

Home Made Pickles & Snacks: Taste the Best

Hardware Required:

Processor: Intel i5 or equivalent (minimum). RAM: 4 GB (8 GB recommended for Full Stack MERN). Storage: 128 GB SSD or 128 GB HDD. Internet Connectivity: High-speed internet (minimum 10 Mbps per system). Additional: Audio-visual setup for interactive sessions (microphone, speakers, etc.).

Software Required:

Processor: Intel i5 or equivalent (minimum). RAM: 4 GB (8 GB recommended for Full Stack MERN). Storage: 128 GB SSD or 128 GB HDD. Internet Connectivity: High-speed internet (minimum 10 Mbps per system). Additional: Audio-visual setup for interactive sessions (microphone, speakers, etc.).

System Required:

Projector and Audio System for presentations in all labs/classrooms Classrooms/Labs are equipped with systems or provisions for students to join sessions with their own laptops.

Description:

Home Made Pickles & Snacks — Taste the Best is a cloud-based culinary platform revolutionizing access to authentic, handcrafted pickles and snacks. Addressing the growing demand for preservative-free, traditional recipes, this initiative combines artisanal craftsmanship with cutting-edge technology to deliver farm-fresh flavors directly to consumers. Built on Flask for backend efficiency and hosted on AWS EC2 for scalable performance, the platform offers seamless browsing, ordering, and subscription management. DynamoDB ensures real-time inventory tracking and personalized user experiences, while fostering sustainability through partnerships with local farmers and eco-friendly packaging. From tangy regional pickles to wholesome snacks, every product celebrates heritage recipes, nutritional integrity, and convenience—proving that tradition and innovation can coexist deliciously. "Preserving

Traditions, One Jar at a Time." **Scenarios:**

Scenario 1: Scalable Order Management for High Demand

A cloud-based system ensures seamless order processing during peak user activity. For instance, during a promotional event, hundreds of users simultaneously access the platform to place orders. The backend efficiently processes requests, updates inventory in real-time, and manages user sessions. The cloud infrastructure handles traffic spikes without performance degradation, ensuring smooth transactions and minimizing wait times.

Scenario 2: Real-Time Inventory Tracking and Updates

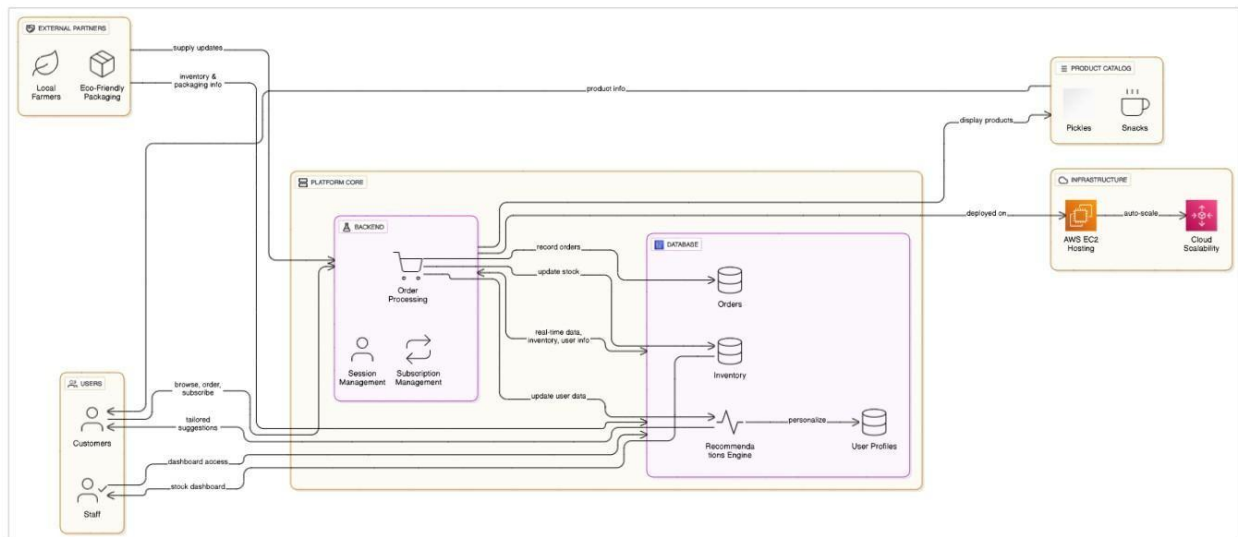
When a customer places an order for a product, the system instantly updates stock levels and records transaction details. For example, a user purchases an item, triggering automatic inventory deduction and order confirmation. Staff members receive updated dashboards to monitor stock availability and fulfillment progress, ensuring timely restocking and minimizing overselling risks.

Scenario 3: Personalized User Experience and Recommendations

The platform leverages user behavior data to enhance engagement. A returning customer, for instance, views tailored recommendations based on past purchases and browsing history. The system dynamically adjusts suggestions in real-time, while maintaining fast response rates even during high traffic, creating a frictionless and intuitive shopping experience.

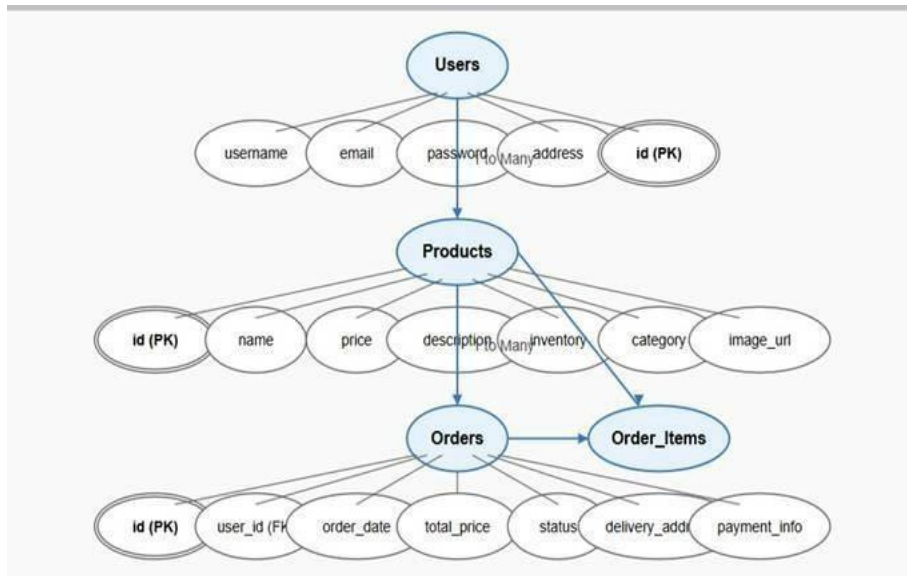
Architecture

This AWS-based architecture powers a scalable and secure web application using Amazon EC2 for hosting the backend, with a lightweight framework like Flask handling core logic. Application data is stored in Amazon DynamoDB, ensuring fast, reliable access, while user access is managed through AWS IAM for secure authentication and control. Real-time alerts and system notifications are enabled via Amazon SNS, enhancing communication and user engagement.



Entity Relationship (ER)Diagram

An ER (Entity-Relationship) diagram visually represents the logical structure of a database by defining entities, their attributes, and the relationships between them. It helps organize data efficiently by illustrating how different components of the system interact and relate. This structured approach supports effective database normalization, data integrity, and simplified query design.



Pre requirements:

AWS Account Setup: <https://docs.aws.amazon.com/accounts/latest/reference/getting-started.html> □

AWS IAM (Identity and Access

Management): <https://docs.aws.amazon.com/IAM/latest/UserGuide/introduction.html> □

AWS EC2 (Elastic Compute Cloud): <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html>

□

AWS DynamoDB: <https://docs.aws.amazon.com/amazondynamodb/Introduction.html> □

Git Documentation: <https://git-scm.com/doc>

VS Code Installation: (download the VS Code using the below link or you can get that in Microsoft store):

<https://code.visualstudio.com/download>

Project WorkFlow

Milestone 1. Backend Development and Application Setup

- Develop the Backend Using Flask.
- Integrate AWS Services Using boto3.

Milestone 2. AWS Account Setup and Login

- Set up an AWS account if not already done.

- Log in to the AWS Management Console

Milestone 3. DynamoDB Database Creation and Setup □ Create a DynamoDB Table.

- Configure Attributes for User Data and Book Requests.

Milestone 4. SNS Notification Setup

- Create SNS topics for book request notifications.
- Subscribe users and library staff to SNS email notifications.

Milestone 5. IAM Role Setup

- Create IAM Role
- Attach Policies

Milestone 6. EC2 Instance Setup

- Launch an EC2 instance to host the Flask application. • Configure security groups for HTTP, and SSH access.

Milestone 7. Deployment on EC2

- Upload Flask Files □ Run the Flask App

Milestone 8. Testing and Deployment

- Conduct functional testing to verify user signup, login, buy/sell stocks and notifications.

Milestone 1 : Web Application Development and Setup

Backend Development and Application Setup focuses on establishing the core structure of the application. This includes configuring the backend framework, setting up routing, and integrating database connectivity. It lays the groundwork for handling user interactions, data management, and secure access.

