# Building a complete server deployment pipeline in Swift

Moritz

# About me

Based in **Germany**

Working at **arconsis** IT-Solutions GmbH

Biggest Problem in my life:

**Too Many** Side Projects,

**Not Enough** Brain:

**Automation as My Second Memory**

# About the Server-App

## simple Todo app

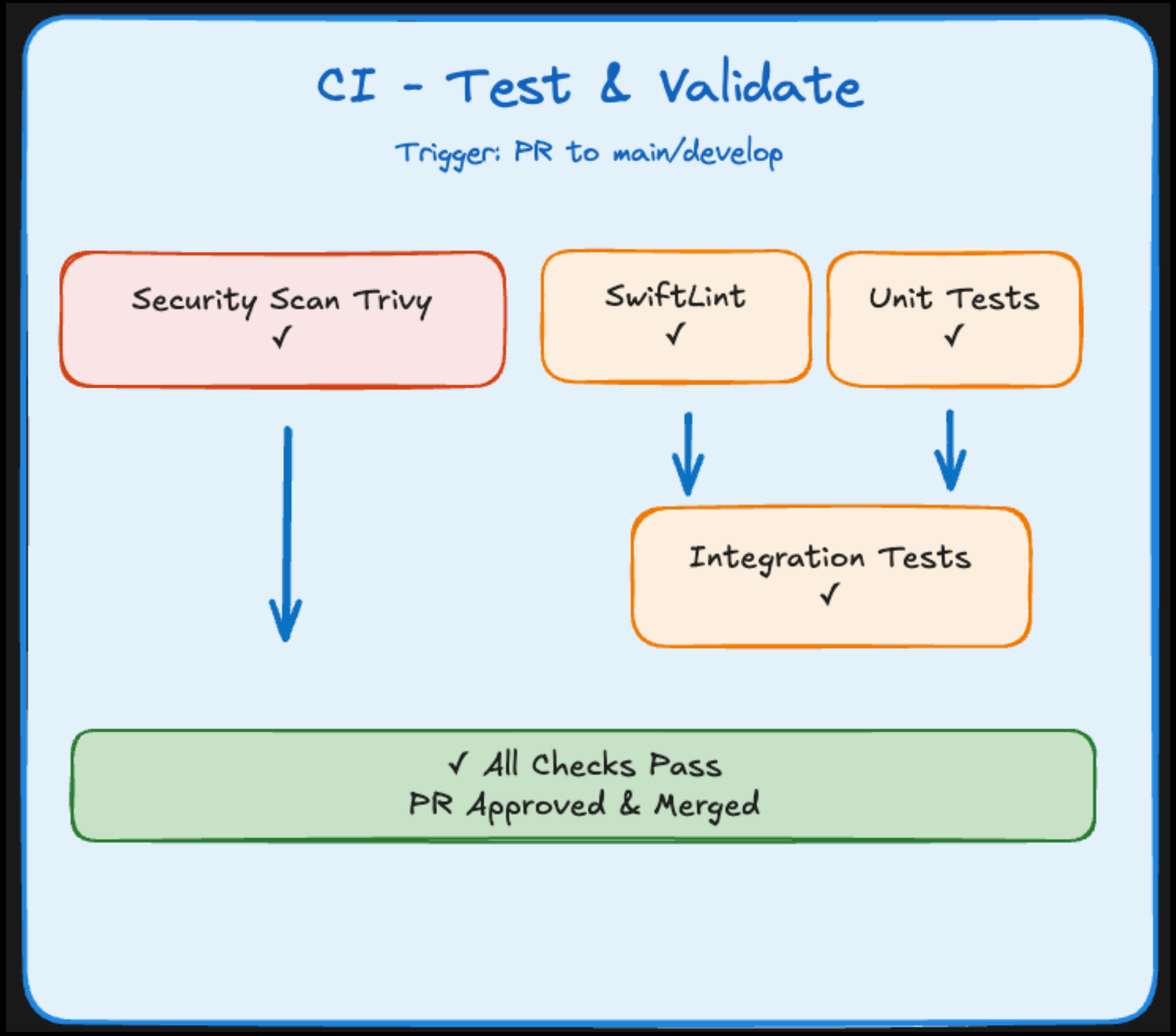
- HummingBird 2.0
  - 3 Routes (endpoints): todos, health, version
  - packaged with docker and docker-compose
- Postgres
- 2 Environment deployments (staging, production)

