SwiftShop

E-Commerce Web Application

With DevOps Integration

Submitted By: Vishal Kumar

Reg.No: 12222786

Roll.No: 30 (KO129)

Submitted To: Arshiya Mam

Course Code: INT332

Table of Contents

2. Introduction

Swiftshop is a web application that enables users to search and compare products across various e-commerce platforms. It gathers real-time data via web scraping and presents a unified comparison based on price, discounts, delivery time, and return policy. The application aims to enhance the online shopping experience by saving time and providing informed choices. The system is built using a microservice architecture, containerized with Docker, and deployed using a CI/CD pipeline in Github Actions, making it highly scalable, maintainable, and efficient.

3. Objectives

- Develop a product search and comparison system across multiple e-commerce platforms.
- Use web scraping tools to extract real-time data.
- Implement microservices for modular and scalable design.
- Containerize the entire application using Docker.
- Integrate Jenkins for automated testing and deployment.
- Deliver a fast, secure, and user-friendly platform to assist users in smart shopping decisions.

The following technologies are utilized in the development of Seiftshop:

- **Frontend:** React.js, Tailwind CSS, Axios **Backend:** Node.js,
- Express.js, MongoDB **Scraping Tools:** Puppeteer, Cheerio
- for web data
  extraction
- **Authentication:** JWT (JSON Web Tokens) for secure user login

4. Implementation

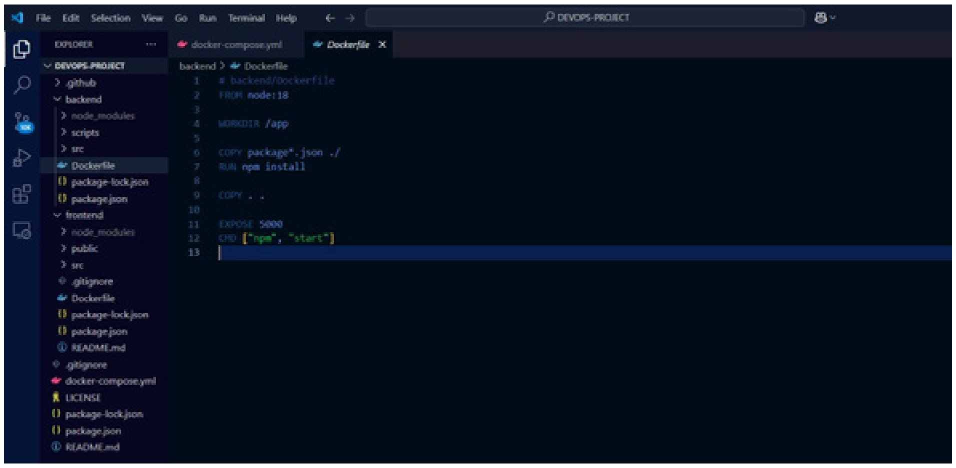
4.1 Microservices Architecture

• Product Scraper Service: Scrapes product details (price, offers, policy) using Puppeteer/Cheerio.
• Comparison Engine Service: Processes scraped data and ranks products.
• User Authentication Service: Manages user login, registration using JWT.
• Frontend Service: Developed in React.js to display data visually.
• Database Service: Uses MongoDB to store user info and scraped data.
Each service is independently developed and deployed, allowing easy scaling and updating.

4.2 Docker Containerization

Each microservice is containerized using Docker. Steps involved:

