Project Overview

Introduction:

The **Inventory Management Application** is a powerful solution designed to centralize and streamline large inventories. Deployed on **Amazon Web Services (AWS)**, it adheres to a **serverless architecture**, ensuring scalability and cost-effectiveness. Leveraging key services such as **AWS Lambda** for business logic, **Amazon RDS (MySQL)** for data storage, **Amazon API Gateway** for secure API exposure, **Amazon Cognito** for user authentication, and **Amazon S3** for static asset storage, the application efficiently manages inventory items. Infrastructure provisioning is handled seamlessly using **Terraform**. Additionally, an **EC2 bastion host** facilitates secure access to the RDS database. All project code resides in the GitHub repository [harjotbasota/InventoryManagementSystem (github.com)]

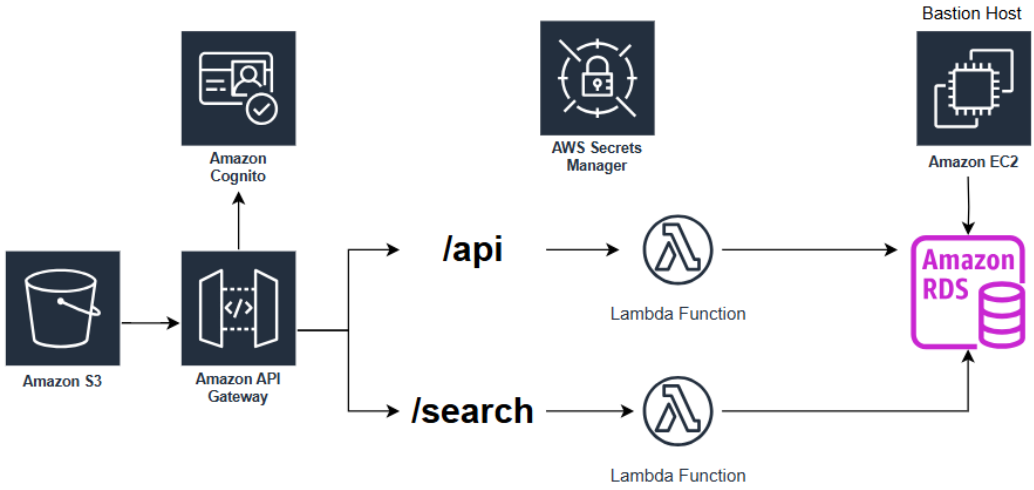
Goals:

1. **Centralization**: Successfully centralized inventory management processes.
2. **Scalability**: Designed a scalable solution using serverless architecture.
3. **Secure Authentication**: Implemented secure user authentication with Amazon Cognito.
4. **Efficient Data Storage**: Leveraged Amazon RDS (MySQL) for efficient data management.
5. **Infrastructure as Code**: Utilized Terraform for seamless infrastructure deployment.
6. **Real-time Tracking**: Enabled real-time visibility into inventory levels.
7. **Cost Optimization**: Optimized resource usage to minimize operational costs.

Technology Used:

- AWS S3
- AWS lambda
- AWS API gateway (REST API)
- AWS Cognito
- Python
- AWS EC2 (Bastion Host)
- Shell Scripting
- Terraform

Architecture:



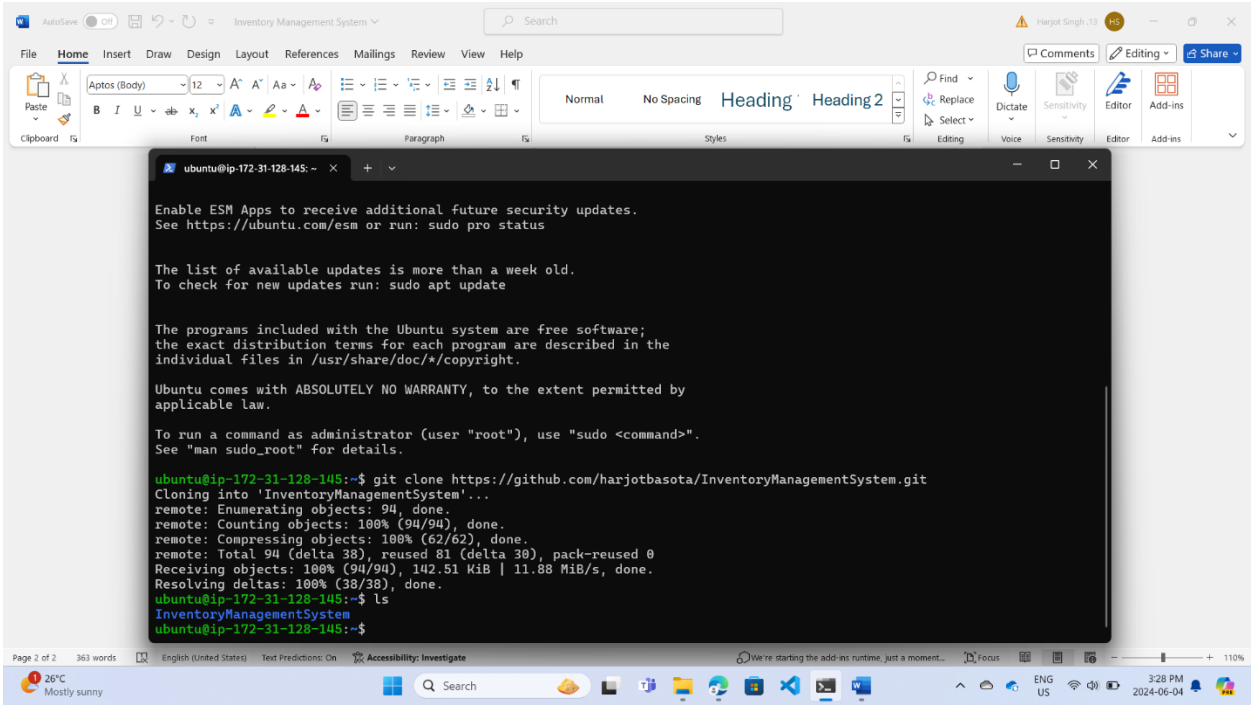
Deployment Overview:

1. Setup the control server by cloning git repository, writing credential files and running setup script. Before you begin you should have ssh keys which you will use for bastion host in this project and you need to change the value of key name in variables file in /bastionhost
2. Execute main.sh to deploy the infrastructure. During deployment you will be prompted for DB password which is available in AWS Secrets Manager
3. Once the bastion host is created ssh into bastion host and use it to setup the RDS server according to queries provided into rds_setup file
4. After all the deployment is complete go to s3 static website URL to access the application
5. Go to Login section and sign up to create user and the Login
6. Your Authentication key will be displayed at home page and using this key you can view, update or delete data from inventory
7. Once you are done execute cleanup.sh to destroy all the resources

Deployment Detail:

1. Create a control server with port 22 open and use ubuntu AMI. Also, make sure it has Public IP
2. SSH into control server and clone this git repository

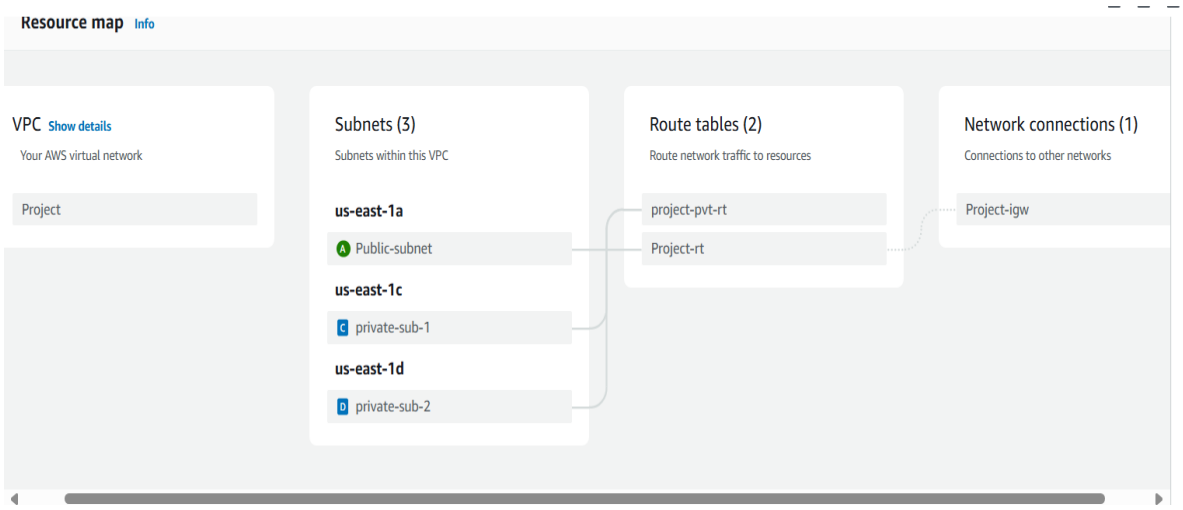
$ git clone https://github.com/harjotbasota/InventoryManagementSystem.git and you need to go to /Infrastructure/bastionhost/terraform.tfvars and change the ssh key name to the name of your key.



