Reflection on Development and Deployment Process

Project Name: Module 14: BREAD Functionality for Calculations

Date: 8/11/2025

Author: Jaydeep Raval

1. Crucial Experiences

- Learned how to use modular code and appropriate endpoints to organize a FastAPI application.
- CRUD/BREAD operations were implemented for authentication-based computations.
- To guarantee consistency across local and production environments, set up Docker for containerization.
- GitHub Actions CI/CD pipeline was integrated to automate Docker image publishing, security scanning (Trivy), and testing.

2. Difficulties with Security Vulnerabilities: At first, Trivy scans were unsuccessful because of serious flaws in the basis image. In order to lower the number of vulnerabilities, I had to look at updated base images and modify dependencies.

Misconfiguration of GitHub Secrets: The deployment failed because the secret names were incorrect (DOCKER_USERNAME rather than DOCKERHUB_USERNAME). The problem was resolved by fixing these and use an access token.

Dependency Compatibility: Careful pinning and additional appropriate packages were needed because several dependencies failed on thin base images.

3. The Approach to Problem-Solving

- Build and deployment errors were extensively traced using GitHub Action logs.
- Modified the Python package dependencies and base image after reviewing Trivy vulnerability findings.
- Configuration of verified secrets was done directly in the GitHub repository settings.
- Builds should be incrementally tested locally before being pushed to CI.

4. Knowledge Acquired

- Always verify environment variables and secret names twice.
- Instead of addressing security scanning at the last stage of deployment, it should be done early.
- Early modular code writing facilitates feature addition and debugging.
- Continuous integration serves as a security and deployment measure in addition to being a testing tool.

5. Upcoming Enhancements

- Automate vulnerability patching with Dependabot or Renovate.
- Add more extensive integration and end-to-end tests.
- Optimize Docker image size using multi-stage builds.