Mid-Term Practical: CI Pipeline with GitHub Actions and Jenkins

Student Name: Nikhil Shankar C S **Course:** PROG8860 - CI/CD
Assignment: Mid-Term Practical

What I'm Building

This project demonstrates a complete CI/CD pipeline for an Android counter application. The goal is to automate the build, test, and deployment process using both GitHub Actions and Jenkins. The final output is a Docker image that serves test reports and lint analysis through a web interface.

Project Structure

```
PROG8860-CICD/
├── Android/                          # Main Android application
│   ├── app/
│   │   └── src/
│   │       ├── main/
│   │       │   └── java/rebirth/nixaclabs/cicd/
│   │       │       ├── Counter.kt           # Counter logic
│   │       │       └── MainActivity.kt      # UI with Compose
│   │       └── test/
│   │           └── java/rebirth/nixaclabs/cicd/
│   │               └── CounterTest.kt       # Unit tests
│   ├── Dockerfile                    # Multi-stage Docker build
│   └── build.gradle
├── .github/
│   └── workflows/
│       └── android-ci.yml            # GitHub Actions pipeline
├── Jenkinsfile                       # Jenkins pipeline
└── README.md

Separate Repository (AndroidReports):
├── server.py                         # Python HTTP server for reports
└── README.md
```

Application Details

What the App Does

Simple counter application with basic functionality:

- Display shows a number (starts at 0)

- "+1" button increases the count
- "-1" button decreases the count
- Built with Kotlin and Jetpack Compose

Test Cases (6 total)

1. Initial counter value is 0
2. Increment increases count by 1
3. Decrement decreases count by 1
4. Multiple increments work correctly
5. Multiple decrements work correctly
6. Increment and decrement together work properly

How to Build and Run

Local Build

```
cd Android
./gradlew assembleDebug
```

APK location: `Android/app/build/outputs/apk/debug/app-debug.apk`

Run Tests

```
cd Android
./gradlew test
```

Test reports generated at: `Android/app/build/reports/tests/testDebugUnitTest/index.html`

Run Lint

```
cd Android
./gradlew lint
```

Lint report generated at: `Android/app/build/reports/lint-results-debug.html`

Docker Setup

Multi-Stage Build

The Dockerfile uses three stages:

Stage 1 - Builder:

- Installs Android SDK and build tools
- Builds the APK
- Runs unit tests and generates HTML report
- Runs lint analysis and generates HTML report

Stage 2 - Server Repository:

- Uses alpine/git image
- Created a gitrepo for serving reports using python and is available in the below repository.
- Clones the report server from GitHub (https://github.com/NikhilShankar/AndroidReports.git)
- Provides server.py for the final stage

Stage 3 - Report Server:

- Uses lightweight Python base image
- Copies test and lint reports from Stage 1
- Copies server.py from Stage 2
- Runs HTTP server on port 9898
- Final image is small (no Android SDK included)

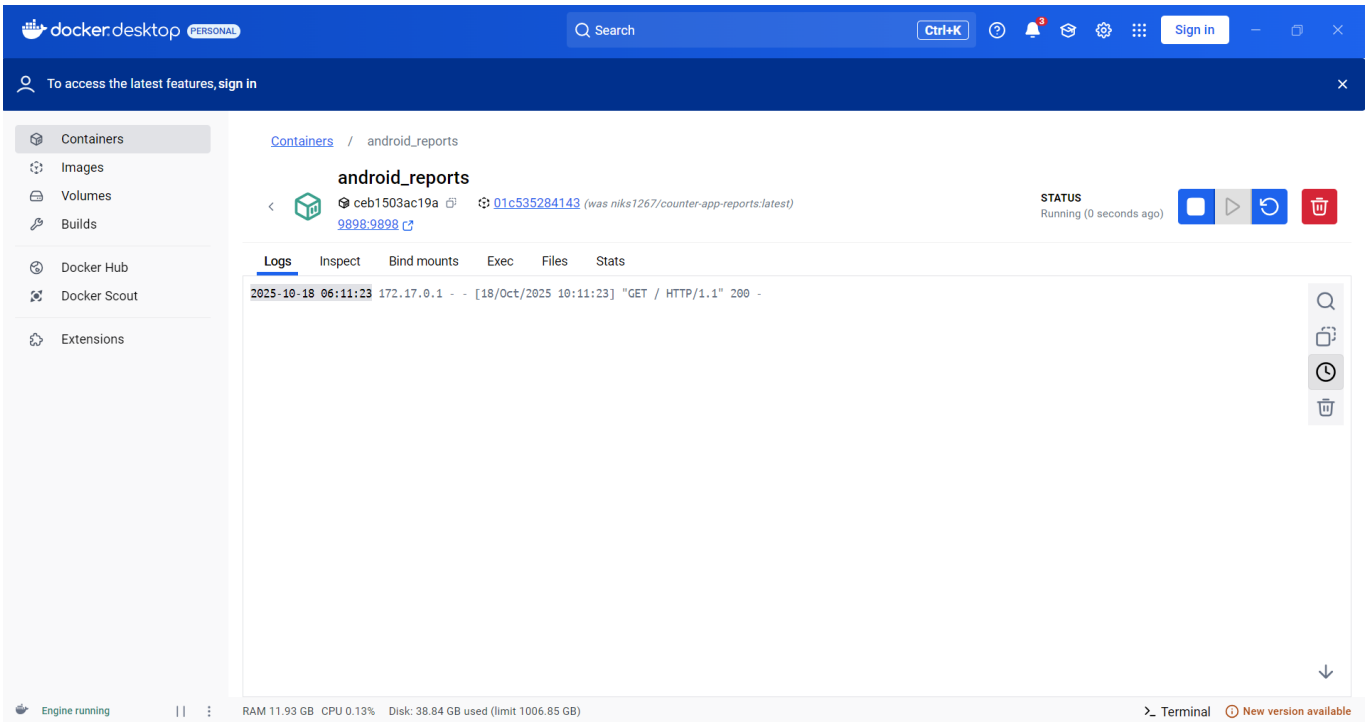
Pull and Run the Image

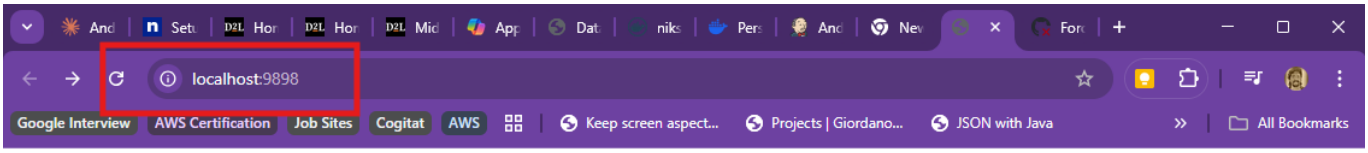
```
docker pull niks1267/counter-app-reports:latest
docker run -p 9898:9898 niks1267/counter-app-reports:latest
```

