DEPLOYMENT STRATEGY & PLAN

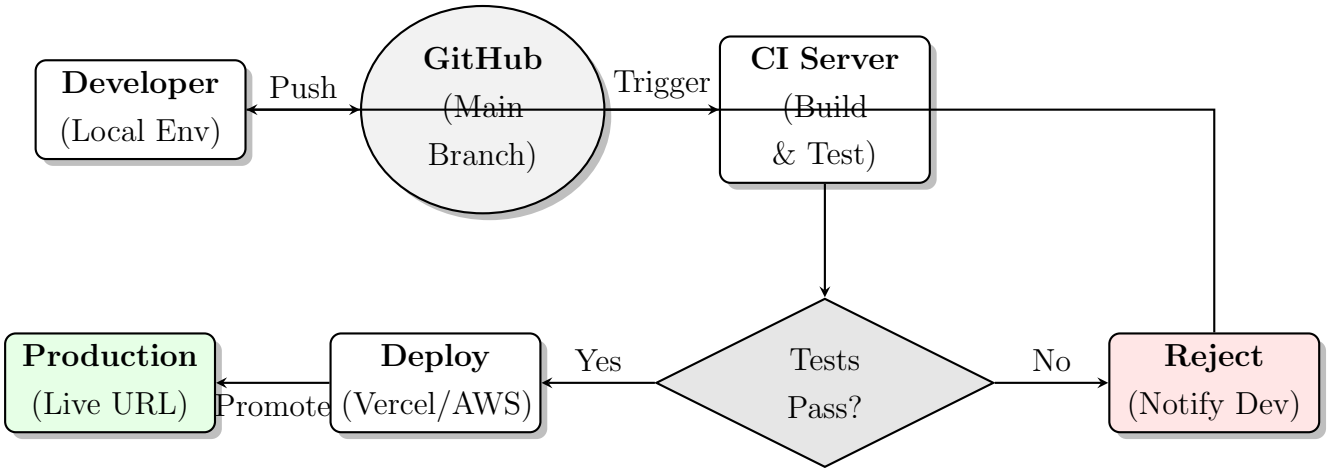
StockFlow: Enterprise Inventory Management System

1 1. Deployment Overview

The deployment strategy for "StockFlow" adheres to modern **DevOps** principles, utilizing a **Continuous Integration and Continuous Deployment (CI/CD)** pipeline. The system is designed to be hosted on serverless infrastructure to ensure high availability, automatic scaling, and zero-downtime updates.

2 2. Deployment Pipeline Diagram

The following diagram illustrates the automated workflow from code commit to production release.



3 3. Environmental Prerequisites

Before initiating the deployment, the following environment variables and dependencies must be configured:

- **Runtime:** Node.js v18.x (LTS) or higher.
- **Package Manager:** npm v9.x or Yarn v1.22+.
- **Environment Variables (.env.production):**

- `REACT_APP_API_URL`: (Optional for backend integration)
- `REACT_APP_VERSION`: v1.0.0-release

4 4. Detailed Execution Phases

4.1 Phase 1: Pre-Deployment (Local Staging)

This phase ensures code quality before it reaches the remote repository.

Actions

- **Linting:** Execute `npm run lint` to identify syntax errors and anti-patterns.
- **Unit Testing:** Run `npm run test` to verify that all critical logic (Calculation, Validation) passes standard test cases.
- **Build Check:** Run `npm run build` locally to ensure no compilation errors occur.

4.2 Phase 2: Build Pipeline (CI)

Triggered automatically upon pushing code to the `main` branch.

Automated Steps

- **Dependency Resolution:** `npm ci` (Clean Install) to install exact dependencies from `package-lock.json`.
- **Optimization:** Webpack bundles the application, performs tree-shaking (removing unused code), and minifies CSS/JS files for performance.
- **Asset Generation:** Static assets (images, icons) are optimized and hashed for caching.

4.3 Phase 3: Production Release (CD)

Hosting & Delivery

- **Platform:** Vercel (preferred for React) or Netlify.
- **Atomic Deployment:** The new build is deployed to a preview URL first. Once health checks pass, the main domain pointer is switched to the new build instantly (Zero Downtime).
- **CDN Propagation:** Static assets are distributed globally via Edge Networks to ensure low latency.

5 5. Rollback Strategy

In the event of a critical failure post-deployment (e.g., white screen of death, critical data bug):

1. **Immediate Revert:** Utilizing the hosting platform's dashboard, instantly revert the "Production" domain to point to the **previous successful deployment hash**.
2. **Hotfix Branch:** Create a `hotfix` branch from `main`, resolve the bug, and push through the full CI/CD pipeline again.

6 6. Post-Deployment Verification

- **Smoke Test:** Verify that the login page loads and the dashboard renders data.
- **Lighthouse Audit:** Run a performance audit to ensure the production build meets Web Vitals scores (LCP $< 2.5s$, CLS < 0.1).