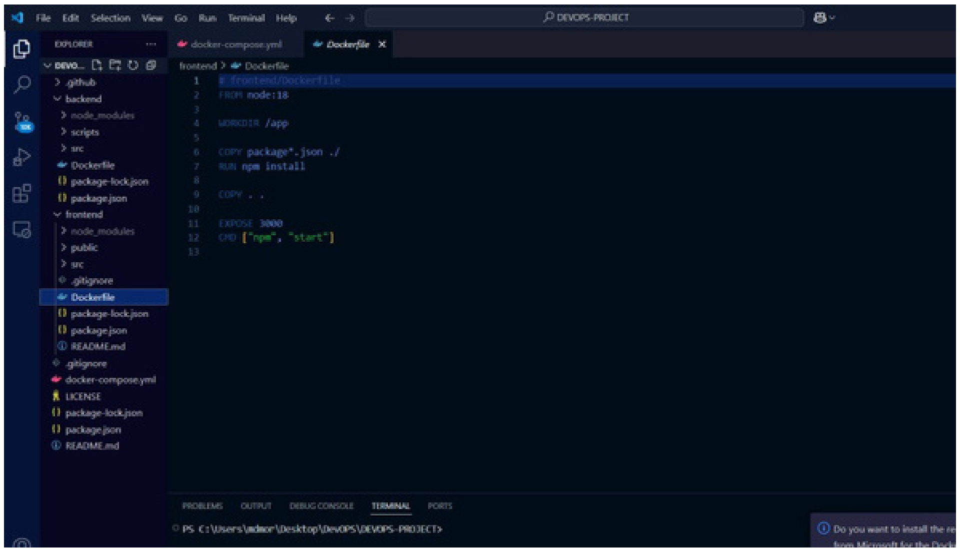
1. Dockerize Backend Services:



```
# backend/Dockerfile
FROM node:18

WORKDIR /app

COPY package*.json ./
RUN npm install

COPY . .

EXPOSE 5000
CMD ["npm", "start"]
```

Build the image: docker build -t Swiftshop-backend .

2. Dockerize React Frontend:



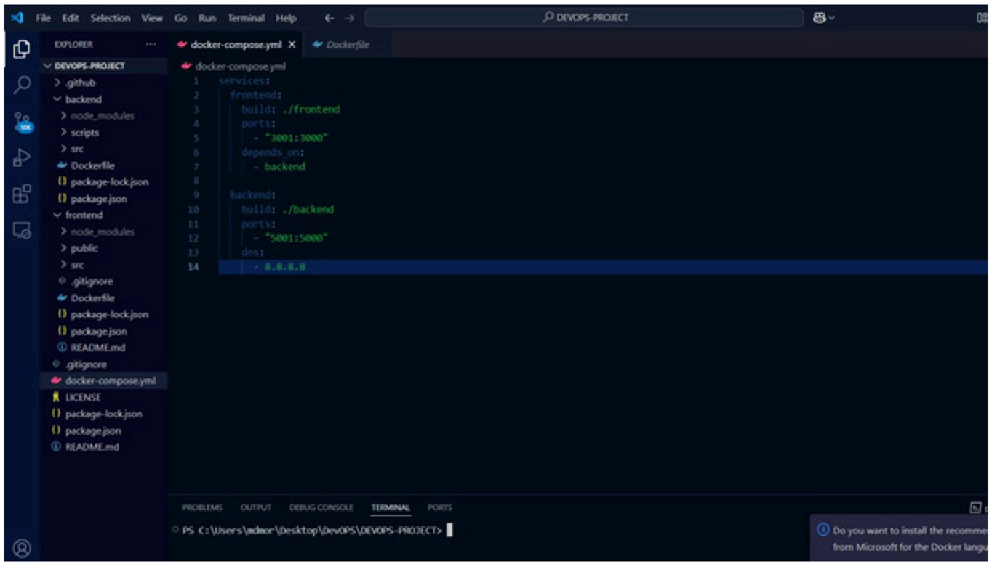
```
# frontend/Dockerfile
FROM node:18

WORKDIR /app

COPY package*.json ./
RUN npm install

COPY . .

EXPOSE 3000
CMD ["npm", "start"]
```

```
PS C:\Users\mdnor\Desktop\DevOPS\DEVOPS-PROJECT>
```

