CMPT 436 Project

Number of People that worked on the project: - 1 person (Only Myself)
Project Worked On: - 3 Projects.

1. **Goals: - I Built an application that converts text to audio and download the text into an audio format using the AWS toolkits (Amazon Polly, S3, Dynamo database, Cloud Formation, Lambda and react JavaScript etc.)**

2. **Goals: - I Built an application that keeps track of the latest change in the price of the exchange of cryptocurrency. I used React JavaScript in designing the frontend of the application, Material UI in react to improve the appearance of the application and the crypto wallet database and Chart JavaScript to display a historical chart of the application in the interface and for the Api I used Coin Gecko to implement the crypto Api for the both the single, historical and trending coins of the crypto wallet and I implemented the AWS S3 for the storage resources of the application on the cloud and AWS Amplify to host the website on the cloud.**

3. **Goals: I Built an application based more on the client side that helps users to navigate through the school portal in terms of viewing the courses offered by the university, the about page for the university and lastly a contact page where users can leave message to the server and can also post a comment or story on the web application. I used Html, CSS, JavaScript and PHP in designing this application so that it can scale very well for users. I also hosted the application on the cloud using AWS Amplify so that the resources can easily be managed on the cloud. I am also in the process of developing the application on the amazon storage S3 so to have a backup in case of any downtime/loss of data in the cloud.**

Project: - I explored numerous AWS services in designing this application. For this project, I built three applications using React JavaScript for the frontend and for the Amazon webservices used include: Amazon Polly for deep learning(to convert text to audio), Amazon S3 Bucket as the storage service, AWS Lambda

for the serverless compute service that would run the code in response to the events and it would automatically manage the resources of the application, AWS Serverless Framework, AWS IAM to manage the access roles in the S3 Bucket like the access key id and the secret access key id, AWS Amplify for hosting the website on the cloud platform, Amazon dynamo database(is a NOSQL database to help run high performance applications at any scale & to store and retrieve any amount of data) and Amazon EC2 to create a virtual computer on the cloud using an instance of the amazon Linux 2 instance.

Define an experimental setup

○ Load Generator (one machine)

Used the Amazon Web Service Gateway Load Balancer GWLB and the load balancer endpoints GWLBE. It helps to improve the security, compliance and policy controls.

○ Web Service host

I used Amazon Web Services in implementing the application for the both the server and client side.

○ Different machines e.g., docker playground, own machine ...

I used the machine provided by the amazon web services platform.