Important Instructions:

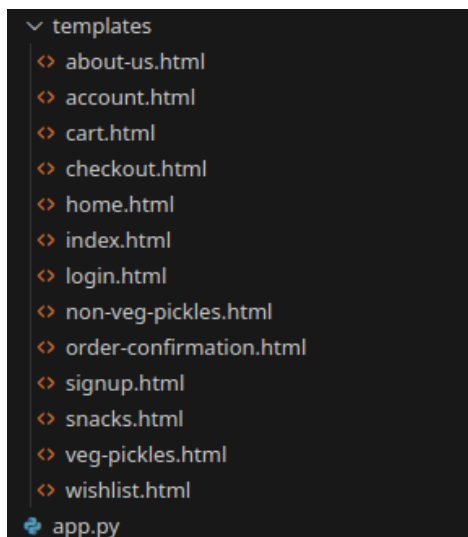
- Start by creating the necessary HTML pages and Flask routes (app.py) to build the core functionality of your application.
- During the initial development phase, store and retrieve data using Python dictionaries or lists locally. This will allow you to design, test, and validate your application logic without external database dependencies.

- Ensure your app runs smoothly with local data structures before integrating any cloud services.

Post Troven Access Activation:

- Once Troven Labs access is provided (valid for 3 hours), you must immediately proceed with Milestone 1 of your Guided Project instructions.
- At this point, modify your app.py and replace local dictionary/list operations with AWS services (such as DynamoDB, RDS, or others as per project requirements).
- Using the temporary credentials provided by Troven Labs, securely connect your application to AWS resources.
- Since the AWS configuration is lightweight and already instructed in the milestones, you should be able to complete the cloud integration efficiently within the allotted time.

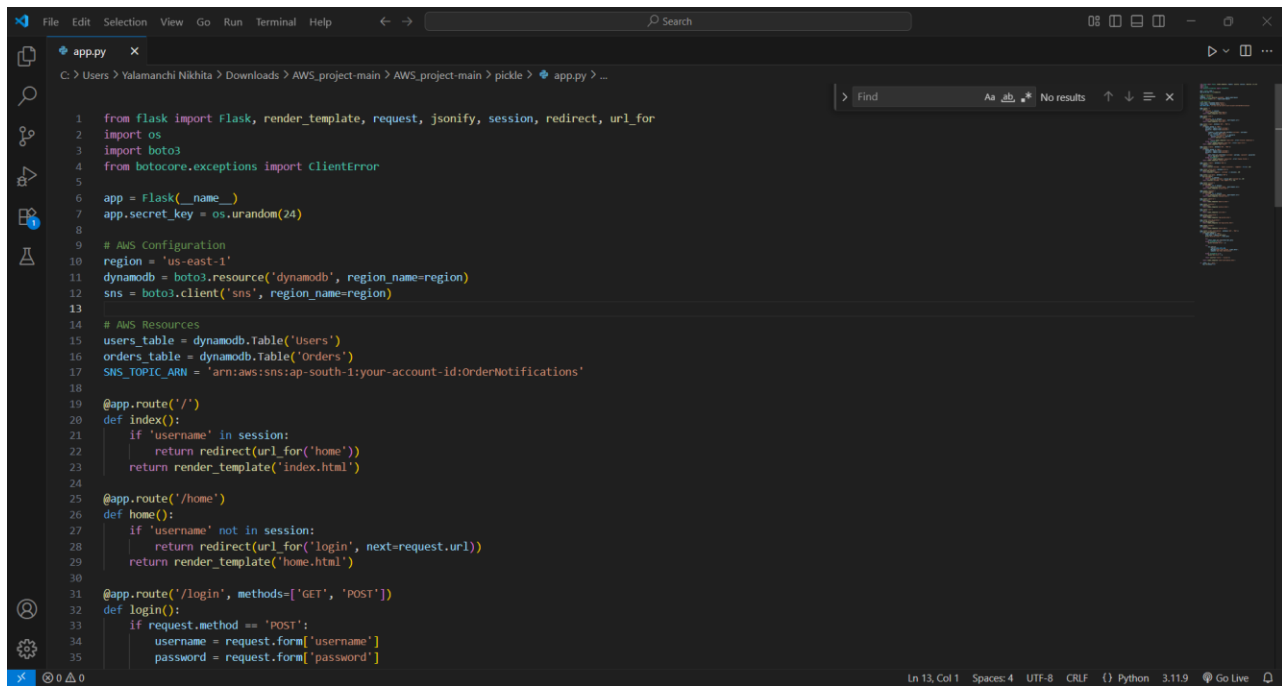
LOCAL DEPLOYMENT



File Explorer Structure

Description of the code :

? Flask App Initialization



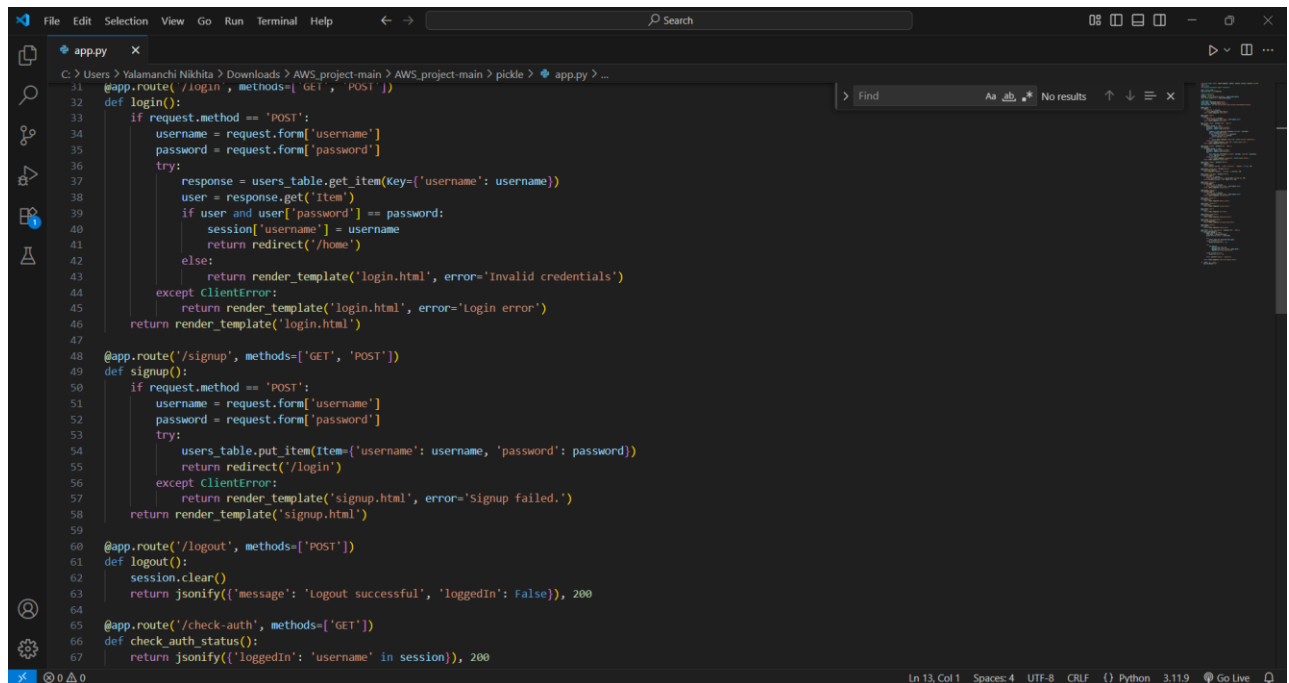
```
1 from flask import Flask, render_template, request, jsonify, session, redirect, url_for
2 import os
3 import boto3
4 from botocore.exceptions import ClientError
5
6 app = Flask(__name__)
7 app.secret_key = os.urandom(24)
8
9 # AWS Configuration
10 region = 'us-east-1'
11 dynamodb = boto3.resource('dynamodb', region_name=region)
12 sns = boto3.client('sns', region_name=region)
13
14 # AWS Resources
15 users_table = dynamodb.Table('Users')
16 orders_table = dynamodb.Table('Orders')
17 SNS_TOPIC_ARN = 'arn:aws:sns:ap-south-1:your-account-id:OrderNotifications'
18
19 @app.route('/')
20 def index():
21     if 'username' in session:
22         return redirect(url_for('home'))
23     return render_template("index.html")
24
25 @app.route('/home')
26 def home():
27     if 'username' not in session:
28         return redirect(url_for('login', next=request.url))
29     return render_template("home.html")
30
31 @app.route('/login', methods=['GET', 'POST'])
32 def login():
33     if request.method == 'POST':
34         username = request.form['username']
35         password = request.form['password']
```

- Use boto3 to connect to DynamoDB for handling user registration, Order details database operations and also mention region_name where Dynamodb tables are created.



```
# DynamoDB tables
dynamodb = boto3.resource('dynamodb', region_name=region)
orders_table = dynamodb.Table('orders')
users_table = dynamodb.Table('users')
contacts_table = dynamodb.Table('contacts')
reviews_table = dynamodb.Table('reviews')
```