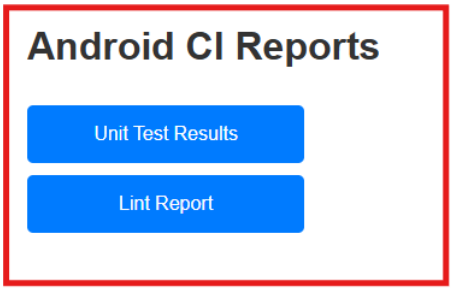
Access Reports

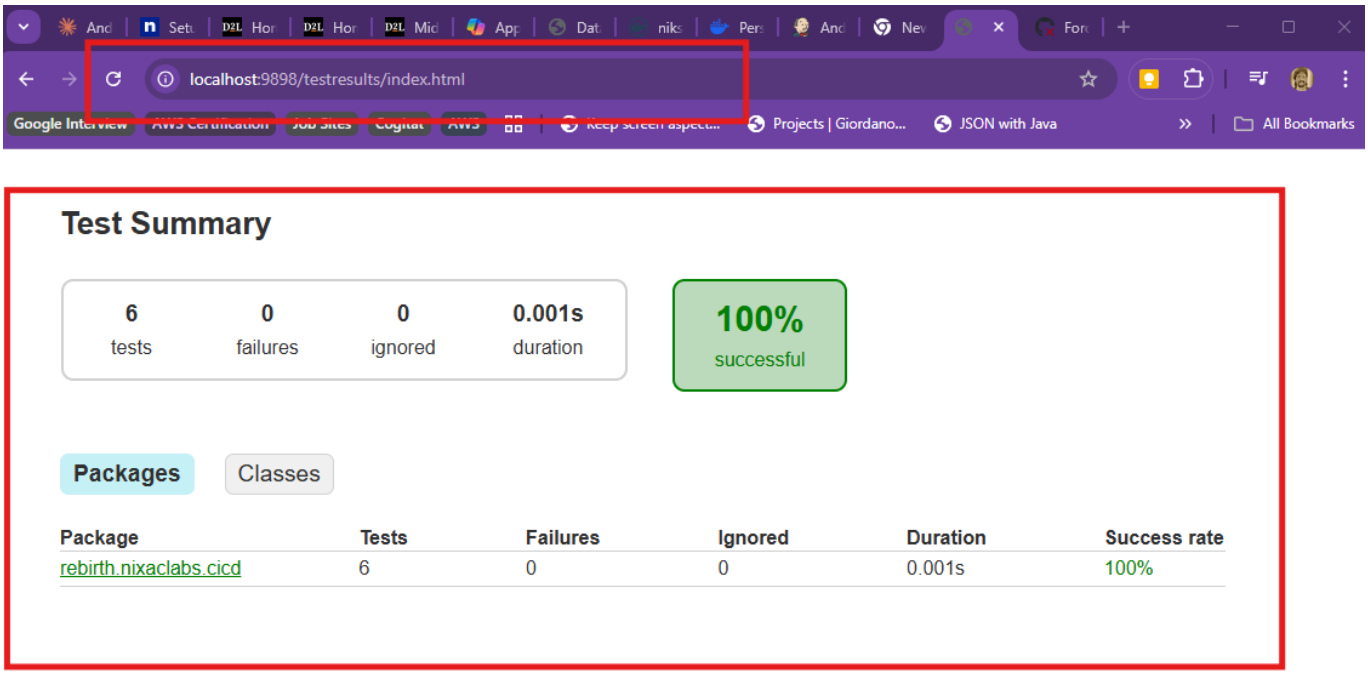
Open your browser:

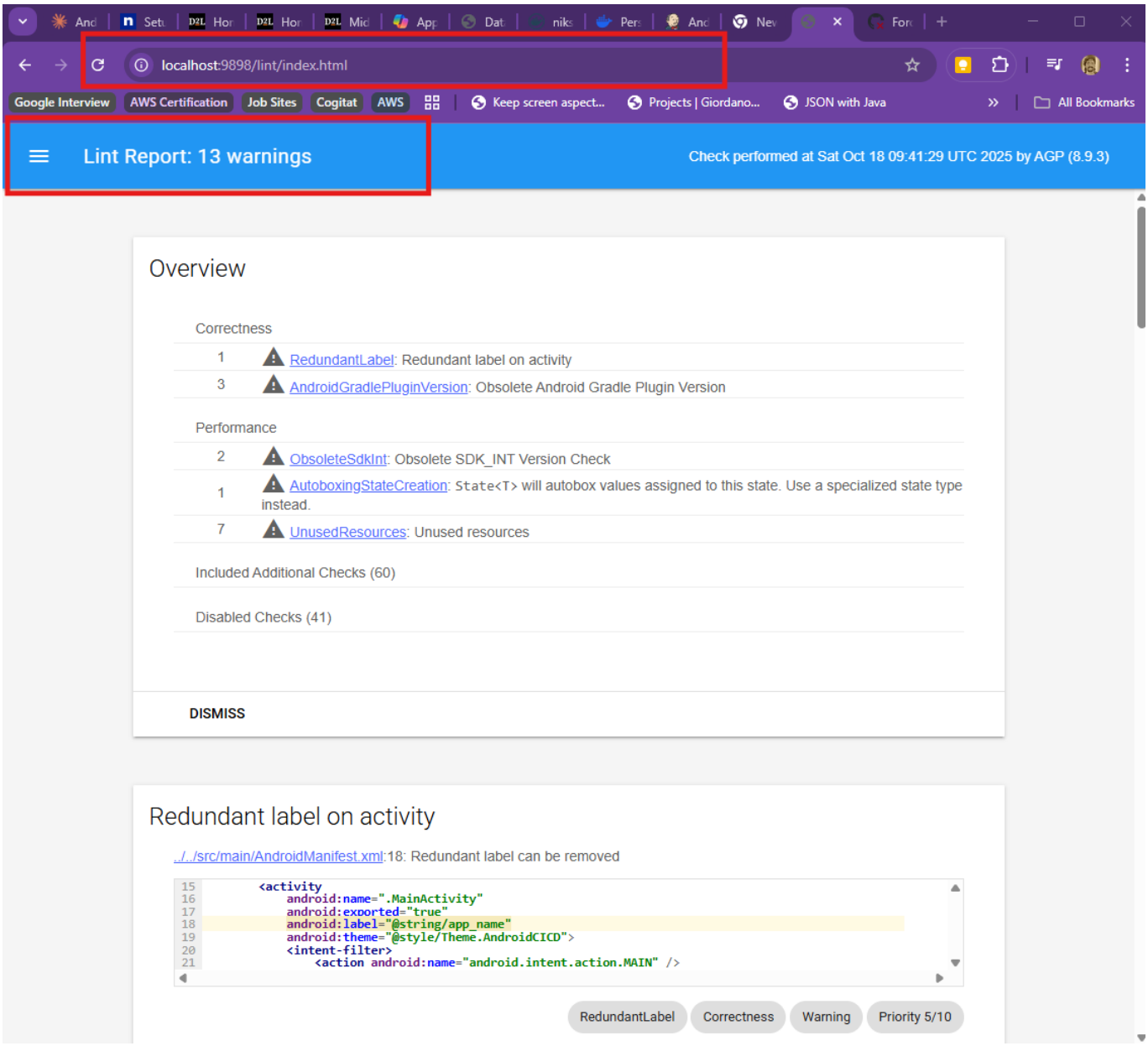
- **Home:** http://localhost:9898
- **Test Results:** http://localhost:9898/testresults/index.html
- **Lint Report:** http://localhost:9898/lint/index.html