Define experiments

○ Test behaviour based on arrival rates, data types, data volume,

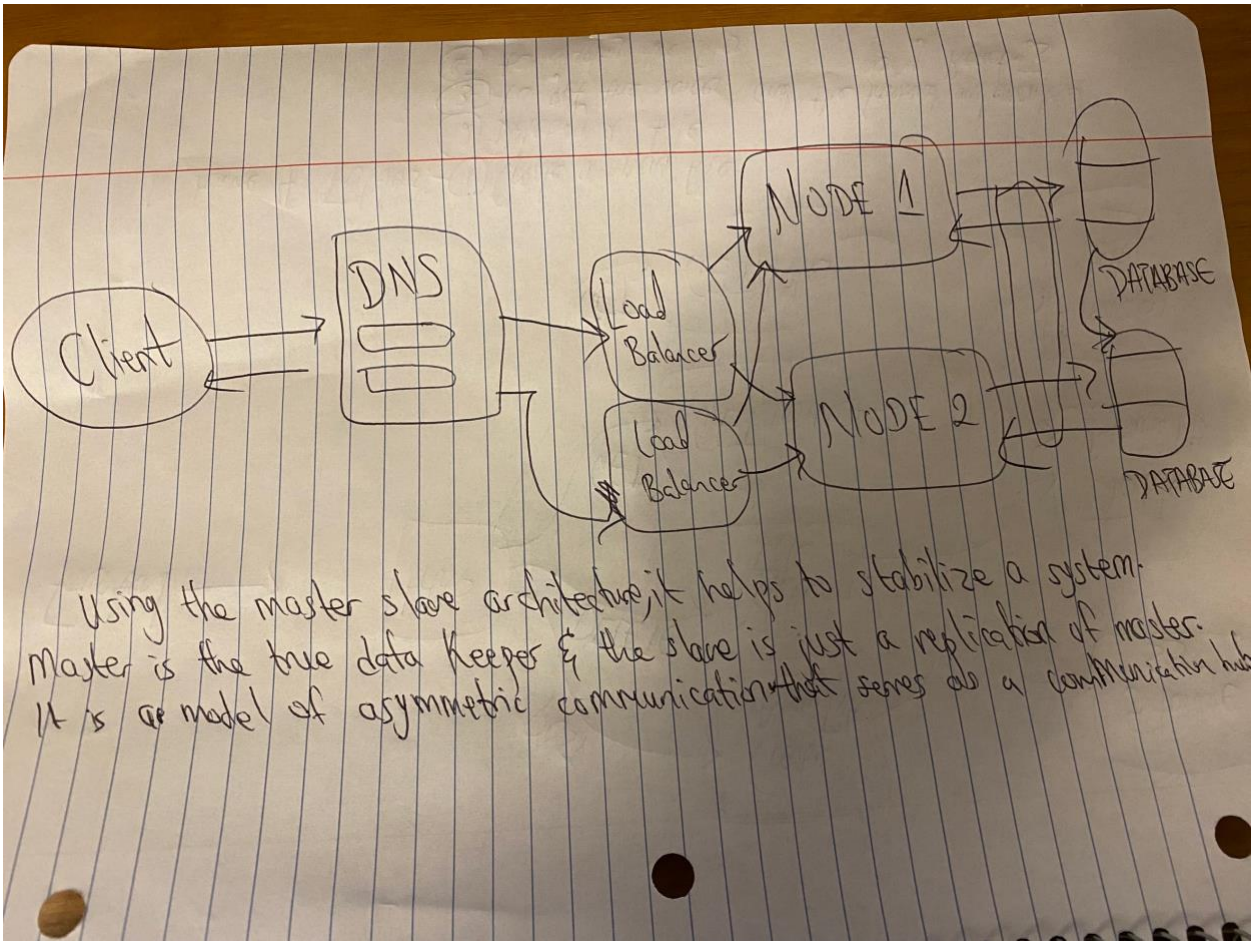
I tested the application based on the arrival rates, data types and the data processed on the application. The application has a high performance, and it does not take too long when processing data inputs in the application. The application size does not affect the overall performance of the data.

○ Rerun experiments at least 3 times: Yes, I rerun the experiments at least 5 times to make sure all the details are well captured. The application does scale very well and has a very high-performance rate.

○ Explore how to handle network and node failures

I could handle the network and node failures by reducing the downtime by having more availability zone in different regions in order to increase the uptime of the application and can also handle the failure by using multiple load

balancers gateway and making the client connect to a DNS which then connects to the load balancer. Below is an example of an architecture that could help handle the network and node failures.