- Routes for Web Pages
- Login Route (GET/POST): Verifies user credentials, increments login count, and redirects to the dashboard on success.



```
11 @app.route('/login', methods=['GET', 'POST'])
12 def login():
13     if request.method == 'POST':
14         username = request.form['username']
15         password = request.form['password']
16         try:
17             response = users_table.get_item(Key={'username': username})
18             user = response.get('Item')
19             if user and user['password'] == password:
20                 session['username'] = username
21                 return redirect('/home')
22             else:
23                 return render_template('login.html', error='Invalid credentials')
24         except ClientError:
25             return render_template('login.html', error='Login error')
26     return render_template('login.html')
27
28 @app.route('/signup', methods=['GET', 'POST'])
29 def signup():
30     if request.method == 'POST':
31         username = request.form['username']
32         password = request.form['password']
33         try:
34             users_table.put_item(Item={'username': username, 'password': password})
35             return redirect('/login')
36         except ClientError:
37             return render_template('signup.html', error='Signup failed.')
38     return render_template('signup.html')
39
40 @app.route('/logout', methods=['POST'])
41 def logout():
42     session.clear()
43     return jsonify({'message': 'Logout successful', 'loggedIn': False}), 200
44
45 @app.route('/check-auth', methods=['GET'])
46 def check_auth_status():
47     return jsonify({'loggedIn': 'username' in session}), 200
```

- SignUp route: Collecting registration data, hashes the password, and stores user details in the database.
- Logout route: The user can Logout so that the user can get back to the Login Page

```
@app.route('/logout', methods=['POST'])
def logout():
    session.clear()
    return jsonify({'message': 'Logout successful', 'loggedIn': False}), 200
```

- Check out Route:

```
@app.route('/checkout')
def checkout_page():
    if 'username' not in session:
        return redirect(url_for('login', next=request.url))
    return render_template('checkout.html')
```

- Home Route: Home page contains the routing for different categories which are Veg_pickles,Non_Veg_pickles,Snacks.

```
@app.route('/home')
def home():
    if 'username' not in session:
        return redirect(url_for('login', next=request.url))
    return render_template('home.html')
```

Milestone 2 : AWS Account Setup

Important Notice: Use Troven Labs for AWS Access

Students are strictly advised not to create their own AWS accounts, as doing so may incur charges. Instead, we have set up a dedicated section called “Labs” on the Troven platform, which provides temporary and cost-free access to AWS services.

Once your website is locally deployed and fully functional, you must proceed with integrating AWS services only through the Troven Labs environment. This ensures secure, controlled access to AWS resources without any risk of personal billing.

All steps involving AWS (such as deploying to EC2, connecting to DynamoDB, or using SNS) must be carried out within the Troven Labs platform, as we've configured temporary credentials for each student.

Reminder: You must complete the Web Development task before gaining access to Troven. Once accessed, the AWS Console via Troven is available for only 3 hours— please plan your work accordingly.

Please follow the provided guidelines and access AWS exclusively through Troven to avoid unnecessary issues.

Please refer the below link -

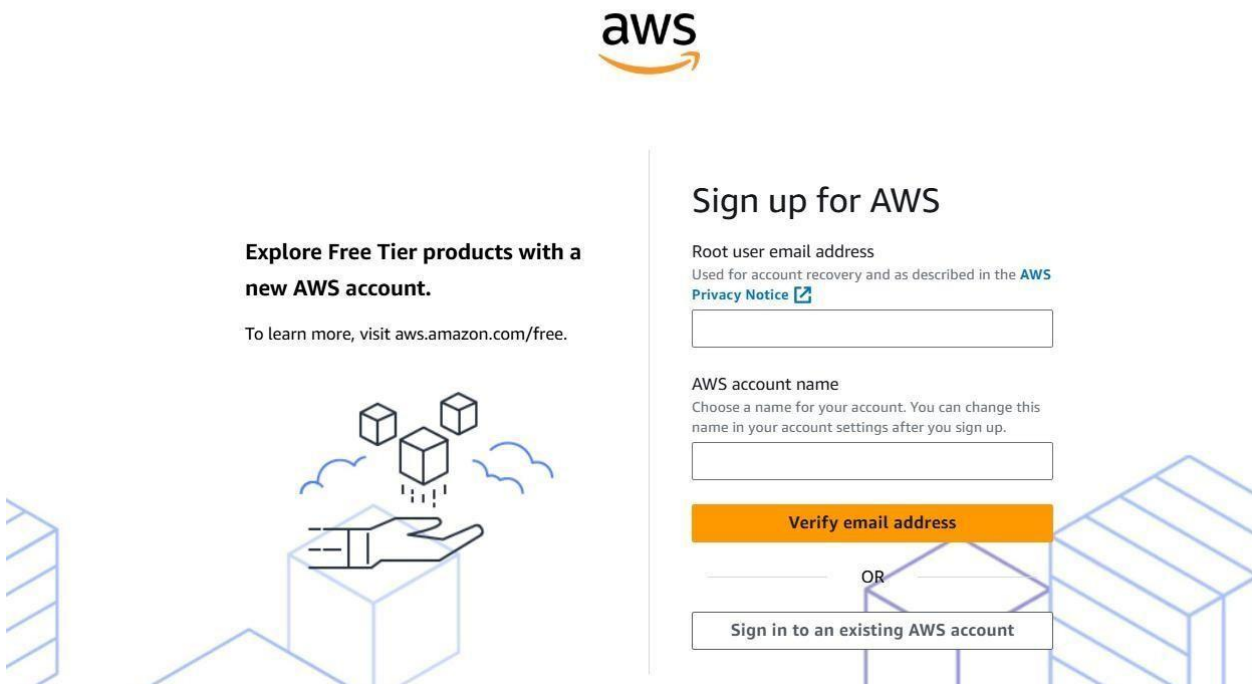
<https://drive.google.com/file/d/1HzWc7AMJ2BrxhVuaw5s0vWtcd28qgl/view?usp=sharing>

AWS Account Setup and Login

This is for your understanding only, please refrain from creating an AWS account. A temporary account will be provided via Troven.

- Go to the AWS website (<https://aws.amazon.com/>).
- Click on the "Create an AWS Account" button.
- Follow the prompts to enter your email address and choose a password.

- Provide the required account information, including your name, address, and phone number.
- Enter your payment information. (Note: While AWS offers a free tier, a credit card or debit card is required for verification.) Complete the identity verification process.
- Choose a support plan (the basic plan is free and sufficient for starting). Once verified, you can sign in to your new AWS accounts.



- Log in to the AWS Management Console
- After setting up your account, log in to the [AWS Management Console](#).

