ASSESSMENT COVERSHEET

Attach this coversheet as the cover of your submission. All sections must be completed.

Section A: Submission Details

Programme	:	BACHELOR OF COMPUTER ENGINEERING TECHNOLOGY
Course Code & Name	:	IBB43203 CLOUD COMPUTING
Course Lecturer(s)	:	ASSOC. PROF. DR. MEGAT NORULAZMI BIN MEGAT MOHAMED NOOR
Submission Title	:	PROJECT AWS SERVICES
Deadline	:	Day 9 Month 1 Year 2025 Time 11:59 PM
Penalties	:	• 5% will be deducted per day to a maximum of four (4) working days, after which the submission will **not** be accepted. • Plagiarised work is an Academic Offence in University Rules & Regulations and will be penalised accordingly.

Section B: Academic Integrity

Tick (√) each box below if you agree:

- [√] I have read and understood the UniKL's policy on Plagiarism in University Rules & Regulations.
- [√] This submission is my own, unless indicated with proper referencing.
- [√] This submission has not been previously submitted or published.
- [√] This submission follows the requirements stated in the course.

Section C: Submission Receipt

Office Receipt of Submission

Date & Time of Submission (stamp)	Student Name(s)	Student ID(s)
9 JAN 2026 2.30 PM	AFIQAH ZAKIRAH BINTI ADNAN AISYAH ALYA HAZIRAH BINTI KAMARUDIN NURSHUHADAH BINTI OMAR	52222122559 52222122213 52222122637

Student Receipt of Submission

This is your submission receipt, the only accepted evidence that you have submitted your work. After this is stamped by the appointed staff & filled in, cut along the dotted lines above & retain this for your record.

Date & Time of Submission (stamp)	Course Code	Submission Title	Student ID(s) & Signature(s)
9 JAN 2026 2.30 PM	IBB43203	PROJECT AWS SERVICES	AFIQAH ZAKIRAH BINTI ADNAN (52222122559) AISYAH ALYA HAZIRAH BINTI KAMARUDIN (52222122213) NURSHUHADAH BINTI OMAR (52222122637)

AWS SERVICES FOR A SIMPLE WEBSITE HOSTING

INTRODUCTION

This project develops a Local Cafe Directory website using AWS cloud services. The website displays information about cafes in Kuala Lumpur, including cafe name, description, location, contact details and images. The web application is hosted on an AWS virtual server, while cafe images are stored using cloud storage services.

The purpose of this project is to demonstrate the use of AWS cloud services and how different components can be implemented to form a simple web application.

METHODOLOGY

The website is hosted on an Amazon EC2 instance, which acts as the compute component of the system. EC2 provides a virtual server where the Apache web server is installed to host the website files. Users can access the website through the EC2 public IP address using a web browser.

Cafe images are displayed on the website where they are stored in Amazon S3. S3 is used to store static image files. When a user visits the website, the browser retrieves the images directly from the S3. Cafe information such as name, description. Location and contact details is managed using Amazon DynamoDB.

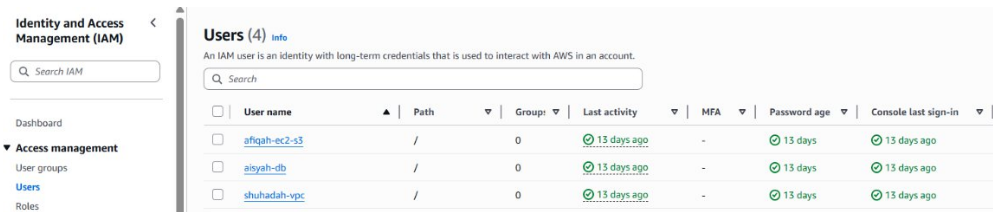
Access control and permissions are managed using AWS Identity and Access Management. IAM is used to create individual users and assign permissions to ensure controlled access. The networking component is handled using the Amazon VPC. Security Groups are configured to allow HTTP traffic for public website access and SSH traffic for server management.

Overall, the integration of IAM, EC2, S3, VOC and DynamoDB enables the website to be securely hosted, publicly accessible with the AWS cloud environment.

STEP-BY-STEP GUIDE

A. Creating users using AWS Identity and Access Management (IAM)

1. Log in to the AWS Management Console using the root account.
2. Navigate to the IAM service.
3. Create IAM user groups based on responsibilities.

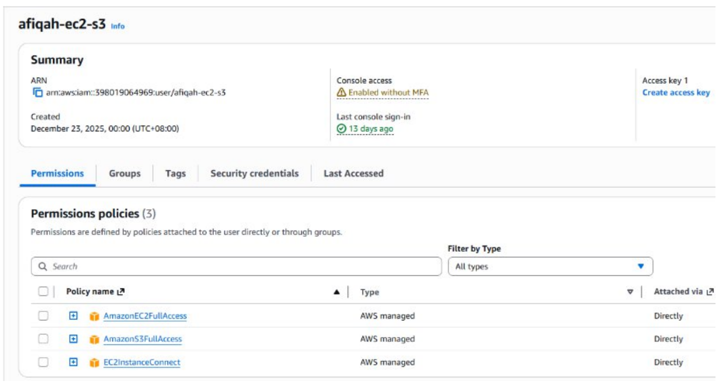


Users (4) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Path	Group:	Last activity	MFA	Password age	Console last sign-in
afiqah-ec2-s3	/	0	13 days ago	-	13 days	13 days ago
aisyah-db	/	0	13 days ago	-	13 days	13 days ago
shuhadah-vpc	/	0	13 days ago	-	13 days	13 days ago

4. Attach required permissions for the service access policies you want to assign to the users.



afiqah-ec2-s3 Info

Summary

ARN
arn:aws:iam::398019064969:user/afiqah-ec2-s3

Created
December 23, 2025, 00:00 (UTC+08:00)

Console access
Enabled without MFA

Last console sign-in
13 days ago

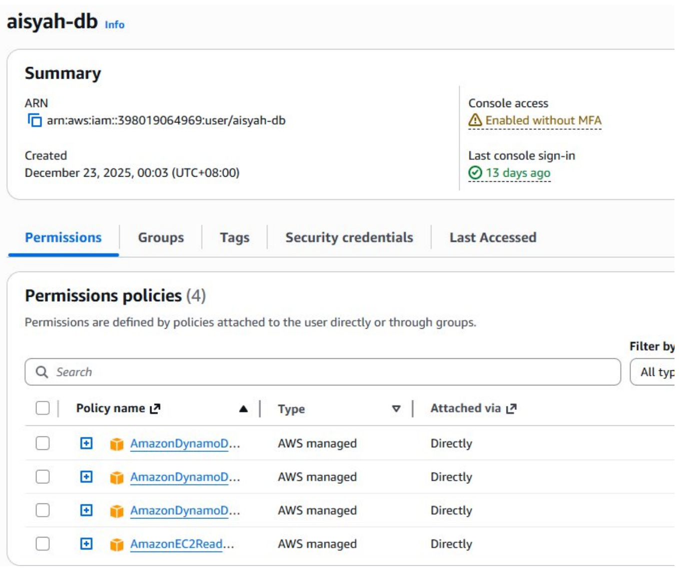
Access key 1
Create access key

Permissions policies (3)

Permissions are defined by policies attached to the user directly or through groups.

