# Report

## Team Member Details

- Sandeep Swami (B19ME072)
- Kshitij Sanodariya (B19ME075)

## Introduction

Our Project is a blogging solution for communities of different sizes or to manage personal stuff where you can read, create and search for different articles. Our solution is different from existing solutions as it is lightweight and easy to deploy on your own server as it is completely containerized. It doesn't depend on any proprietary software or library so you can customize it as per your need as it is open sources on Github. Frontend is build using React and Tailwind CSS, It has features of Google Login as well as Login with Email and Password. You can write posts with a rich text editor and also upload a Image which will be stores in Amazon S3(Simple Storage Service) bucket to be shown on the feed. Frontend talks to backend with a library axios(a Promise-based HTTP client).

Backend is build using Flask(web application framework written in python) and uses mongoDB database (a NoSQL general purpose document database). As mentioned previously backend is completely containerized with Docker, flask server runs with waitress(a production-quality pure-Python Web Server Gateway Interface server) and Nginx(Nginx is a web server that can be used as a reverse proxy, load balancer etc.) for load balance and to serve static files.

Server is built using Publisher-Subscriber pattern which enable an application to announce events to multiple interested consumers asynchronously, without coupling the senders to the receivers. Author will publish the article and all the use on the site will receive the article.

## Summary of Work done

Started with research and thinking about possible ways of solving the problem. As it was a group project we divided it into two major part Frontend and Backend. Frontend consist of user experience, user interface, user interactions etc.

Backend consist of server, database management, load balancer etc. Server created with python and flask framework. We used MongoDB as a database as it is a general purpose database and works well with

We also used version control with git and Github.

## Work done

Work Done By Kshitij

- Design architecture of backend. Ended up choosing Pub-Sub architecture.

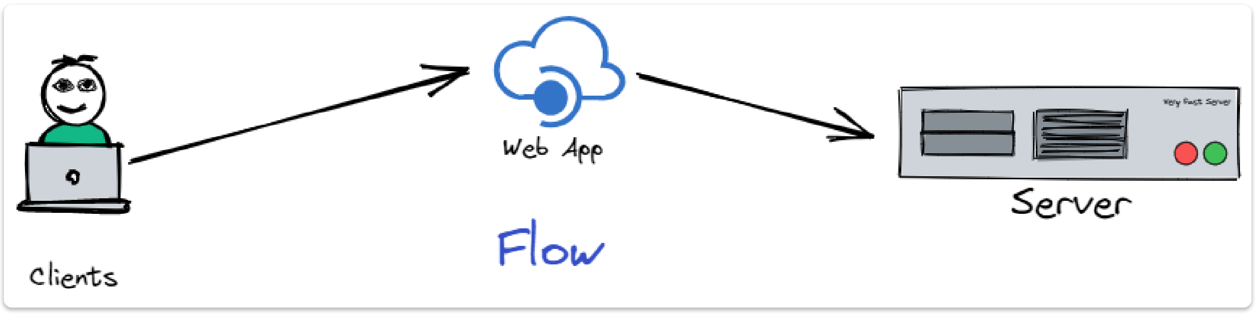


Figure 1: Flow of the app. Client using webapp requests server for content.