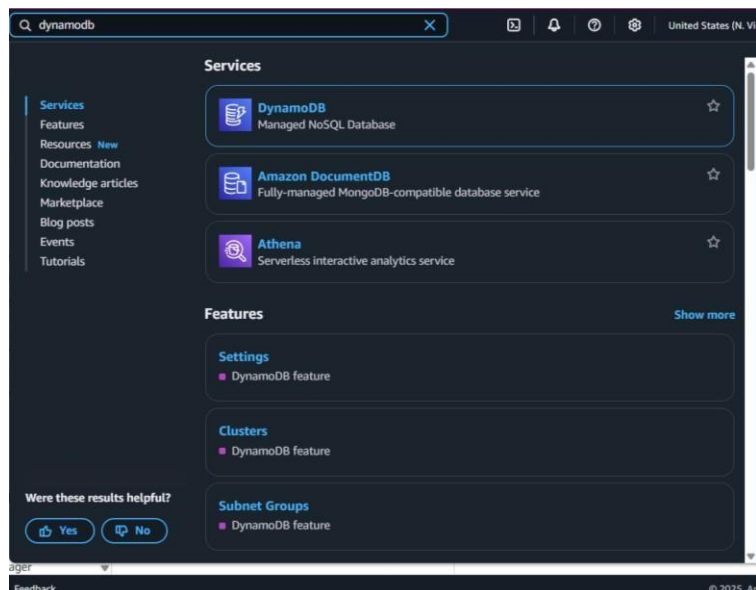


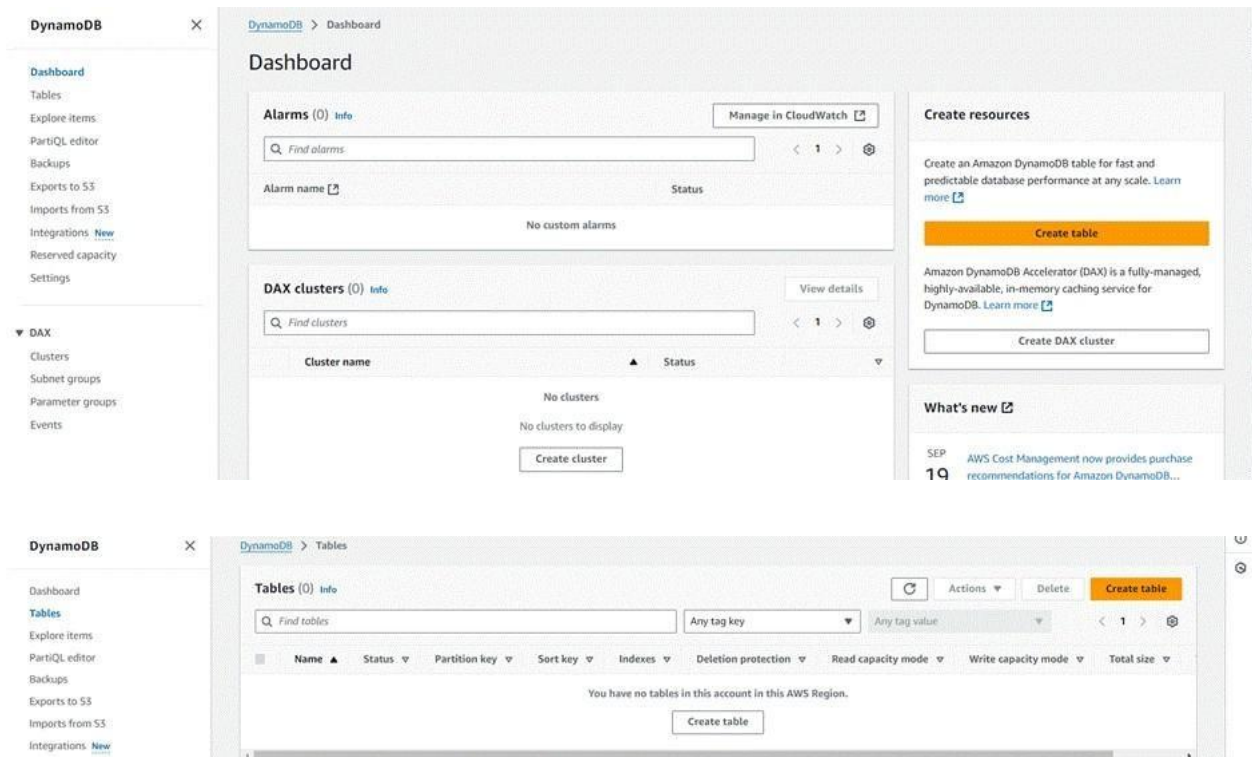
Milestone 3 : DynamoDB Database Creation and Setup

Database Creation and Setup involves initializing a cloud-based NoSQL database to store and manage application data efficiently. This step includes defining tables, setting primary keys, and configuring read/write capacities. It ensures scalable, highperformance data storage for seamless backend operations.

Navigate to the DynamoDB

- In the AWS Console, navigate to DynamoDB and click on create tables.





Create a DynamoDB table for storing data

- Create Users table with partition key "Username" with type String and click on create tables.



Search

[Alt+S]

≡ [DynamoDB](#) > [Tables](#) > Create table

Create table

Table details [Info](#)

DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

Table name

This will be used to identify your table.

Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).

Partition key

The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and

String



1 to 255 characters and case sensitive.

Sort key - optional

You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.

String



1 to 255 characters and case sensitive.

| | | |
|----------------------------|--------------------------|-----|
| Table class | DynamoDB Standard | Yes |
| Capacity mode | Provisioned | Yes |
| Provisioned read capacity | 5 RCU | Yes |
| Provisioned write capacity | 5 WCU | Yes |
| Auto scaling | On | Yes |
| Local secondary indexes | - | No |
| Global secondary indexes | - | Yes |
| Encryption key management | Owned by Amazon DynamoDB | Yes |
| Deletion protection | Off | Yes |
| Resource-based policy | Not active | Yes |

Tags

Tags are pairs of keys and optional values, that you can assign to AWS resources. You can use tags to control access to your resources or track your AWS spending.

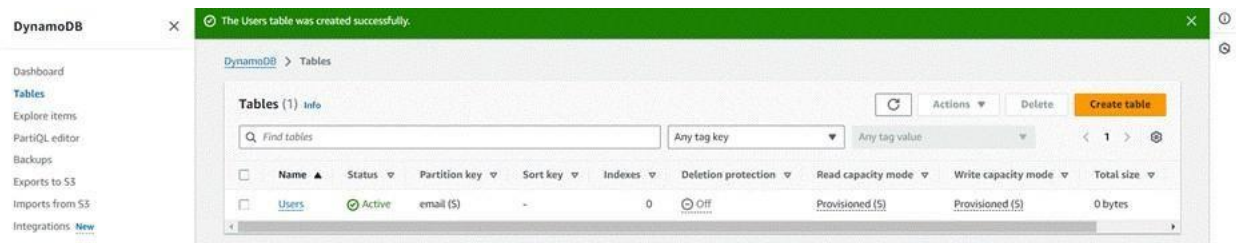
No tags are associated with the resource.

Add new tag

You can add 50 more tags.

Cancel

Create table



- Follow the same steps to create an Orders table with Order_id as the primary key to store Order details.



Search

[Alt+S]

DynamoDB > Tables > Create table

Create table

Table details [Info](#)

DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.

Table name

This will be used to identify your table.

Orders

Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).

Partition key

The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability.

Order_id

String

1 to 255 characters and case sensitive.

Sort key - optional

You can use a sort key as the second part of a table's primary key. The sort key allows you to sort or search among all items sharing the same partition key.

Enter the sort key name

String

1 to 255 characters and case sensitive.



| | | |
|----------------------------|--------------------------|-----|
| Table class | DynamoDB Standard | Yes |
| Capacity mode | Provisioned | Yes |
| Provisioned read capacity | 5 RCU | Yes |
| Provisioned write capacity | 5 WCU | Yes |
| Auto scaling | On | Yes |
| Local secondary indexes | - | No |
| Global secondary indexes | - | Yes |
| Encryption key management | Owned by Amazon DynamoDB | Yes |
| Deletion protection | Off | Yes |
| Resource-based policy | Not active | Yes |

Tags

Tags are pairs of keys and optional values, that you can assign to AWS resources. You can use tags to control access to your resources or track your AWS spending.

No tags are associated with the resource.

Add new tag

You can add 50 more tags.

Cancel

Create table

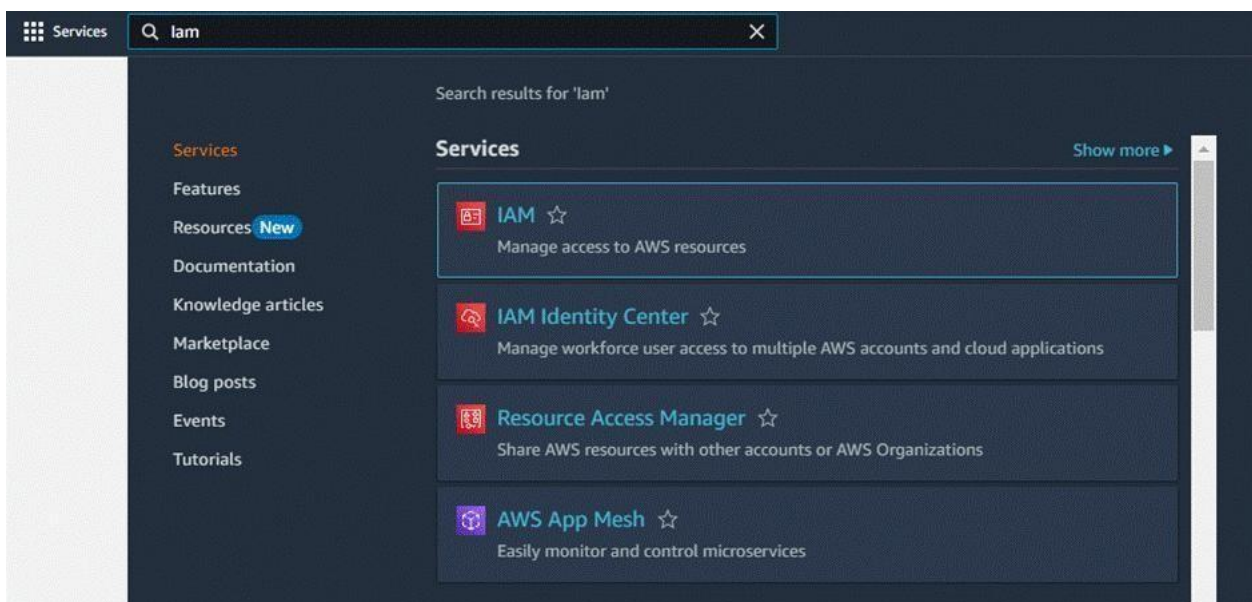
| Tables (2) Info | | | | | | | |  Actions |
|--|------------------------|----------|-----------------|--|-----------|--|---------------------|---|
| <input type="text" value="Find tables"/> | | | | <input type="text" value="Any tag key"/> | | <input type="text" value="Any tag value"/> | | |
| <input type="checkbox"/> | Name ▲ | Status ▼ | Partition key ▼ | Sort key ▼ | Indexes ▼ | Replication Regions ▼ | Deletion protection | |
| <input type="checkbox"/> | Orders | Active | order_id (S) | - | 0 | 0 | Off | |
| <input type="checkbox"/> | Users | Active | username (S) | - | 0 | 0 | Off | |

Milestone 4 : IAM Role Setup

IAM (Identity and Access Management) role setup involves creating roles that define specific permissions for AWS services. To set it up, you create a role with the required policies, assign it to users or services, and ensure the role has appropriate access to resources like EC2, S3, or RDS. This allows controlled access and ensures security best practices in managing AWS resources.

