AWS Marketplace Web Application Project Documentation

# 1. Introduction

This project demonstrates the design and deployment of a Marketplace Web Application using AWS Academy Sandbox. The goal was to build a cloud-native infrastructure, deploy a dynamic web app, and implement authentication, while overcoming sandbox limitations.

# 2. Cloud Infrastructure Design

The infrastructure was deployed across two AWS regions (N. Virginia and Ohio) to ensure high availability and regional failover. The setup includes:  
- Two VPCs (one in N. Virginia, one in Ohio).  
- Public and private subnets in each VPC.  
- EC2 instances running a Flask-based web application.  
- Internet Gateway and NAT Gateway for internet access.  
- Transit Gateway and Peering Connection to connect VPCs across regions.  
- DynamoDB (planned for production) for user data storage.  
- S3 bucket (planned for product images), mirrored across regions.

# 3. EC2 Instances

Two EC2 instances were deployed:  
- \*\*N. Virginia EC2\*\*: Hosted the main Marketplace web application.  
- \*\*Ohio EC2\*\*: Backup instance for failover.  
  
Access between instances was tested via PuTTY using Pageant to load multiple SSH keys, allowing private IP communication between regions.

# 4. Authentication System

The original plan was to use Amazon Cognito for user authentication. However, the AWS Academy Sandbox does not support IAM role creation, which made Cognito integration unavailable.  
  
To overcome this limitation, a custom authentication system was implemented using Flask and a local JSON file (`users.json`) stored on the EC2 instance:  
- Users can \*\*register\*\* with email and password.  
- Users can \*\*login\*\*, receiving a JWT token if credentials are valid.  
- User accounts are saved and validated against the `users.json` file.

# 5. Marketplace Web Application

The front-end was built with HTML, CSS, and JavaScript, while the back-end was developed using Flask.  
  
Main Features:  
- \*\*Homepage\*\* with marketplace overview.  
- \*\*Login Page\*\* connected to Flask backend.  
- \*\*Register Page\*\* storing new users in `users.json`.  
- \*\*Products Page\*\* as a placeholder for marketplace items.

# 6. Challenges and Solutions

Several challenges were faced during the project due to AWS Sandbox restrictions:  
1. \*\*Cognito not supported\*\* → Solution: Custom Flask + JSON authentication.  
2. \*\*IAM role limitations\*\* → Solution: Local user storage on EC2.  
3. \*\*S3 access issues\*\* → Solution: Planned fallback with local static files.  
4. \*\*Cross-region communication\*\* → Solution: Configured Transit Gateway + VPC peering.

# 7. Screenshots Overview

The following screenshots were taken to demonstrate the system setup and functionality:  
1. EC2 Details (N. Virginia)  
2. EC2 Details (Ohio)  
3. PuTTY + Pageant dual key setup  
4. Marketplace Homepage  
5. Marketplace Products Page  
6. VPC Resource Map (Ohio)  
7. VPC Resource Map (N. Virginia)  
8. Login Page  
9. Register Page  
10. users.json file with registered users

# 8. Conclusion

This project provided hands-on experience with AWS cloud infrastructure, multi-region architecture, and web application deployment. By working around sandbox limitations, a production-like system was simulated, demonstrating both technical skills and problem-solving abilities.