3. MongoDB Docker Setup:

• Use official MongoDB image in docker-compose.yml

4. Docker Compose for Orchestration:

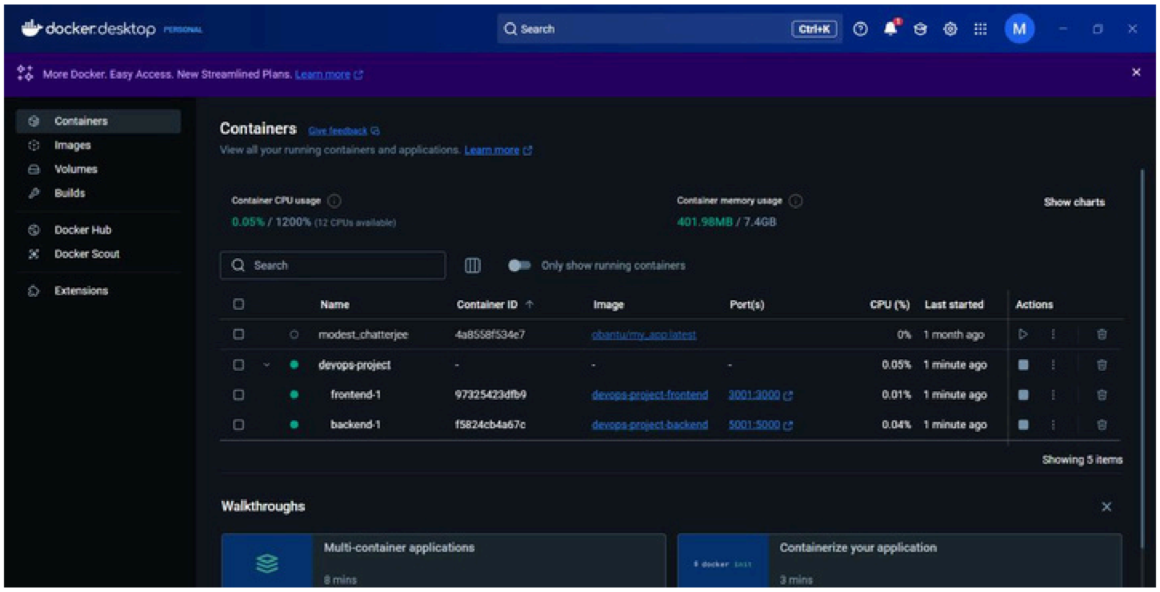


```yaml
services:
  frontend:
    build: ./frontend
    ports:
      - "3001:3000"
    depends_on:
      - backend

  backend:
    build: ./backend
    ports:
      - "5001:5000"
    dns:
      - 8.8.8.8
```

```
PS C:\Users\mdmor\Desktop\DevOPS\DEVOPS-PROJECT>
```

5. Run All Services:

docker-compose up --build



Containers

View all your running containers and applications.

Container CPU usage 0.05% / 1200% (12 CPUs available)

Container memory usage 401.98MB / 7.4GB

Name	Container ID	Image	Port(s)	CPU (%)	Last started
modest_chatterjee	4a8558f534e7	obantu/my_app:latest		0%	1 month ago
devops-project	-	-	-	0.05%	1 minute ago
frontend-1	97325423dfb9	devops-project-frontend	3001:3000	0.01%	1 minute ago
backend-1	f5824cb4a67c	devops-project-backend	5001:5000	0.04%	1 minute ago