Filter by Type: All types

Policy name	Type	Attached via
AmazonEC2FullAccess	AWS managed	Directly
AmazonS3FullAccess	AWS managed	Directly
EC2InstanceConnect	AWS managed	Directly



aisyah-db Info

Summary

ARN
arn:aws:iam::398019064969:user/aisyah-db

Created
December 23, 2025, 00:03 (UTC+08:00)

Console access
Enabled without MFA

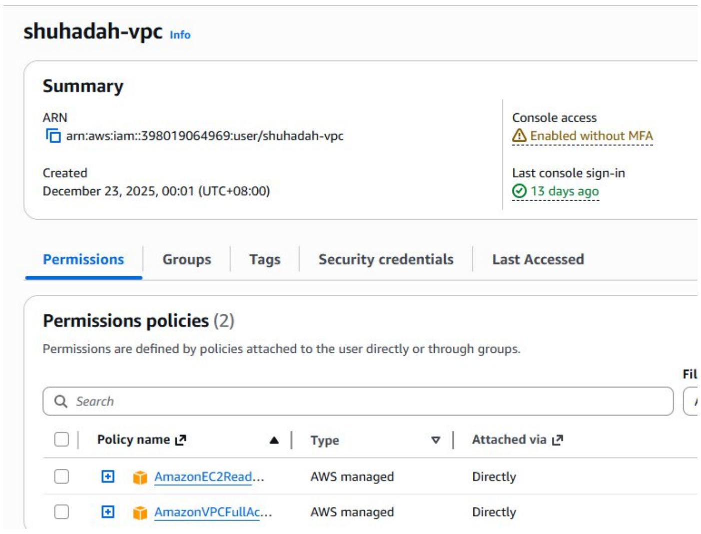
Last console sign-in
13 days ago

Permissions policies (4)

Permissions are defined by policies attached to the user directly or through groups.

Filter by: All typ

Policy name	Type	Attached via
AmazonDynamoD...	AWS managed	Directly
AmazonDynamoD...	AWS managed	Directly
AmazonDynamoD...	AWS managed	Directly
AmazonEC2Read...	AWS managed	Directly



shuhadah-vpc Info

Summary

ARN arn:aws:iam::398019064969:user/shuhadah-vpc	Console access Enabled without MFA
Created December 23, 2025, 00:01 (UTC+08:00)	Last console sign-in 13 days ago

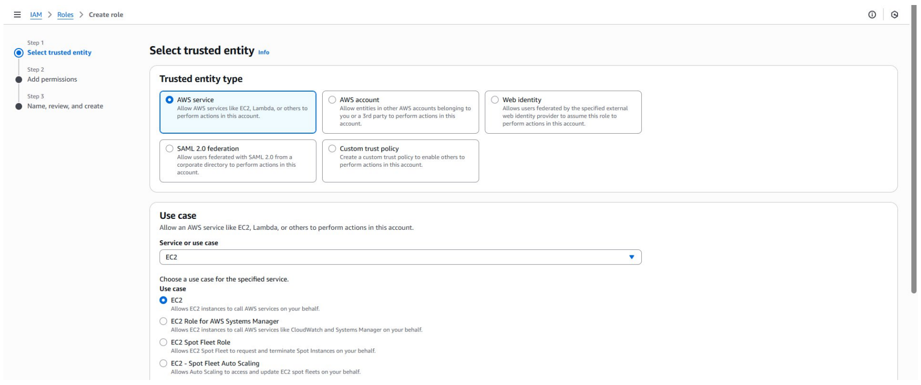
Permissions | Groups | Tags | Security credentials | Last Accessed

Permissions policies (2)

Permissions are defined by policies attached to the user directly or through groups.

Policy name	Type	Attached via
AmazonEC2Read...	AWS managed	Directly
AmazonVPCFullAc...	AWS managed	Directly

5. Users can now log in using the IAM user credentials.
6. Next, create a role for EC2 access to DynamoDB.



IAM > Roles > Create role

Step 1 Select trusted entity
Step 2 Add permissions
Step 3 Name, review, and create

Select trusted entity Info

Trusted entity type

- [✓ AWS service] Allow AWS services like EC2, Lambda, or others to perform actions in this account.
- AWS account: Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
- Web identity: Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
- SAML 2.0 federation: Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
- Custom trust policy: Create a custom trust policy to enable others to perform actions in this account.

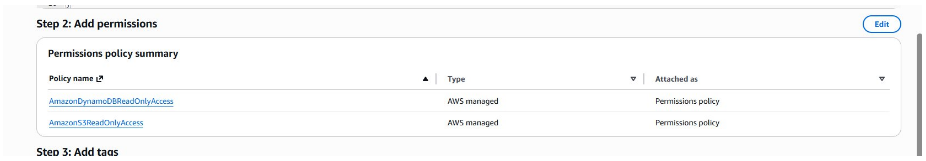
Use case

Allow an AWS service like EC2, Lambda, or others to perform actions in this account.

Service or use case: EC2

Choose a use case for the specified service.
Use case
- [✓ EC2] Allows EC2 instances to call AWS services on your behalf.
- EC2 Role for AWS Systems Manager: Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.
- EC2 Spot Fleet Role: Allows EC2 Spot Fleet to request and terminate Spot Instances on your behalf.
- EC2 - Spot Fleet Auto Scaling: Allows Auto Scaling to access and update EC2 spot fleets on your behalf.

7. Add permissions for read only DynamoDB access, for data fetching to be available later during website hosting.

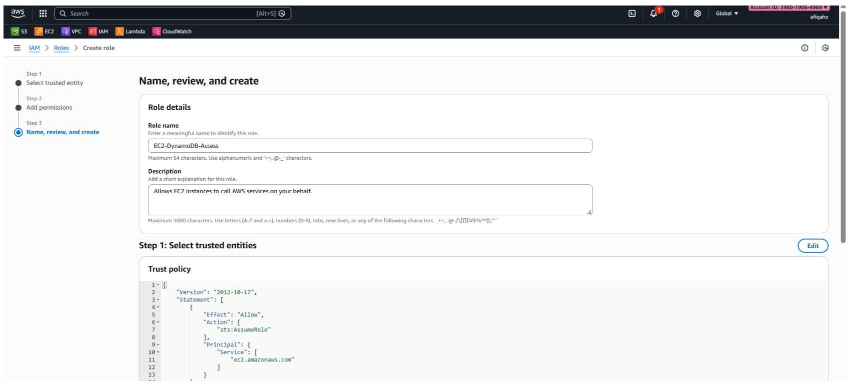


Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AmazonDynamoDBReadOnlyAccess	AWS managed	Permissions policy
AmazonS3ReadOnlyAccess	AWS managed	Permissions policy

Step 3: Add tags



IAM > Roles > Create role

Step 1 Select trusted entity
Step 2 Add permissions
Step 3 Name, review, and create

Name, review, and create

Role details

Role name: EC2-DynamoDB-Access
Maximum 64 characters. Use alphanumeric and '+=,.@-_' characters.

Description: Allows EC2 instances to call AWS services on your behalf.
Maximum 1000 characters. Use letters (A-Z and a-z), numbers (0-9), tabs, new lines, or any of the following characters: _+=,. @-/\[{}]!#$%^*();"' `