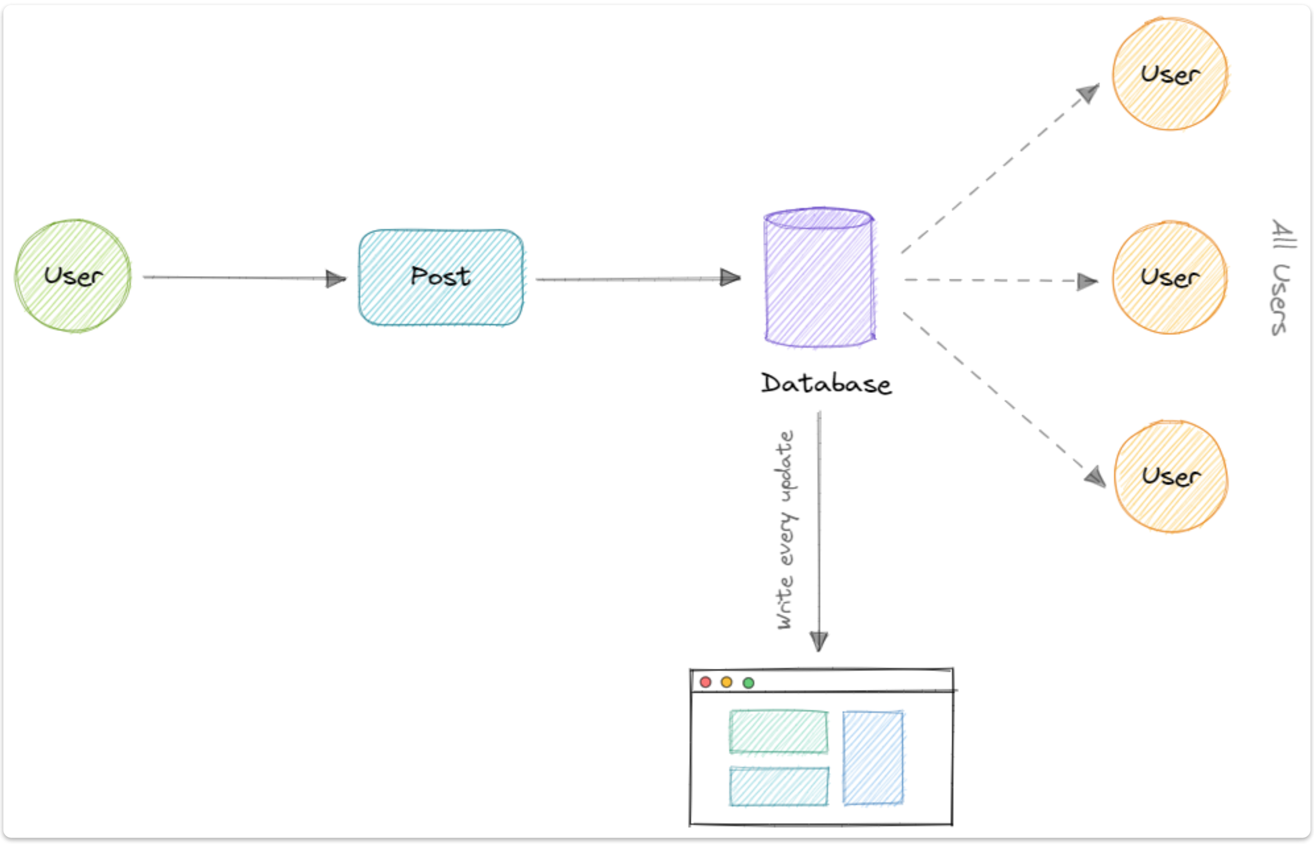


Figure 2: Flow of the app. From Author(user) to database -> all users of the app.