docker desktop PERSONAL
Search Ctrl+K
Sign in
To access the latest features, sign in
Containers
Images
Volumes
Builds
Docker Hub
Docker Scout
Extensions
Containers / android_reports
android_reports
ceb1503ac19a
01c535284143 (was niks1267/counter-app-reports:latest)
9898:9898
STATUS
Running (0 seconds ago)
Logs Inspect Bind mounts Exec Files Stats
2025-10-18 06:11:23 172.17.0.1 - - [18/Oct/2025 10:11:23] "GET / HTTP/1.1" 200 -
Engine running
RAM 11.93 GB CPU 0.13% Disk: 38.84 GB used (limit 1006.85 GB)
Terminal
New version available


localhost:9898
Google Interview
AWS Certification
Job Sites
Cogitat
AWS
Keep screen aspect...
Projects | Giordano...
JSON with Java
All Bookmarks


Android CI Reports
Unit Test Results
Lint Report


localhost:9898/testresults/index.html
Projects | Giordano...
JSON with Java
All Bookmarks
Test Summary
6
tests
0
failures
0
ignored
0.001s
duration
100%
successful
Packages
Classes
Package
Tests
Failures
Ignored
Duration
Success rate
rebirth.nixaclabs.cicd
6
0
0
0.001s
100%
Generated by Gradle 8.11.1 at Oct 18, 2025, 9:40:44 AM



GitHub Actions CI Pipeline

Pipeline Stages

The workflow file is located at `.github/workflows/android-ci.yml`

What it does:

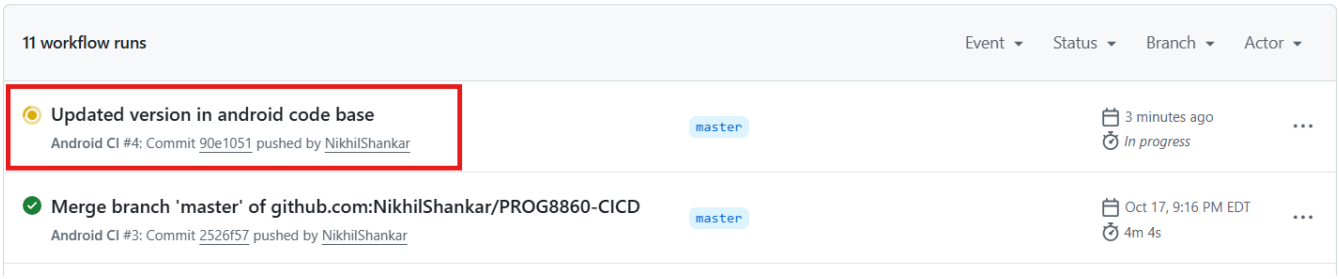
- Triggers on push to Android folder
- Builds the multi-stage Docker image
- Pushes image to Docker Hub
- All build, test, and lint steps happen inside Docker

