ACEest Fitness - Final Report

Student: Uday Kumar Katipaga
Roll No: 2024ht66532

1. CI/CD Architecture Overview

The CI/CD pipeline uses:
- Version control: GitHub
- CI: Jenkins (Jenkinsfile included) and GitHub Actions (simple CI)
- Testing: Pytest
- Quality: SonarQube (config included)
- Containerization: Docker (Dockerfile included)
- Orchestration: Kubernetes (manifests included for rolling, blue-green, canary, shadow, A/B)

Build flow: Commit -> CI (tests + sonar) -> Docker build -> push to registry -> Deploy to K8s

2. Challenges & Mitigations

- Pytest import path errors: solved by setting PYTHONPATH in CI and adding __init__.py where needed.
- Line ending differences: handled via core.autocrlf in git config.
- Remote repo conflicts: initial force push recommended for first upload; avoid later unless necessary.
- SonarQube requires server endpoint/access: properties file prepared for Jenkins Sonar plugin.

3. Key Automation Outcomes

- Automated testing with Pytest integrated into CI.
- Docker build and image tagging automated in pipeline.
- Kubernetes manifests prepared for multiple deployment strategies enabling rollback and staged rollouts

4. How to run locally

- Install Python, pip, Docker, kubectl (for k8s), and optionally minikube.
- Create venv and install requirements: python -m venv venv; venv\Scripts\activate; pip install -r requirem
- Run tests: set PYTHONPATH (Windows PowerShell: $env:PYTHONPATH=(Get-Location).Path; Git CM
- Build Docker: docker build -t <dockerhub-user>/aceest-fitness:V1.0 .

5. Deliverables

- app.py, tests, Dockerfile, Jenkinsfile, SonarQube config, k8s manifests, README, this report.