve our machine learning model as an app for others to interact with.

### TECHNOLOGIES & RECOURCES USE

- GCP ( Google App Engine )
- Streamlit
- Dockerfile Container
- Programing Language: Python

### DOCKER CONTAINER

Imagine you built an app and shared all the project files to run that app on another machine. If you are lucky it will run as-is. But the majority of the time, you will have to do some debugging to make the app work. Usually, it's to do with missing libraries or incompatible library versions, or missing files.



**Deploy a machine learning model on App Engine, we need to follow the steps below:**

1. **Train and save the machine learning model:**
   Before deploying a machine learning model, we need to train it and save it in a format that can be used by our application. We can use popular machine learning libraries such as TensorFlow, Scikit-learn, and Keras to train and save model
2. **Create an application:**
   To deploy our machine learning model on App Engine, we need to create a web application that can receive requests and provide predictions. We can use popular web frameworks such as to create the application(We have made use of Streamlit).
3. **Deploy the application to the App Engine:**
   Once we have created the application and defined the API endpoints, we can deploy the application to App Engine. To deploy the application, we must create a new project on the Google Cloud Console and enable the App Engine API. We can then use the gcloud command-line tool to deploy the application. The deployment process involves uploading our application code, defining the runtime environment, and configuring the scaling behaviour.

4. **Test and scale the application:**
After deploying the application, we can test it using various tools such as Curl or Postman. We can also monitor the application using the Google Cloud Console, and scale it based on traffic and demand.

**Dialogflow Code (ML Model)**

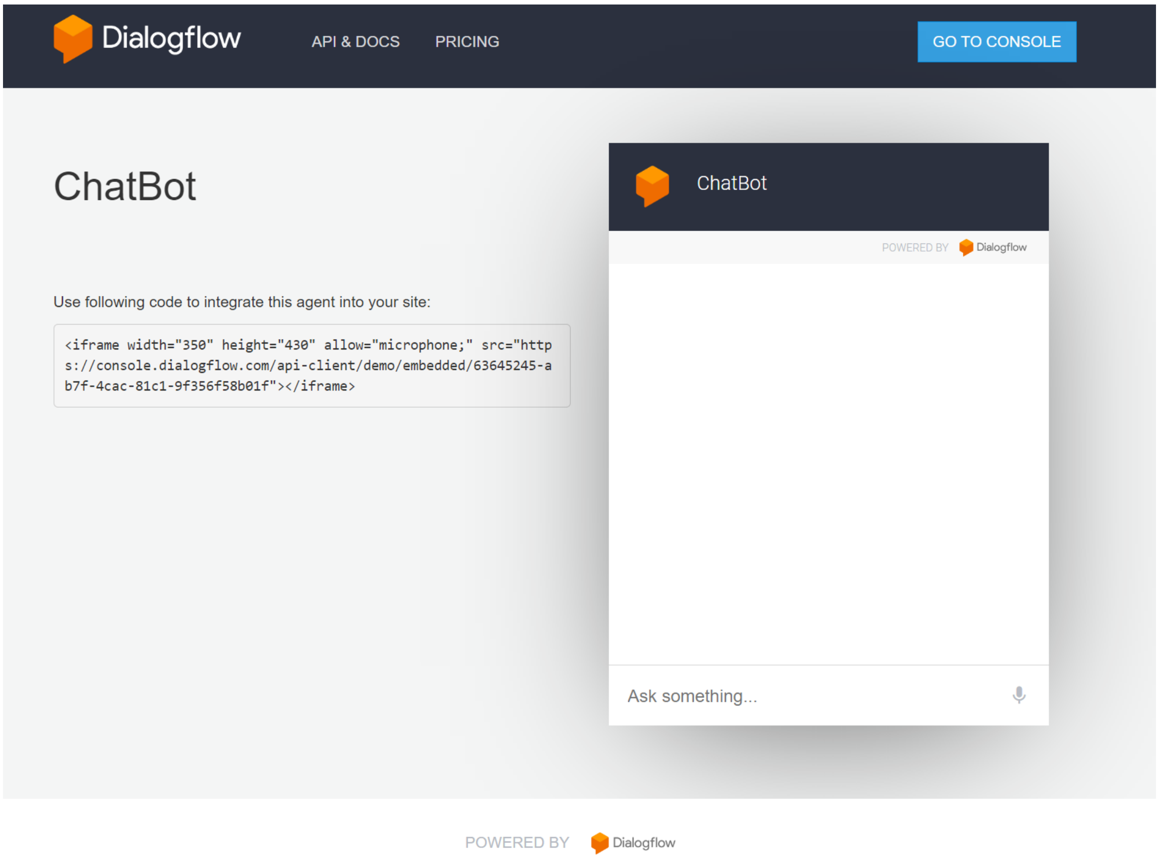
```
<iframe
    height="430"
    width="350"
    src="https://bot.dialogflow.com/63645245-ab7f-4cac-81c1-9f356f58b01f">
</iframe>
```