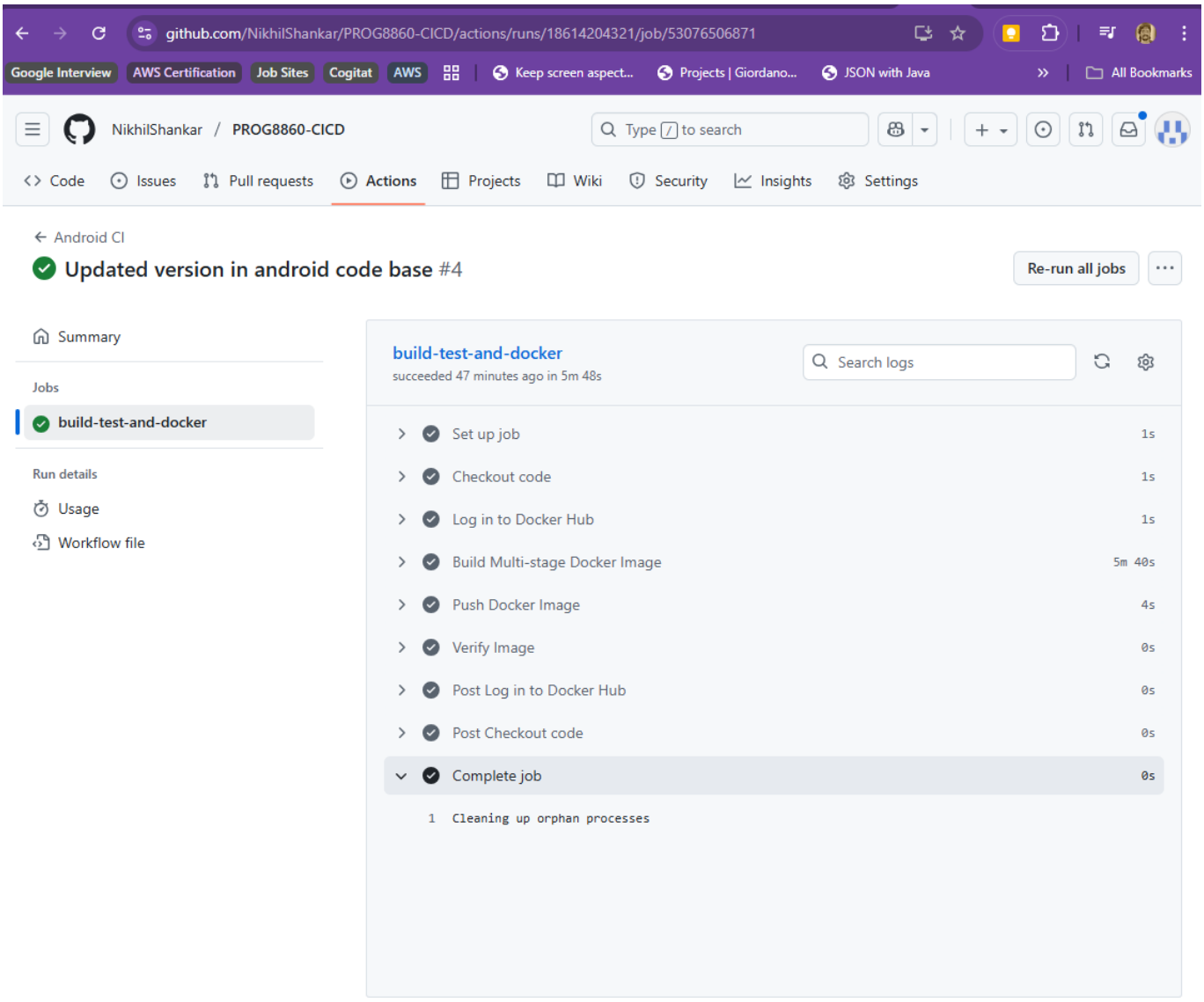
How to Test the Pipeline

Push any change to the Android folder:

```
git add .
git commit -m "Update Android app"
git push
```

GitHub Actions automatically runs the pipeline.





Demonstrating Success and Failure

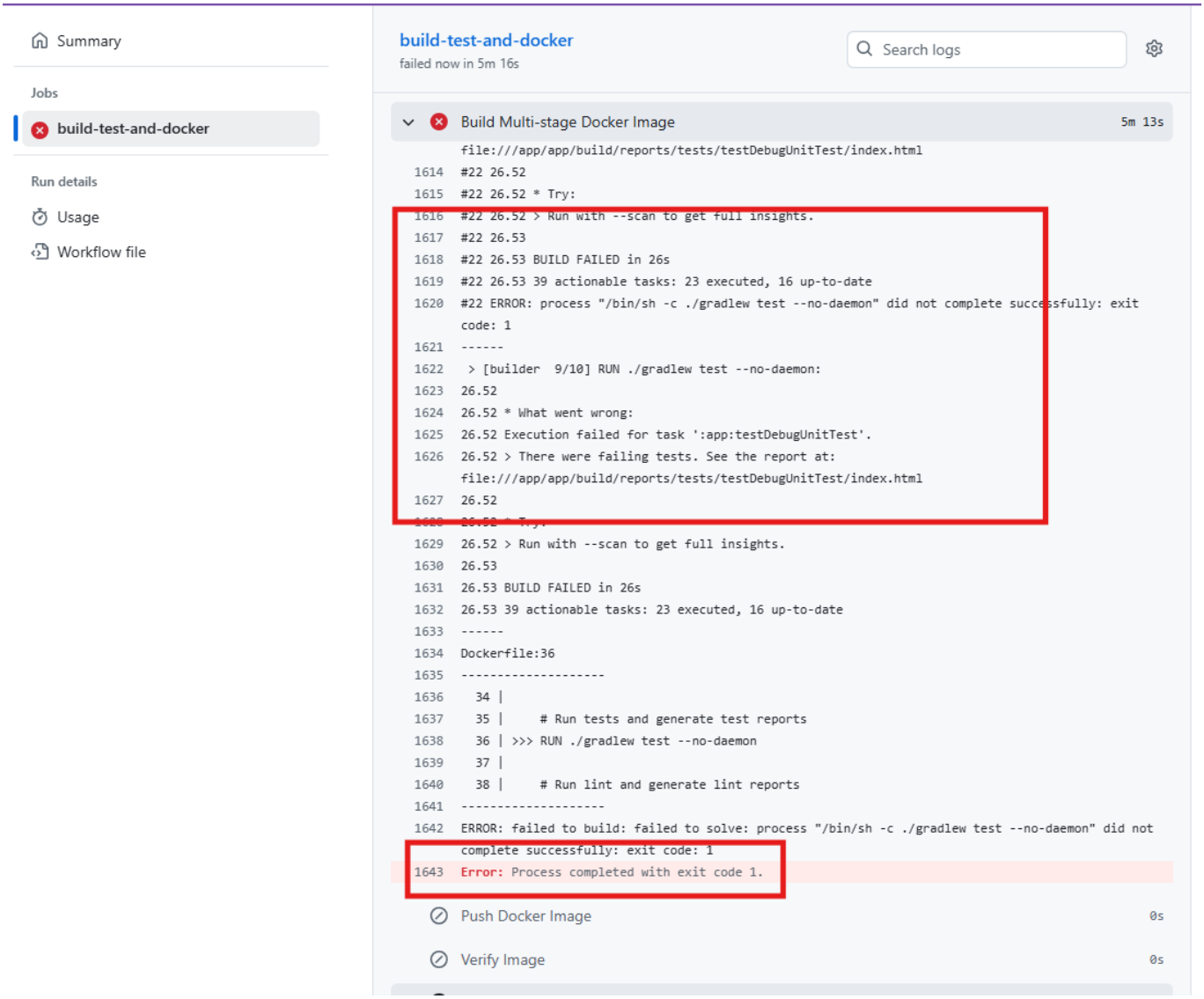
Success scenario: All tests pass (current state)

Failure scenario:

1. Open `Android/app/src/test/java/rebirth/nixaclabs/cicd/CounterTest.kt`
2. Change line in `testInitialCountIsZero()`:

```
assertEquals(1, counter.getCount())  // Change 0 to 1
```

3. Push the change
4. Pipeline fails because test fails
5. Revert the change
6. Pipeline passes again



Jenkins Pipeline

Setup

Jenkins runs in Docker:

```
docker run -d \
  --name jenkins \
```

```
  -p 8080:8080 \
  -p 50000:50000 \
  -v jenkins_home:/var/jenkins_home \
  -v /var/run/docker.sock:/var/run/docker.sock \
  jenkins/jenkins:lts
```

Pipeline Configuration

The Jenkinsfile is located at the root of the repository.