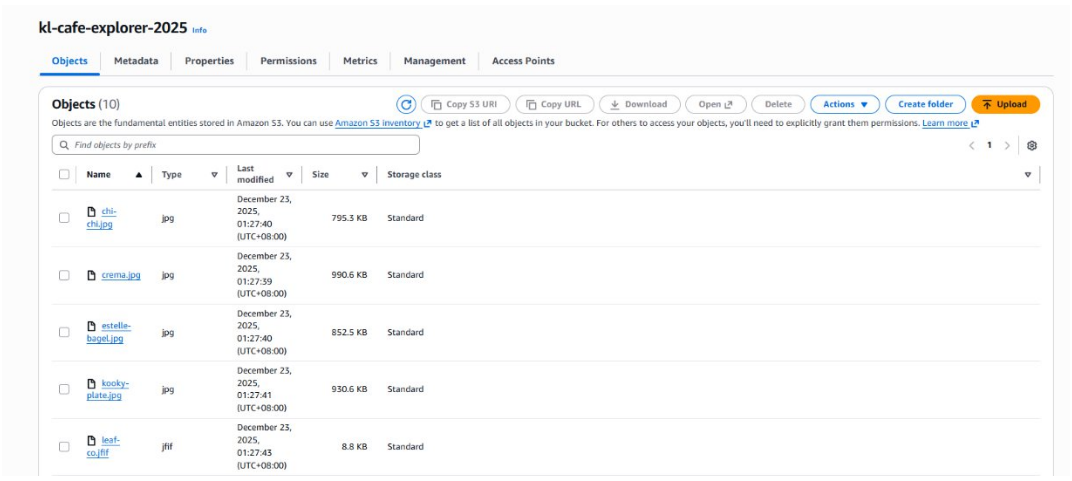
Step 1: Select trusted entities

Trust policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
```

B. Image Storage using Amazon S3

1. Navigate to the Amazon S3 service.
2. Create a new S3 bucket in the same region as the EC2 instance.
3. Disable block public access to allow public image viewing.
4. Upload cafe images to the S3 bucket.



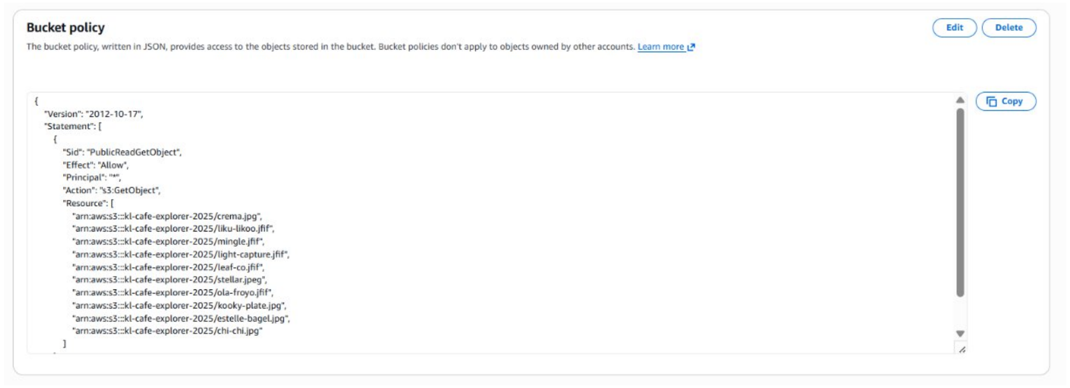
kl-cafe-explorer-2025

Objects (10)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
chi-chi.jpg	jpg	December 23, 2025, 01:27:40 (UTC+08:00)	795.3 KB	Standard
crema.jpg	jpg	December 23, 2025, 01:27:39 (UTC+08:00)	990.6 KB	Standard
estelle-bagel.jpg	jpg	December 23, 2025, 01:27:40 (UTC+08:00)	852.5 KB	Standard
kooky-plate.jpg	jpg	December 23, 2025, 01:27:41 (UTC+08:00)	930.6 KB	Standard
leaf-co.jfif	jfif	December 23, 2025, 01:27:43 (UTC+08:00)	8.8 KB	Standard

5. Configure the bucket policy to allow public read access.



Bucket policy

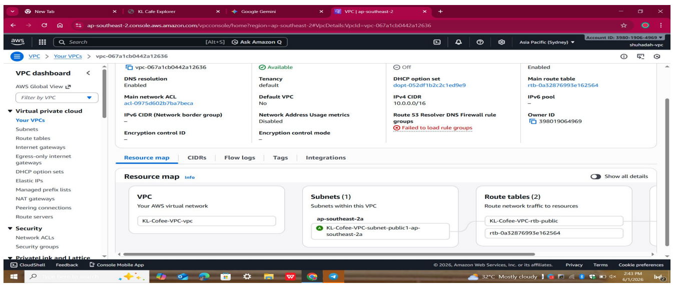
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": [
        "arn:aws:s3:::kl-cafe-explorer-2025/crema.jpg",
        "arn:aws:s3:::kl-cafe-explorer-2025/liku-likoo.jfif",
        "arn:aws:s3:::kl-cafe-explorer-2025/mingle.jfif",
        "arn:aws:s3:::kl-cafe-explorer-2025/light-capture.jfif",
        "arn:aws:s3:::kl-cafe-explorer-2025/leaf-co.jfif",
        "arn:aws:s3:::kl-cafe-explorer-2025/stellar.jpeg",
        "arn:aws:s3:::kl-cafe-explorer-2025/ola-froyo.jfif",
        "arn:aws:s3:::kl-cafe-explorer-2025/kooky-plate.jpg",
        "arn:aws:s3:::kl-cafe-explorer-2025/estelle-bagel.jpg",
        "arn:aws:s3:::kl-cafe-explorer-2025/chi-chi.jpg"
      ]
```

6. Embed the S3 image URLs in the DynamoDB database.

C. Network configuration using Amazon VPC

1. Create a new custom VPC to provide a private, isolated network space.
2. Set the name tag “KL-Coffee-VPC" and IPv4 CIDR block 10.0.0.0/16



ap-southeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-southeast-2#VpcDetails:VpcId=vpc-067a1cb0442a12636

VPC > Your VPCs > vpc-067a1cb0442a12636

vpc-067a1cb0442a12636	Available	Off	Enabled
DNS resolution: Enabled	Tenancy: default	DHCP option set: dopt-052df1b2c2c1ed9e9	Main route table: rtb-0a32876993e162564
Main network ACL: acl-0975d602b7ba7beca	Default VPC: No	IPv4 CIDR: 10.0.0.0/16	IPv6 pool: –
IPv6 CIDR (Network border group): –	Network Address Usage metrics: Disabled	Route 53 Resolver DNS Firewall rule groups: Failed to load rule groups	Owner ID: 398019064969
Encryption control ID: –	Encryption control mode: –		

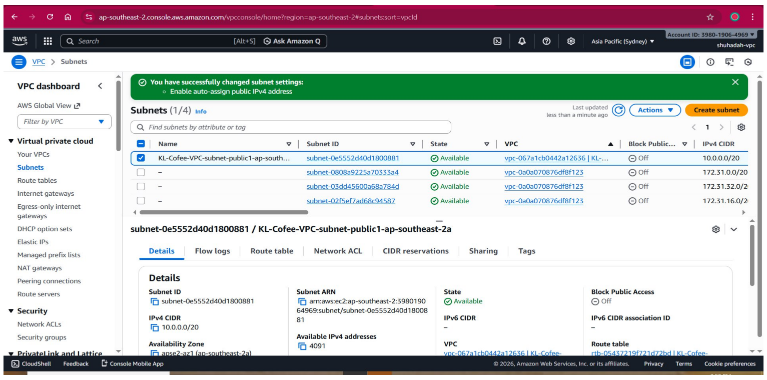
Resource map

VPC: Your AWS virtual network — KL-Cofee-VPC-vpc

Subnets (1): Subnets within this VPC — ap-southeast-2a — KL-Cofee-VPC-subnet-public1-ap-southeast-2a

Route tables (2): Route network traffic to resources — KL-Cofee-VPC-rtb-public — rtb-0a32876993e162564

3. Edit subnet settings.



ap-southeast-2.console.aws.amazon.com/vpcconsole/home?region=ap-southeast-2#subnets:sort=vpcId

VPC > Subnets

You have successfully changed subnet settings:
- Enable auto-assign public IPv4 address

Subnets (1/4)