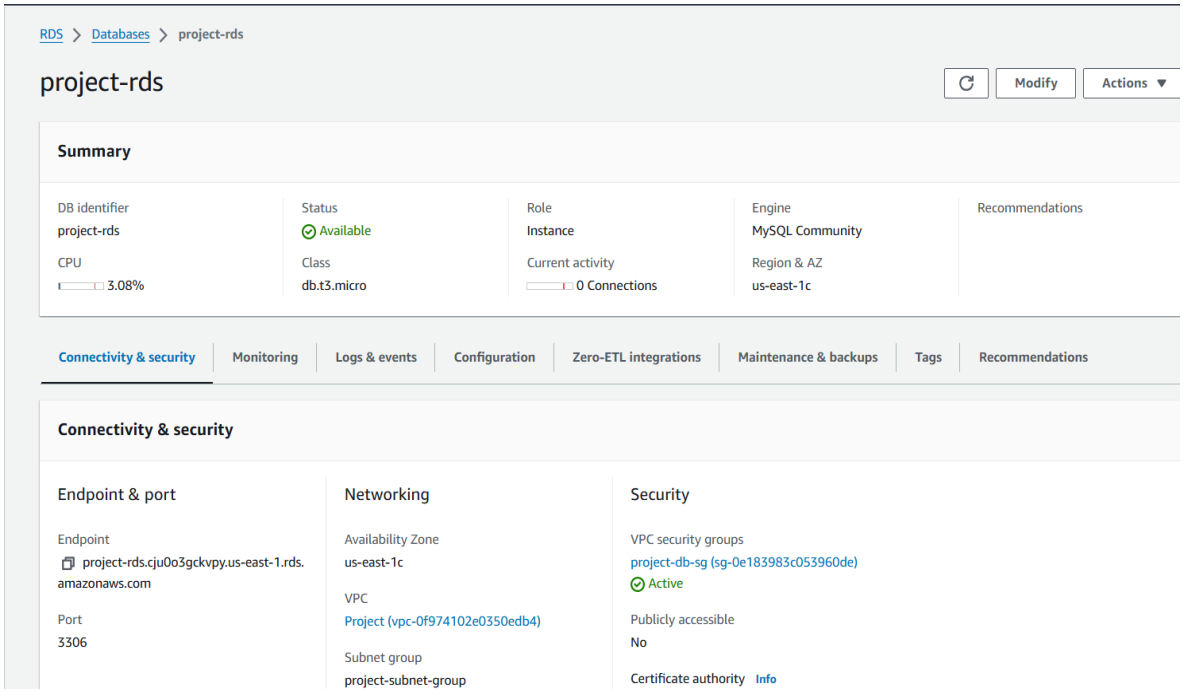
3. Create a credentials file at ~/.aws/credentials
   [default]
   aws_access_key_id= “YOUR ACCESS KEY HERE”
   aws_secret_access_key= “YOUR SECRET KEY HERE”
4. Make setup.sh file execute file and run bash setup.sh

```
ubuntu@ip-172-31-128-145:~$ mkdir .aws
ubuntu@ip-172-31-128-145:~$ vi ~/.aws/credentials
ubuntu@ip-172-31-128-145:~$ ls
InventoryManagementSystem
ubuntu@ip-172-31-128-145:~$ cd Inventory*
ubuntu@ip-172-31-128-145:~/InventoryManagementSystem$ ls
Infrastructure  README.md  cleanup.sh  frontend  insertlinks.sh  main.sh  rds_setup  setup.sh
ubuntu@ip-172-31-128-145:~/InventoryManagementSystem$ chmod +x setup.sh
ubuntu@ip-172-31-128-145:~/InventoryManagementSystem$ bash setup.sh
```

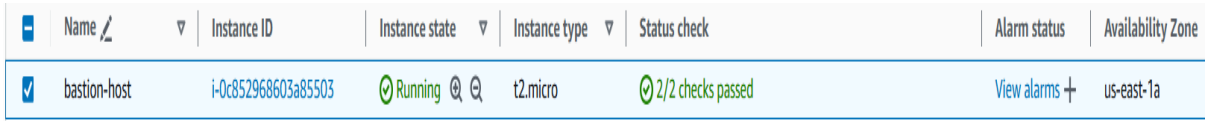
   This script will update the cache, install terraform and make other required scripts executable
5. Execute bash main.sh to start deploying the infrastructure for this project. This script will
   - Create VPC for this project will the required components such as ACLs, Route Tables, Internet gateway , subnets, security groups etc.



- Create an RDS instance with MYSQL server engine and its password will be managed by AWS Secrets Manager



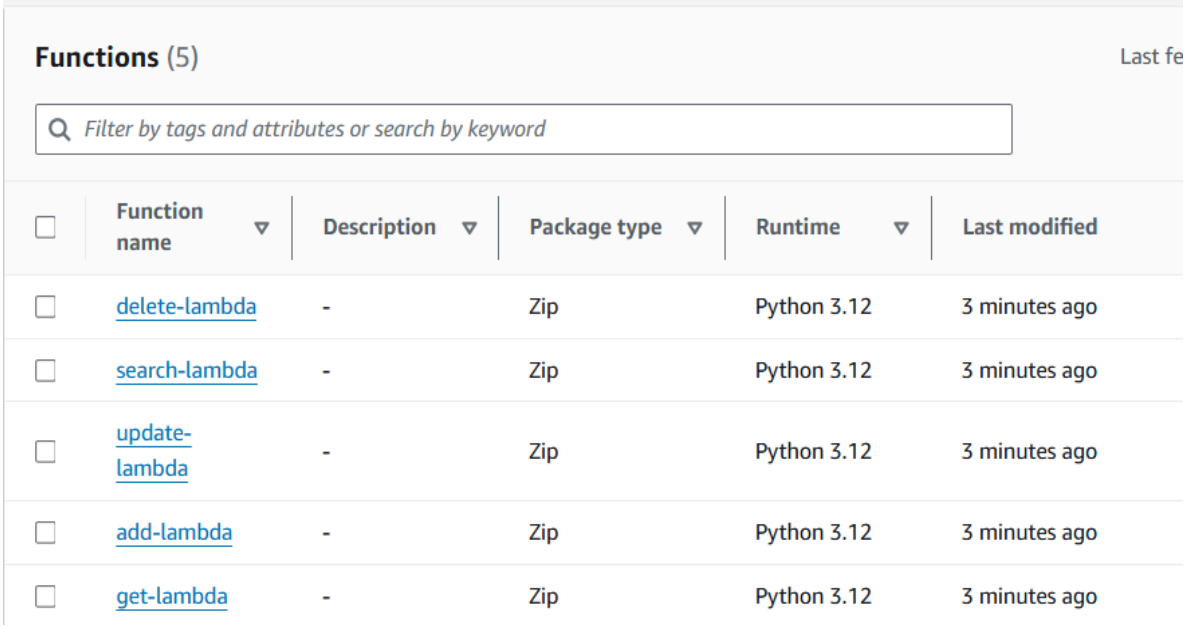
- Create the bastion host for accessing our MYSQL Server



