DevOps CA 2 Report

Group Members:

Siddharth Sandeep - 22070122210

Shreyas Tambe - 22070122221

Rohit Raj - 22070122248

1. Project Overview

Project Name: PathoScanAI

GitHub Repo: https://github.com/shreyastambe103/PathoScanAI
Deployed at: https://pathoscanai-50rd.onrender.com/

Objective:

To implement a full DevOps pipeline for PathoScanAI, including:

- Automated deployment using CI/CD tools
- Configuration management and environment setup using Ansible
- Containerization and orchestration with Docker & Kubernetes
- Monitoring and logging with Prometheus (or Nagios)
- Reflection on challenges and lessons learned

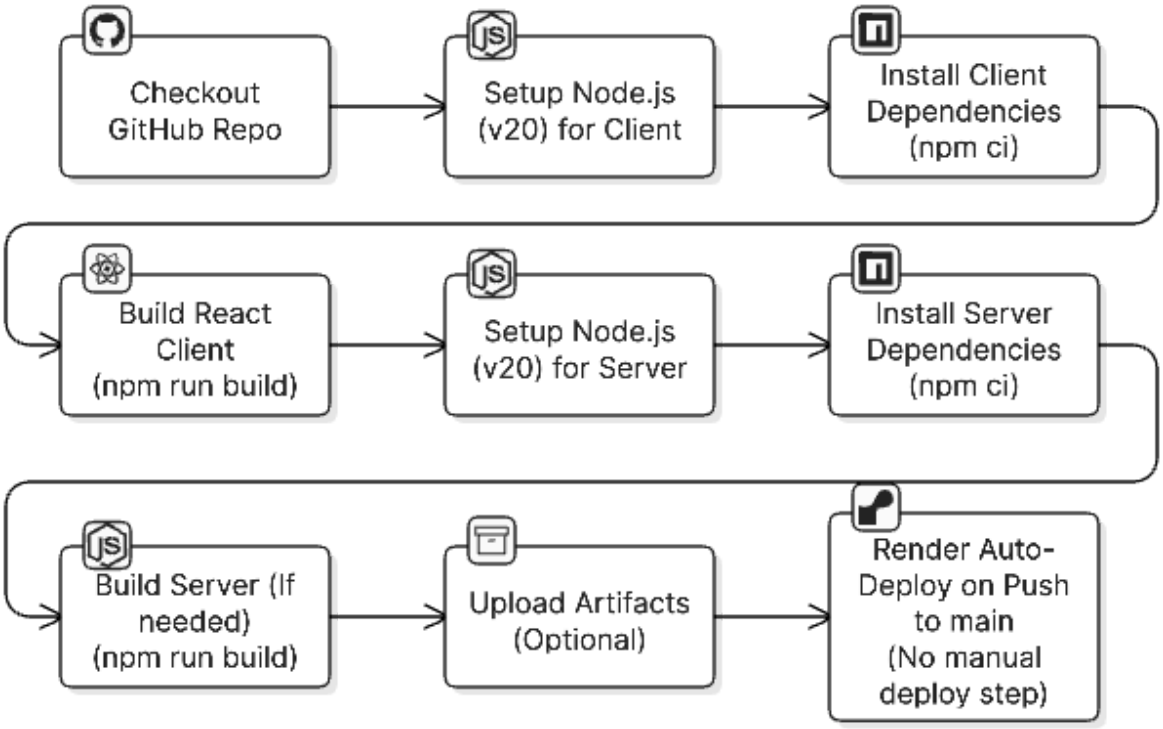
Scope:

- All steps were implemented locally using WSL2 / Docker / Windows binaries where necessary.
- The pipeline ensures reproducibility, automation, and observability of the PathoScanAI application.