# Deploy using swift

- familiar language (no bash-script)
    - everyone can understand it
- Type safety in infrastructure
- Github supports Swift
- 18 Swift based GitHub actions

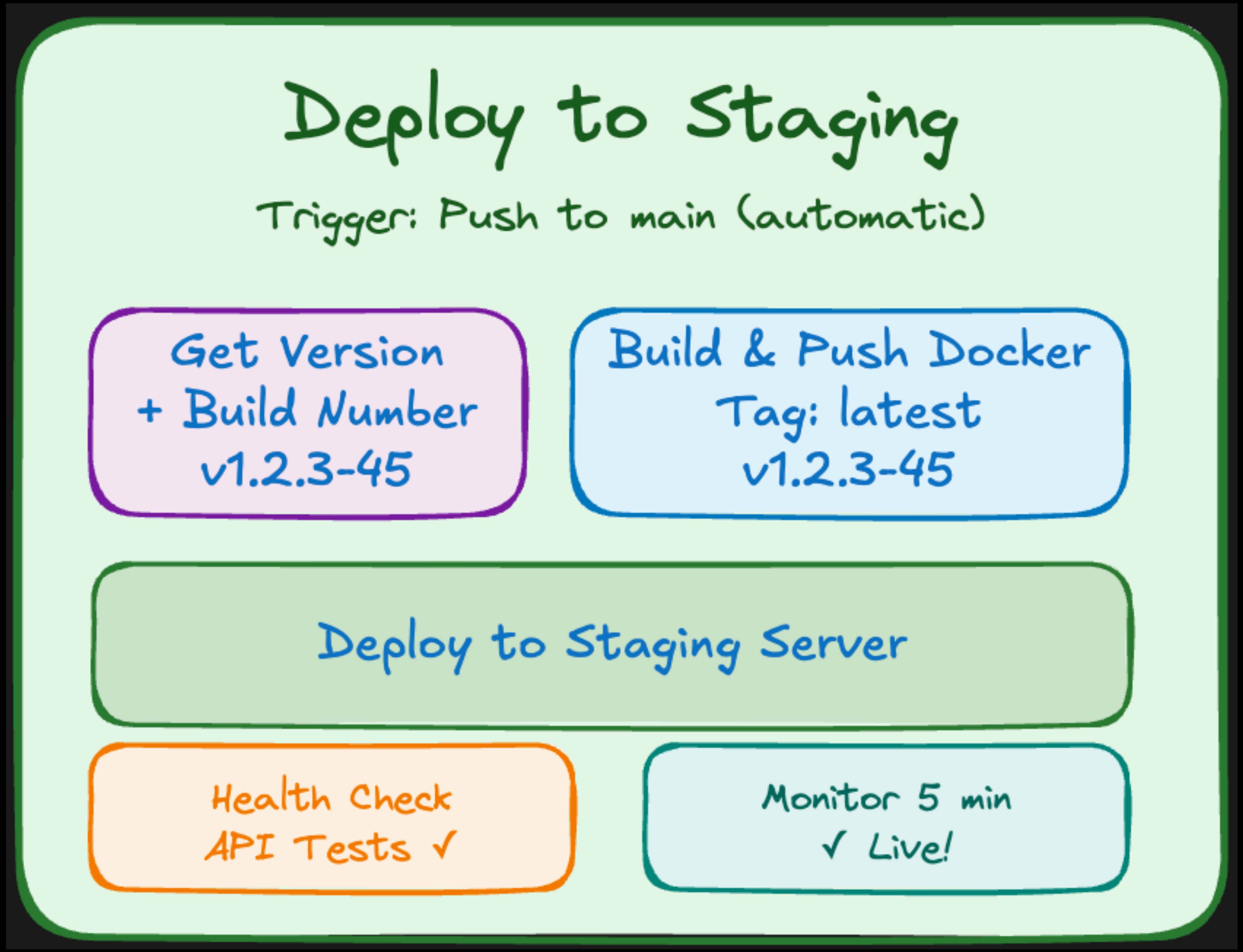
# 3 Workflows

- ci.yml -> Verification

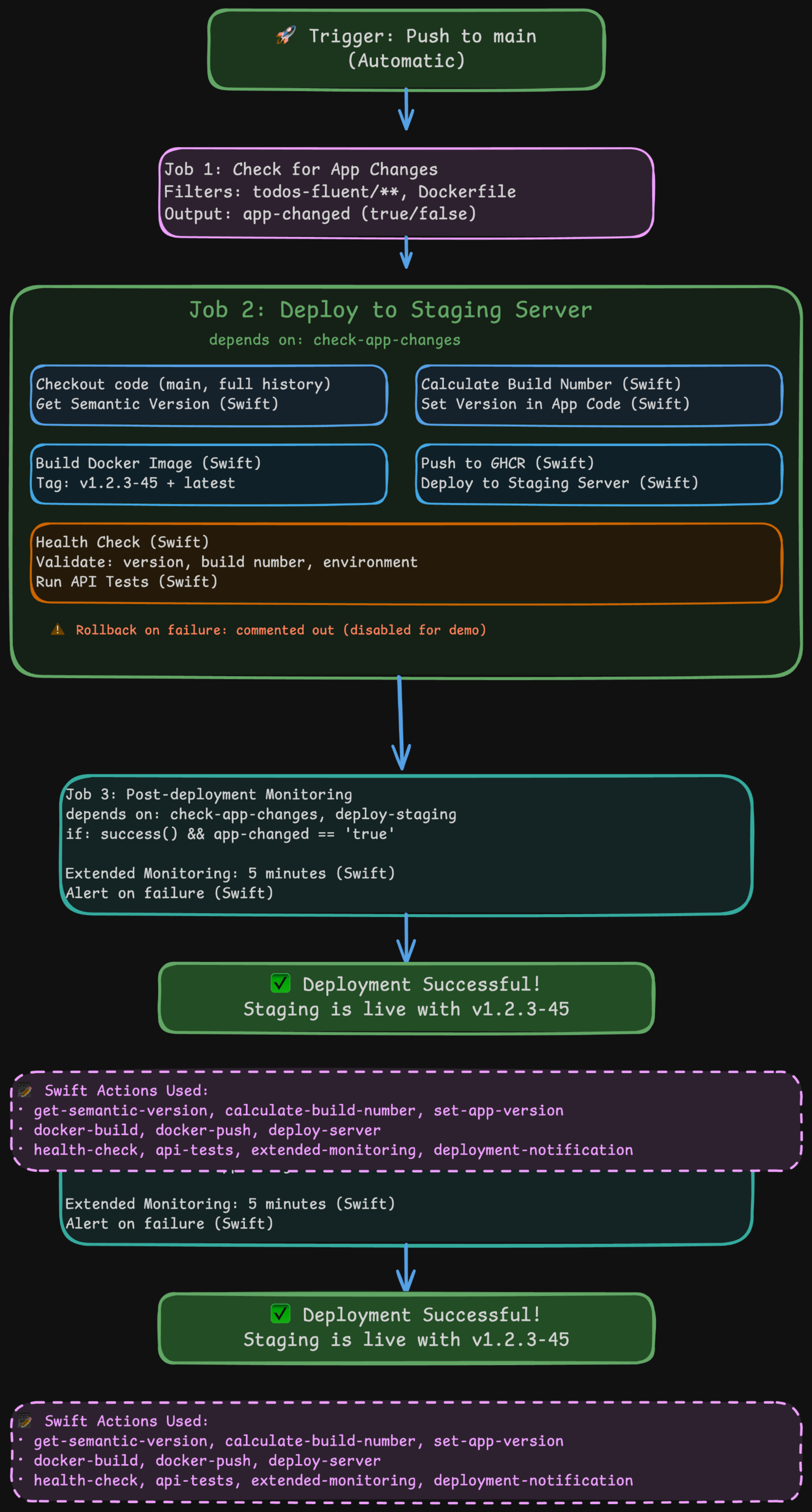


# 3 Workflows

- ci.yml -> Verification
- deploy-staging.yml -> Auto-deploy



# Deploy to Staging Workflow



🚀 Trigger: Push to main
(Automatic)

Job 1: Check for App Changes
Filters: todos-fluent/**, Dockerfile
Output: app-changed (true/false)

Job 2: Deploy to Staging Server
depends on: check-app-changes

Checkout code (main, full history)
Get Semantic Version (Swift)

Calculate Build Number (Swift)
Set Version in App Code (Swift)

Build Docker Image (Swift)
Tag: v1.2.3-45 + latest

Push to GHCR (Swift)
Deploy to Staging Server (Swift)

Health Check (Swift)
Validate: version, build number, environment
Run API Tests (Swift)

⚠️ Rollback on failure: commented out (disabled for demo)

Job 3: Post-deployment Monitoring
depends on: check-app-changes, deploy-staging
if: success() && app-changed == 'true'

Extended Monitoring: 5 minutes (Swift)
Alert on failure (Swift)

✅ Deployment Successful!
Staging is live with v1.2.3-45

📝 Swift Actions Used:
- get-semantic-version, calculate-build-number, set-app-version
- docker-build, docker-push, deploy-server
- health-check, api-tests, extended-monitoring, deployment-notification