- Create Lambda functions for our backend logic. In this step it will ask you for db_password. This is the password which is stored in AWS Secrets Manager and is not asking for creating password for your database as it has already been created in previous step. So, copy that password here

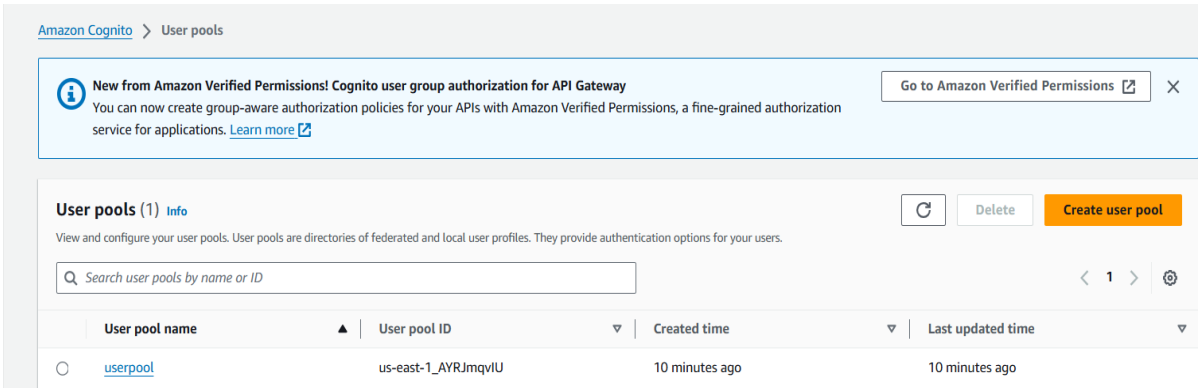
```
If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
var.db_password
  Enter a value: U>je.lz:g+9UQRCG.~{BW4g+Mm5N
```

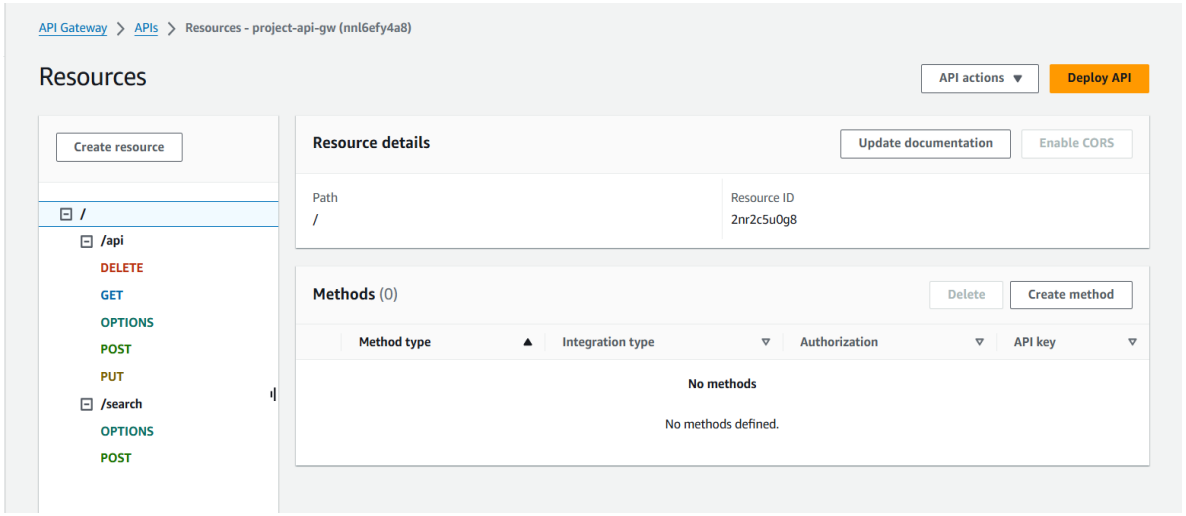
  This step is for setting up our environment variables for lambda functions so that they can connect to our MYSQL Server.