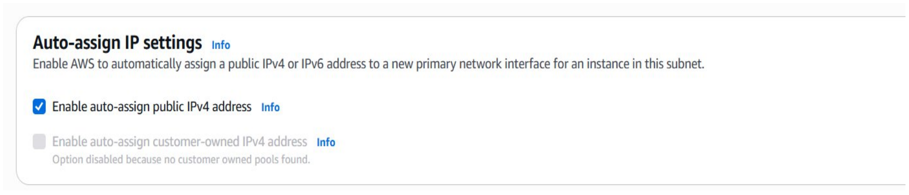
Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
KL-Cofee-VPC-subnet-public1-ap-south...	subnet-0e5552d40d1800881	Available	vpc-067a1cb0442a12636 | KL-...	Off	10.0.0.0/20
–	subnet-0808a9225a70333a4	Available	vpc-0a0a070876df8f123	Off	172.31.0.0/20
–	subnet-03dd45600a68a784d	Available	vpc-0a0a070876df8f123	Off	172.31.32.0/2
–	subnet-02f5ef7ad68c94587	Available	vpc-0a0a070876df8f123	Off	172.31.16.0/2

subnet-0e5552d40d1800881 / KL-Cofee-VPC-subnet-public1-ap-southeast-2a

Details

Subnet ID: subnet-0e5552d40d1800881	Subnet ARN: arn:aws:ec2:ap-southeast-2:398019064969:subnet/subnet-0e5552d40d1800881	State: Available	Block Public Access: Off
IPv4 CIDR: 10.0.0.0/20	Available IPv4 addresses: 4091	IPv6 CIDR: –	IPv6 CIDR association ID: –
Availability Zone: apse2-az1 (ap-southeast-2a)		VPC: vpc-067a1cb0442a12636 | KL-Cofee-	Route table: rtb-05437219f721d72bd | KL-Cofee-

4. Enable auto-assign public IPv4 address and save the new setting to ensure any EC2 instance launched in this subnet is reachable via the internet.



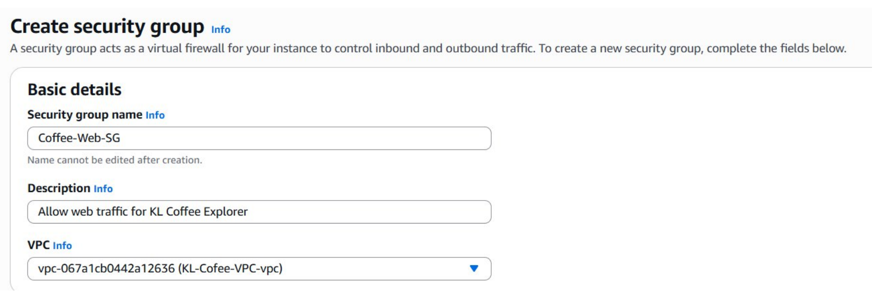
Auto-assign IP settings Info

Enable AWS to automatically assign a public IPv4 or IPv6 address to a new primary network interface for an instance in this subnet.

☑ Enable auto-assign public IPv4 address Info

☐ Enable auto-assign customer-owned IPv4 address Info
Option disabled because no customer owned pools found.

5. Create a new security group that name “coffee-web-sg" and select new vpc that just created.



Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below.

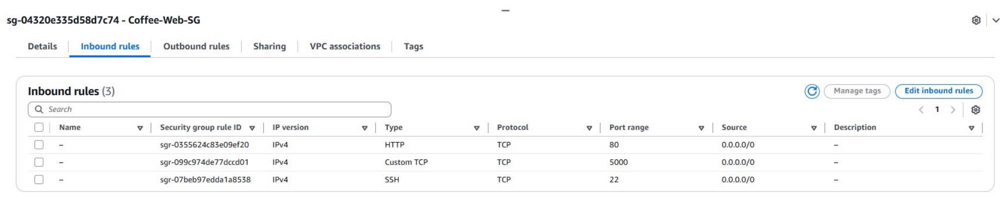
Basic details

Security group name Info: Coffee-Web-SG
Name cannot be edited after creation.

Description Info: Allow web traffic for KL Coffee Explorer

VPC Info: vpc-067a1cb0442a12636 (KL-Cofee-VPC-vpc)

6. Add rule HTTP, SSH and custom TCP to allows anyone to see the web and team to log in to server.

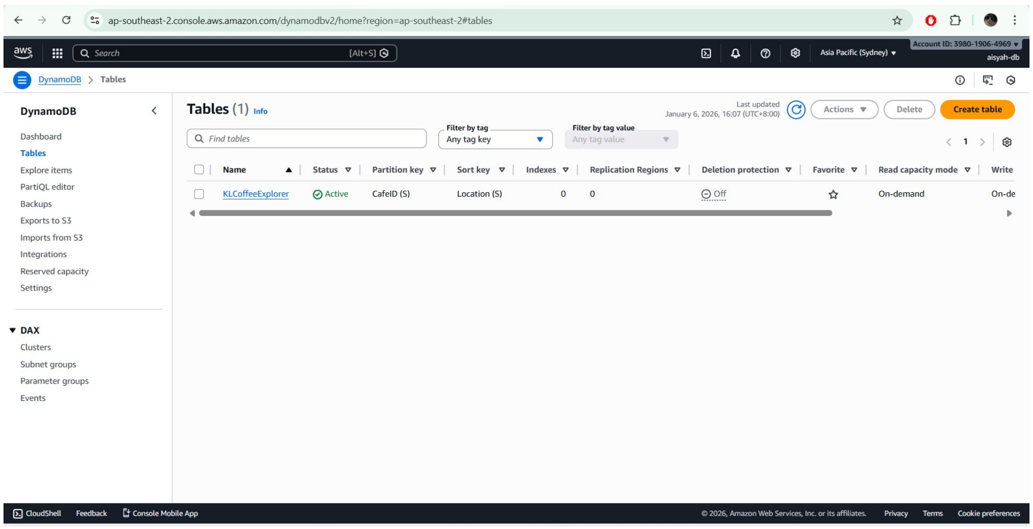

sg-04320e335d58d7c74 - Coffee-Web-SG

Inbound rules (3)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
–	sgr-0355624c83e09ef20	IPv4	HTTP	TCP	80	0.0.0.0/0	–
–	sgr-099c974de77dccd01	IPv4	Custom TCP	TCP	5000	0.0.0.0/0	–
–	sgr-07beb97edda1a8538	IPv4	SSH	TCP	22	0.0.0.0/0	–

D. Database using Amazon DynamoDB

1. Table Configuration: *KLCoffeeExplorer*
   - Partitione key (Primary Key): CafeID (string) - chosen to ensure each cafe has a unique identifier for fast lookups.

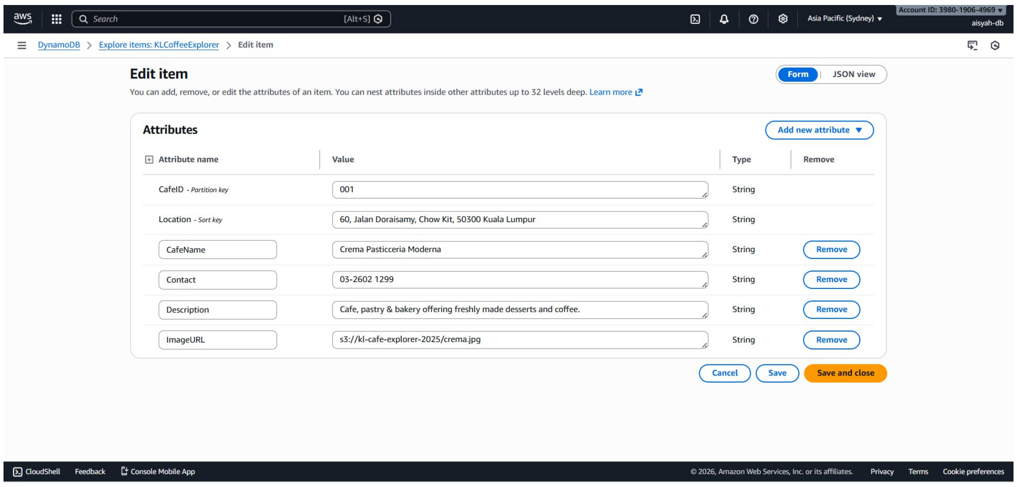

ap-southeast-2.console.aws.amazon.com/dynamodbv2/home?region=ap-southeast-2#tables

Tables (1)

Last updated January 6, 2026, 16:07 (UTC+8:00)

Name	Status	Partition key	Sort key	Indexes	Replication Regions	Deletion protection	Favorite	Read capacity mode	Write
KLCoffeeExplorer	Active	CafeID (S)	Location (S)	0	0	Off	☆	On-demand	On-de

2. Since DynamoDB is schema-less, we added custom attributes for each cafe to match the website's front-end requirements:
   - CafeName: The display title
   - Description: A short summary of the cafe.
   - Location: The physical location
   - Contact : The telephone number
   - ImageURL : A direct link to the logo stored in out S3 bucket.
3. Implementation method: Used the Form method to manually input cafe details into the database using the AWS Management Console's visual interface.