Create IAM Role.

- In the AWS Console, go to IAM and create a new IAM Role for EC2 to interact with DynamoDB.



Attach Policies

Attach the following policies to the role:

- AmazonDynamoDBFullAccess: Allows EC2 to perform read/write operations on DynamoDB.

The screenshot shows the 'Name, review, and create' page for a new IAM role. The role name is 'sns_dynamodb_role'. The description is 'Allows EC2 instances to call AWS services on your behalf'. The trust policy is displayed as JSON code. The permissions policy summary shows two attached policies: 'AmazonDynamoDBFullAccess' and 'AmazonSNSFullAccess'. The 'Add tags' section is empty.

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": {
7         "Service": "ec2.amazonaws.com"
8       },
9       "Action": "sts:AssumeRole"
10    }
11  ]
12 }
```

| Policy name | Type | Attached as |
|--------------------------|-------------|--------------------|
| AmazonDynamoDBFullAccess | AWS managed | Permissions policy |
| AmazonSNSFullAccess | AWS managed | Permissions policy |

The screenshot shows the 'sns_dynamodb_role' page in the AWS IAM console. The summary section displays the creation date (October 13, 2024, 23:06 UTC+05:30), last activity (6 days ago), ARN (arn:aws:iam::557690616836:role/sns_dynamodb_role), and instance profile ARN (arn:aws:iam::557690616836:instance-profile/sns_dynamodb_role). The permissions section shows two attached policies: 'AmazonDynamoDBFullAccess' and 'AmazonSNSFullAccess'.

| Policy name | Type | Attached entities |
|--------------------------|-------------|-------------------|
| AmazonDynamoDBFullAccess | AWS managed | 4 |
| AmazonSNSFullAccess | AWS managed | 2 |

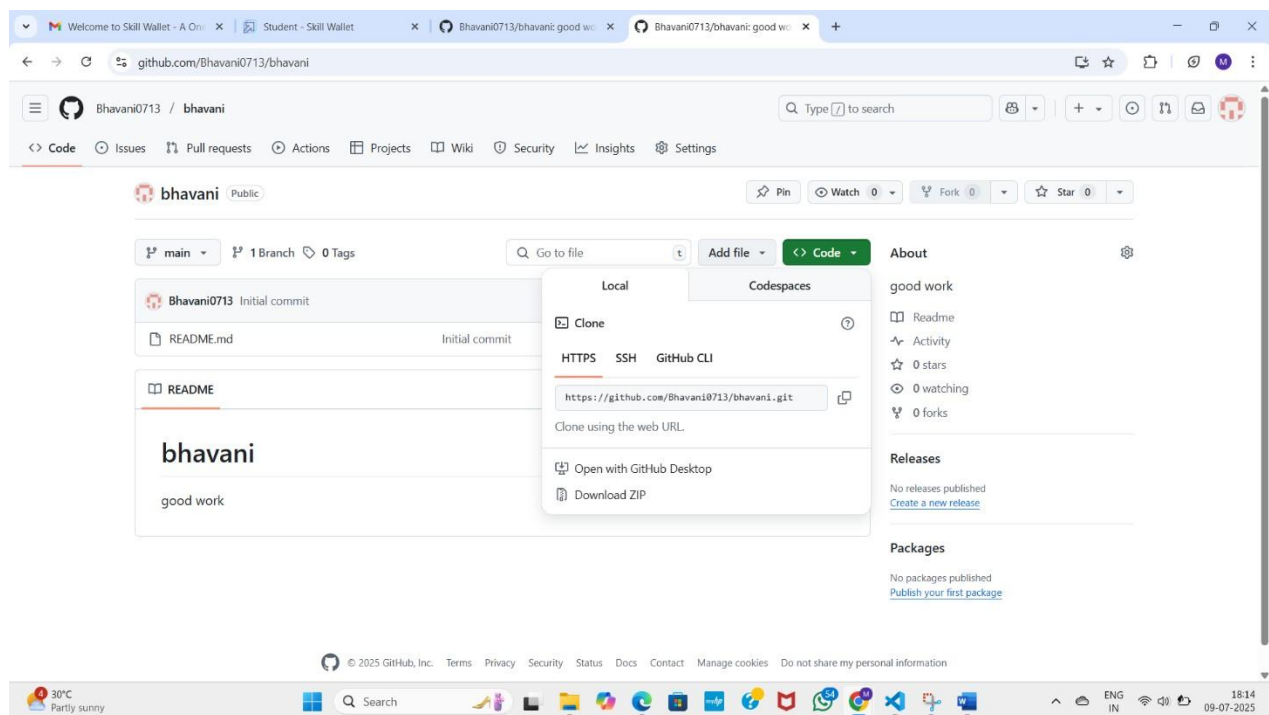
Milestone 5 : EC2 Instance Setup

To set up a public EC2 instance, choose an appropriate Amazon Machine Image (AMI) and instance type. Ensure the security group allows inbound traffic on necessary ports

(e.g., HTTP/HTTPS for web applications). After launching the instance, associate it with an Elastic IP for consistent public access, and configure your application or services to be publicly accessible.

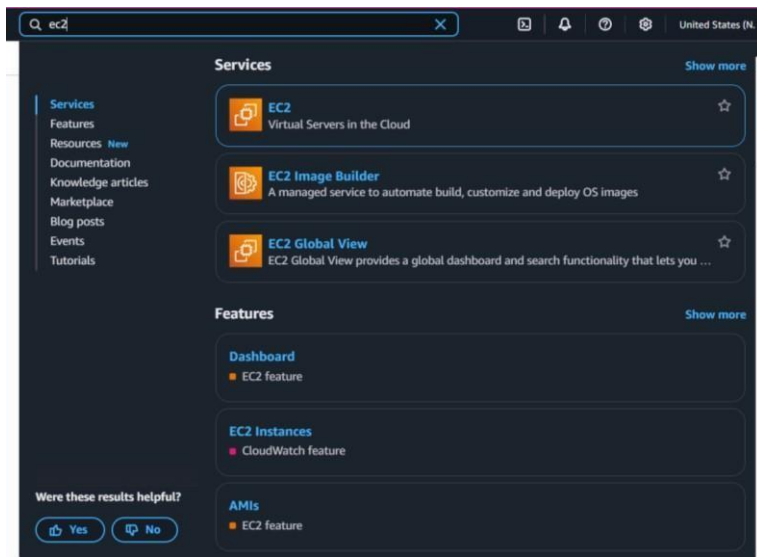
- Note: Load your Flask app and Html files into GitHub repository.

| | |
|---|------------------------|
|  static | Initial commit |
|  templates | Update statistics.html |
|  app.py | Update app.py |

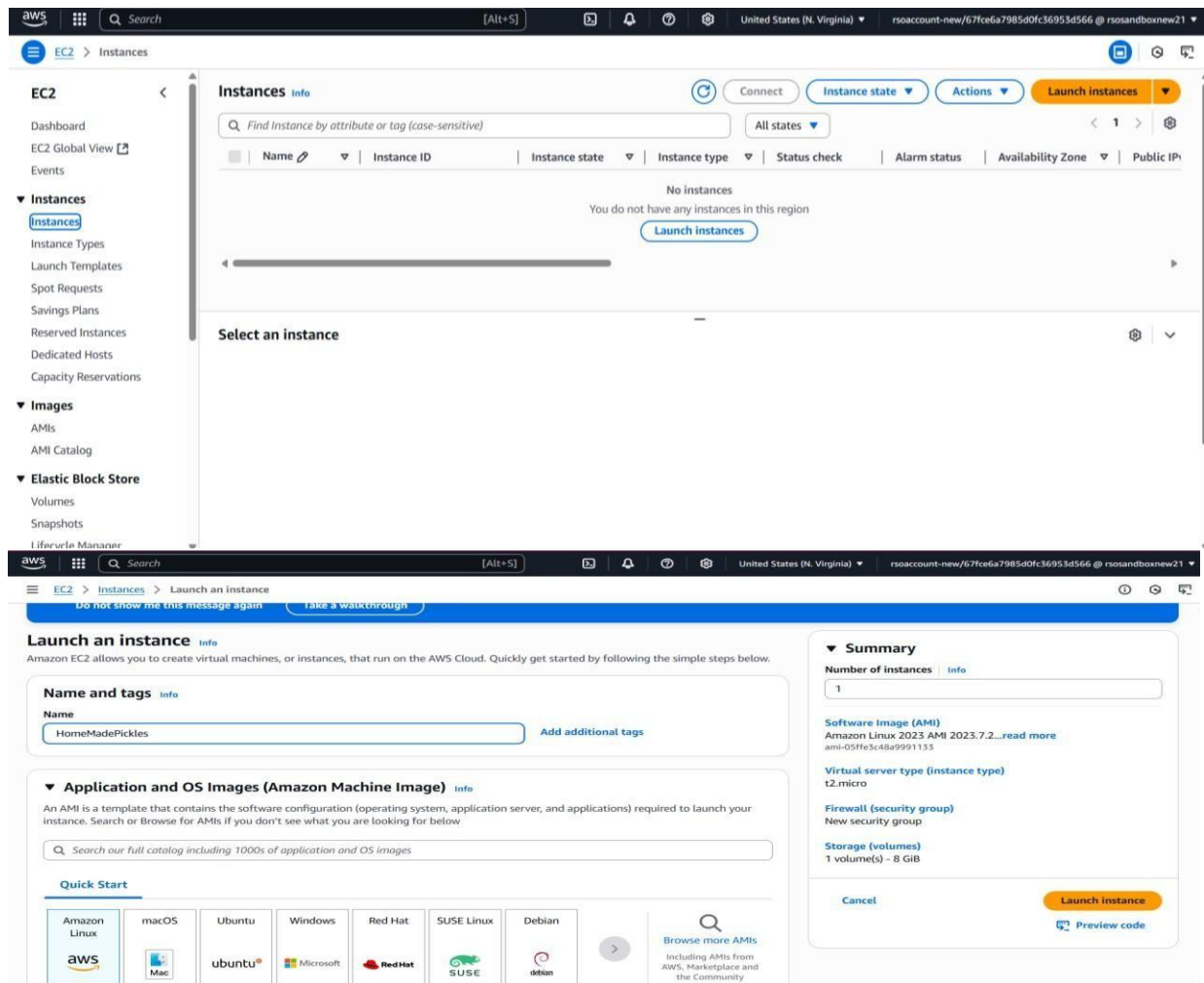


Launch an EC2 instance to host the Flask

- Launch EC2 Instance
- In the AWS Console, navigate to EC2 and launch a new instance.



- Click on Launch instance to launch EC2 instance



? Choose Amazon Linux 2 or Ubuntu as the AMI and t2.micro as the instance type (free-tier eligible).

The screenshot shows the AWS IAM console's 'Amazon Machine Image (AMI)' selection page. At the top, there are tabs for different operating systems: Amazon Linux, macOS, Ubuntu, Windows, and Red Hat. The 'Amazon Linux' tab is selected. Below the tabs, the 'Amazon Linux 2023 AMI' is highlighted. It shows the AMI ID 'ami-02b49a24cfb95941c' for 64-bit (x86) and 'ami-04ad8c7fcc828fad4' for 64-bit (Arm). The AMI is marked as 'Free tier eligible'. A description states: 'Amazon Linux 2023 is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.' Below the description, there are fields for 'Architecture' (64-bit (x86)), 'Boot mode' (uefi-preferred), and 'AMI ID' (ami-02b49a24cfb95941c). A green 'Verified provider' badge is also visible.

- Create and download the key pair for Server access.

The screenshot shows the AWS IAM console's 'Create key pair' dialog box. The dialog has a title 'Create key pair' and a close button. It contains the following fields and options: 'Key pair name' (HomeMade), 'Key pair type' (RSA selected, ED25519 also available), and 'Private key file format' (.pem selected, .ppk also available). A warning message states: 'When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance. Learn more'. At the bottom, there are 'Cancel' and 'Create key pair' buttons.

The screenshot shows the AWS IAM console's 'Launch an instance' page. A green banner at the top indicates 'Success' with the message 'Successfully initiated launch of instance (i-0d5ebf9d2ceab48c1)'. Below the banner, there is a 'Launch log' section and a 'Next Steps' section. The 'Next Steps' section shows a document icon.

HomeMade.pem

Configure security groups for HTTP, and SSH access.

▼ Network settings [Info](#)

VPC - *required* [Info](#)

vpc-03cdc7b6f19dd7211
172.31.0.0/16

(default) ▼



Subnet [Info](#)

No preference ▼



Create new subnet [↗](#)

Auto-assign public IP [Info](#)

Enable ▼

Additional charges apply when outside of free tier allowance

Firewall (security groups) [Info](#)

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

Security group name - *required*

launch-wizard

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-./()#,@[]+=&:{}!\$*

Description - *required* [Info](#)