Showing 5 items

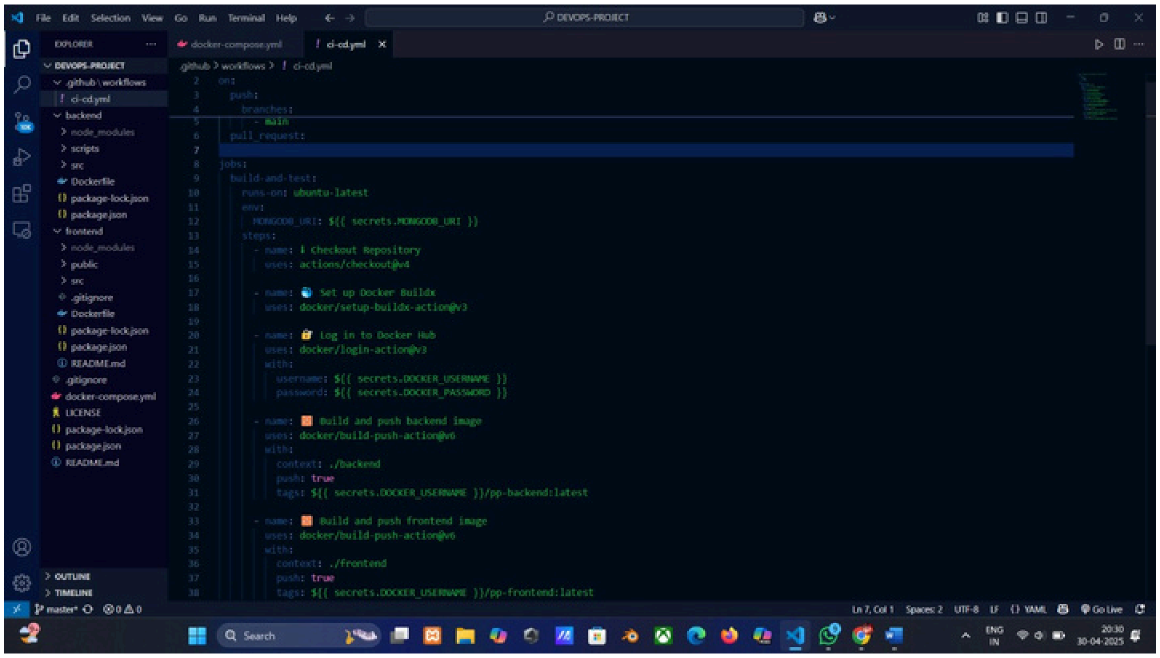
4.3 Jenkins CI/CD Pipeline

Steps to automate testing and deployment using Jenkins:

• Set up Jenkins on server/VM.

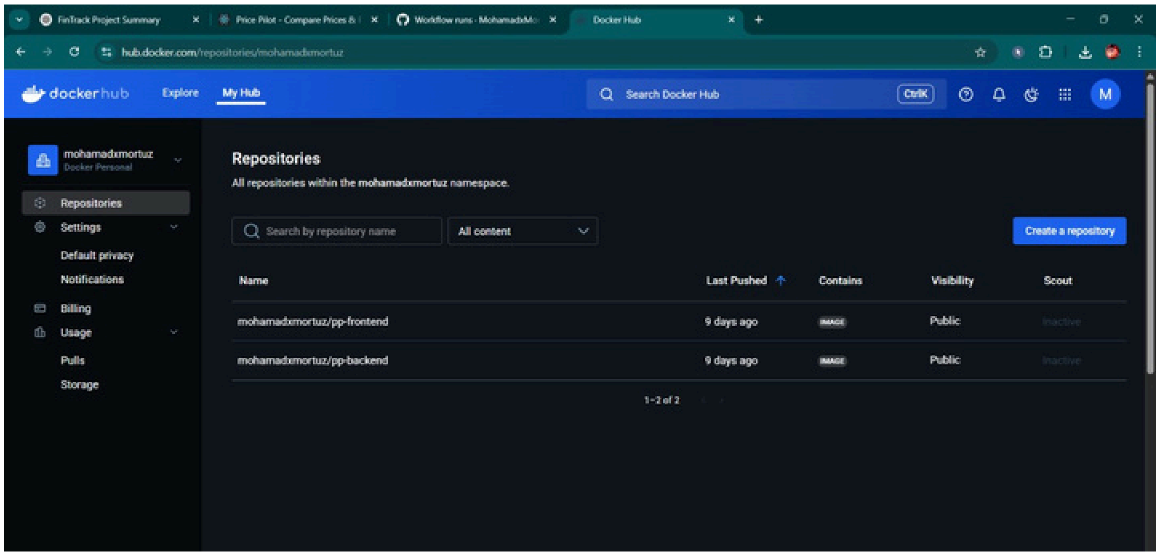
• Create Jenkins Pipeline (Declarative or Freestyle).