Pipeline stages:

1. **Checkout:** Gets code from GitHub
2. **Build & Test:** Builds Docker image (includes tests and lint)
3. **Push to Docker Hub:** Publishes image to registry

Webhook Setup

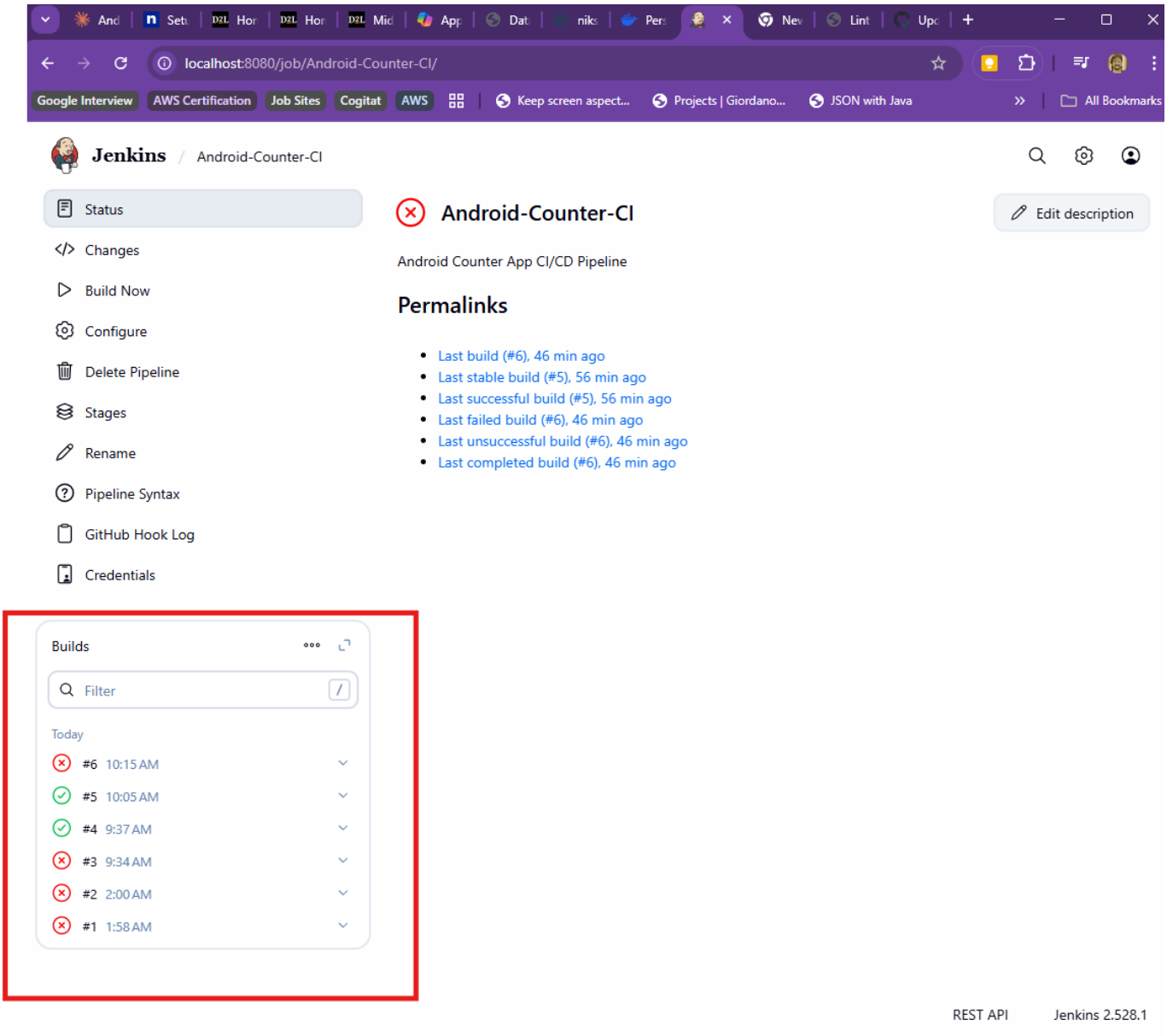
Jenkins is triggered automatically when code is pushed to GitHub.