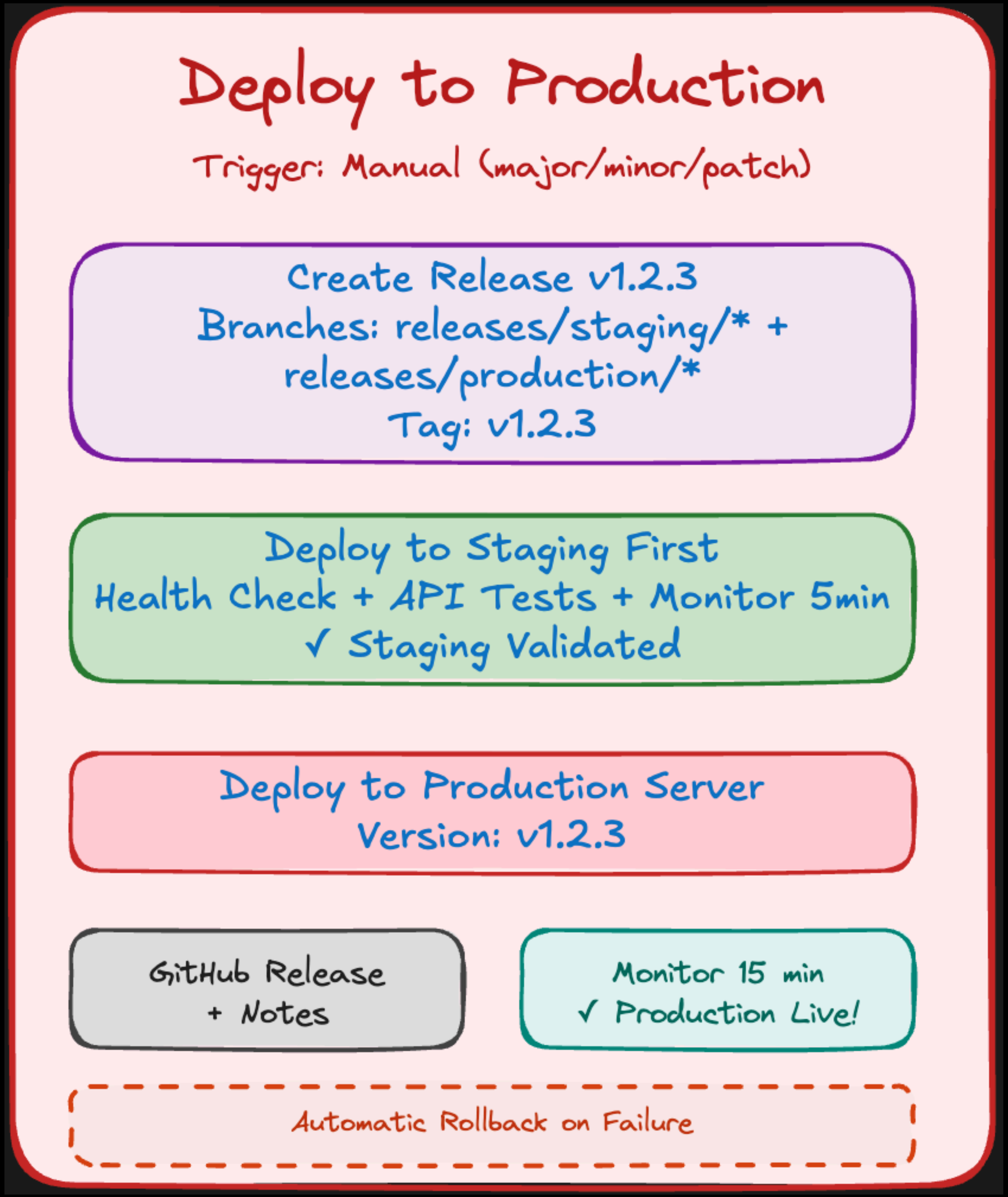
Extended Monitoring: 5 minutes (Swift)
Alert on failure (Swift)

✅ Deployment Successful!
Staging is live with v1.2.3-45

📝 Swift Actions Used:
- get-semantic-version, calculate-build-number, set-app-version
- docker-build, docker-push, deploy-server
- health-check, api-tests, extended-monitoring, deployment-notification

# 3 Workflows

- ci.yml -> Verification
- deploy-staging.yml -> Auto-deploy
- deploy-production -> Manual release



Deploy to Production

Trigger: Manual (major/minor/patch)

Create Release v1.2.3
Branches: releases/staging/* + releases/production/*
Tag: v1.2.3

Deploy to Staging First
Health Check + API Tests + Monitor 5min
✓ Staging Validated