- It will create s3 bucket with all the required configurations and policies. Bucket will be named harjotbasota. You can change the name to your desired value by editing the s3.tf file

- Then it will setup the Cognito userpool for our application and also create the API gateway with all the desired resources, methods, authorizers etc.





- It will execute insertlinks.sh. This is to supply the API endpoints and cognito login, logout URLs to our frontend code. In our index.html

```
<a id="authlinks" href="#1"> Login</a>
<a id="authlinks" href="#2"> Logout</a>
```

This will be changed to (will retrieve values from terraform for our endpoints)

```
        <h1> INVENTORY MANAGEMENT SYSTEM</h1>
        <a id="authlinks" href="https://basota.auth.us-east-1.amazoncognito.com/login?client_id=5m3dsph7nav15e4li1lleecg
lc&response_type=token&scope=aws.cognito.signin.user.admin+email+openid+phone+profile&redirect_uri=https%3A%2F%2Fharjotb
asota.s3.amazonaws.com%2Findex.html"> Login</a>
        <a id="authlinks" href="https://basota.auth.us-east-1.amazoncognito.com/logout?client_id=5m3dsph7nav15e4li1lleec
glc&response_type=token&redirect_uri=https%3A%2F%2Fharjotbasota.s3.amazonaws.com%2Findex.html"> Logout</a>
```

And in other html files

```
// make API call with parameters and use promises to get response
fetch("#", requestOptions)
    .then(response => response.json())
    .then(data => {
        const mydata= data.body;
```

Will be changed to (will replace # with API invoke URL)

```
// make API call with parameters and use promises to get response
fetch("https://nnl6efy4a8.execute-api.us-east-1.amazonaws.com/stage/api", requestOptions)
   .then(response => response.json())
   .then(data => {
        const mydata= data.body;
```

- In final step will be upload the updated frontend files to s3

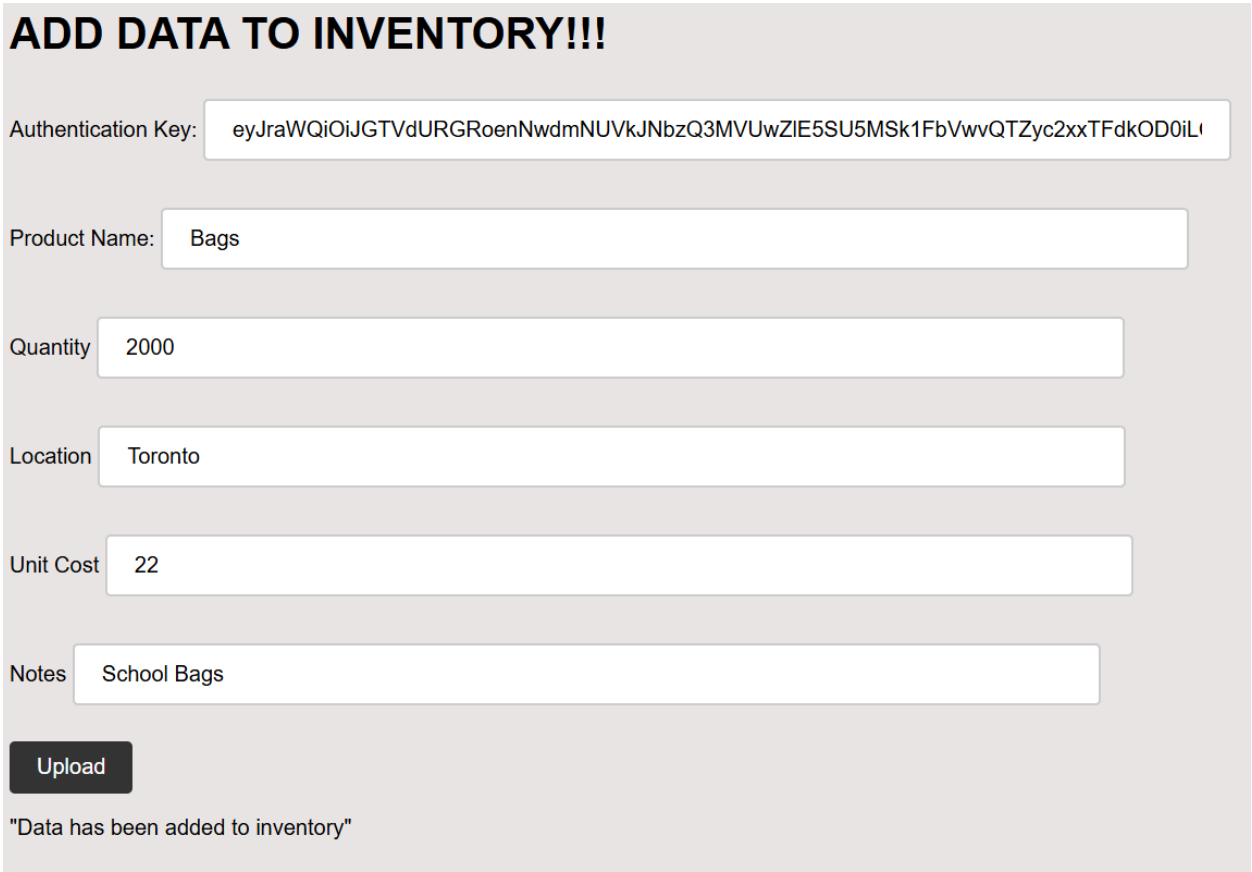
```
Plan: 6 to add, 0 to change, 0 to destroy.
aws_s3_object.delete: Creating...
aws_s3_object.upload: Creating...
aws_s3_object.getfiltered: Creating...
aws_s3_object.get: Creating...
aws_s3_object.index: Creating...
aws_s3_object.update: Creating...
aws_s3_object.delete: Creation complete after 0s [id=delete.html]
aws_s3_object.update: Creation complete after 0s [id=update.html]
aws_s3_object.upload: Creation complete after 0s [id=upload.html]
aws_s3_object.get: Creation complete after 0s [id=get.html]
aws_s3_object.getfiltered: Creation complete after 0s [id=getfiltered.html]
aws_s3_object.index: Creation complete after 0s [id=index.html]
```