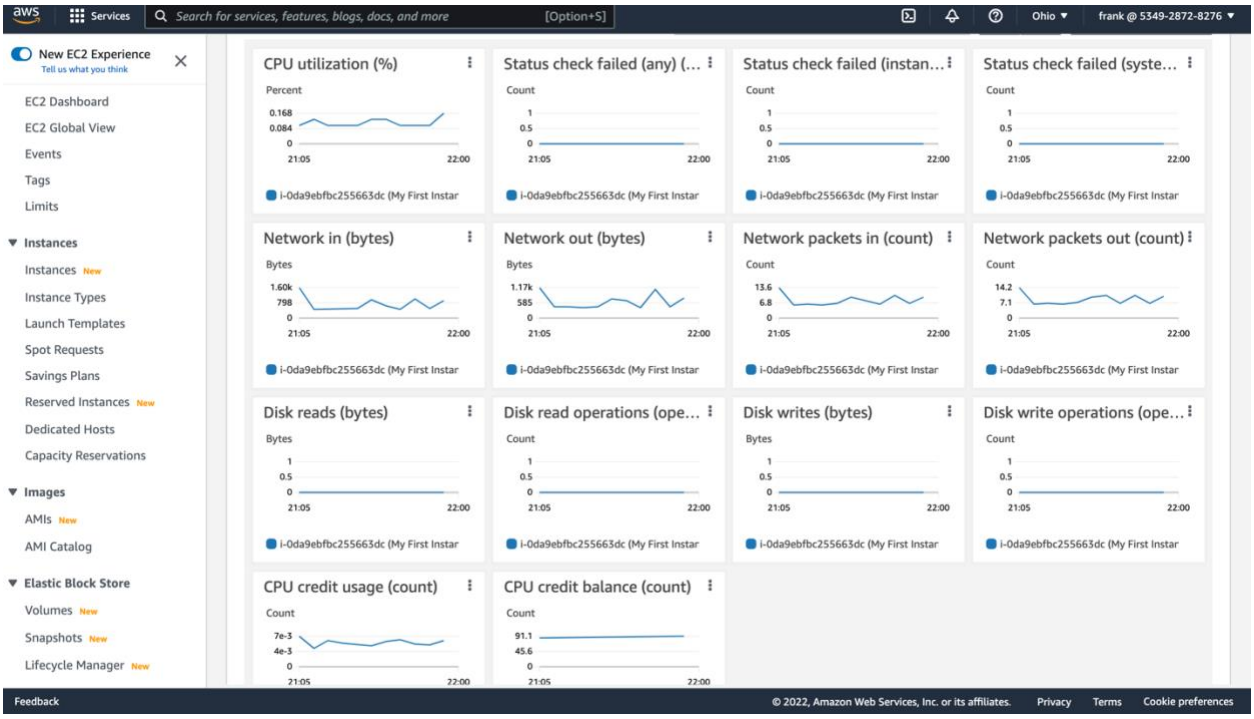


○ Explore Scalability:

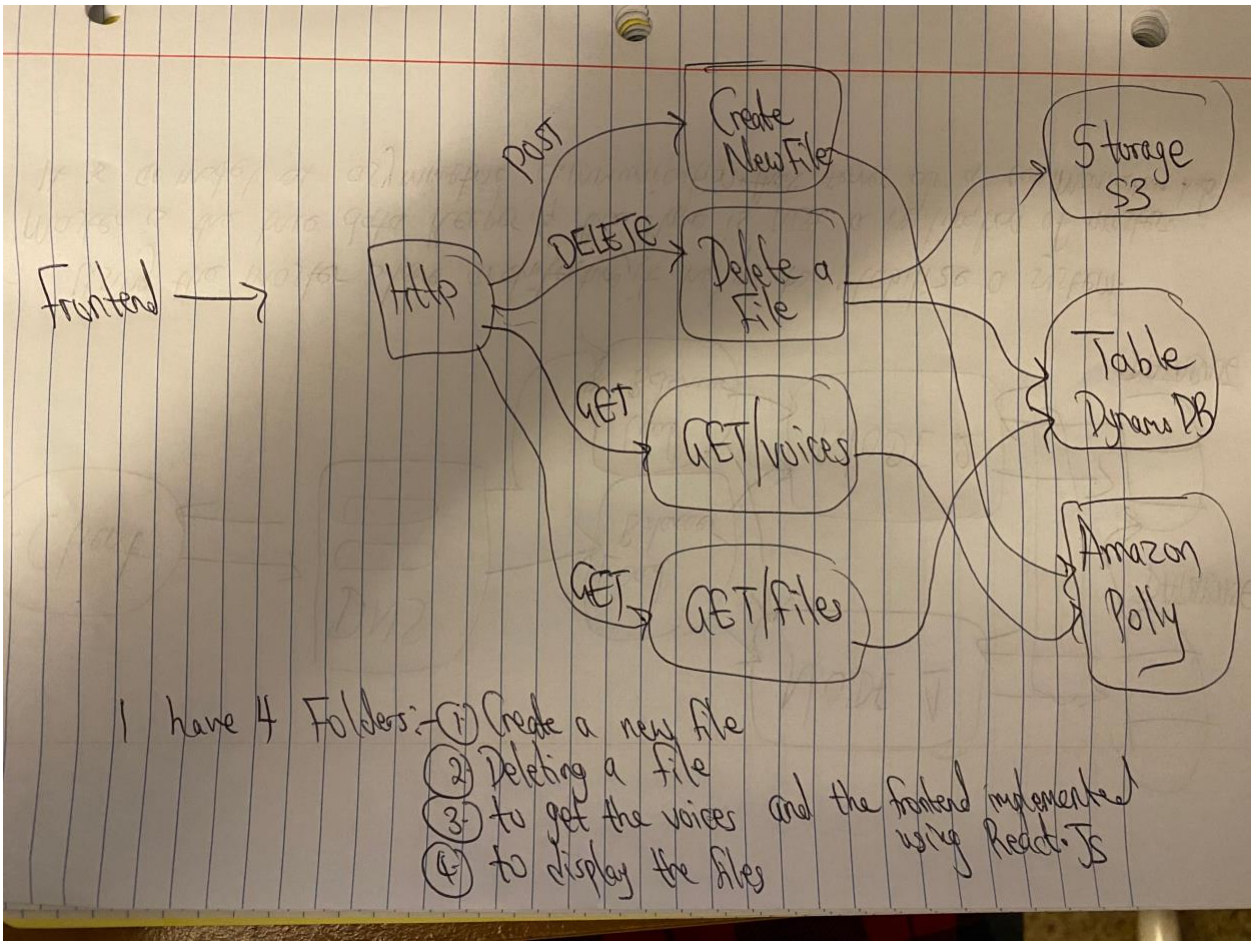
Scalability is the ability of the workload to perform it's agreed function when the load or scope changes. For example, of scalability would be how well your system performs when there is an increase in the number of user and how well a database could handle an increase in the query of data. One of the main advantages of scalable system is that it is adaptable to changes in the needs and demands of consumers/users. Another best word to describe scalability is adaptability or flexibility. To wrap it up, Scalability is increasing the capacity to meet the increasing workload. The main aspects of Scalability is the ability to scale up when there is an increase in users who are using the application and also the last aspect is the ability to scale down when there is a decrease number of users in order to save Cost.

Present results in a graph/spreadsheet

Below is a graph representation of the EC2 Instance of the CPU utilization and the network packets over the virtual machine on the cloud.



First Project: Built an application that converts text to audio using deep learning provided by Amazon Polly and I used S3 Bucket as a means of storage and Amazon dynamo database to scale the throughput and concurrency for the application. It has a lower latency that helps to also scale down the latency of the application. Dynamo database scales seamlessly no matter the traffic and it has an unlimited storage space. Used Amazon IAM (Identity and Access Management) to restrict access on the S3 Bucket, AWS EC2, Amplify, AWS Lambda for the serverless framework and used React JavaScript for the frontend of the application. Below is an architecture of the application and how it works.



Second Project: Built an application that keeps track of the current price of crypto currency using coin gecko for my api to keep track of both the single coin and trending coin currently. I used Material UI to design the theme of the application, Chart JS to handle the graph chart of both the previous and current price of the market and React JS in building the product of the application and material UI for the table and table row of the crypto application. I also used Amazon Amplify and S3 Bucket to host the application on the cloud.

Third Project: Built an application for a university website where users can read about the university, leave a message or comment on the application and a blog post where users can post a message. It was designed using HTML, CSS, JavaScript and PHP. Used AWS Amplify to host the website on the cloud.