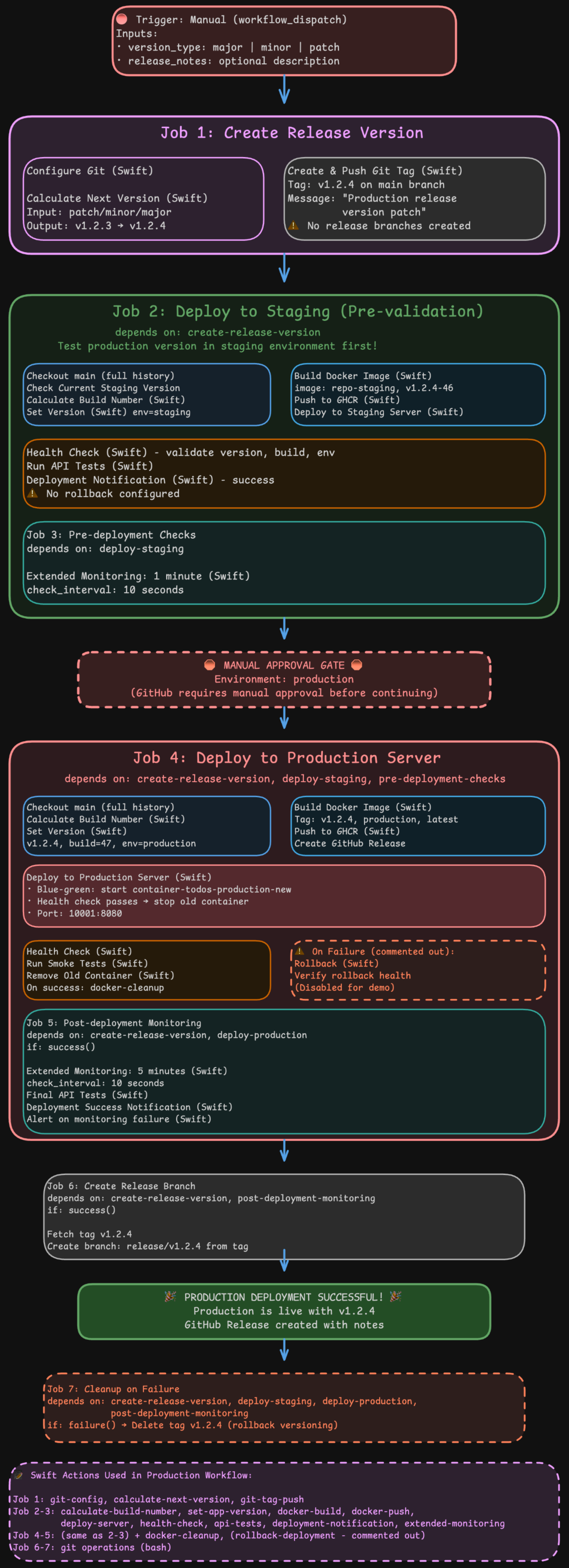
Deploy to Production Server
Version: v1.2.3

GitHub Release
+ Notes

Monitor 15 min
✓ Production Live!

Automatic Rollback on Failure

# Deploy to Production Workflow



🔴 Trigger: Manual (workflow_dispatch)
Inputs:
• version_type: major | minor | patch
• release_notes: optional description

## Job 1: Create Release Version

Configure Git (Swift)

Calculate Next Version (Swift)
Input: patch/minor/major
Output: v1.2.3 → v1.2.4

Create & Push Git Tag (Swift)
Tag: v1.2.4 on main branch
Message: "Production release
version patch"
⚠️ No release branches created

## Job 2: Deploy to Staging (Pre-validation)

depends on: create-release-version
Test production version in staging environment first!

Checkout main (full history)
Check Current Staging Version
Calculate Build Number (Swift)
Set Version (Swift) env=staging

Build Docker Image (Swift)
image: repo-staging, v1.2.4-46
Push to GHCR (Swift)
Deploy to Staging Server (Swift)

Health Check (Swift) - validate version, build, env
Run API Tests (Swift)
Deployment Notification (Swift) - success
⚠️ No rollback configured

Job 3: Pre-deployment Checks
depends on: deploy-staging

Extended Monitoring: 1 minute (Swift)
check_interval: 10 seconds

🛑 MANUAL APPROVAL GATE 🛑
Environment: production
(GitHub requires manual approval before continuing)

## Job 4: Deploy to Production Server

depends on: create-release-version, deploy-staging, pre-deployment-checks

Checkout main (full history)
Calculate Build Number (Swift)
Set Version (Swift)
v1.2.4, build=47, env=production

Build Docker Image (Swift)
Tag: v1.2.4, production, latest
Push to GHCR (Swift)
Create GitHub Release

Deploy to Production Server (Swift)
• Blue-green: start container-todos-production-new
• Health check passes → stop old container
• Port: 10001:8080

Health Check (Swift)
Run Smoke Tests (Swift)
Remove Old Container (Swift)
On success: docker-cleanup