Step 1 – Deployment Strategy

Tool Chosen: GitHub Actions

Pipeline Overview:



Pipeline Steps:

1. Trigger: Push or Pull Request to main branch
2. Checkout repository
3. Install dependencies (Node.js, npm)
4. Run tests (if any)
5. Deploy application to local environment / Docker container

Files Submitted:

- `deploy.yml` – GitHub Actions workflow file

Step 2 – Configuration Management (Ansible)

Objective: Automate environment setup for PathoScanAI

Flow:

WSL2 Ubuntu (Ansible Host) → Ansible Playbook → Localhost Environment

Tasks Performed:

- Installed Node.js, npm, git, build tools
- Cloned PathoScanAI repository
- Ran npm install and npm run build

Files Submitted:

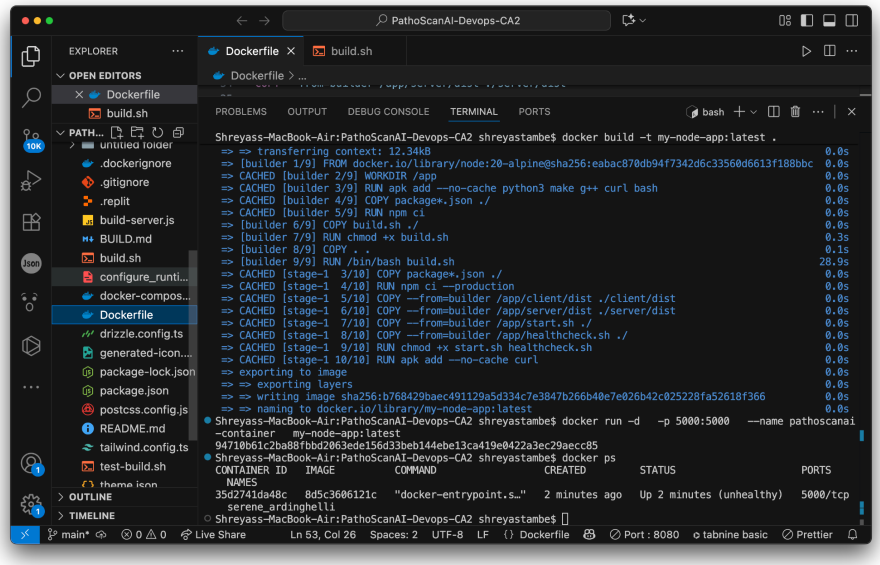
- `playbook.yml` – main playbook
- `inventory.ini` – localhost target

Outcome:
Environment fully configured automatically, no manual steps required

---

Step 3 – Containerization & Orchestration

Tools Used: Docker & Kubernetes



Tasks Performed:

- Dockerized PathoScanAI application
- Created Kubernetes manifests:
    - deployment.yaml
    - service.yaml

Outcome:
Application containerized and orchestrated with Kubernetes successfully

---

Step 4 – Monitoring & Logging

Tools Used: Prometheus + Node Exporter (or Nagios)

Architecture Diagram:

Tasks Performed:

- Installed Prometheus (Docker / Windows binaries)
- Installed Node Exporter for system metrics
- Configured Prometheus to scrape application and system metrics
- Created basic dashboard showing:
    - Uptime
    - Latency
    - Error rate

Files Submitted:

- prometheus.yml
- Node Exporter setup scripts

Outcome:
Application and system metrics visible in real-time, ensuring observability

---

Reflection

Challenges Faced:

- Learning curve for Ansible syntax & YAML
- Running Ansible on Windows localhost (not cloud VM)
- Docker Desktop issues / Prometheus setup
- Kubernetes manifests & rolling update commands
- Mapping metrics to meaningful dashboards

Lessons Learned:

- Automation ensures consistent and reproducible environment setup
- Containerization & orchestration simplify deployment and scaling
- Monitoring is essential for understanding app and system health
- Documentation and screenshots are crucial for reproducibility

Tools Mastered:

- GitHub Actions
- Ansible
- Docker & Kubernetes
- Prometheus + Node Exporter

Final Pipeline Architecture