**Dockerize App and Test Locally**
We will now begin to dockerize this app, to run as a container on any machine with a Docker engine. To do this we need to first build a Docker image from the instructions in the Dockerfile in the current directory. These instructions dictate how to build the container.

**Build Docker Image**
With the virtual environment active, choose a suitable name for the docker image and run docker build -t <docker_image_name> . , in the terminal to build a docker image from the Dockerfile.

**Running a Container**
Once the image build is completed, you can run the app as a container, from the image using docker run -p 8080:8080 <docker_image_name> .

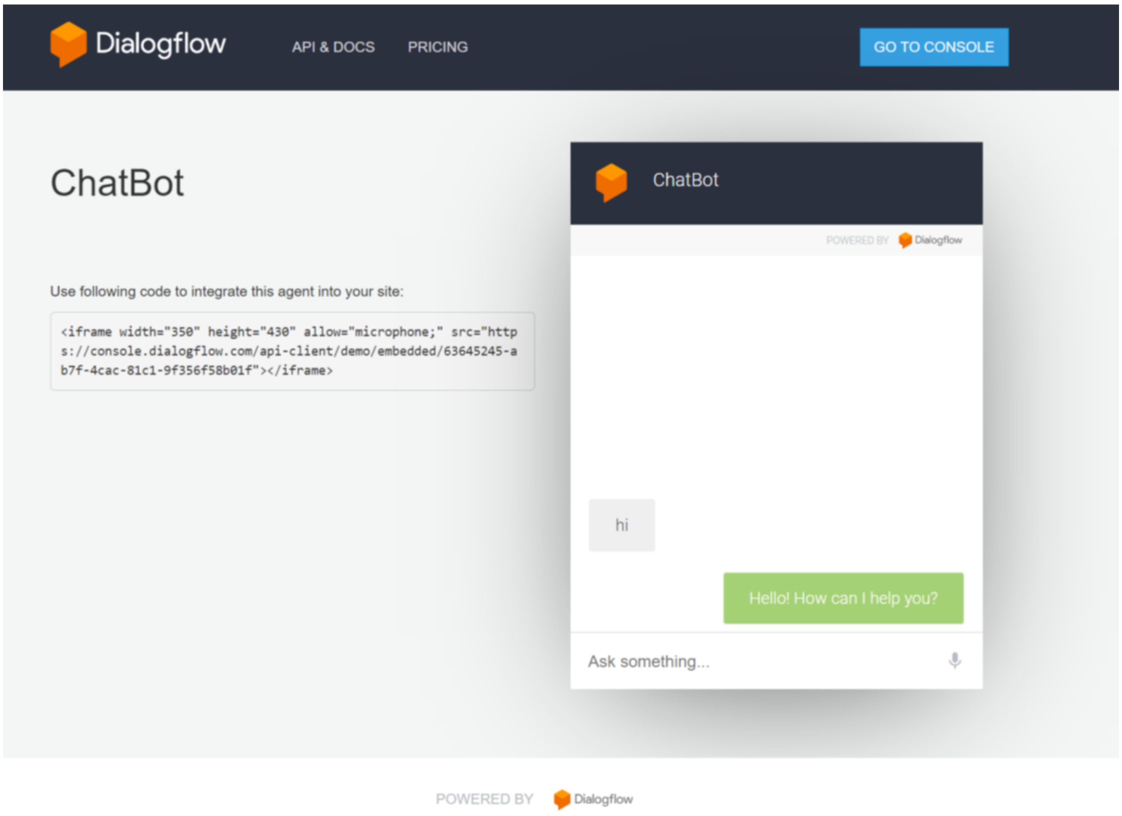
**Deploy App to Google App Engine**
Before deploying the app to App Engine, we need to add another file to the project directory: app.yaml. This is a configuration file that basically contains settings for the app engine.