launch-wizard created 2024-10-13T17:49:56.622Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Remove

Type [Info](#)

ssh ▼

Protocol [Info](#)

TCP

Port range [Info](#)

22

Source type [Info](#)

Anywhere ▼

Source [Info](#)

[Add CIDR, prefix list or security](#)

0.0.0.0/0 ✕

Description - *optional* [Info](#)

e.g. SSH for admin desktop

▼ Security group rule 2 (TCP, 80, 0.0.0.0/0)

Remove

Type [Info](#)

HTTP ▼

Protocol [Info](#)

TCP

Port range [Info](#)

80

Source type [Info](#)

Custom ▼

Source [Info](#)

[Add CIDR, prefix list or security](#)

0.0.0.0/0 ✕

Description - *optional* [Info](#)

e.g. SSH for admin desktop

▼ Security group rule 3 (TCP, 5000, 0.0.0.0/0)

Remove

Type [Info](#)

Custom TCP ▼

Protocol [Info](#)

TCP

Port range [Info](#)

5000

Source type [Info](#)

Custom ▼

Source [Info](#)

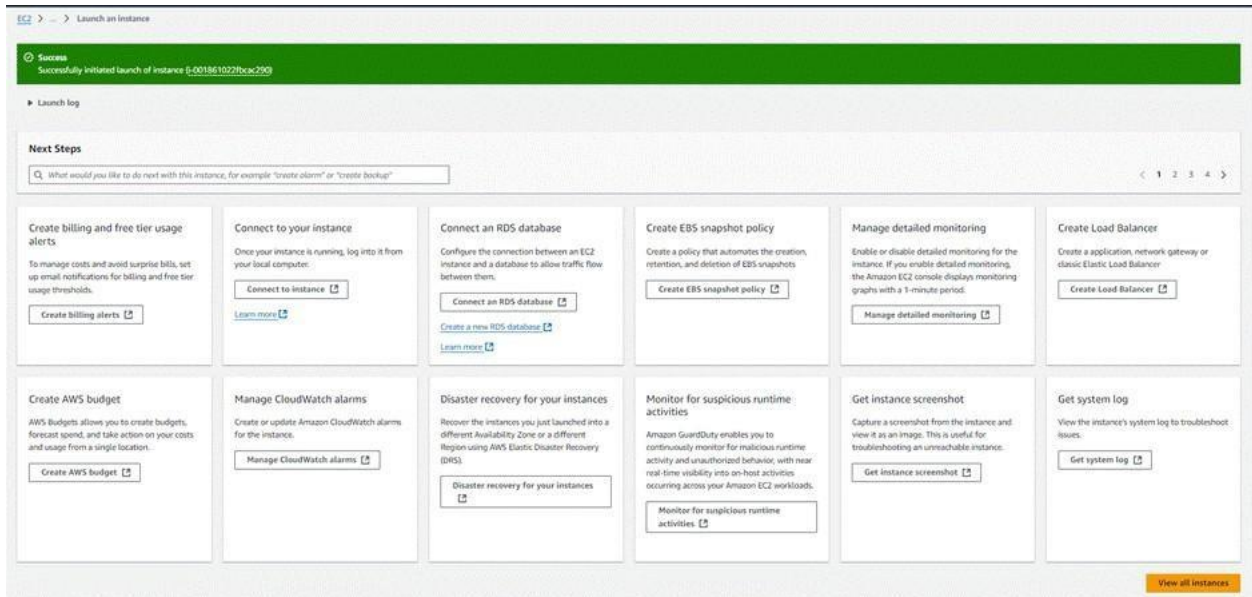
[Add CIDR, prefix list or security](#)

0.0.0.0/0 ✕

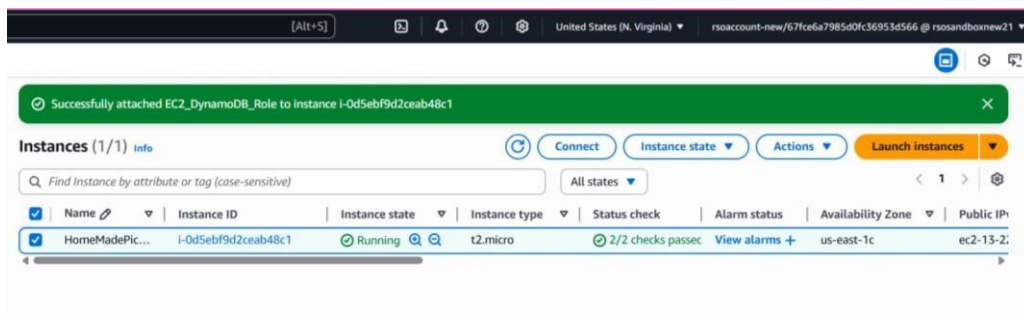
Description - *optional* [Info](#)

e.g. SSH for admin desktop

Add security group rule



- To connect to EC2 using EC2 Instance Connect, start by ensuring that an IAM role is attached to your EC2 instance. You can do this by selecting your instance, clicking on Actions, then navigating to Security and selecting Modify IAM Role to attach the appropriate role. After the IAM role is connected, navigate to the EC2 section in the AWS Management Console. Select the EC2 instance you wish to connect to. At the top of the EC2 Dashboard, click the Connect button. From the connection methods presented, choose EC2 Instance Connect. Finally, click Connect again, and a new browser-based terminal will open, allowing you to access your EC2 instance directly from your browser.



EC2 > Instances > i-001861022fbcac290

Instance summary for i-001861022fbcac290 (InstantLibraryApp) info

Connect

Instance state

Actions

Updated less than a minute ago

Instance ID
i-001861022fbcac290

IPv6 address
-

Hostname type
IP name: ip-172-31-3-5-ap-south-1-compute.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
-

IAM Role
sns_Dynamodb_role

IMDSv2
Required

Public IPv4 address
-

Instance state
Stopped

Private IP DNS name (IPv4 only)
ip-172-31-3-5-ap-south-1-compute.internal

Instance type
t2.micro

VPC ID
vpc-03cdc7b6f196d7211

Subnet ID
subnet-0d9fa3144480cc9a9

Instance ARN
arn:aws:ec2:ap-south-1:557690616836:instance/i-001861022fbcac290

Private IPv4 addresses
172.31.3.5

Public IPv4 DNS
-

Elastic IP addresses
-

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name
-

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

EC2 > Instances > i-001861022fbcac290

Instance summary for i-001861022fbcac290 (InstantLibraryApp) info

Connect

Instance state

Actions

Updated less than a minute ago

Instance ID
i-001861022fbcac290

IPv6 address
-

Hostname type
IP name: ip-172-31-3-5-ap-south-1-compute.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
-

IAM Role
sns_Dynamodb_role

IMDSv2
Required

Public IPv4 address
-

Instance state
Stopped

Private IP DNS name (IPv4 only)
ip-172-31-3-5-ap-south-1-compute.internal

Instance type
t2.micro

VPC ID
vpc-03cdc7b6f196d7211

Subnet ID
subnet-0d9fa3144480cc9a9

Instance ARN
arn:aws:ec2:ap-south-1:557690616836:instance/i-001861022fbcac290

Private IPv4 addresses
172.31.3.5

Public IPv4 DNS
-

Elastic IP addresses
-

AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name
-

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

Connect

Manage instance state

Instance settings

Networking

Security

Image and templates

Monitor and troubleshoot

EC2 > Instances > i-001861022fbcac290 > Modify IAM role

Modify IAM role info

Attach an IAM role to your instance.

Instance ID
i-001861022fbcac290 (InstantLibraryApp)

IAM role

Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.

sns_Dynamodb_role

Create new IAM role

Cancel

Update IAM role

Connect to instance Info


Connect to your instance i-001861022fbcac290 (InstantLibraryApp) using any of these options


EC2 Instance Connect

Session Manager

SSH client

EC2 serial console


**Port 22 (SSH) is open to all IPv4 addresses**
Port 22 (SSH) is currently open to all IPv4 addresses, indicated by **0.0.0.0/0** in the inbound rule in [your security group](#). For increased security, consider restricting access to only the EC2 Instance Connect service IP addresses for your Region: 13.233.177.0/29. [Learn more.](#)

Instance ID
 i-001861022fbcac290 (InstantLibraryApp)

Connection Type


☒ **Connect using EC2 Instance Connect**
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 or IPv6 address.

☐ **Connect using EC2 Instance Connect Endpoint**
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

☒ **Public IPv4 address**
 13.200.229.59

☐ IPv6 address
—

Username
Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ec2-user.

 **Note:** In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Cancel

Connect