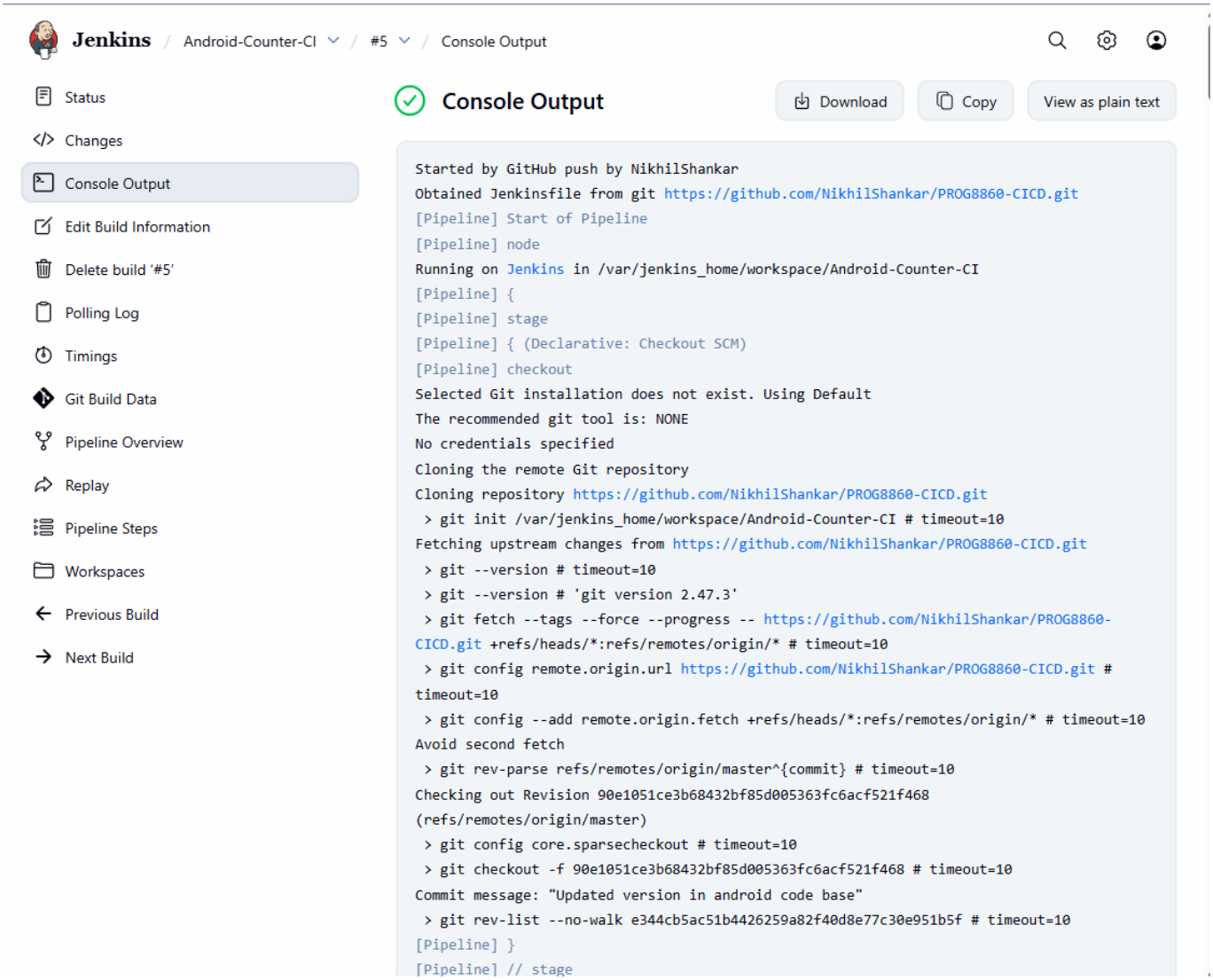
Using ngrok for local Jenkins:

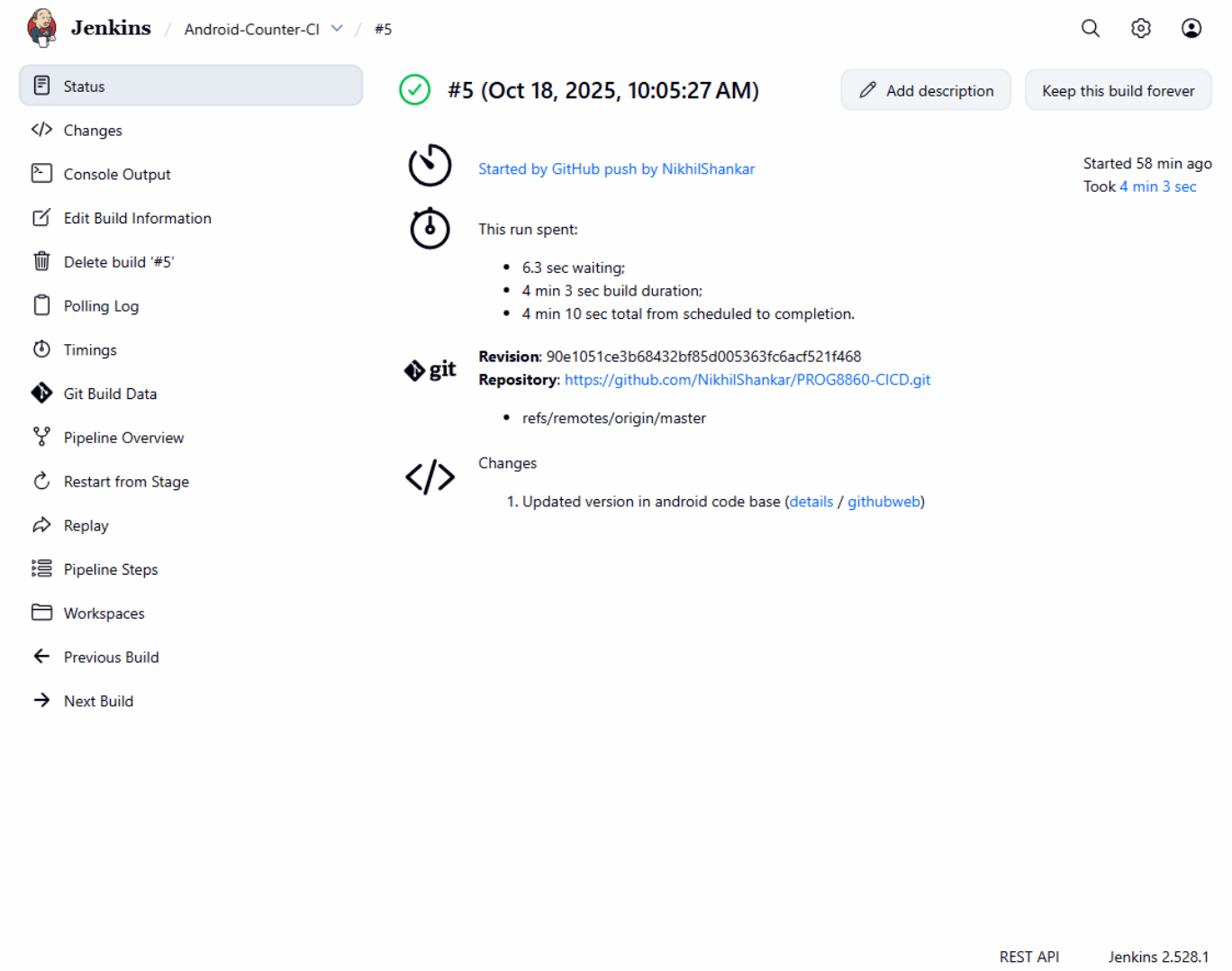
1. Install ngrok: https://ngrok.com/download
2. Run: `ngrok http 8080`
3. Add webhook to GitHub: `https://your-ngrok-url/github-webhook/`
4. Configure Jenkins job to use "GitHub hook trigger for GITScm polling"