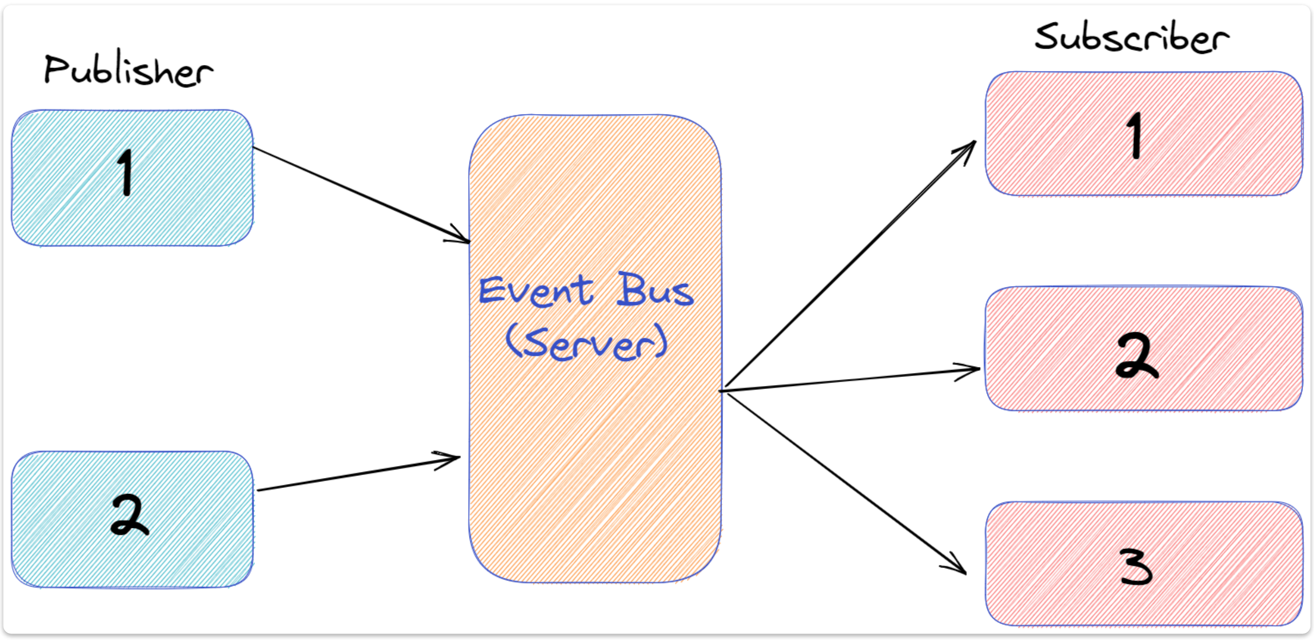


Figure 1: Showcase of how Publisher-Subscriber pattern works. Request from publisher goes to through server and then to server.

- Designed Models to store in database.
- Containerized backend with docker.
- Deployed Nginx and MongoDB with Docker to work with Flask.