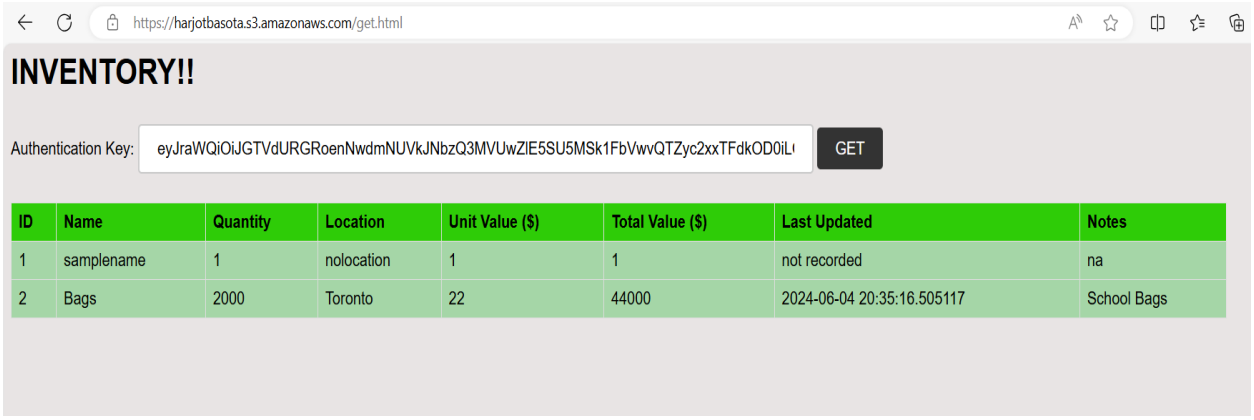
6. Now we will setup the RDS instance. First ssh the EC2 instance which is named bastionhost. Now open the rds_setup file in control server and follow the instructions to setup the RDS

```
ubuntu@ip-20-0-1-183:~$ mysql -h project-rds.cju0o3gckvpy.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 33
Server version: 8.0.35 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> use inventory;
Database changed
MySQL [inventory]> CREATE TABLE inventory (
    ->   id INT AUTO_INCREMENT PRIMARY KEY,
    ->   name VARCHAR(255),
    ->   quantity VARCHAR(255),
    ->   location VARCHAR(255),
    ->   unitcost VARCHAR(255),
    ->   totalcost VARCHAR(255),
    ->   lastupdate VARCHAR(255),
    ->   notes VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.031 sec)
```

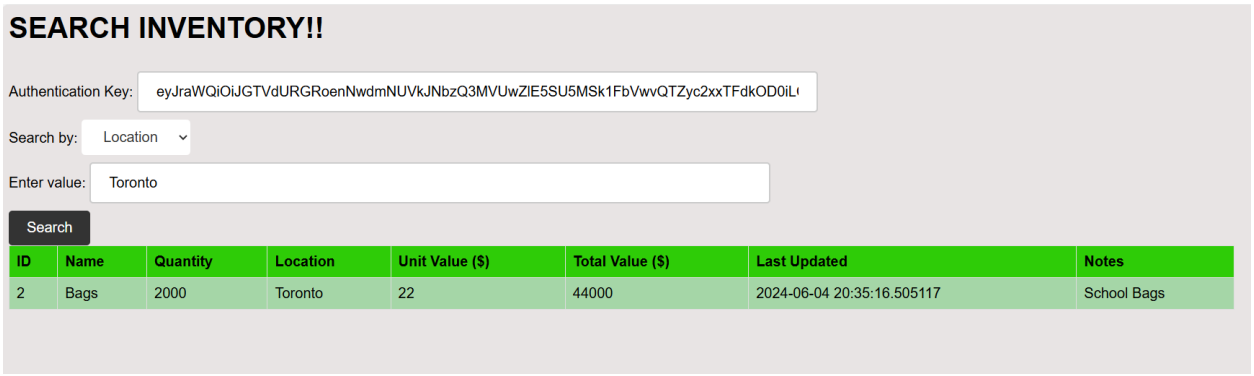
7. Open your browser and go to s3 bucket website endpoint. Click green login button and the click sign up. Complete all the steps and verify your email. After everything you will be redirected to homepage and your authentication key will be available