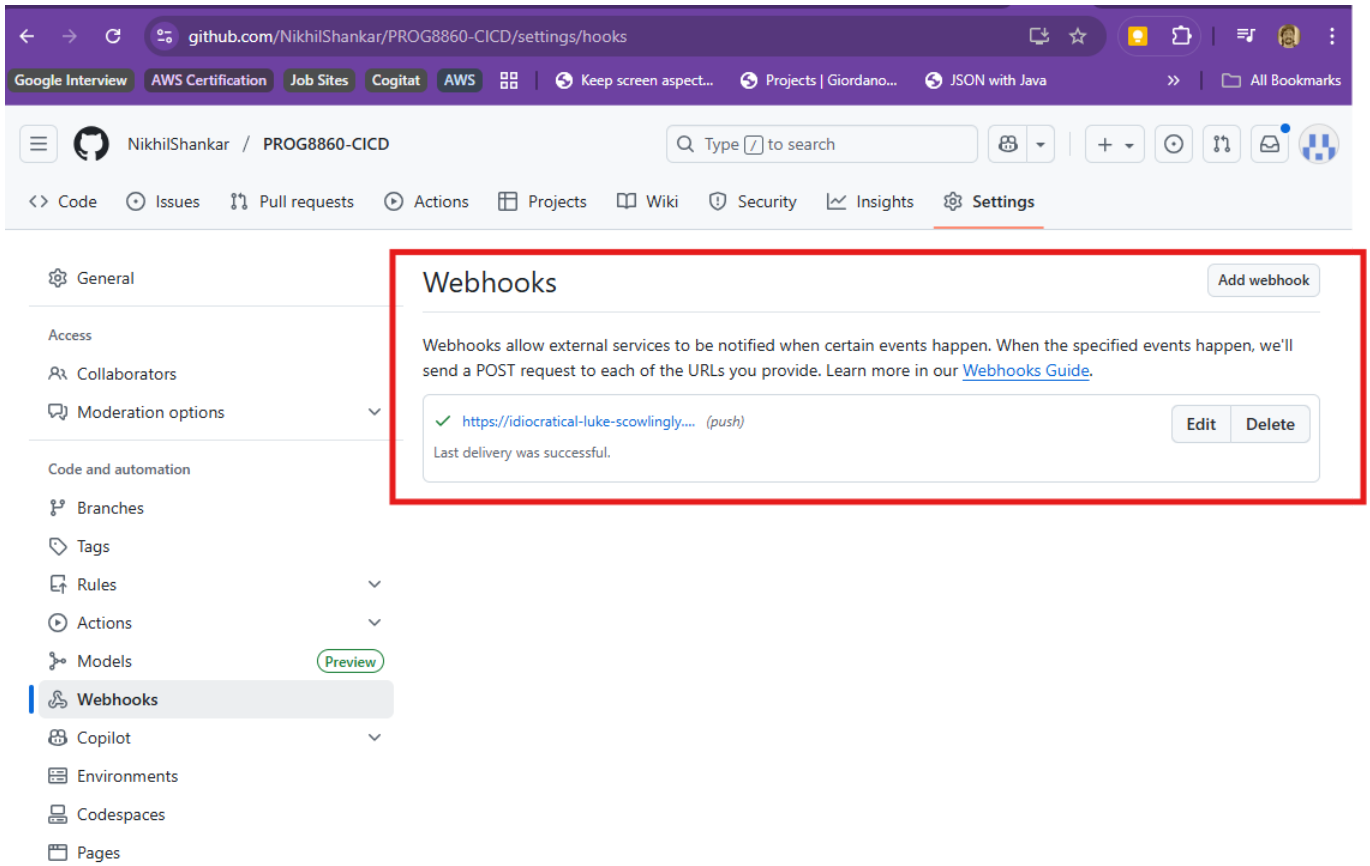
How to Run Jenkins Pipeline

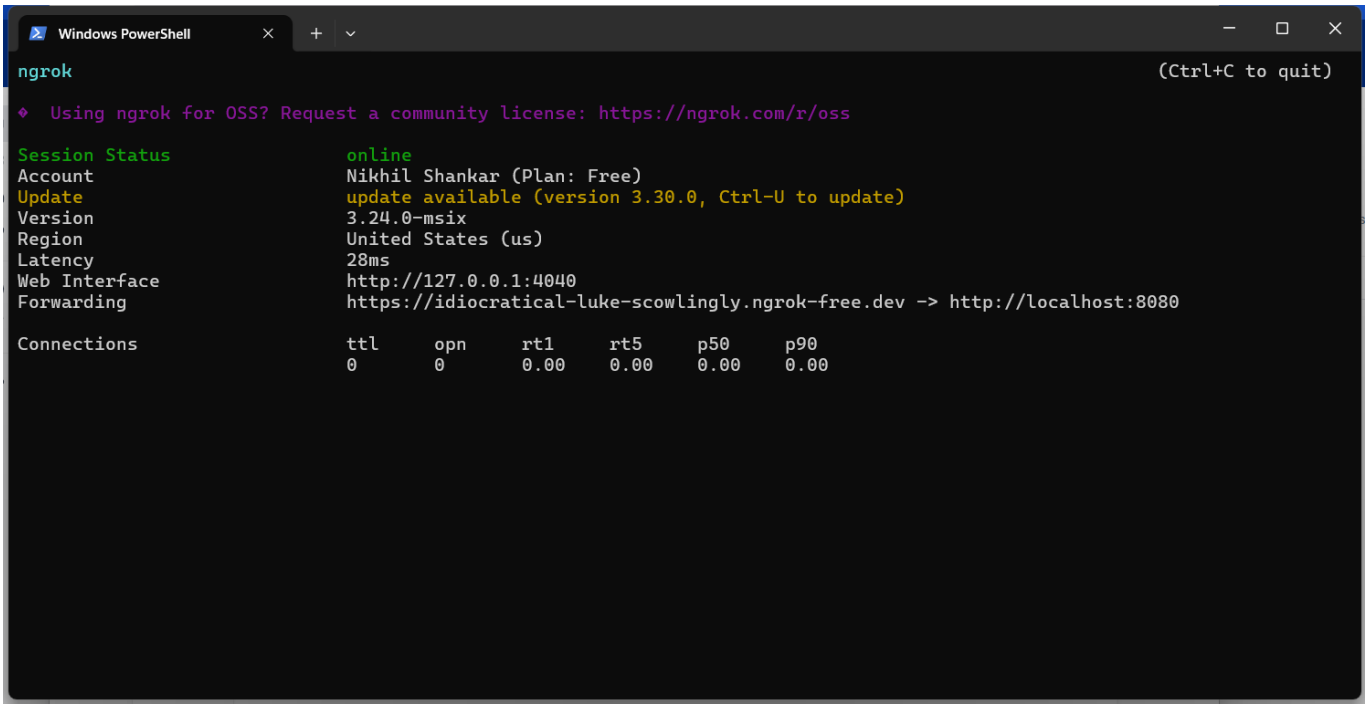
1. Access Jenkins: http://localhost:8080
2. Open "Android-Counter-CI" job
3. Click "Build Now"
4. Or push code to GitHub (webhook triggers automatically)