```
A newer release of "Amazon Linux" is available.
Version 2023.6.20241010:
Run "/usr/bin/dnf check-release-update" for full release and version update info

Amazon Linux 2023

https://aws.amazon.com/linux/amazon-linux-2023

Last login: Tue Oct 15 04:17:59 2024 from 13.233.177.3
ec2-user@ip-172-31-3-5 ~]$
```

Milestone 6 : Deployment on EC2

Deployment on an EC2 instance involves launching a server, configuring security groups for public access, and uploading your application files. After setting up

necessary dependencies and environment variables, start your application and ensure it's running on the correct port. Finally, bind your domain or use the public IP to make the application accessible online.

Install Software on the EC2 Instance

Install Python3, Flask, and Git:

On Amazon Linux 2:

- `sudo yum update -y`
- `sudo yum install python3 git`
- `sudo pip3 install flask boto3`

Verify Installations:

- `flask --version`
- `git --version`

Clone Your Flask Project from GitHub

Clone your project repository from GitHub into the EC2 instance using Git.

- Run: `"git clone https://github.com/nikhitayalamanchi/AWS_project.git"`
- Note: change your-github-username and your-repository-name with your credentials

here: `"https://github.com/nikhitayalamanchi/AWS_project.git"`

- This will download your project to the EC2 instance.

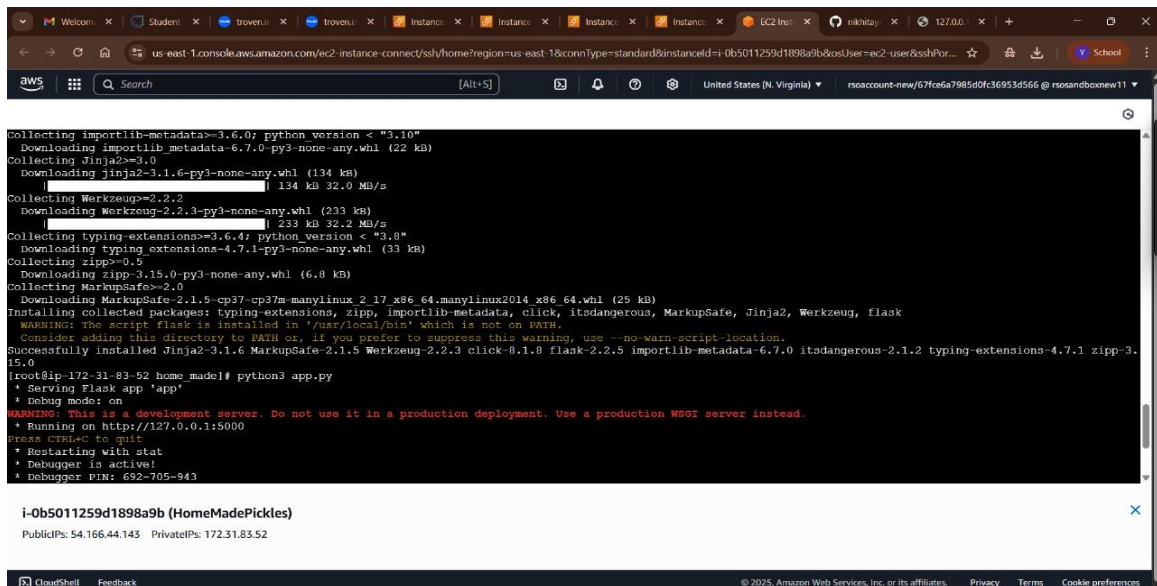
To navigate to the project directory, run the following command:

- `cd Homemadepicklesandsnacks`
- `cd "Home Made Pickles1" Create a Virtual Environment:`
- `python3 -m venv venv`
- `source venv/bin/activate`
- `sudo yum install python3 git`
- `sudo pip3 install flask boto3`

Once inside the project directory, configure and run the Flask application by executing the following command with elevated privileges:

- [illegible]

- Run the Flask app on the EC2 instance



Access the w

ebsite through:

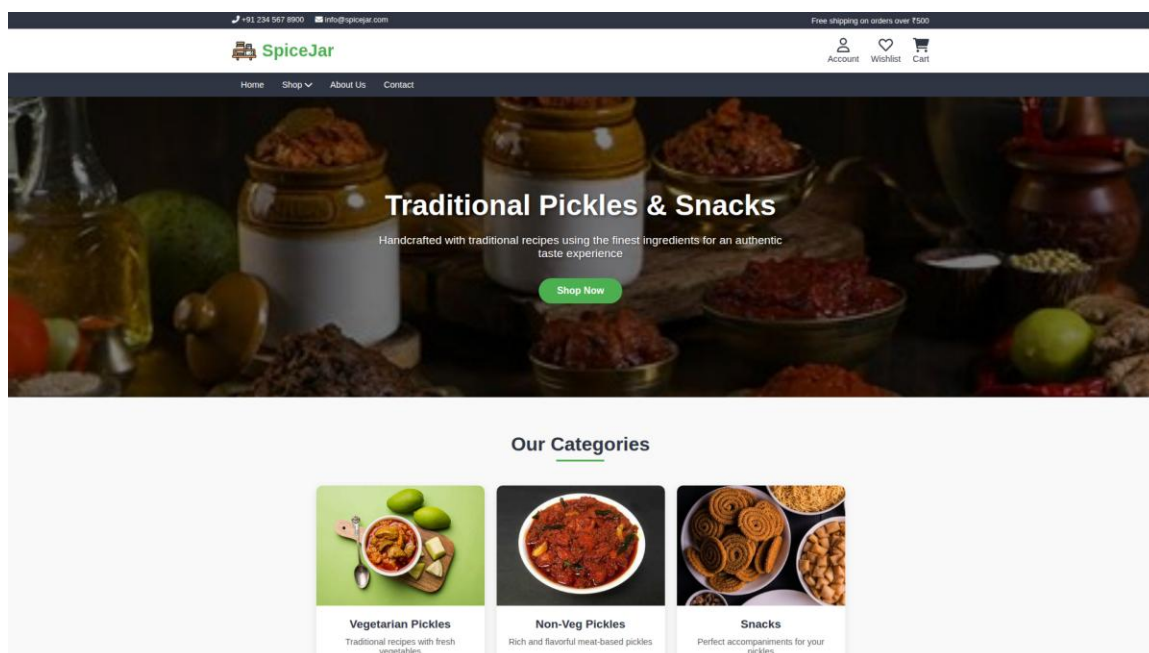
PublicIPs: http://54.166.44.143:5000/

Milestone 7 : Testing and Deployment

Testing and deployment involve verifying that your application works as expected before making it publicly accessible. Start by testing locally or on a staging environment to catch bugs and ensure functionality. Once tested, deploy the application to an EC2 instance, configure necessary services, and perform a final round of live testing to confirm everything runs smoothly in the production environment.

Functional testing to verify the Project

- Welcome page:











- veg pickles:

Premium Veg Pickles Collection

Authentic homemade pickles with traditional recipes

Our Vegetarian Pickle Varieties (15 Options)

*Rates are mentioned per kg

| | | | |
|---|--|--|---|
|  <p>Traditional Mango Pickle Spicy raw mango pickle with mustard oil and spices</p> <p>₹199</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |  <p>Lemon Pickle Tangy lemon preserved with spices and oil</p> <p>₹179</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |  <p>Mixed Vegetable Pickle Assorted vegetables in spicy mustard oil</p> <p>₹229</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |  <p>Garlic Pickle Pungent garlic cloves in spicy marinade</p> <p>₹249</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |
|  <p>Green Chilli Pickle Fiery green chillies preserved with spices</p> <p>₹189</p> |  <p>Carrot Pickle Crunchy carrots in tangy spice mix</p> <p>₹199</p> |  <p>Ginger Pickle Zesty ginger slices in spicy oil</p> <p>₹219</p> |  <p>Amla (Gooseberry) Pickle Nutritious amla in traditional spices</p> <p>₹239</p> |









- non-veg pickles :

Premium Non-Veg Pickles Collection

Authentic homemade meat pickles with traditional recipes

Our Non-Vegetarian Pickle Varieties (10 Options)

*Rates are mentioned per kg

| | | | |
|--|---|---|---|
|  <p>Spicy Chicken Pickle Tender chicken pieces in fiery spices</p> <p>₹349</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |  <p>Mutton Pickle Juicy mutton chunks in rich gravy</p> <p>₹399</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |  <p>Prawn Pickle Succulent prawns in tangy spice mix</p> <p>₹379</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |  <p>Fish Pickle Sea fish pieces in traditional spices</p> <p>₹329</p> <p>Add to Cart</p> <p>Add to Wishlist</p> |
|  <p>Chicken Keema Pickle Minced chicken in spicy oil</p> <p>₹299</p> |  <p>Crab Pickle Delicate crab meat infused with bold, rustic masalas</p> <p>₹449</p> |  <p>Egg Pickle Boiled eggs in spicy gravy</p> <p>₹279</p> |  <p>Bombay Duck(Fish) Pickle Tender bombay duck pieces in special masala</p> |


- snacks:

Premium Homemade Snacks Collection

Traditional and innovative snack options for every taste

Snacks (12 Options)

*Rates are mentioned per kg




Traditional Murukku

Crispy rice flour spiral snacks

₹129

Add to Cart

Add to Wishlist




Thattai

Crunchy lentil flour discs

₹119

Add to Cart

Add to Wishlist




Mixture

Assorted fried snacks mix