ChatBot

Use following code to integrate this agent into your site:

```
<iframe width="350" height="430" allow="microphone;" src="https://console.dialogflow.com/api-client/demo/embedded/63645245-ab7f-4cac-81c1-9f356f58b01f"></iframe>
```



ChatBot

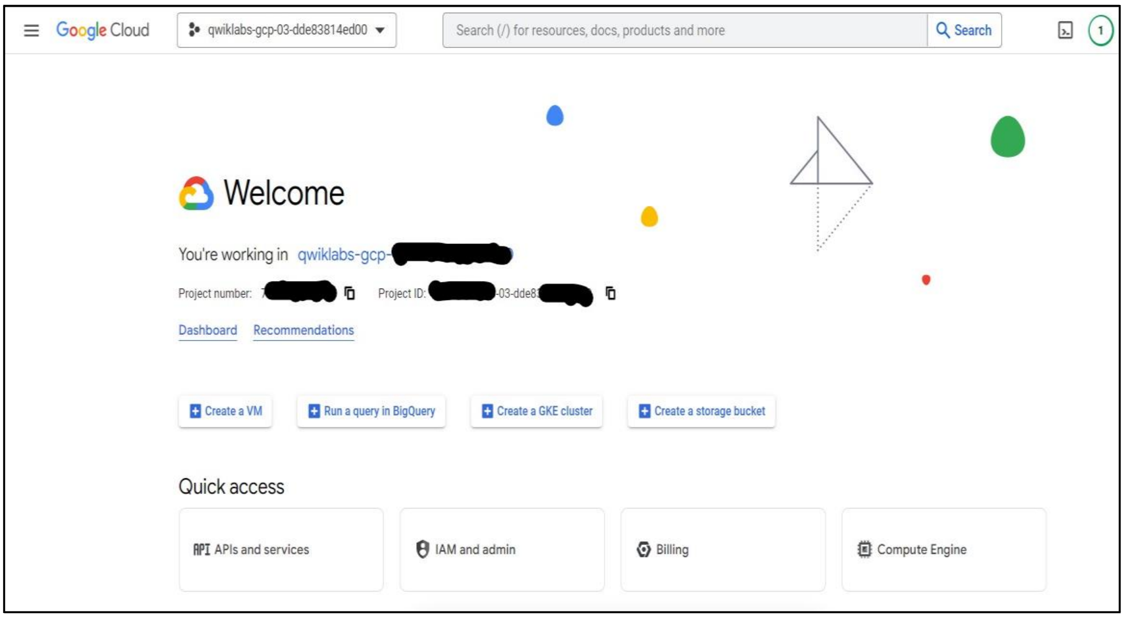
Use following code to integrate this agent into your site:

```
<iframe width="350" height="430" allow="microphone;" src="http
s://console.dialogflow.com/api-client/demo/embedded/63645245-a
b7f-4cac-81c1-9f356f58b01f"></iframe>
```

ChatBot

hi

Hello! How can I help you?


Google Cloud
qwiklabs-gcp-03-dde83814ed00
Search (/) for resources, docs, products and more
Search

Welcome

You're working in qwiklabs-gcp-[redacted]

Project number: 7[redacted] Project ID: [redacted]-03-dde8[redacted]

## CONCLUSION:

In conclusion, building a machine learning model as a serverless app using Google App Engine provides several benefits, including automatic scaling, cost-effectiveness, and ease of deployment. Google App Engine offers a range of tools and services to streamline the development process, making it a great choice for building machine learning applications.