# Maintenance Plan for Inventory Automation Bot

## 1. Patch and Release Strategy

Releases follow semantic versioning and are tagged in GitHub using annotated tags (e.g., v1.2.0). GitHub Actions handles CI/CD, running tests, building containers, and pushing to the registry. Each release is tracked with release notes and rollback capability.

## 2. Dependency Management

Dependencies are managed via pip-tools (requirements.in → requirements.txt). Dependabot is enabled to scan for outdated or vulnerable packages and propose automated pull requests.

## 3. Scaling Strategy

- 5× Scale: Increase batch size, enable multithreading.  
- 10× Scale: Use multiprocessing or containerized workers.  
- 100× Scale: Deploy horizontally using Kubernetes pods or container orchestration.  
Load metrics and logs are used to trigger horizontal scaling thresholds.

## 4. Recovery and Failover

All failed operations are retried up to 3 times with exponential backoff. Persistent failures are logged to a dead-letter queue (DLQ) JSON file. Scripts monitor the DLQ and send alerts if the size exceeds thresholds.

## 5. Python Examples

Retries are implemented with try/except blocks and time.sleep for backoff. CI/CD flows use GitHub Actions YAMLs to build/test/deploy automatically.