DynamoDB > Explore items: KLCoffeeExplorer > Edit item

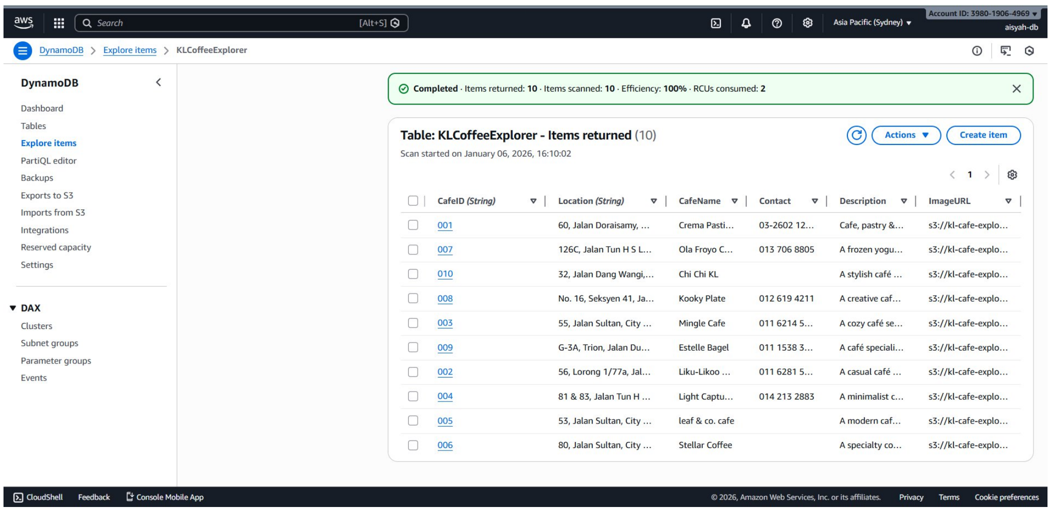
Edit item

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more

Attributes

Attribute name	Value	Type
CafeID - *Partition key*	001	String
Location - *Sort key*	60, Jalan Doraisamy, Chow Kit, 50300 Kuala Lumpur	String
CafeName	Crema Pasticceria Moderna	String
Contact	03-2602 1299	String
Description	Cafe, pastry & bakery offering freshly made desserts and coffee.	String
ImageURL	s3://kl-cafe-explorer-2025/crema.jpg	String

4. Data Verification : Before saving, ensure that each attribute name matched the frontend code requirements exactly (case-sensitive).



DynamoDB > Explore items > KLCoffeeExplorer

Completed · Items returned: **10** · Items scanned: **10** · Efficiency: **100%** · RCUs consumed: **2**

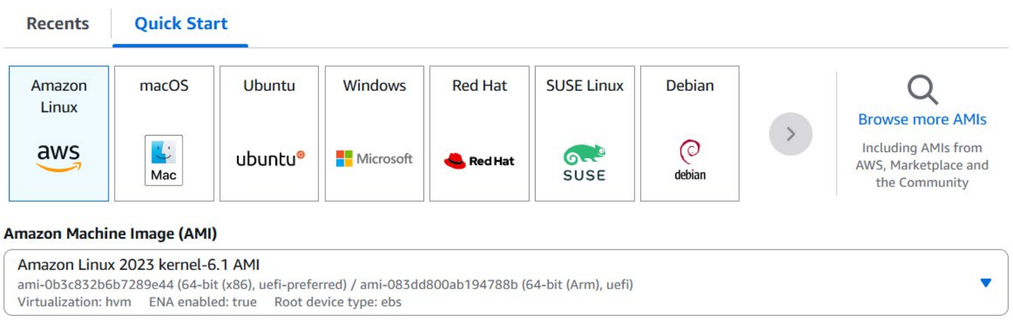
Table: KLCoffeeExplorer - Items returned (10)

Scan started on January 06, 2026, 16:10:02

CafeID *(String)*	Location *(String)*	CafeName	Contact	Description	ImageURL
001	60, Jalan Doraisamy, ...	Crema Pasti...	03-2602 12...	Cafe, pastry &...	s3://kl-cafe-explo...
007	126C, Jalan Tun H S L...	Ola Froyo C...	013 706 8805	A frozen yogu...	s3://kl-cafe-explo...
010	32, Jalan Dang Wangi,...	Chi Chi KL		A stylish café ...	s3://kl-cafe-explo...
008	No. 16, Seksyen 41, Ja...	Kooky Plate	012 619 4211	A creative caf...	s3://kl-cafe-explo...
003	55, Jalan Sultan, City ...	Mingle Cafe	011 6214 5...	A cozy café se...	s3://kl-cafe-explo...
009	G-3A, Trion, Jalan Du...	Estelle Bagel	011 1538 3...	A café speciali...	s3://kl-cafe-explo...
002	56, Lorong 1/77a, Jal...	Liku-Likoo ...	011 6281 5...	A casual café ...	s3://kl-cafe-explo...
004	81 & 83, Jalan Tun H ...	Light Captu...	014 213 2883	A minimalist c...	s3://kl-cafe-explo...
005	53, Jalan Sultan, City ...	leaf & co. cafe		A modern caf...	s3://kl-cafe-explo...
006	80, Jalan Sultan, City ...	Stellar Coffee		A specialty co...	s3://kl-cafe-explo...

E. Amazon EC2 (Website Hosting)

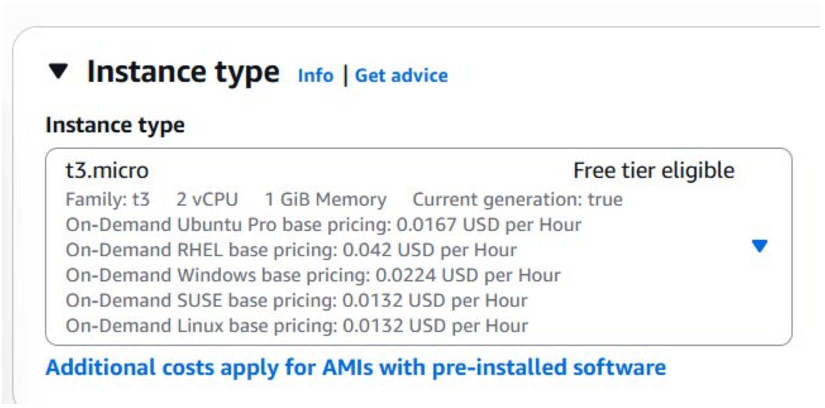
1. Navigate to the EC2 service and launch a new instance.
2. Select Amazon Linux as the operating system.