8. Copy this key and go to Add Data, Search , Update Inventory, View and perform the operations

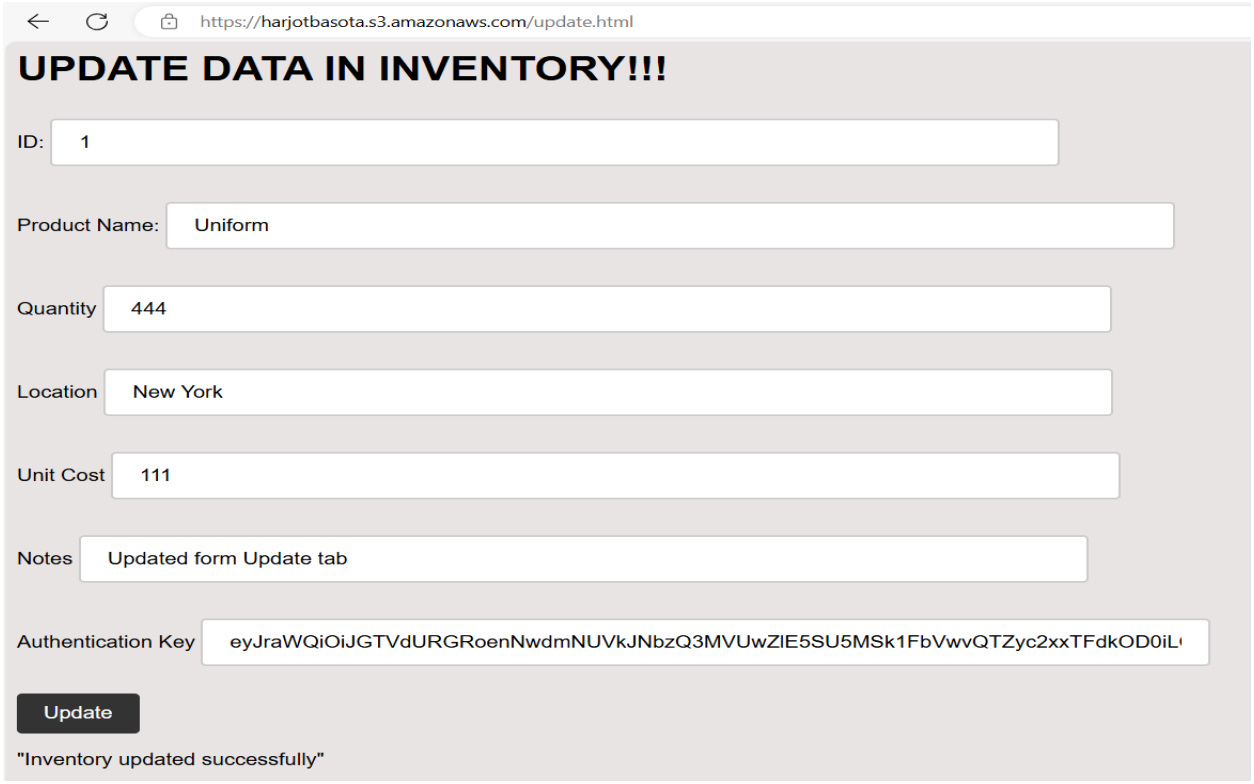


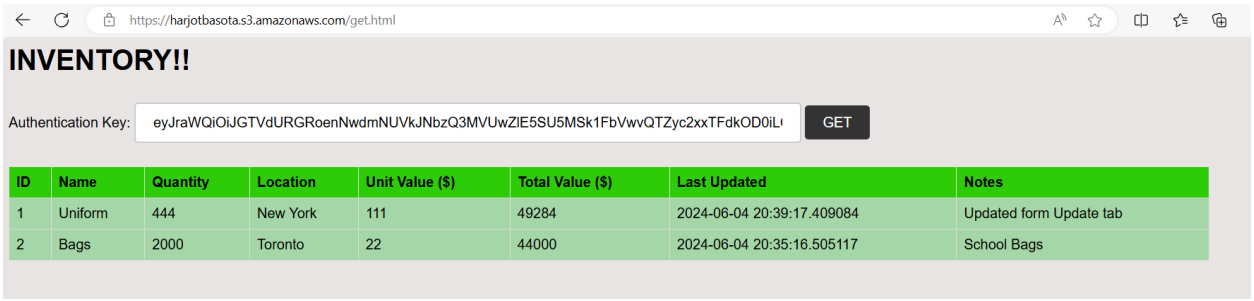


ID	Name	Quantity	Location	Unit Value ($)	Total Value ($)	Last Updated	Notes
1	samplename	1	nolocation	1	1	not recorded	na
2	Bags	2000	Toronto	22	44000	2024-06-04 20:35:16.505117	School Bags

