CSCI 5411

Cloud Architecting

Term Project Report

Professor : Lu Yang

Name : Luv Patel

Banner ID : B00950942

Fall 2024

Table of Contents :

Open Source Project :

SplitApp : https://github.com/tuzup/SplitApp/

Why use this open source project ?

i). Full Stack Web Application :

- Built with frameworks of JavaScript, which makes it easy for development and maintenance to have one programming language in front-end and back-end code.

ii). Scalability:

- MERN components are developed in a way so as to support morphing with an aptitude of serving high-traffic applications when coupled with an extensive scalable infrastructure that is provided by AWS itself.

iii). For Friends:

- SplitApp is platform for friends, to get engaged and manage their finances and split bills among friends community.

Chosen AWS Services :

Category	Services
1). Compute	EC2, EC2 Auto Scaling
2). Storage	AWS S3
3). Network & Content Delivery	Amazon VPC, Elastic Load Balancing
4). Database	Amazon DocumentDB
5). Management & Governance	AWS cloudFormation, AWS Systems Manager Parameter

6). Security, identity and compliance	AWS Secrets Manager

Explanation of the AWS Services Used :

1). Compute - Amazon EC2 (Elastic Compute Cloud) :

- ⇨ **Usage :** Hosting backend in an availability zone.
- ⇨ **Scalability :**
  - Helps in scalability as it gives the flexibility of removing or adding new instances as and when required.
  - This is achieved by the backend Auto Scaling Group that is attached for this purpose, that performs scaling based on the demand.
  - Also, it provides the features of using different sizes and load of instances that will help match the incoming load requirements.
- ⇨ **Cost:**
  - Another advantage is the pay-as-you-go model, that allows for cost optimization.
  - To further reduce the cost, we can also use spot instances, which will help save upto 72% cost. [1]
- ⇨ While AWS Lambda and container services such as ECS offer serverless options, EC2 provides a great deal of freedom over the operating system and runtime environment. [2]

Why EC2 ?

- ⇨ By providing these features as well as it ideal pricing model, it makes ideal choice for running and hosting my backend with auto scaling group.

How will it help with my project SplitApp ?

- ⇨ The elastic compute cloud (EC2) instances are over cloud virtual servers . Using it provides flexibility of scalable and resizable

computing power, thus enabling me with scaling up and/or scaling down based on the application's needs.

⇨ For hosting backend, an EC2 instances in **us-east-1b** provides a cost effective and stable environment by using t2.micro which is as per application requirements.

⇨ Using EC2 with an **Auto Scaling Group** allows for dynamic scalability which helps in stability and performance while ensuring the lowest possible cost.

⇨ In my cloudformation template script, I have chosen the instance type as "**t2.micro".**

2). Elastic Load Balancer (ELB): (Network and Content Delivery)

⇨ **Usage :** For distributing traffic to the backend EC2 instances.

But how does it help in SplitApp?

⇨ Following the cloud best practices, I have chosen Application Load Balancer (ALB), as it operates in the application layer.

⇨ Also the load balancer has a feature of health check for the port of **3001** for target group, where the node and express js is running.

3). Storage- Simple Storage Service (S3):

Usage : To host the frontend of the application for enabling the users to access the web application.

Cost:

- It is cost effective because it is a service that charges us for the amount of storage that I use, with no minimum fee for accessing the service.
- S3 provides multiple storage classes such as Infrequent Access, Amazon S3 Glacier are available to store data is not frequently accessed, thus helping reduce the storage cost. [3]

Performance:

- S3 provides 99.99% availability.
- Provides low latency access to data, thus helps in frequent retrieval which will allow the users seamless access to the frontend.

Security :

- S3 utilizes the SSE-S3 (Server Side Encryption) to encrypt the data at rest. [4].

Scalability :

- It provides the feature of the unlimited storage space (virtually). And is capable of auto scaling based on needs.

Why use S3 and how it helps SplitApp ?

- Compared to other AWS services like ECS or EC2, S3 is better suited for static site content and large files.
- It offers better scalability as S3 provides auto-scaling feature and does not require any additional setting for performing auto scaling.
- Also, hosting the website on S3 is much cheaper than hosting the website frontend on EC2, and as compared to EC2 it is much cheaper**.**
- S3 is able to auto scale, hence no need for any autoscaling group or any other scaling components for scaling based on the user traffic inbound.

4). Database : DocumentDB (MongoDB Compatible):

Usage : For NoSQL database storage for the SplitApp application.

Cost : Pay-as-you-go pricing with no additional cost for using the service. [5]

Performance :

- Compatible with MongoDB, which will have easy migration from mongoDB to documentDB.

Scalability :

- MongoDB provides high scalability as it provides features for creating DB instances and replicas instances and read replicas. The DB instances must be created in another availability zones ensuring fault tolerance .
- If the main cluster goes down, then the traffic is automatically redirected to the secondary instances or other available instances.
- It provides high availability by specifying to choose atleast one instance as mandatory and that too in another availability zone.
- DocumentDB offers a fully managed, MongoDB-compatible database service. Placing it in a private subnet enhances security.[6]

Why choose DocumentDB ?

- Compared to other similar database like DynamoDB, DocumentDB I believe is a better choice as SpitApp is a MERN application which required MongoDB compatibility and handles complex queries.
- On the other hand, DynamoDB is better suited or key-value pair databases.

How DocumentDB helps in SplitApp ?

- To connect using the mongoose package, we need to add a global-bundle file for authenticating the DocumentDB host URL as parameter.
- For this MERN stack application, we do not need to change queries as DocumentDB is MongoDB-compatible. The DocumentDB allows only backend launch template inbound rule to further strengthen security.

- This enhancement strengthens security allowing an incoming rule access through only the backend launch template concerning DocumentDB.

5). Network : VPC (Virtual Private Cloud):

Usage :

- I have used the VPC to divide the network into multiple public and private subnets and isolating the network from outside world.

