CSCI 5253: Project Proposal

Krishna Sai Chemudupati

1 Title

Scalable File OCR Storage System

2 Project Participants

Krishna Sai Chemudupati (myself)

3 Project Goals

The goal of this project is to:
1. Have a simple front-end and web-server to be able to login to your profile.
2. Have a feature where the user can upload scans or images of data, and in return get the text.
3. Provide the user the capability to store this document and the OCR data.
4. Provide the user the capability of retrieving this data when the user logs back in.

Note:
I have considered using ideas of Gradescope and using the FinCen Files, but I wasn't able to find ways/APIs to differentiate between questions and answers in a given pdf of the scan of the exam/assignment.
Furthermore, I explored the FinCen data available on the website provided but I couldn't find the pdf or scans of files. I only found a csv of transactions.
I plan to attend office hours and refine my idea a bit.

4 List of Software/Hardware Components

4.1 Software

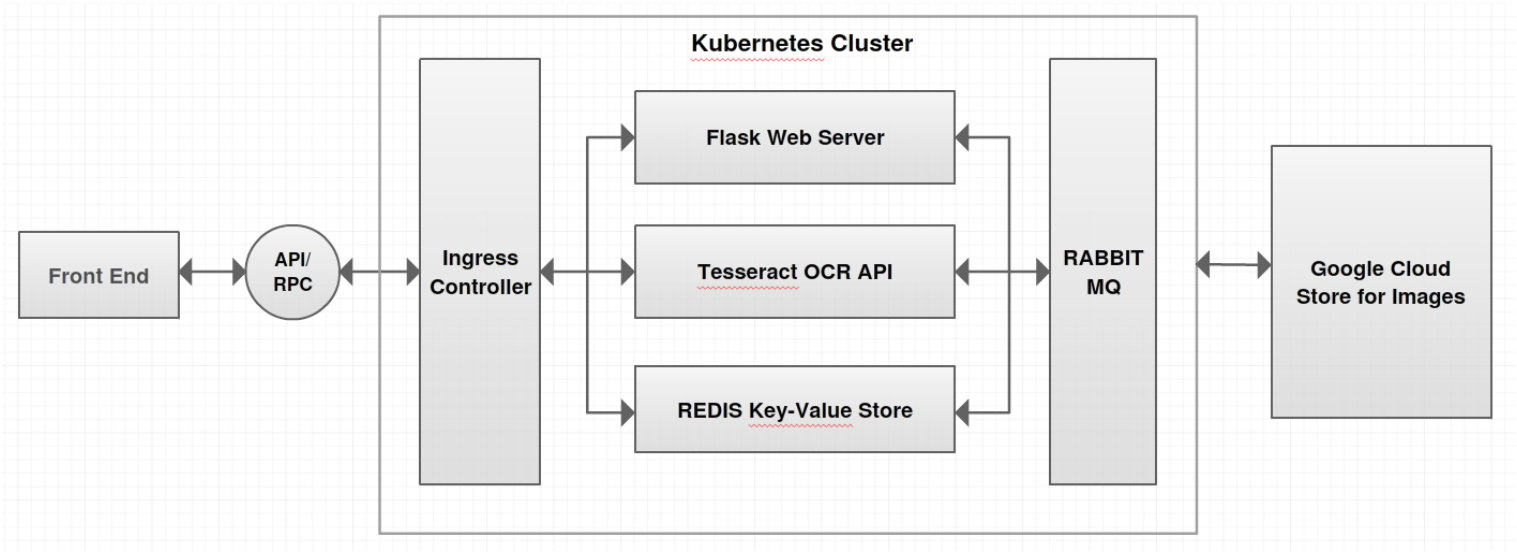
1. Docker
2. Kubernetes
3. Google Cloud Platform
4. Tesseract OCR
5. RabbitMQ
6. Flask
7. Redis key-value store database
8. Maybe Google Cloud Store to store the actual images of scan
9. REST APIs/Google Protobufs to send images to the flask web server, and other requests

4.2 Hardware

1. My laptop
2. Machines used by Google Cloud Platform

5 Architectural Diagram

Figure 1: Software Architecture



6 Description of Interaction

1. Front end and the web-server is made with Flask. Front-end will send REST api requests or GRPC requests using the protobufs to the web-server whenever the user tries to login, or add images of documents, or retrieve documents.

2. Then the web-server is then going to:

   - **For user authentication:**
     Get information (password hash) from the Redis key-value store which has persistent data, and checks if the user is valid, and sends appropriate message back to the front-end, which could be html code to show the next page (register, or image dashboard)
   - **For adding an image:**
     (a) Hash the image and check with the Redis database if the user has already uploaded this image and send back to the the front-end the text stored in the Redis database for that image hash.
     (b) If the hash is not found, send the image to the Tesseract OCR Service which then returns the text found back to the web-server. Then the web-server will add the scanned image of the document to the Cloud Storage and store the user's username (unique), image hash, link to the bucket where the image is stored, and the text into the Redis database. It then sends a message back to the front-end that the image has been stored along with the text found in the image.
   - **For registering profile:**
     Store the username and the hash of the password in the Redis database, and send back to the front-end a message that the user has been registered or the username already exists.

3. The pod running Tesseract OCR Service is going to receive requests with an image and sends back the text found.

4. The pod running the Redis database will have multiple databases for user authentication and image information. It will also have a persistent volume mounted to it so profiles are not lost. Lastly it will receive requests from the web-server for values associated with a key, and this service will send back this information.

5. All the inter-service communication is done through RabbitMQ.
6. The Google Cloud Storage is going to be used to store scanned images of documents in buckets.
7. The ingress controller is responsible to exposing the endpoint for external connections, and then handles some load balancing.
8. Kubernetes is going to handle the automatic scaling part of this project by handling pod and container creation.

7 Description of Debugging Methods

I will have to debug this one container at a time. If the individual services work well, I will connect everything and check if the entire application works well together. To debug if the application is able to scale well, I might need to create a script to make new profiles and upload new images repeatedly, or do this manually. I will need to discuss more about this with the professor though. I can also probably have a port for logs for each container and check the logs as the application is running. Lastly, I can always exec into the container to check for issues.

8 Does this meet all requirements?

The project will use the following cloud technologies (still needs to be confirmed):
1. RPC / API interfaces – REST/GRPC + Protobufs
2. Message queues – RabbitMQ
3. Key-Value Stores – Redis
4. Virtual Machines, containers, or "functions as a service" – Docker, Kubernetes, GCP
5. Storage services (s3, object storage, etc) – Google Cloud Storage

Therefore, I believe this project satisfies the requirement of using various cloud technologies.

9 Is this interesting to me?

I am interested in the field of Computer Vision, and Optical Character Recognition (OCR) is a very important, and practical use case. Also, I believe finishing this project will help me gain very useful experience and the skills to complete more scalable computer vision applications in the future. Furthermore, this project will be an amazing add to my resume for recruiters, as skills in these technologies are highly sought after in the industry!

10 Does this project meet eventual project requirements?

Yes, I do believe that as explained above.

11 Is this within the scope of the class?

Yes, the technologies used are well within the scope of the class. Given, I am the only one working on this, I am a little worried about if I'll be able to finish this in time, that's all.