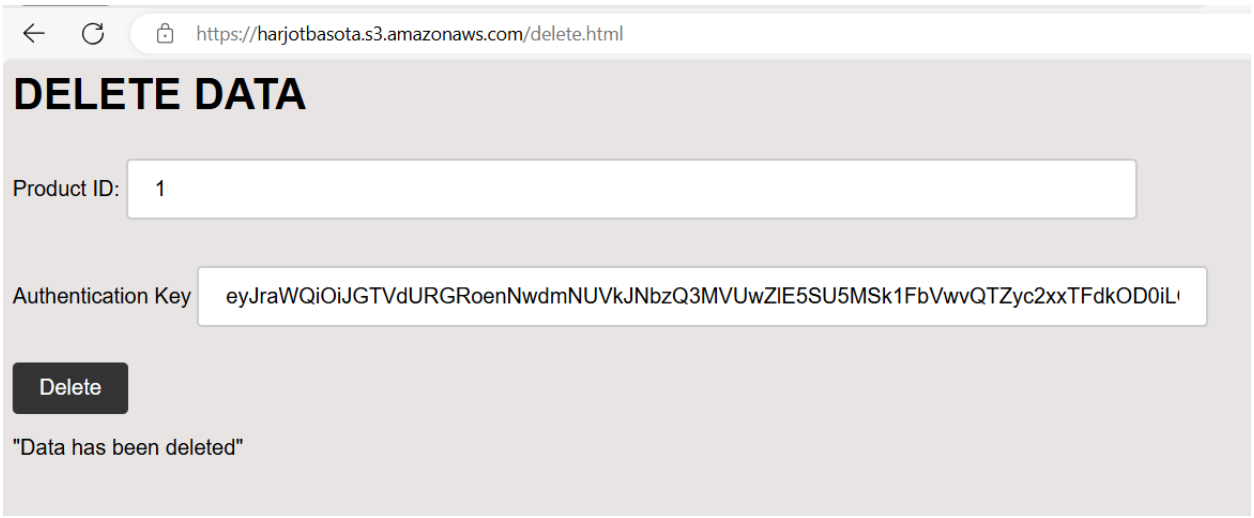


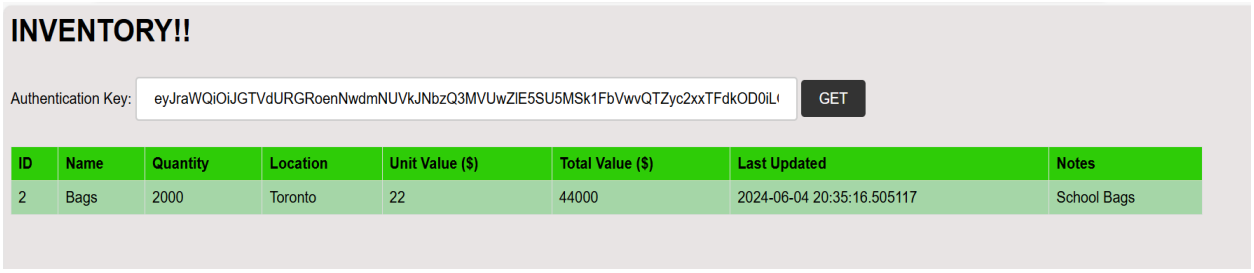
ID	Name	Quantity	Location	Unit Value ($)	Total Value ($)	Last Updated	Notes
2	Bags	2000	Toronto	22	44000	2024-06-04 20:35:16.505117	School Bags



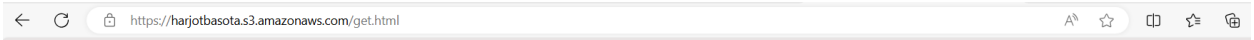


ID	Name	Quantity	Location	Unit Value ($)	Total Value ($)	Last Updated	Notes
1	Uniform	444	New York	111	49284	2024-06-04 20:39:17.409084	Updated form Update tab
2	Bags	2000	Toronto	22	44000	2024-06-04 20:35:16.505117	School Bags





ID	Name	Quantity	Location	Unit Value ($)	Total Value ($)	Last Updated	Notes
2	Bags	2000	Toronto	22	44000	2024-06-04 20:35:16.505117	School Bags



INVENTORY!!

ID	Name	Quantity	Location	Unit Value ($)	Total Value ($)	Last Updated	Notes
2	Bags	2000	Toronto	22	44000	2024-06-04 20:35:16.505117	School Bags
3	shoes	100	London	65	6500	2024-06-04 20:42:42.068893	School Shoes
4	Sports Kit	1090	Toronto	150	163500	2024-06-04 20:43:19.848812	Sports Kit
5	Hair bands	33	Toronto	15	495	2024-06-04 20:44:11.775238	NA
6	Hair bands	28	New York	12	336	2024-06-04 20:44:28.888890	NA
7	Uniform	100	New York	15	1500	2024-06-04 20:44:42.826725	NA

9. Execute cleanup.sh to destroy all the resources and make sure all the resources are deleted and also delete the control server.