Justification:

- Helps in isolating the network from outside world, thus improving the security of the VPC.
- We have control over IP address ranges, subnets, route tables, internet gateways and network gateways.
- Enables creation of multi-tier architecture for our cloud implementation. [7]

Why choose VPC ?

- VPC is important for creating a private network for my application, which is isolated and protected from outside world.

VPC for SplitApp:

○ I made my backend ec2 and document DB by keeping them separate in a public and private subnet.

○ Total 4 Subnets will be there, 2 Public subnets where the EC2 instance are hosted and other 2 are private subnets where the DocumentDB database is hosted.

○ Public subnets will be able to access the internet with an internet gateway.

- Each subnet is associated with a security group and a route table association.

6.) AWS Secrets Manager (Security, Identity & Compliance):

Usage :

- I have used secret manager in my architecture to securely store DocumentDB Credentials.

Justification :

- With a secret manager, it will help to protect application sensitive information such as credentials, without the need for hard-coding the credential values in the SplitApp application.
- It also provides a feature, where we can automatically rotate secrets by means of specific dates.
- Encryption: Allow encryption of secret keys, giving extra security for crucial data.

How AWS Secrets Manager helps in SplitApp?

⇨ AWS Secrets Manager stores credentials for DocumentDB securely, ensuring no credentials are hardcoded in the application code.
⇨ Using the AWS SDK, the application fetches secrets programmatically at runtime, improving security and flexibility.
⇨ Secrets are rotated automatically using pre-configured policies with minimal developer intervention. This ensures SplitApp always uses updated and secure credentials.
⇨ Fine-grained IAM policies restrict access to secrets, ensuring that only authorized components or users can retrieve sensitive information.

Management & Governance

7). AWS CloudFormation:

Usage :

- I have used CloudFormation for setting up the infrastructure of the project in AWS environment.

- It simplifies the creation, management, and updating of AWS resources using YAML or JSON templates.

Justification :

i). Infrastructure as Code (IaC):

CloudFormation has empowered SplitApp to manage its infrastructure in a reproducible and automated fashion through the minimization of manual setup errors.

ii). Simplified Resource Management:

The deployment of architectures with a number of resources so that all resources can be created in the correct order and proper dependencies can be established and in one go.

iii). Rollback:

Automatic rollback upon unforeseen failures during deployment will ensure the integrity of the infrastructure.

iv). Cost Efficient:

Such usage of templates would require no additional setup costs for duplication of environments whether in development, testing, or production.

8). AWS Systems Manager

Usage:

Parameter Store is used to securely store and manage configuration data, such as database connection strings, API endpoints, and feature flags, required by the SplitApp application.

It provides a centralized repository for configuration management, allowing retrieval during runtime.

Architecture Diagram :

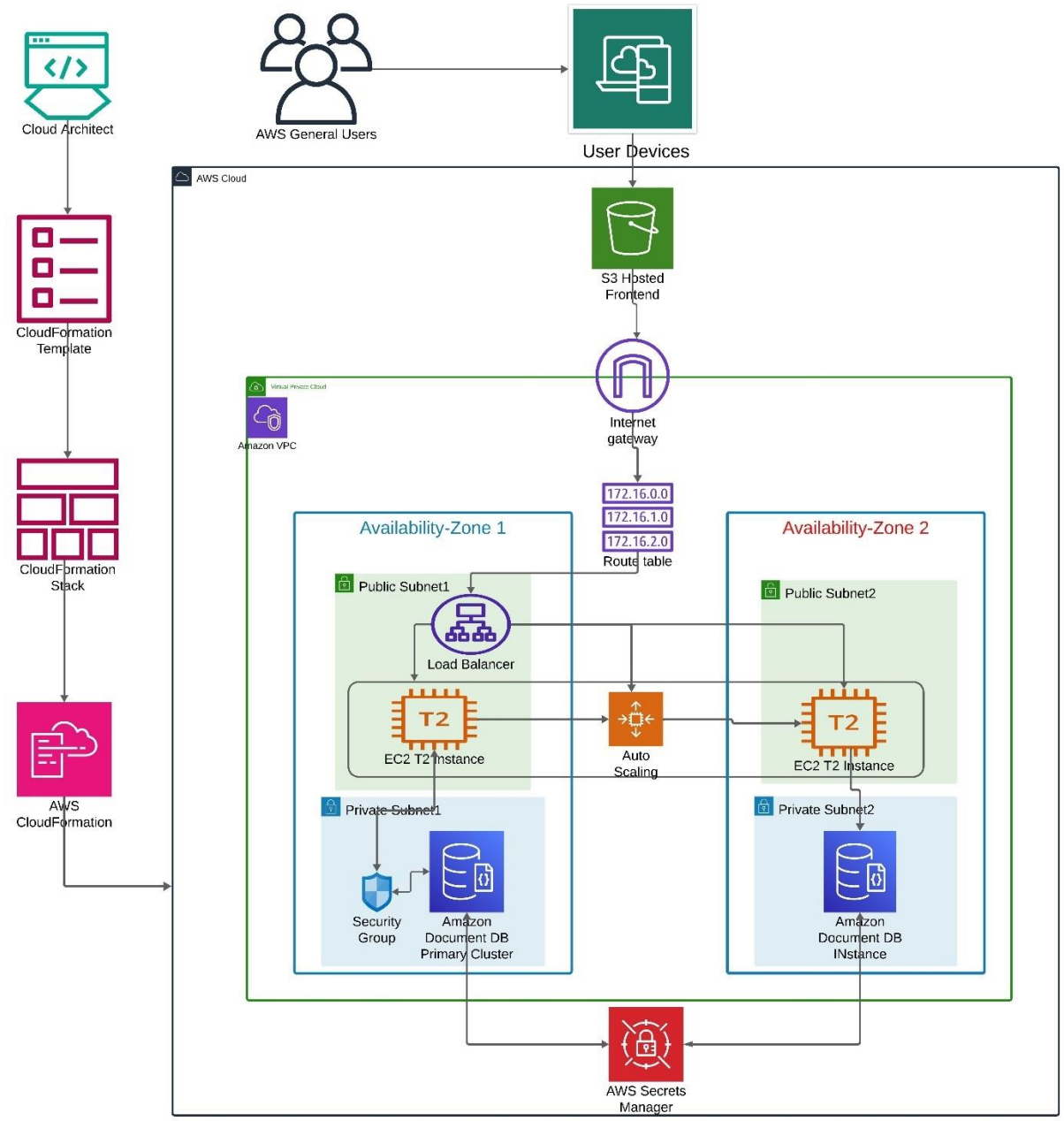

Figure 1: Architecture Diagram

Application Flow :