• Sample Jenkinsfile:



```yaml
on:
  push:
    branches:
      - main
  pull_request:

jobs:
  build-and-test:
    runs-on: ubuntu-latest
    env:
      MONGODB_URI: ${{ secrets.MONGODB_URI }}
    steps:
      - name: ⬇ Checkout Repository
        uses: actions/checkout@v4

      - name: 🌐 Set up Docker Buildx
        uses: docker/setup-buildx-action@v3

      - name: 🔐 Log in to Docker Hub
        uses: docker/login-action@v3
        with:
          username: ${{ secrets.DOCKER_USERNAME }}
          password: ${{ secrets.DOCKER_PASSWORD }}

      - name: 🧱 Build and push backend image
        uses: docker/build-push-action@v6
        with:
          context: ./backend
          push: true
          tags: ${{ secrets.DOCKER_USERNAME }}/pp-backend:latest

      - name: 🧱 Build and push frontend image
        uses: docker/build-push-action@v6
        with:
          context: ./frontend
          push: true
          tags: ${{ secrets.DOCKER_USERNAME }}/pp-frontend:latest
```

Created Repositories for Integrateing CI/CD:



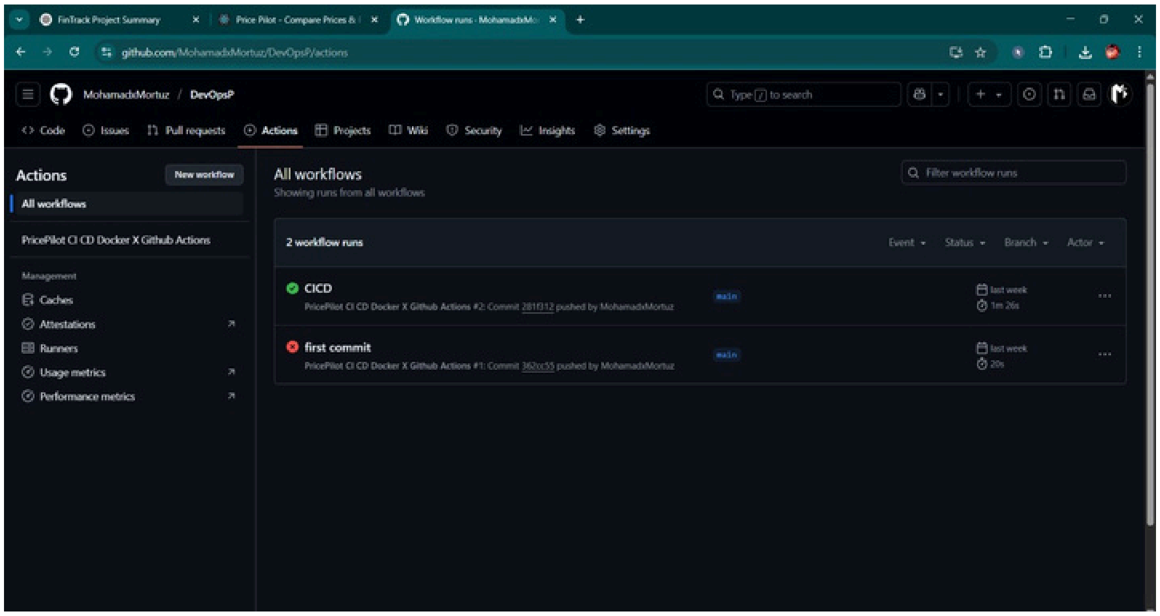
Repositories

All repositories within the mohamadxmortuz namespace.

Name	Last Pushed	Contains	Visibility	Scout
mohamadxmortuz/pp-frontend	9 days ago	IMAGE	Public	Inactive
mohamadxmortuz/pp-backend	9 days ago	IMAGE	Public	Inactive