Amazon Machine Image (AMI)

Amazon Linux 2023 kernel-6.1 AMI
ami-0b3c832b6b7289e44 (64-bit (x86), uefi-preferred) / ami-083dd800ab194788b (64-bit (Arm), uefi)
Virtualization: hvm   ENA enabled: true   Root device type: ebs

3. Choose t3.micro instance type.

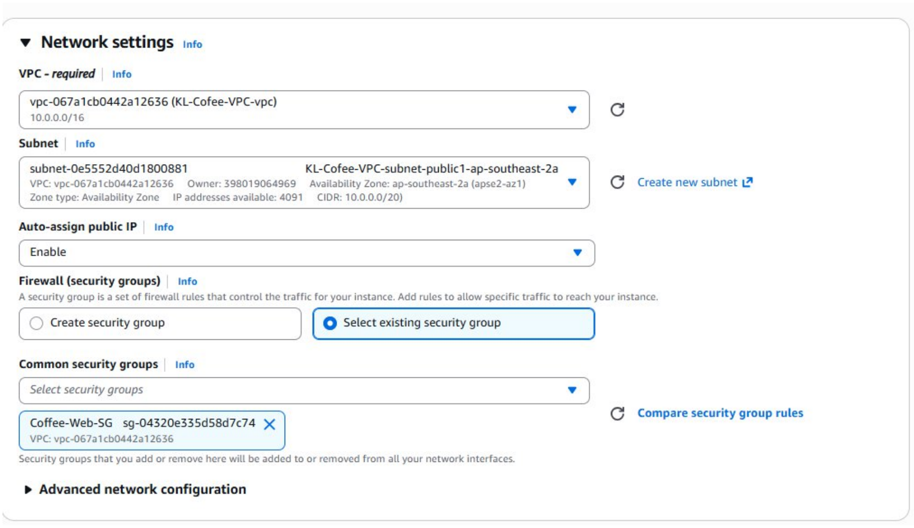


Instance type

t3.micro   Free tier eligible
Family: t3   2 vCPU   1 GiB Memory   Current generation: true
On-Demand Ubuntu Pro base pricing: 0.0167 USD per Hour
On-Demand RHEL base pricing: 0.042 USD per Hour
On-Demand Windows base pricing: 0.0224 USD per Hour
On-Demand SUSE base pricing: 0.0132 USD per Hour
On-Demand Linux base pricing: 0.0132 USD per Hour

Additional costs apply for AMIs with pre-installed software

4. Create a key pair for SSH access.
5. For the network settings, select the created custom VPC and security group.



VPC - *required*: vpc-067a1cb0442a12636 (KL-Cofee-VPC-vpc) 10.0.0.0/16

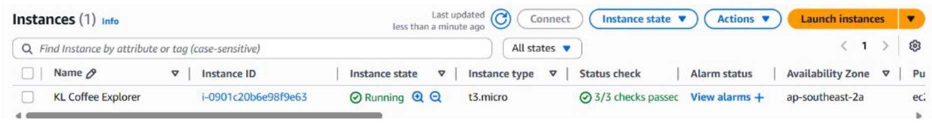
Subnet: subnet-0e5552d40d1800881   KL-Cofee-VPC-subnet-public1-ap-southeast-2a
VPC: vpc-067a1cb0442a12636   Owner: 398019064969   Availability Zone: ap-southeast-2a (apse2-az1)
Zone type: Availability Zone   IP addresses available: 4091   CIDR: 10.0.0.0/20)

Auto-assign public IP: Enable

Firewall (security groups)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
Create security group [✓ Select existing security group]

Common security groups: *Select security groups*
Coffee-Web-SG   sg-04320e335d58d7c74
VPC: vpc-067a1cb0442a12636
Security groups that you add or remove here will be added to or removed from all your network interfaces.

6. Launch the EC2 instance and wait until it is running.

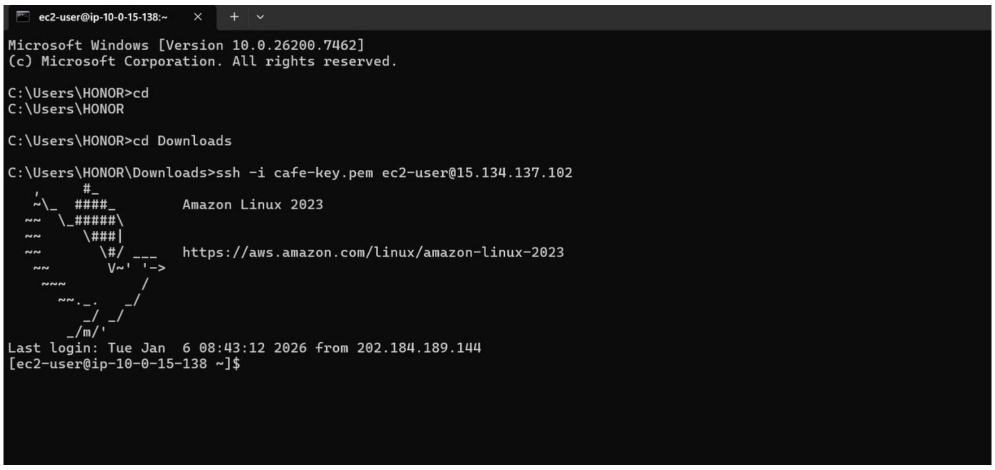


Instances (1) Info

Last updated less than a minute ago
Connect
Instance state
Actions
Launch instances
Find Instance by attribute or tag (case-sensitive)
All states

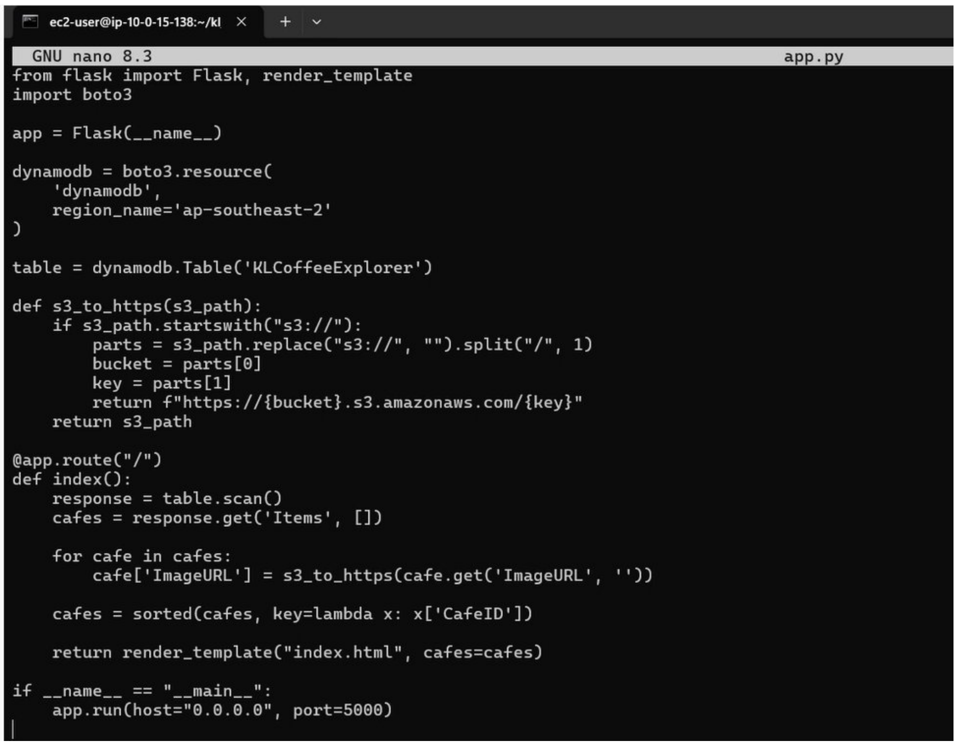
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Pu
KL Coffee Explorer	i-0901c20b6e98f9e63	Running	t3.micro	3/3 checks passec	View alarms +	ap-southeast-2a	ec:

7. Connect to the EC2 instance using SSH using the created keypair.



```
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HONOR>cd
C:\Users\HONOR

C:\Users\HONOR>cd Downloads

C:\Users\HONOR\Downloads>ssh -i cafe-key.pem ec2-user@15.134.137.102
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Jan  6 08:43:12 2026 from 202.184.189.144
[ec2-user@ip-10-0-15-138 ~]$
```

8. Install required libraries which is Flask and Boto3, for website hosting and data fetching.
9. Update the app.py file. This file communicates with DynamoDB, sends data to the website and connect AWS services together.



```
  GNU nano 8.3                                                        app.py
from flask import Flask, render_template
import boto3

app = Flask(__name__)

dynamodb = boto3.resource(
    'dynamodb',
    region_name='ap-southeast-2'
)

table = dynamodb.Table('KLCoffeeExplorer')

def s3_to_https(s3_path):
    if s3_path.startswith("s3://"):
        parts = s3_path.replace("s3://", "").split("/", 1)
        bucket = parts[0]
        key = parts[1]
        return f"https://{bucket}.s3.amazonaws.com/{key}"
    return s3_path

@app.route("/")
def index():
    response = table.scan()
    cafes = response.get('Items', [])

    for cafe in cafes:
        cafe['ImageURL'] = s3_to_https(cafe.get('ImageURL', ''))

    cafes = sorted(cafes, key=lambda x: x['CafeID'])

    return render_template("index.html", cafes=cafes)

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)
```

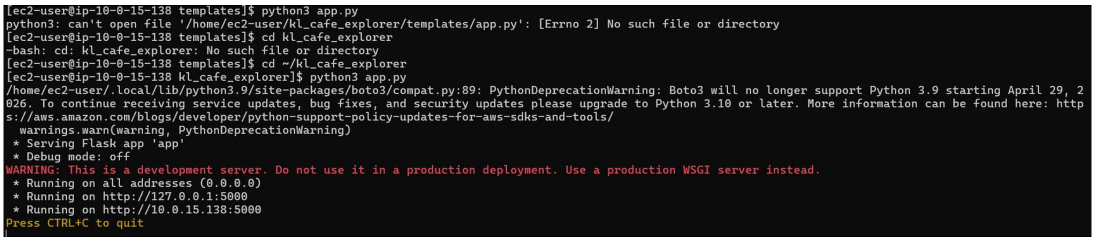
10. Upload the index.html file to host the cafe directory website.

    Full index.html file:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>KL Cafe Explorer</title>

  <style>
    body {
      margin: 0;
      font-family: 'Segoe UI', Arial, sans-serif;
      background-color: #fff1f4;
      color: #5a3a42;
    }

    header {
      background-color: #f7c6d0;
      padding: 30px;
      text-align: center;
      border-bottom: 3px solid #f1aebd;
    }

    header h1 {
      margin: 0;
      font-size: 36px;
      color: #7a2e3a;
    }

    header p {
      margin-top: 8px;
      font-size: 15px;
      color: #8a4b55;
    }

    .container {
      max-width: 1100px;
      margin: 40px auto;
      padding: 0 20px;
      display: grid;
      grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
      gap: 25px;
    }

    .cafe-card {
      background-color: white;
      border-radius: 16px;
      padding: 18px;
      box-shadow: 0 8px 20px rgba(247,198,208,0.5);
      transition: transform 0.2s;
    }

    .cafe-card:hover {
      transform: translateY(-6px);
    }

    .cafe-card img {
      width: 100%;
      height: 180px;
      object-fit: cover;
      border-radius: 12px;
      margin-bottom: 12px;
    }

    .cafe-card h2 {
      margin: 6px 0 10px;
      color: #c04b67;
      font-size: 22px;
    }

    .cafe-card p {
      margin: 4px 0;
      font-size: 14px;
      line-height: 1.6;
    }

    .label {
      font-weight: bold;
      color: #a33a52;
    }

    footer {
      text-align: center;
      padding: 20px;
      color: #9b6b74;
      font-size: 13px;
    }
  </style>
</head>

<body>

<header>
  <h1>☕ KL Cafe Explorer</h1>
  <p>Discover lovely cafes around Kuala Lumpur</p>
</header>

<div class="container">
  {% for cafe in cafes %}
```

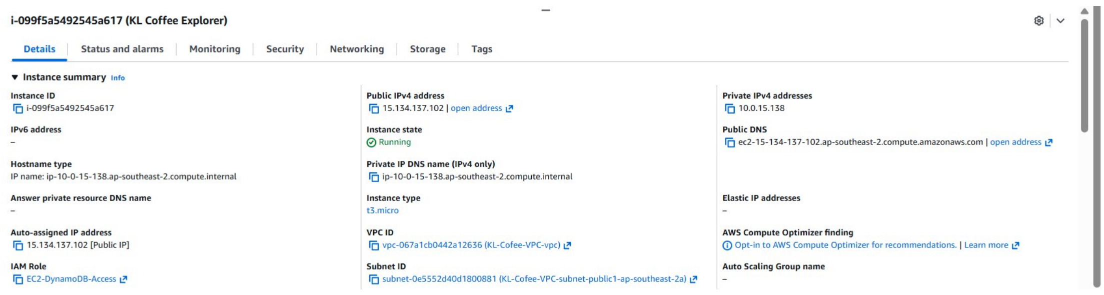
```
                <div class="cafe-card">
                  <img src="{{ cafe.ImageURL }}" alt="{{ cafe.CafeName }}">
                  <h2>{{ cafe.CafeName }}</h2>
                  <p><span class="label">Description:</span> {{ cafe.Description }}</p>
                  <p><span class="label">Location:</span> {{ cafe.Location }}</p>
                  {% if cafe.Contact %}
                    <p><span class="label">Contact:</span> {{ cafe.Contact }}</p>
                  {% endif %}
                </div>
                {% endfor %}
              </div>

              <footer>
                © 2025 KL Cafe Explorer · Powered by AWS
              </footer>

              </body>
</html>
```

11. Run the app.py file on SSH.