₹149

Add to Cart

Add to Wishlist




Sev

Thin gram flour noodles

₹99

Add to Cart

Add to Wishlist




Banana Chips

Crispy raw banana slices

₹109

Add to Cart




Kara Boondhi

Tiny fried gram flour pearls

₹89

Add to Cart




Spiced Peanuts

Roasted peanuts with spices

₹79

Add to Cart



Chakli

Spiral shaped crispy snack

₹139

Add to Cart

- Cart page:

Your Shopping Cart

Traditional Mango Pickle

Spicy raw mango pickle with mustard oil and spices

₹199.00

✕

Qty: 1

Spicy Chicken Pickle

Tender chicken pieces in fiery spices

₹349.00

✕

Qty: 1

Traditional Murukku

Crispy rice flour spiral snacks

₹129.00

✕

Qty: 1

Subtotal

₹677.00

Shipping

FREE

Total

₹677.00

Continue Shopping

Proceed to Checkout

© 2023 Homemade Pickles & Snacks. All rights reserved.

- Checkout page:

Checkout

Complete your purchase

Shipping Information

First Name

Last Name

Address

City

State

ZIP Code

Phone Number

Payment Method

☒ Credit Card
 ☐ PayPal
 ☐ Cash on Delivery

Card Number

1234 5678 9012 3456

Expiry Date

MM/YY

CVV

123

Order Notes (optional)

Order Summary


| | |
|---------------------|----------------|
| Lemon Pickle Qty: 1 | ₹179.00 |
| Prawn Pickle Qty: 1 | ₹379.00 |
| Subtotal | ₹558.00 |
| Shipping | FREE |
| Total | ₹558.00 |

Place Order

- order confirmationi page:

Homemade Pickles & Snacks

Order Confirmation



Order Confirmed!

Thank you for your purchase. We've sent a confirmation email to your registered address.

Continue Shopping

© 2023 Homemade Pickles & Snacks. All rights reserved.

About page: “We chose to give the About page a different background to make it stand out, as it tells the unique story of our homemade pickles.



This helps users clearly understand what makes our products special."

Dynamodb Database updations :

1. Users table :

✓ Completed. Read capacity units consumed: 2

| Items returned (3) | | | | Actions | Create item |
|--------------------------|-------------------|---------------|---|---------|-------------|
| | username (String) | email | password | | |
| <input type="checkbox"/> | Shiva | kilarukusu... | scrypt:32768:8:1\$W5tA59Z7nQjLXbtX\$d6bfef2b3e14bbe9d3d3e3f1c... | | |
| <input type="checkbox"/> | kusuma | <empty> | <empty> | | |
| <input type="checkbox"/> | Alekhya | alekhya@g... | scrypt:32768:8:1\$EwCDTl0iaGcKutw3\$cd5dbf5c12ec17cb518f7c15cd... | | |

2. Orders table :

Completed. Read capacity units consumed: 2

Items returned (4)

Actions

Create item

< 1 > ⚙️

| <input type="checkbox"/> | order_id (String) | address | items | name | payment_met... | phone |
|--------------------------|--|-------------|---------------|--------------|----------------|---------|
| <input type="checkbox"/> | 7c6bd84e-f2c7-4fe0-... | Kothur | [{"M": {"n... | Siri | cod | 8187810 |
| <input type="checkbox"/> | 3de0fe0c-9539-4fb6-... | chatanpally | [{"M": {"n... | KILARU KU... | cod | 9849889 |
| <input type="checkbox"/> | fbc41d6d-d6f2-4158-... | chatanpally | [{"M": {"n... | KILARU KU... | cod | 9849889 |
| <input type="checkbox"/> | 1q | <empty> | [] | <empty> | <empty> | 0 |

Conclusion

The Homemade Pickles and Snacks platform has been meticulously crafted to deliver a seamless and delightful experience for food enthusiasts seeking authentic, handcrafted flavors. By leveraging modern web technologies such as Flask for backend logic, secure user authentication, and dynamic cart management, the platform ensures a user-friendly interface for browsing, customizing, and ordering artisanal pickles and snacks.

The integration of cloud-ready architecture (e.g., AWS for future scalability) and robust session management allows the platform to handle high traffic efficiently while maintaining real-time updates for orders and inventory. Features like weight-based pricing, category-specific searches, and instant checkout streamline the shopping process, empowering customers to explore a diverse range of traditional and innovative recipes with ease.

This project addresses the growing demand for homemade, preservative-free food products by bridging the gap between small-scale producers and discerning customers. The platform's intuitive design and secure payment workflows enhance trust and convenience, while backend tools enable effortless inventory tracking and order fulfillment for administrators.

By combining time-honored recipes with modern e-commerce capabilities, this website not only preserves culinary heritage but also adapts to the digital age, ensuring that every jar of pickle or snack reaches customers with the same care and quality as a homemade meal. As the platform evolves, it stands ready to scale, introduce new product lines, and foster a community of food lovers united by a passion for authentic flavors.

In essence, this project redefines the way homemade delicacies are shared and enjoyed, offering a flavorful bridge between tradition and technology.