Frontend S3 :

- When users will try to access the frontend of the website, they will be using the url of the s3 static hosting site.
- S3 is able to auto scale, hence no need for any autoscaling group or any other scaling components for scaling based on the user traffic inbound.

LoadBalancer :

- LoadBalancer will accept inbound traffic from the frontend hosted application, and then forward it to the ec2 instance, and from there, returns the response along with necessary data from the database.

Auto Scaling Group:

- Auto scaling groups will automatically scale EC2 instances if a health check fails.

EC2Launch Template :

- This contains user data script for running the backend application and moreover, it also consists of the auto-scaling group within a VPC region.

DocumentDB:

- This database cluster is in the a private subnet, which consist of two instance of the database, one as main data base cluster, while the second working as a secondary instance. This multi AZ deployment ensures that in event of failover of main cluster, the load is dynamically shifted to

the secondary instance without any requirement of manual intervention.

Secrets Manager:

- Stores the sensitive database credentials of the document DB which is in the private subnet.

AWS Well Architected Framework :

1). Operational Excellence Pillar :

- The Operational Excellence pillar of the AWS Well-Architected Framework focuses on the ability to develop and run workloads efficiently, effectively and gain insight into their operations. [8]

Principle of Operational Excellence :

i). Operation as Code:

Infrastructure as Code (IaC):

- The entire application is being provisioned using the AWS Cloudformation script template. Using this template, I have defined the whole architecture, ensuring that infrastructure is managed as code.
- This IaC allows that the whole infrastructure can be deployed multiple times or at different places while maintaining the automated management of infrastructure changes thus leading to minimal human intervention in infrastructure deployment and eliminates the human errors.
- Cloudformation Template Script link :

https://github.com/LuvPatel/SplitApp-AWS

Cloudformation Flow :

Figure 2: Cloudformation Fiow diagram

Monitoring and Logging :

● It provides a means for one's application performance to be observed as well as troubleshooting of issues can be done much faster.

● At the same time, various services can be enabled such as CloudWatch monitoring applications and logging all important data.

● Provides real-time insight into application performance and operational health, enabling issues to be detected and resolved more quickly.

2). Security Pillar :

The Security pillar of the AWS Well-Architected Framework is aimed at protecting data, systems, and security measures as well as best practices.

On infrastructure protection: Infrastructure include VPC in order to create a private subnet, which will be : secured, and a scaled environment in which we can define our topology—including gateways, routing tables, and public and private subnets.

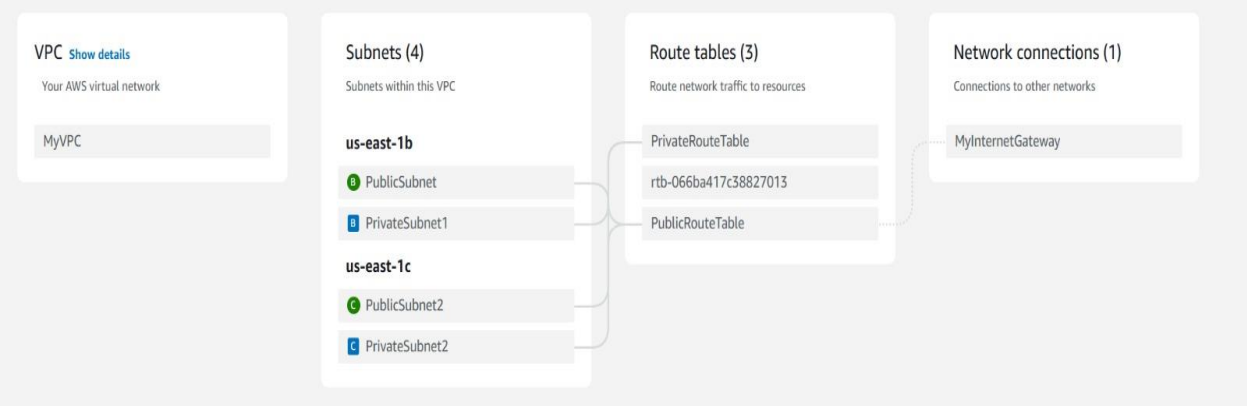

Figure 3: VPC structure diagram

- In SplitApp I have VPC, with a 2 public and 2 private subnet, containing DocumentDB.
- Security group of DocumentDB, which is in a private subnet : only allows inbound rule from security group of EC2, all others will be rejected.
- Security group of load balancer will also allow only HTTP port 80, which will control traffic for load balancer.

3). Reliability Pillar :

- The pillar of Reliability in the AWS Well-Architected Framework is all about making sure that a workload does what it was designed

to do at the right time and in the right way- usually, but not always.

High Availability and Fault Tolerance

- Our backend EC2 instance is in Auto scaling group with a load balancer in a public subnet (us-east-1b and us-east-1c) ensuring high availability.