localhost:8080/job/Android-Counter-CI/
Google Interview
AWS Certification
Job Sites
Cogitat
AWS
Keep screen aspect...
Projects | Giordano...
JSON with Java
All Bookmarks
Jenkins
Android-Counter-CI
Status
Changes
Build Now
Configure
Delete Pipeline
Stages
Rename
Pipeline Syntax
GitHub Hook Log
Credentials
Android-Counter-CI
Edit description
Android Counter App CI/CD Pipeline
Permalinks
Last build (#6), 46 min ago
Last stable build (#5), 56 min ago
Last successful build (#5), 56 min ago
Last failed build (#6), 46 min ago
Last unsuccessful build (#6), 46 min ago
Last completed build (#6), 46 min ago
Builds
Filter
Today
#6 10:15 AM
#5 10:05 AM
#4 9:37 AM
#3 9:34 AM
#2 2:00 AM
#1 1:58 AM
REST API
Jenkins 2.528.1


Jenkins / Android-Counter-CI / #5 / Console Output
Status
Changes
Console Output
Edit Build Information
Delete build '#5'
Polling Log
Timings
Git Build Data
Pipeline Overview
Replay
Pipeline Steps
Workspaces
Previous Build
Next Build
Console Output
Download
Copy
View as plain text
Started by GitHub push by NikhilShankar
Obtained Jenkinsfile from git https://github.com/NikhilShankar/PROG8860-CICD.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/Android-Counter-CI
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/NikhilShankar/PROG8860-CICD.git
> git init /var/jenkins_home/workspace/Android-Counter-CI # timeout=10
Fetching upstream changes from https://github.com/NikhilShankar/PROG8860-CICD.git
> git --version # timeout=10
> git --version # 'git version 2.47.3'
> git fetch --tags --force --progress -- https://github.com/NikhilShankar/PROG8860-CICD.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/NikhilShankar/PROG8860-CICD.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 90e1051ce3b68432bf85d005363fc6acf521f468 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 90e1051ce3b68432bf85d005363fc6acf521f468 # timeout=10
Commit message: "Updated version in android code base"
> git rev-list --no-walk e344cb5ac51b4426259a82f40d8e77c30e951b5f # timeout=10
[Pipeline] }
[Pipeline] // stage


Jenkins / Android-Counter-CI / #5
Status
Changes
Console Output
Edit Build Information
Delete build '#5'
Polling Log
Timings
Git Build Data
Pipeline Overview
Restart from Stage
Replay
Pipeline Steps
Workspaces
Previous Build
Next Build
#5 (Oct 18, 2025, 10:05:27 AM)
Add description
Keep this build forever
Started by GitHub push by NikhilShankar
Started 58 min ago
Took 4 min 3 sec
This run spent:
6.3 sec waiting;
4 min 3 sec build duration;
4 min 10 sec total from scheduled to completion.
Revision: 90e1051ce3b68432bf85d005363fc6acf521f468
Repository: https://github.com/NikhilShankar/PROG8860-CICD.git
refs/remotes/origin/master
Changes
1. Updated version in android code base (details / githubweb)
REST API
Jenkins 2.528.1


github.com/NikhilShankar/PROG8860-CICD/settings/hooks
Google Interview
AWS Certification
Job Sites
Cogitat
AWS
Keep screen aspect...
Projects | Giordano...
JSON with Java
All Bookmarks
NikhilShankar / PROG8860-CICD
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Models
Preview
Webhooks
Copilot
Environments
Codespaces
Pages
Webhooks
Add webhook
Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our Webhooks Guide.
https://idiocratical-luke-scowlingly.... (push)
Last delivery was successful.
Edit
Delete



Technologies Used

- **Language:** Kotlin
- **UI Framework:** Jetpack Compose
- **Build Tool:** Gradle
- **Testing:** JUnit
- **CI/CD:** GitHub Actions, Jenkins
- **Containerization:** Docker (multi-stage builds)
- **Report Server:** Python HTTP Server
- **Registry:** Docker Hub
- **Webhook Tunneling:** ngrok

Key Learnings

Multi-Stage Docker Builds

- Separates build environment from runtime environment
- Reduces final image size significantly
- Clones external repositories during build

CI/CD Automation

- Automated testing catches errors early
- Docker ensures consistent build environment
- Webhooks enable automatic pipeline triggers

Pipeline Design

- Build and test stages run independently
- Reports are generated and served automatically

- Same pipeline works in both GitHub Actions and Jenkins

Repository Links

- **Main Repository:** https://github.com/NikhilShankar/PROG8860-CICD
- **Report Server Repository:** https://github.com/NikhilShankar/AndroidReports
- **Docker Hub Image:** https://hub.docker.com/r/niks1267/counter-app-reports

---

Conclusion

This project demonstrates a complete CI/CD pipeline with automated testing, Docker containerization, and deployment to a registry. The multi-stage build approach and automated webhooks show real-world CI/CD practices. Both GitHub Actions and Jenkins pipelines successfully build, test, and deploy the application.