Ecommerce Microservices Application Proposal

Final Project Submission

Team Leader: Mahmoud Nasser

Email: Mahmoudnasser3.14pi@gmail.com

Submitted on: October 11, 2025

1 Project Description

The Ecommerce Microservices Application is a cloud-native platform designed to enable seamless online shopping. The system comprises multiple microservices communicating via REST APIs, each managing a specific domain such as user management, product listings, cart logic, and notifications. This project aims to demonstrate a scalable microservice architecture, leveraging CI/CD automation and cloud deployment with modern DevOps practices. It incorporates containerization, automated pipelines, monitoring, and logging to ensure high reliability and performance.

2 Group Members and Roles

Name	Role	Responsibilities
Mahmoud Nasser	Development + GitHub Actions	Leads backend architecture design and implements microservices.
Moayad Darwish	Development	Develops responsive UI for users and restaurant owners.
Basma Hassan	IAC using Terraform	Designs and manages AWS infrastructure.
Ali Yasser	Monitoring and Logging	Creates a logging pipeline in Elastic Search or Grafana.
Mohannad Hesham	Containerization using Docker	Containerizes microservices and ensures proper communication between containers.
Khaled Ali	Kubernetes Cluster Management	Manages Kubernetes and ensures proper deployment on the cloud.

3 Objectives

- Design and implement a scalable microservice-based e-commerce system.
- Automate the build, test, and deployment process using Jenkins or GitHub Actions, Docker, and Ansible.
- Deploy the system on a cloud environment (AWS or Azure) using Docker and Kubernetes.
- Implement observability (monitoring, logging, and alerts) for improved reliability.
- Ensure continuous integration and delivery through version-controlled pipelines.

4 Tools and Technologies

- **Backend**: .NET 8 Web API, Fast API (for async services)
- **Database**: PostgreSQL, Redis (or Aurora)
- **Frontend**: React.js / Next.js
- **CI/CD & DevOps**: Jenkins, GitHub Actions, Docker, Docker Hub, Ansible, Kubernetes
- **Cloud Platform**: AWS EC2, ECR, S3, RDS
- **Monitoring**: Prometheus, Grafana, ELK Stack
- **Version Control**: Git & GitHub

5 Milestones and Timeline

Milestone	Description
M1: Requirements Gathering & System Design	Define architecture, microservice boundaries, and tech stack.
M2: Backend Service Development	Implement user, products, cart, and purchase services.
M3: Database Integration	Connect microservices to PostgreSQL.
M4: CI/CD Pipeline Configuration	Setup Jenkins, Docker, and Ansible for automated builds and deployments.
M5: Cloud Deployment	Deploy services to AWS using Docker & Kubernetes.
M6: Monitoring & Logging Integration	Configure Prometheus, Grafana, and ELK for metrics and logs.
M7: Final Testing & Presentation	Conduct full system testing and final project demo.

Note: Specific deadlines for milestones are not provided. Tentative dates (e.g., M1: Week 1-2, M2: Week 3-4) are recommended for planning purposes.

6 Key Performance Indicators (KPIs)

6.1 Infrastructure & Automation

- Jenkins pipeline configured with automated build and test triggers on Git commits or GitHub Actions.
- Docker images built and pushed to Docker Hub automatically.
- Ansible playbooks for environment setup and deployment automation.

6.2 Pipeline Efficiency & Performance

- CI/CD pipeline completes under 10 minutes from code commit to deployment.
- Zero downtime during rolling updates.

6.3 Code Integration & Testing

- Unit test coverage across all microservices.
- Automated integration tests run after every successful build.
- GitHub integration for pull request validation and status checks.

6.4 Deployment & Cloud Management

- Application deployed successfully to AWS using Docker and Kubernetes clusters.

6.5 Monitoring & Reliability

- Real-time monitoring with Prometheus & Grafana dashboards.
- Centralized logging with ELK Stack.