Term Project
CS 436

Group Members
Ayhan Salih Öner
Eren Çavuş
Gürsel Yiğit Pekgöz
Yarkın Alpmen Akyosun

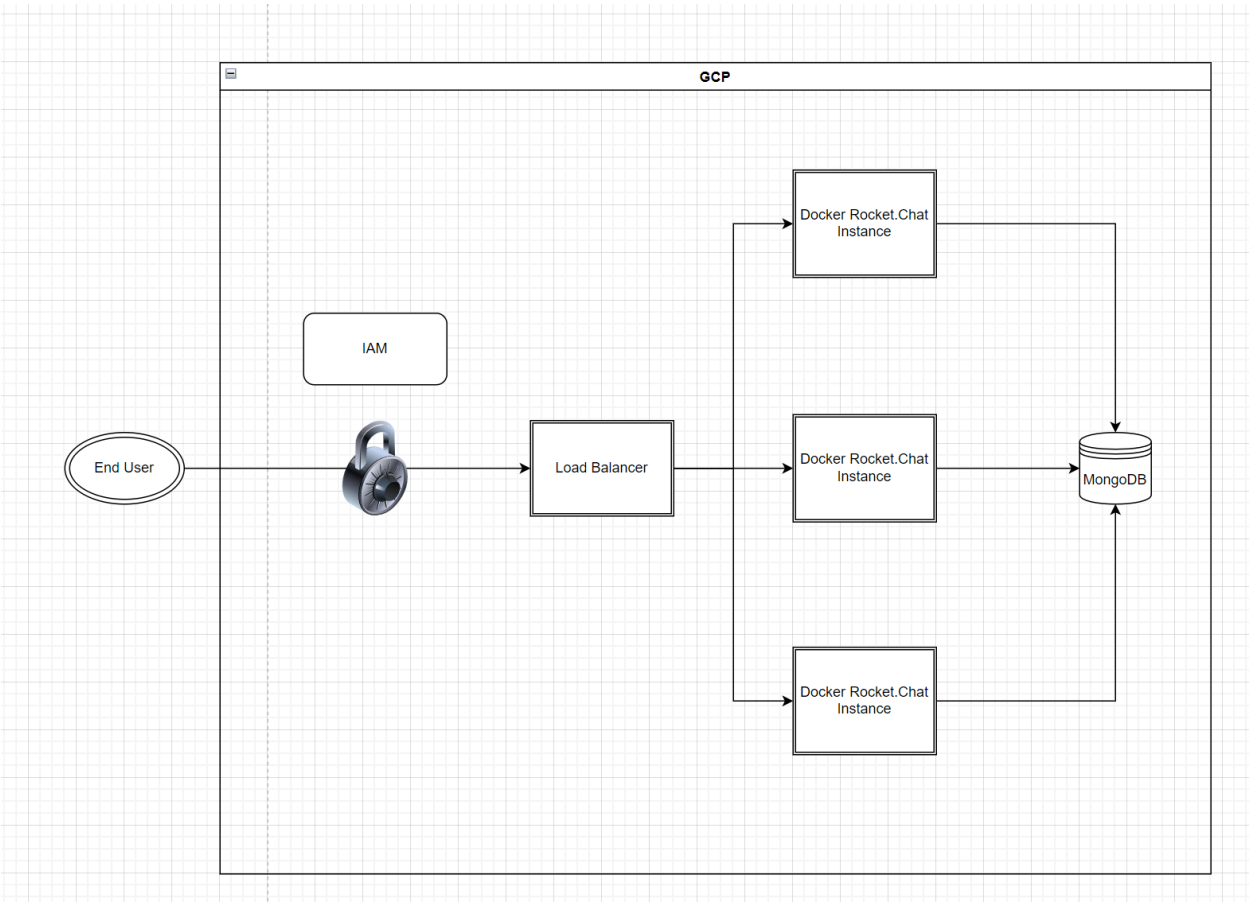
Link to Github Repository:
https://github.com/onerayhan/cs436_project/

Plan to Proceed

We would like to use Rocket.Chat as our backend application. Our aim is to use a Google Compute Engine with Docker swarm to orchestrate containerized Rocket.Chat instances and related services and run it in a managed Docker environment. As a database Rocket chat needs us to use MongoDB as a Database so we might use Cloud SQL or MongoDB atlas which are available on GCP marketplace and also if time permits we might implement an automated backup / disaster recovery solution. We have no aim to handle media uploads because of our price restrictions but we plan on storing text messages sent by the users. We are eager to familiarize ourselves with IAM policies to manage user access to our resources securely. Also we aim to use an internal load balancer to divide the load into multiple instances and test how it behaves under heavy traffic. In order to prevent concurrency issues in the database we aim to connect each server to the same database. We also would like to implement auto scaling features and load balancing as IaC with terraform or Ansible. If time permits we might consider building a CI/CD pipeline for continuous deployment into our backend application.

Draft Architecture

End User will reach its data through our balanced docker containers after its authentication. Containerized Instances will be used to run multiple Rocket Chat instances and it will be allowed to scale automatically. Since there may be concurrency issues in the database as stated above and Rocket.Chat requires MongoDb to run. A single database instance will run on MongoDb Atlas or a Google Compute Engine. Here is a draft Architecture diagram:


GCP
Docker Rocket.Chat Instance
IAM
End User
Load Balancer
Docker Rocket.Chat Instance
MongoDB
Docker Rocket.Chat Instance