Vehicle Diagnostics Project Deployment Process

# 1. Pre-Deployment Checklist

- Ensure all code is merged into the main branch.  
- All tests (unit, integration, contract) must pass.  
- Static code analysis (e.g., SonarQube) reports no critical issues.  
- Environment variables and secrets are securely configured.  
- Tag the build version for deployment.

# 2. Build Phase

- Compile and package each Spring Boot microservice using Maven or Gradle.  
- Build Docker images for all microservices.  
- Push Docker images to a container registry (e.g., DockerHub, AWS ECR).

# 3. CI/CD Pipeline

- Automate build, test, Dockerization, and deployment processes using Jenkins/GitHub Actions.  
- Deploy first to Dev environment and run integration tests.  
- Promote to UAT/Stage environment.  
- Deploy to Production using canary, rolling, or blue-green strategy.

# 4. Deployment Target

- Local Development: Docker Compose  
- Dev/UAT/Prod: Kubernetes (EKS/GKE/AKS) or AWS ECS

# 5. Deployment to Kubernetes

- Use kubectl or Helm to deploy services.  
- Example: kubectl apply -f auth-service-deployment.yaml

# 6. Environment Configuration

- Use Spring profiles for environment-specific configs.  
- Use Config Server, environment variables, or secret mounts for secure configurations.

# 7. Monitoring & Logging

- Integrate Prometheus and Grafana for monitoring.  
- Use ELK or EFK stack for centralized logging.  
- Enable distributed tracing (e.g., Zipkin, Jaeger).

# 8. Rollback Strategy

- Maintain previous stable image tags for rollback.  
- Use Helm or Kubernetes rollback commands.  
- Monitor deployments and trigger rollback on failure.

# Deployment Process Flowchart