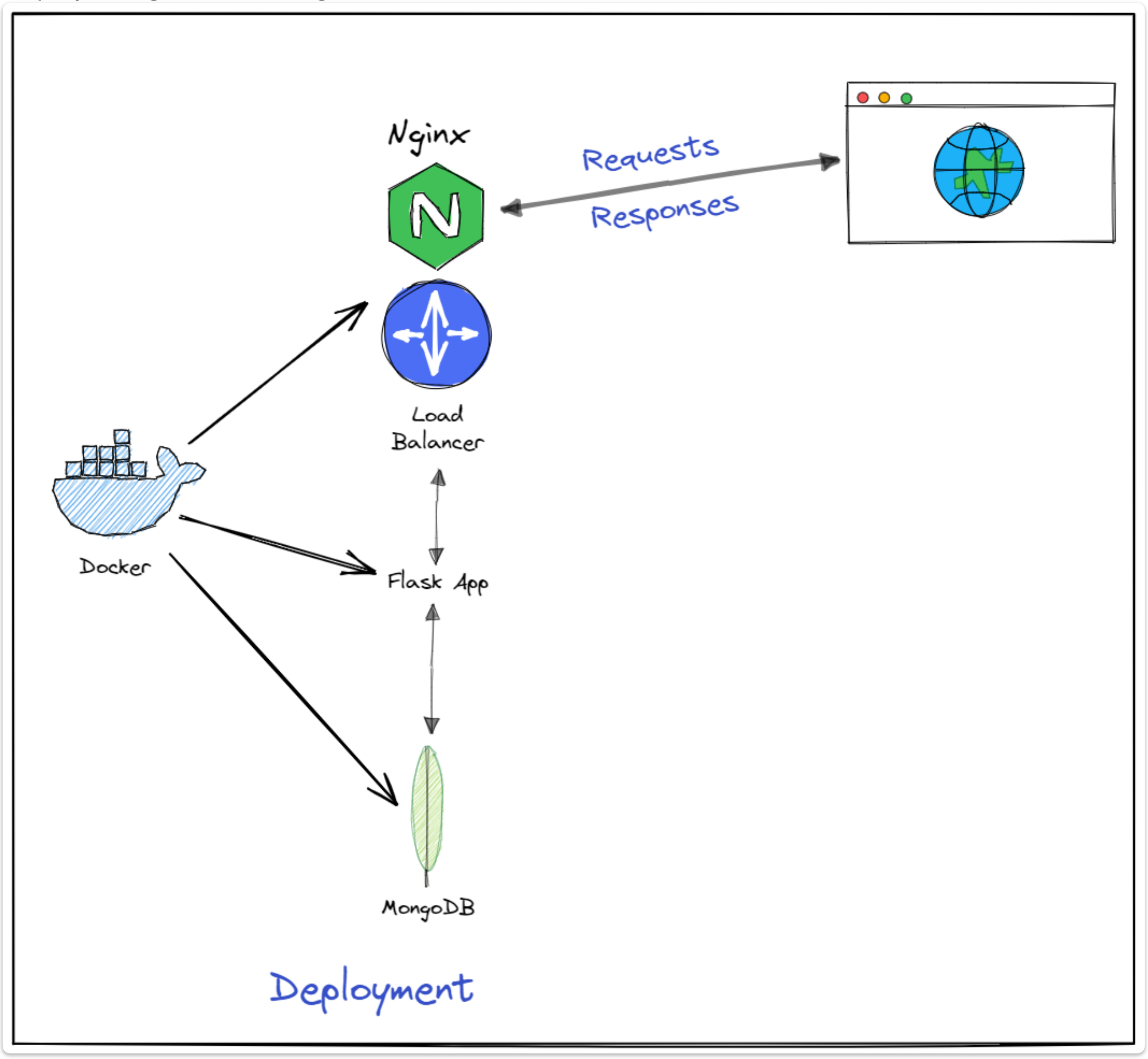
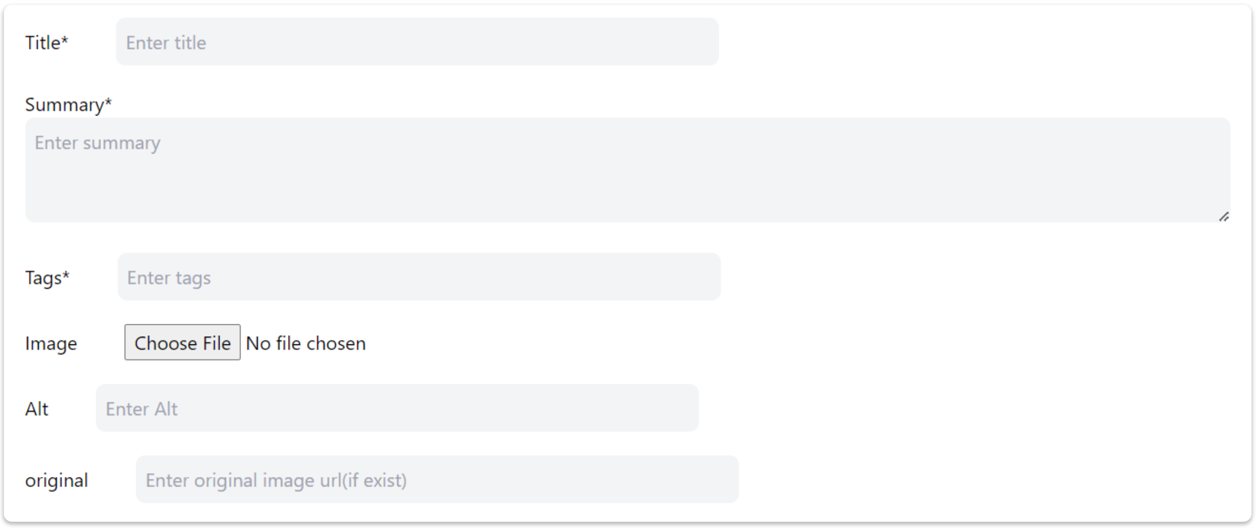


Figure 2: Architecture of server

- Implemented image storage with S3.
- In frontend, implemented communication with backend using axios.

- Designed Form to publish post.



| | |
|---|---|
| Title* | Enter title |
| Summary* | Enter summary |
| Tags* | Enter tags |
| Image | Choose File No file chosen |
| Alt | Enter Alt |
| original | Enter original image url(if exist) |

Work Done By Sandeep:

## Results

## Conclusion

## Reference