```
[ec2-user@ip-10-0-15-138 templates]$ python3 app.py
python3: can't open file '/home/ec2-user/kl_cafe_explorer/templates/app.py': [Errno 2] No such file or directory
[ec2-user@ip-10-0-15-138 templates]$ cd kl_cafe_explorer
-bash: cd: kl_cafe_explorer: No such file or directory
[ec2-user@ip-10-0-15-138 templates]$ cd ~/kl_cafe_explorer
[ec2-user@ip-10-0-15-138 kl_cafe_explorer]$ python3 app.py
/home/ec2-user/.local/lib/python3.9/site-packages/boto3/compat.py:89: PythonDeprecationWarning: Boto3 will no longer support Python 3.9 starting April 29, 2
026. To continue receiving service updates, bug fixes, and security updates please upgrade to Python 3.10 or later. More information can be found here: http
s://aws.amazon.com/blogs/developer/python-support-policy-updates-for-aws-sdks-and-tools/
  warnings.warn(warning, PythonDeprecationWarning)
 * Serving Flask app 'app'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5000
 * Running on http://10.0.15.138:5000
Press CTRL+C to quit
```

12. Access the website using the EC2 public IP address. Ensure that the created IAM Role is selected to get the access to DynamoDB.



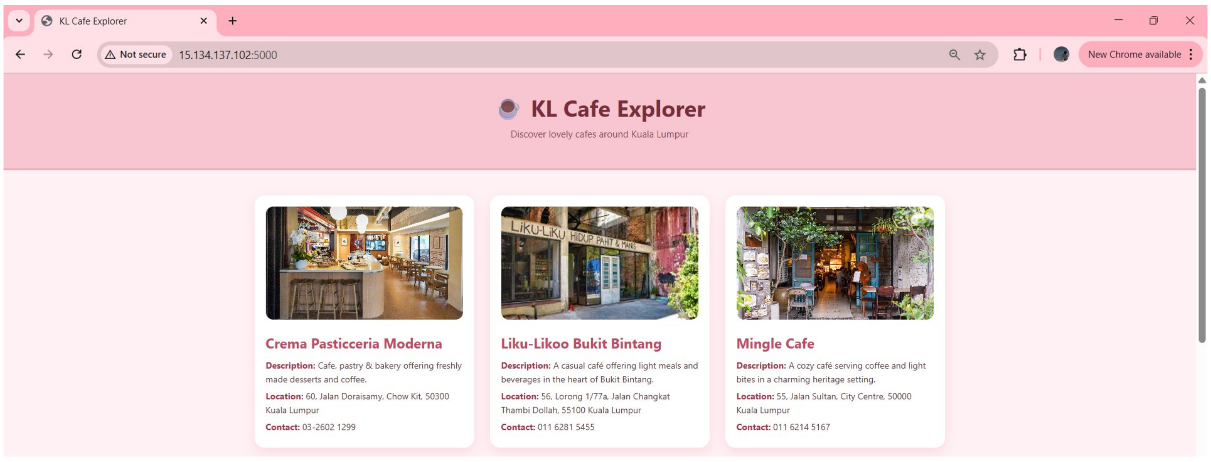
i-099f5a5492545a617 (KL Coffee Explorer)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

▼ Instance summary Info

Instance ID i-099f5a5492545a617	**Public IPv4 address** 15.134.137.102 | open address	**Private IPv4 addresses** 10.0.15.138
IPv6 address –	**Instance state** Running	**Public DNS** ec2-15-134-137-102.ap-southeast-2.compute.amazonaws.com | open address
Hostname type IP name: ip-10-0-15-138.ap-southeast-2.compute.internal	**Private IP DNS name (IPv4 only)** ip-10-0-15-138.ap-southeast-2.compute.internal	
Answer private resource DNS name –	**Instance type** t3.micro	**Elastic IP addresses** –
Auto-assigned IP address 15.134.137.102 [Public IP]	**VPC ID** vpc-067a1cb0442a12636 (KL-Cofee-VPC-vpc)	**AWS Compute Optimizer finding** Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role EC2-DynamoDB-Access	**Subnet ID** subnet-0e5552d40d1800881 (KL-Cofee-VPC-subnet-public1-ap-southeast-2a)	**Auto Scaling Group name** –

WEBSITE LAYOUT



KL Cafe Explorer

Discover lovely cafes around Kuala Lumpur

Crema Pasticceria Moderna

Description: Cafe, pastry & bakery offering freshly made desserts and coffee.

Location: 60, Jalan Doraisamy, Chow Kit, 50300 Kuala Lumpur

Contact: 03-2602 1299

Liku-Likoo Bukit Bintang

Description: A casual café offering light meals and beverages in the heart of Bukit Bintang.

Location: 56, Lorong 1/77a, Jalan Changkat Thambi Dollah, 55100 Kuala Lumpur

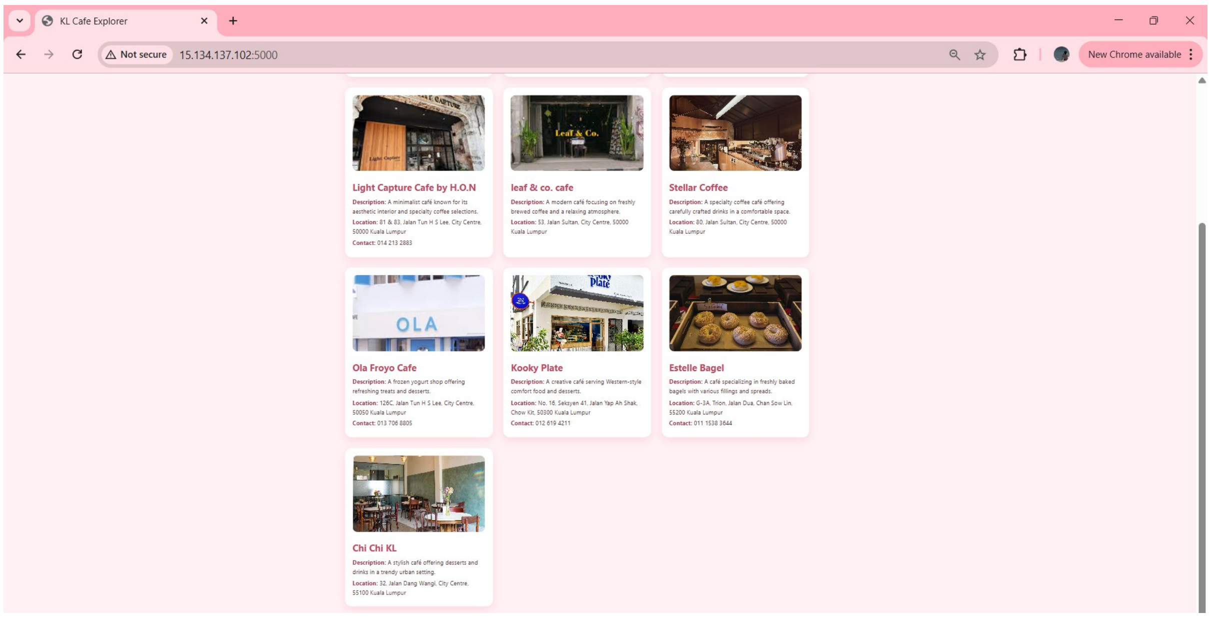
Contact: 011 6281 5455

Mingle Cafe

Description: A cozy café serving coffee and light bites in a charming heritage setting.

Location: 55, Jalan Sultan, City Centre, 50000 Kuala Lumpur

Contact: 011 6214 5167



Light Capture Cafe by H.O.N

Description: A minimalist café known for its aesthetic interior and specialty coffee selections.

Location: 81 & 83, Jalan Tun H S Lee, City Centre, 50000 Kuala Lumpur

Contact: 014 213 2883

leaf & co. cafe

Description: A modern café focusing on freshly brewed coffee and a relaxing atmosphere.

Location: 53, Jalan Sultan, City Centre, 50000 Kuala Lumpur

Stellar Coffee

Description: A specialty coffee café offering carefully crafted drinks in a comfortable space.

Location: 80, Jalan Sultan, City Centre, 50000 Kuala Lumpur

Ola Froyo Cafe

Description: A frozen yogurt shop offering refreshing treats and desserts.

Location: 126C, Jalan Tun H S Lee, City Centre, 50050 Kuala Lumpur

Contact: 013 706 8805

Kooky Plate

Description: A creative café serving Western-style comfort food and desserts.

Location: No. 16, Seksyen 41, Jalan Yap Ah Shak, Chow Kit, 50300 Kuala Lumpur

Contact: 012 619 4211

Estelle Bagel

Description: A café specializing in freshly baked bagels with various fillings and spreads.

Location: G-3A, Trion, Jalan Dua, Chan Sow Lin, 55200 Kuala Lumpur

Contact: 011 1538 3644

Chi Chi KL

Description: A stylish café offering desserts and drinks in a trendy urban setting.

Location: 32, Jalan Dang Wangi, City Centre, 55100 Kuala Lumpur

VIDEO PRESENTATION LINK

https://www.youtube.com/watch?v=-Sipp-6VIgY

GITHUB REPOSITORY LINK

https://github.com/afiqahz0/AWS-Cloud-Based-Cafe-Directory

CONCLUSION

In conclusion, this project effectively used Amazon Web Services (AWS) to create a basic cloud-based website. Amazon EC2 for web hosting, Amazon S3 for image storage, Amazon DynamoDB for cafe data management, IAM for access control, and Amazon VPC for secure networking are all integrated on the KL Cafe Explorer website. Important cloud computing ideas like service integration, security, and scalability were put to use in this project. All things considered, the met its goal and showed how AWS services can be used to create a secure and useful online application.