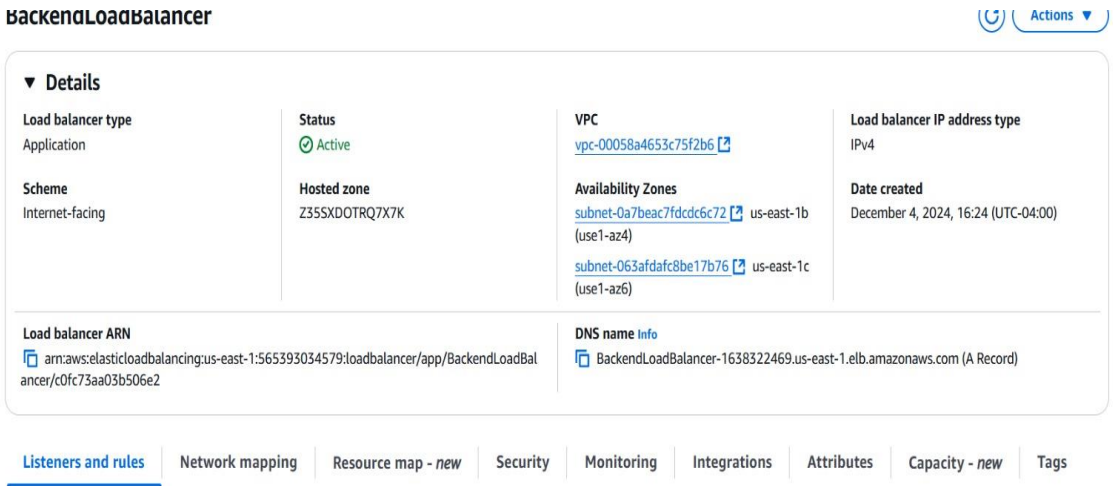

Figure 4: Multiple Availability Zones

Auto Scaling :

- Automates the scaling of backend EC2 instances into or out from time to time based on demand while keeping optimal application performance and availability despite the failure of some instances.

- The unhealthy instances in an auto-scaling group are automatically replaced so that the impact of instance failures is minimized.

Load Balancer :

- The spreading of traffic over several back-end instances according to the application's capacity, so that it does not go down if any of these individual instances go down.

4). Performance Efficiency Pillar :

Using the cloud resource optimally to meet system requirements and to enhance performance defines the Performance Efficiency pillar of the AWS Well-Architected Framework.

EC2 Instances: The application architecture has used EC2 Instances for back-end. It would allow dynamically changing the number and sizes of the instances according to the traffic and workload. Thus, using the appropriate instance types and sizes for your applications would be assured to have zero to negligible downtime.

Thus, we are able to achieve correct sizing of EC2 instances by the Auto-scaling feature.

Advanced Technologies :

- The complexity of database management - backups, patching, and scaling - is offloaded by using the managed, more advanced services of AWS, such as DocumentDB, thus enabling me to focus more on application development.

Global availability:

Multi-AZ Deployment :

- This implies the presence of multi-AZ deployment given the fact that it caters applications being accessed from various data centers and regions. It gives the applications regional resilience, hence making it perform consistently across various geographical areas.

5). Cost Optimization Pillar :

- This pillar of the AWS Well-Architected Framework speaks of winterizing costs and optimizing the value of one's investment. It shall be concerned with cost control and maximizing the value of investment through the use of AWS services tremendously.

- The Cost Optimization pillar of the AWS Well-Architected Framework lets the owner manage and maximize value for investments made in AWS.

Adopting a consumption model:

- Auto-Scaling: Determining how many instances will be in the running state, contingent upon traffic and load, means that i'll only be paying for what I've consumed. Reduces the cost of running instances when demand is low by scaling down the instances running at such times.

Application Screenshots:

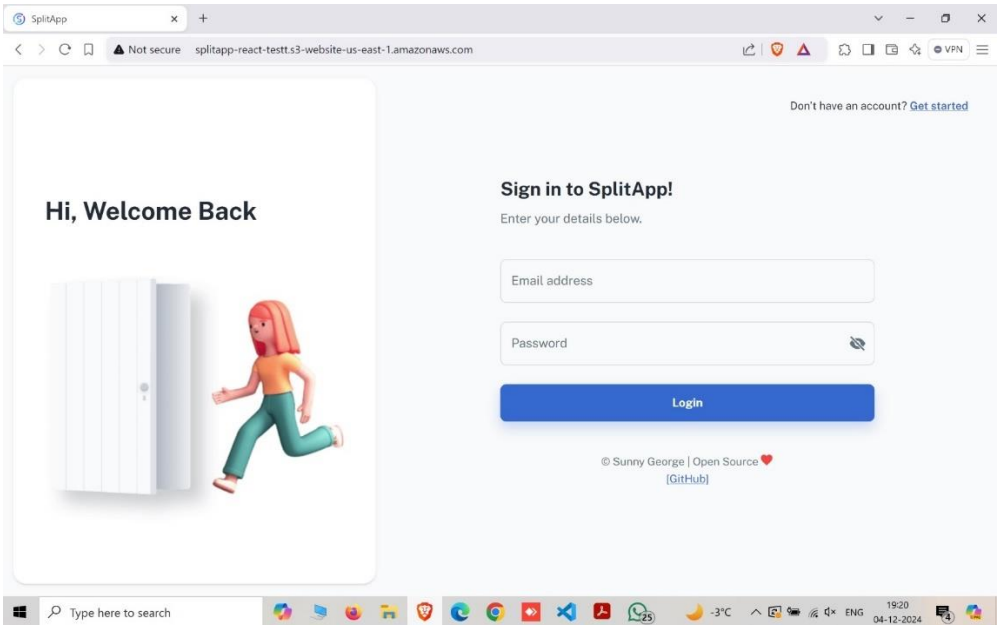

Figure 5: Application Login Page

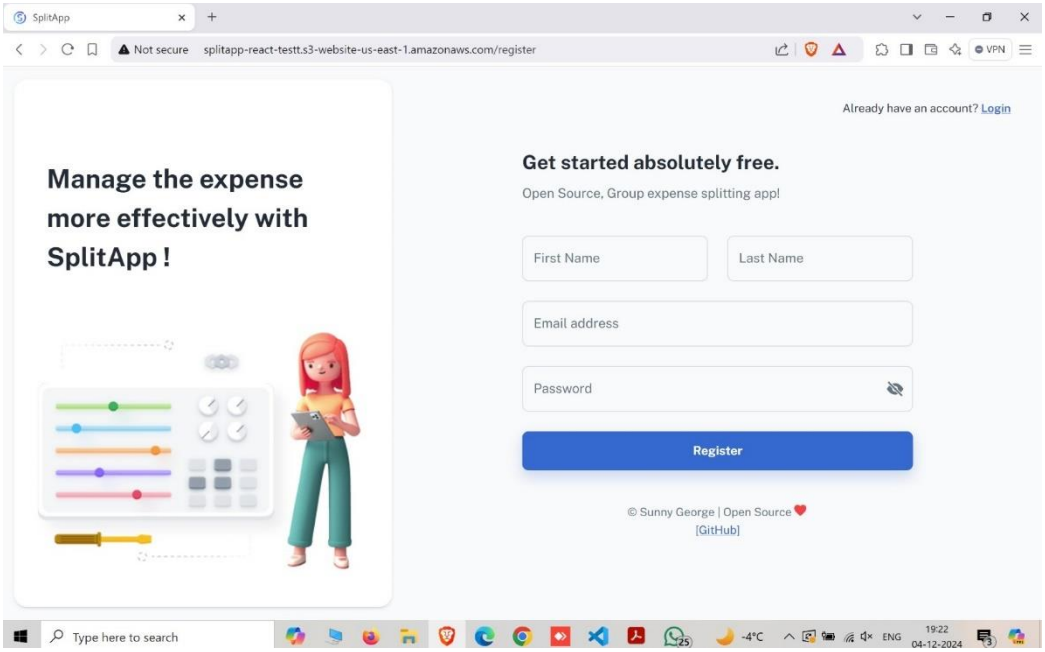

Figure 6: Signup Page

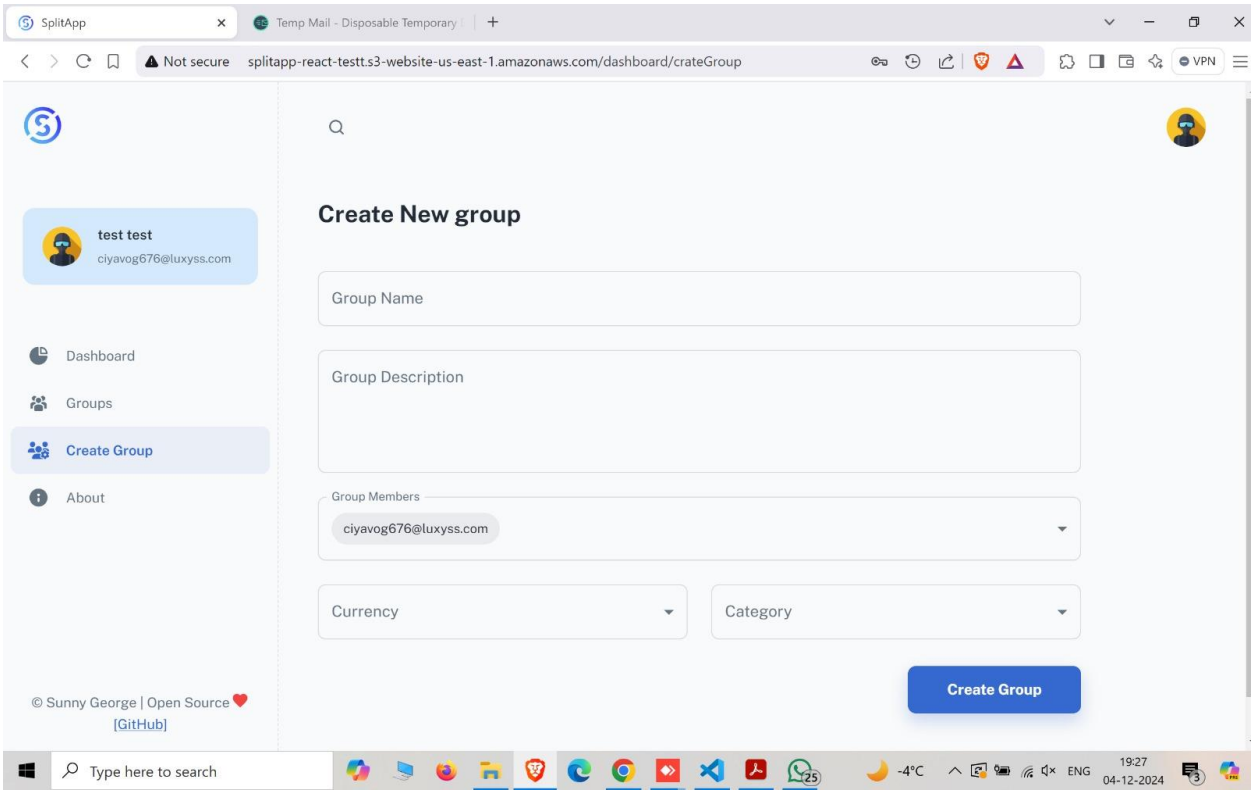

Figure 7: Create Group Page

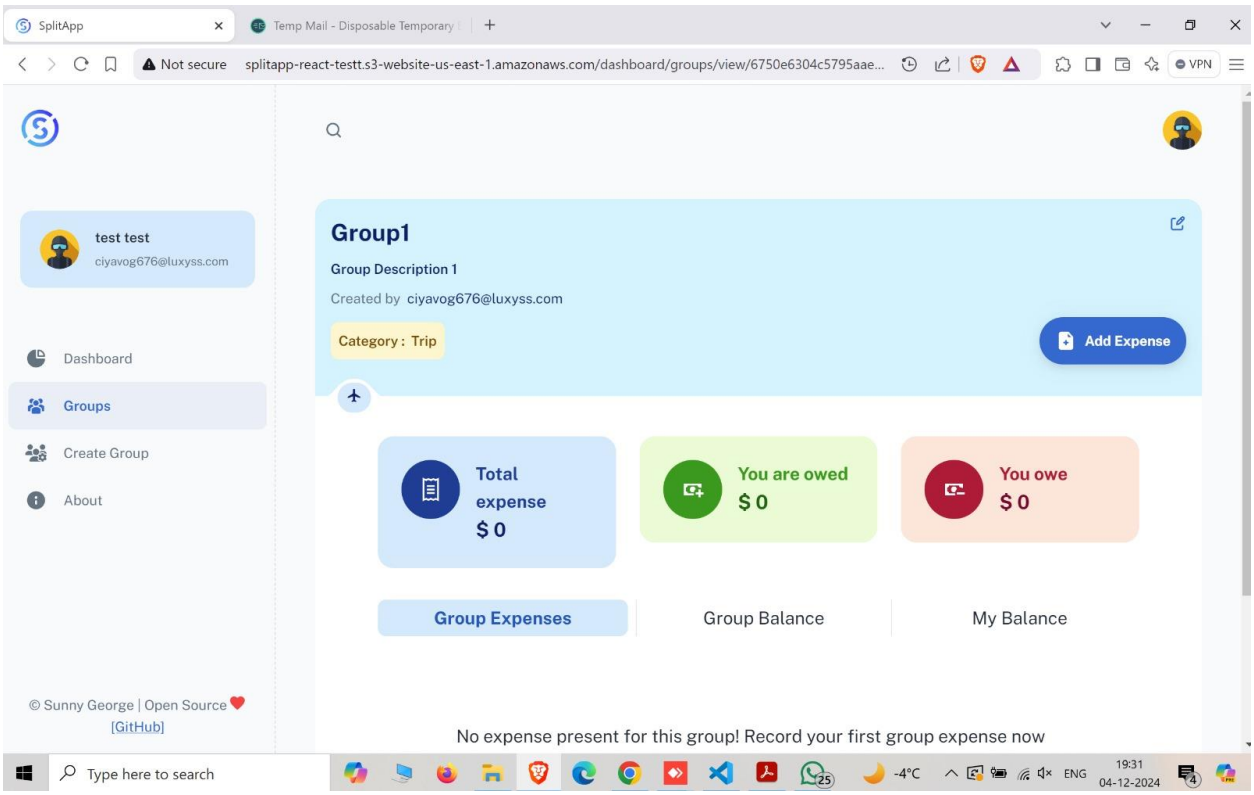

Figure 8 : Group Created Successfully

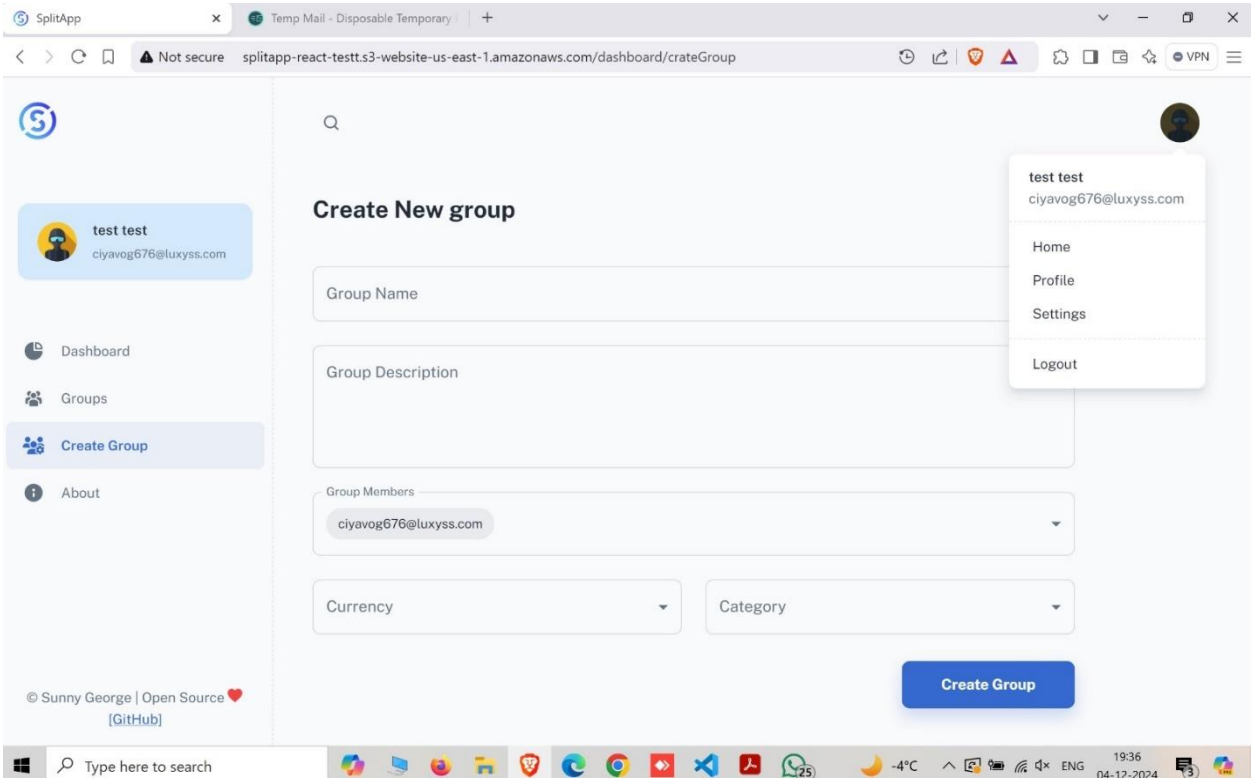

Figure 9: Login Successful

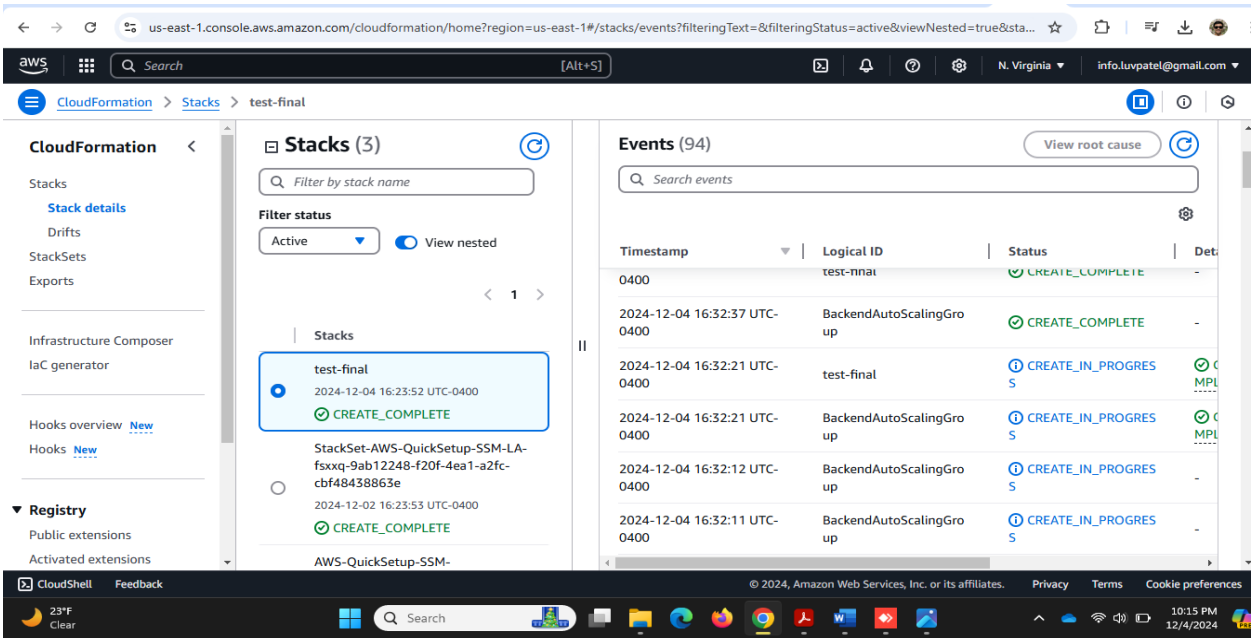

Figure 10: Successful stack creation

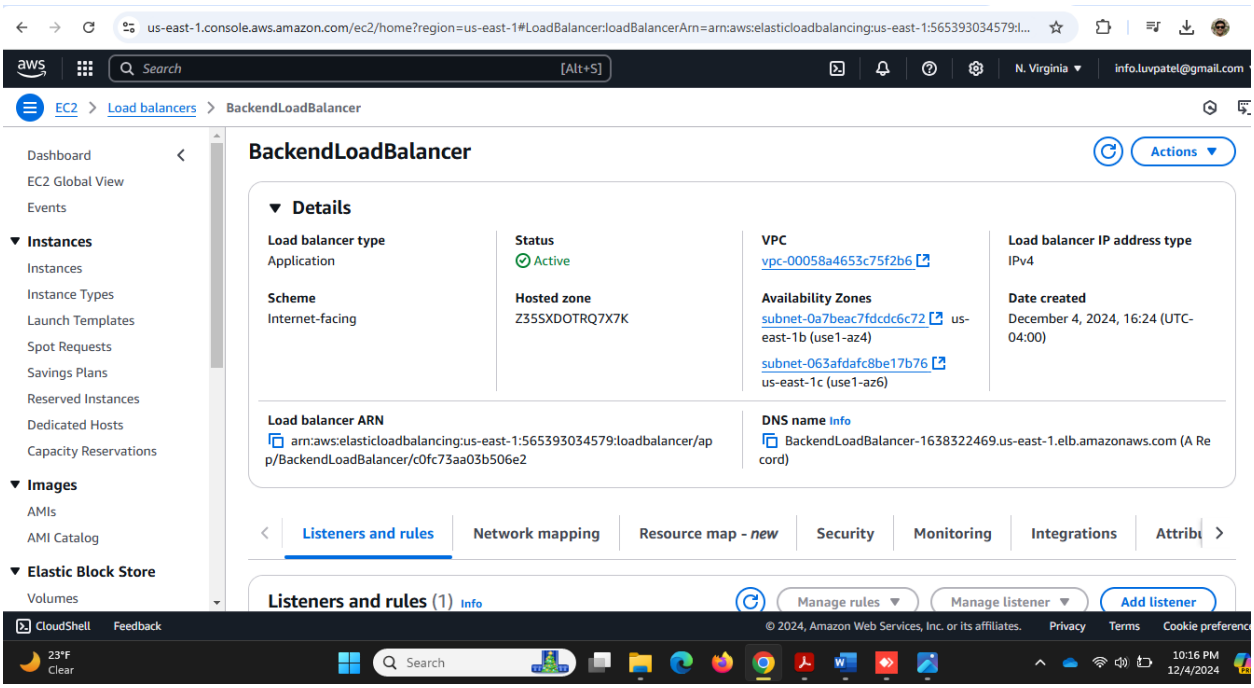