1–2 of 2

1. Integrate with GitHub/GitLab Webhooks for auto-trigger on push.
2. Monitor Logs and Deployment from Jenkins UI.



All workflows
Showing runs from all workflows

2 workflow runs

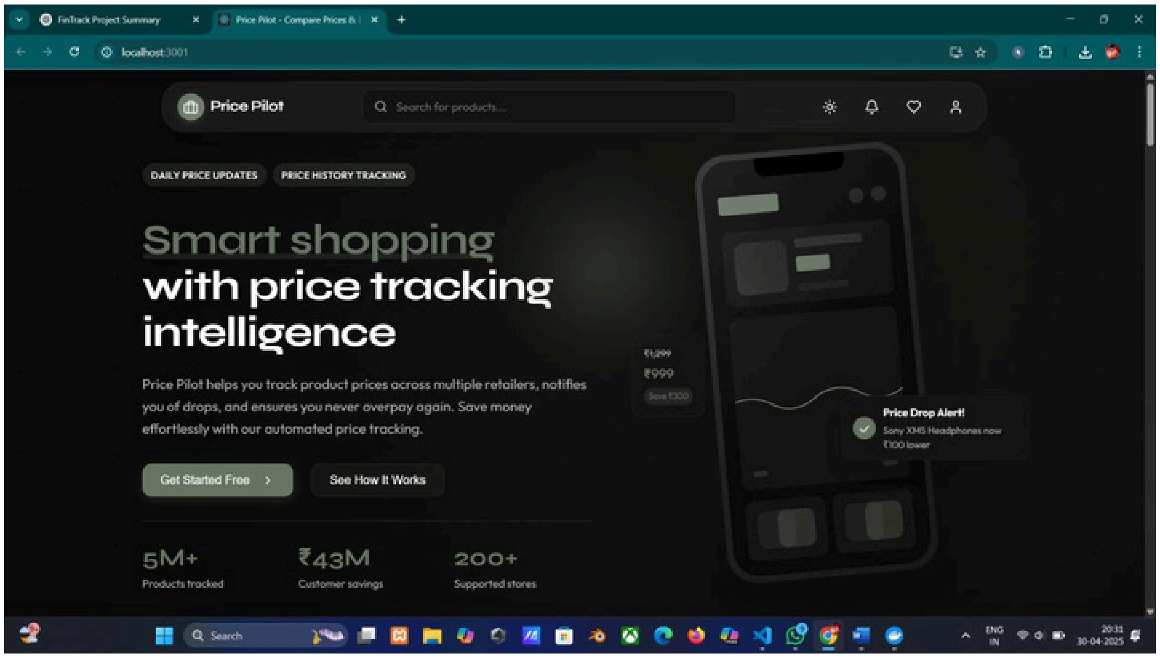
CICD
PricePilot CI CD Docker X Github Actions #2: Commit 281f312 pushed by MohamadxMortuz — main — last week — 1m 26s

first commit
PricePilot CI CD Docker X Github Actions #1: Commit 362cc55 pushed by MohamadxMortuz — main — last week — 20s

5. Outcomes

- Fully functional containerized application for e-commerce comparison.
- Microservices-based architecture for better scalability and modularity.
- Docker ensures platform-independent deployment with consistent environments.
- CI/CD pipeline with Jenkins automates build, test, and deployment.
- Helps users make informed shopping decisions quickly and efficiently.

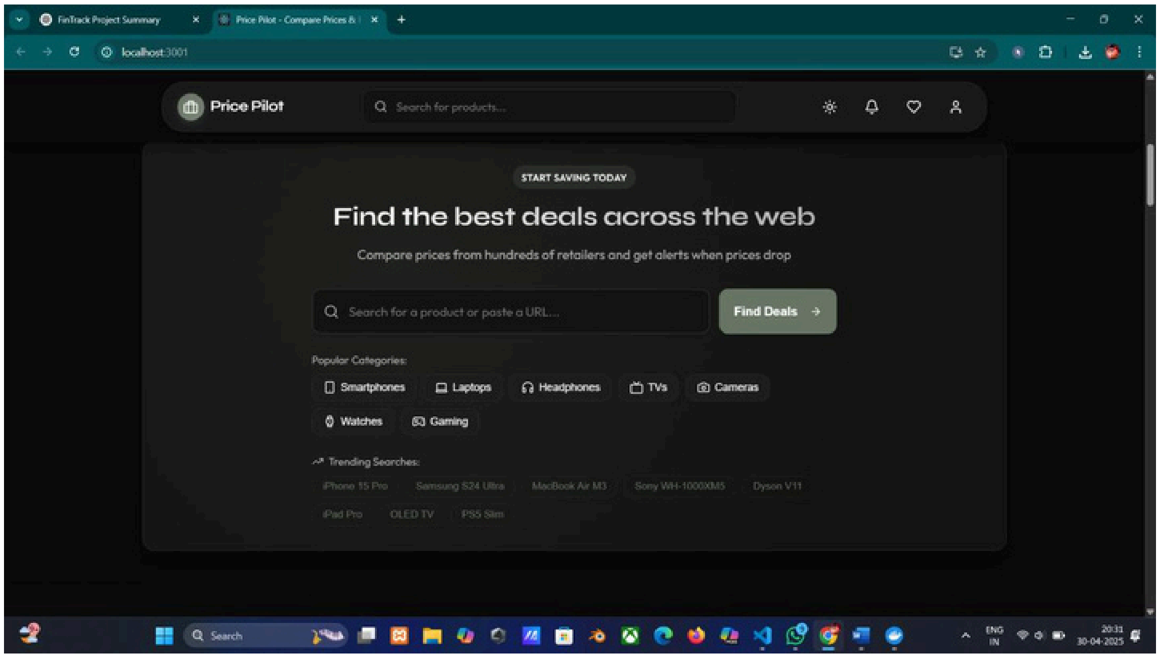
6. Project Visuals:



Smart shopping with price tracking intelligence

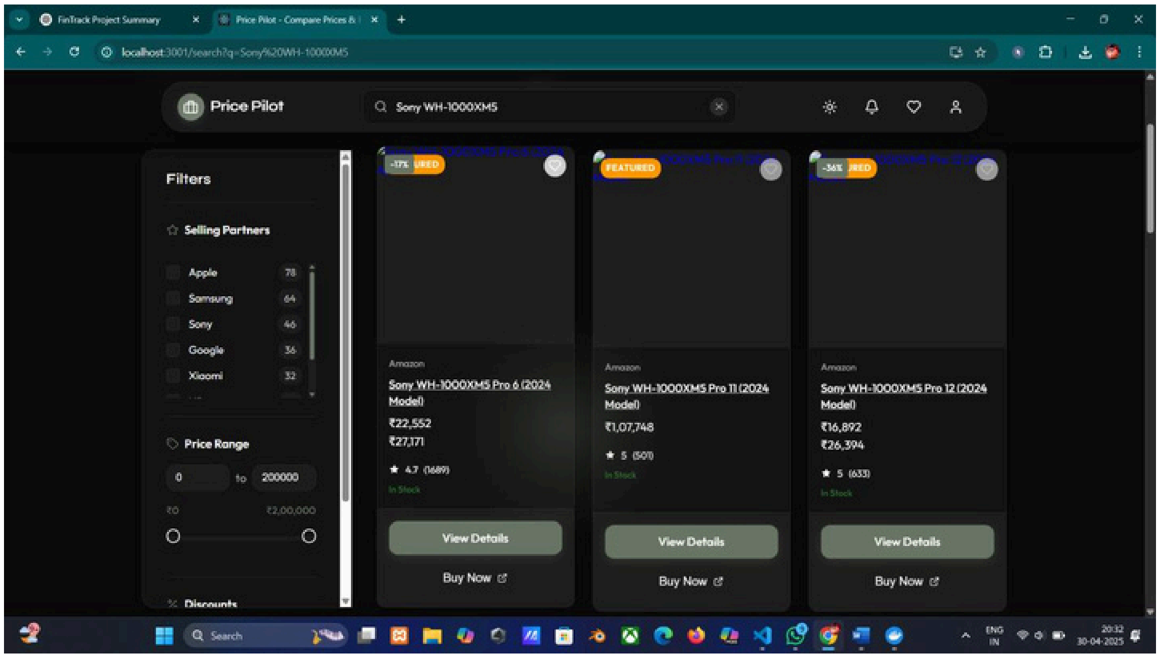
Price Pilot helps you track product prices across multiple retailers, notifies you of drops, and ensures you never overpay again. Save money effortlessly with our automated price tracking.

Price Drop Alert! Sony XM5 Headphones now ₹100 lower



Find the best deals across the web

Compare prices from hundreds of retailers and get alerts when prices drop



7.Conclusion

Swiftshop successfully demonstrates the power of microservices and modern DevOps tools to build a robust, real-time, and scalable web application. The use of Docker and Jenkins ensures smooth development workflows and consistent deployments, while the core functionality empowers users to make smarter shopping decisions by simplifying e-commerce comparisons. The project reflects effective integration of backend services, frontend UI, automation, and deployment strategies.

GitHub: https://github.com/vk28122812/SwiftShop/