Figure 11: Loadbalancer up and running

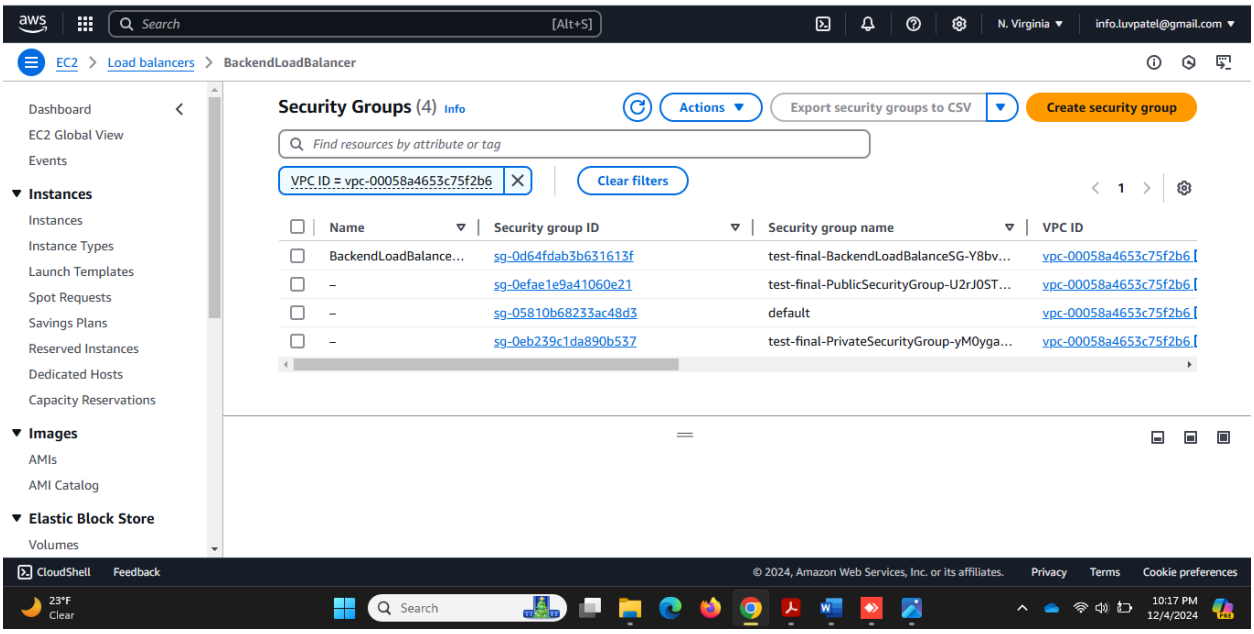

Figure 12: Security groups in VPC

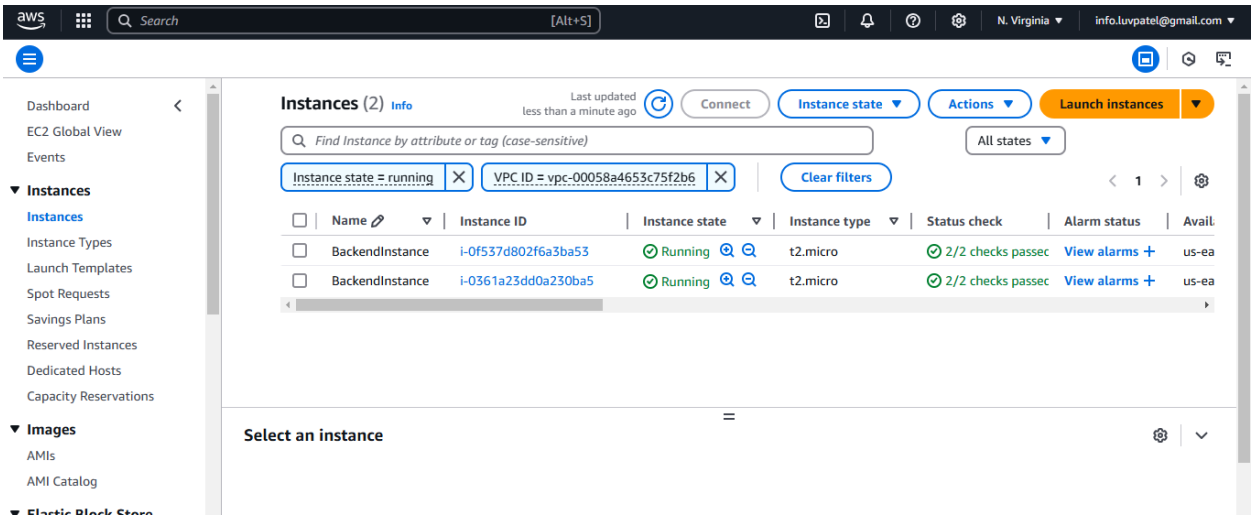

Figure 13: EC2 up and running

References :

[1] Amazon Web Services, "Amazon EC2 Reserved Instances Pricing," Amazon Web Services. [Online]. Available: https://aws.amazon.com/ec2/pricing/reserved-instances [Accessed: December 4, 2024].

[2] Amazon Web Services, "Amazon EC2," Amazon Web Services. [Online]. Available: https://aws.amazon.com/ec2/ . [Accessed: December 4, 2024].

[3] Amazon Web Services, "S3 Storage Classes," Amazon Web Services. [Online]. Available: https://aws.amazon.com/s3/storage-classes/ . [Accessed: December 4, 2024].

[4]. Amazon Web Services, "Using Encryption with Amazon S3," Amazon Web Services. [Online]. Available: https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingEncryption.html . [Accessed: December 4, 2024].

[5] Amazon Web Services, "Amazon DocumentDB Pricing," Amazon Web Services. [Online]. Available: https://aws.amazon.com/documentdb/pricing/ . [Accessed: December 4, 2024].

[6] Amazon Web Services, "Amazon DocumentDB," Amazon Web Services. [Online]. Available: https://aws.amazon.com/documentdb/ .[Accessed: December 4, 2024].

[7] Amazon Web Services, "What is Amazon VPC?" Amazon Web Services. [Online]. Available: https://docs.aws.amazon.com/vpc/latest/userguide/what-is-amazon-vpc.html .[Accessed: December 4, 2024]