⚠️ On Failure (commented out):
Rollback (Swift)
Verify rollback health
(Disabled for demo)

Job 5: Post-deployment Monitoring
depends on: create-release-version, deploy-production
if: success()

Extended Monitoring: 5 minutes (Swift)
check_interval: 10 seconds
Final API Tests (Swift)
Deployment Success Notification (Swift)
Alert on monitoring failure (Swift)

Job 6: Create Release Branch
depends on: create-release-version, post-deployment-monitoring
if: success()

Fetch tag v1.2.4
Create branch: release/v1.2.4 from tag

🎉 PRODUCTION DEPLOYMENT SUCCESSFUL! 🎉
Production is live with v1.2.4
GitHub Release created with notes

Job 7: Cleanup on Failure
depends on: create-release-version, deploy-staging, deploy-production,
post-deployment-monitoring
if: failure() → Delete tag v1.2.4 (rollback versioning)

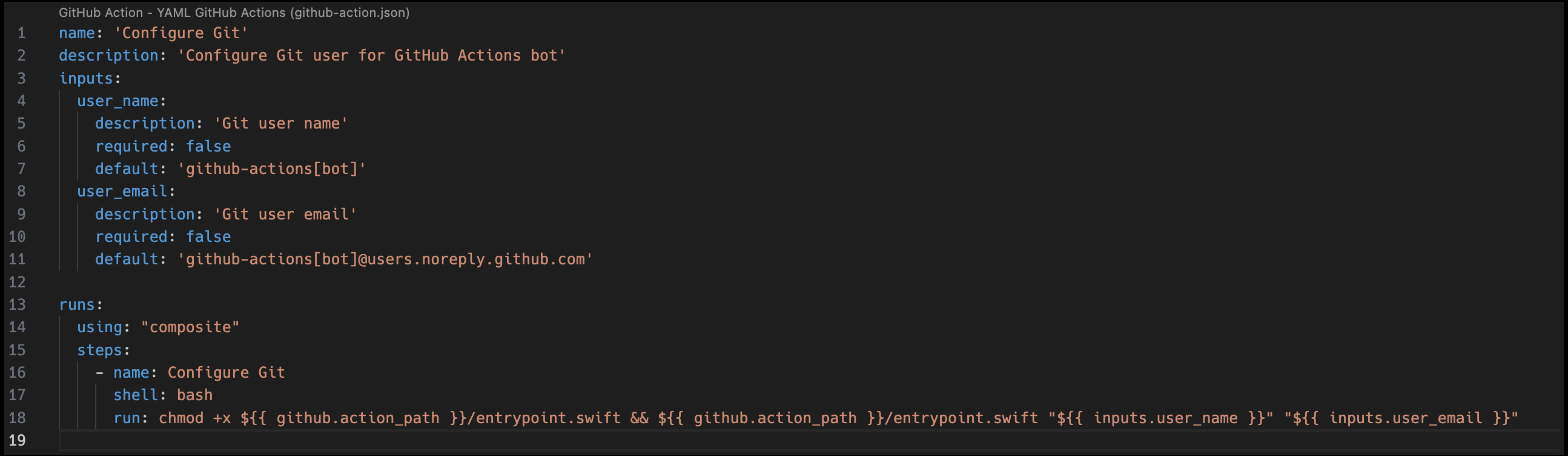
📝 Swift Actions Used in Production Workflow:

Job 1: git-config, calculate-next-version, git-tag-push
Job 2-3: calculate-build-number, set-app-version, docker-build, docker-push,
deploy-server, health-check, api-tests, deployment-notification, extended-monitoring
Job 4-5: (same as 2-3) + docker-cleanup, (rollback-deployment - commented out)
Job 6-7: git operations (bash)

# Core principle

## Swift GitHub Action

- Define your arguments in `action.yml`
- one liner to execute the Swift-script (permission + run)
- with safe argument parsing



GitHub Action - YAML GitHub Actions (github-action.json)

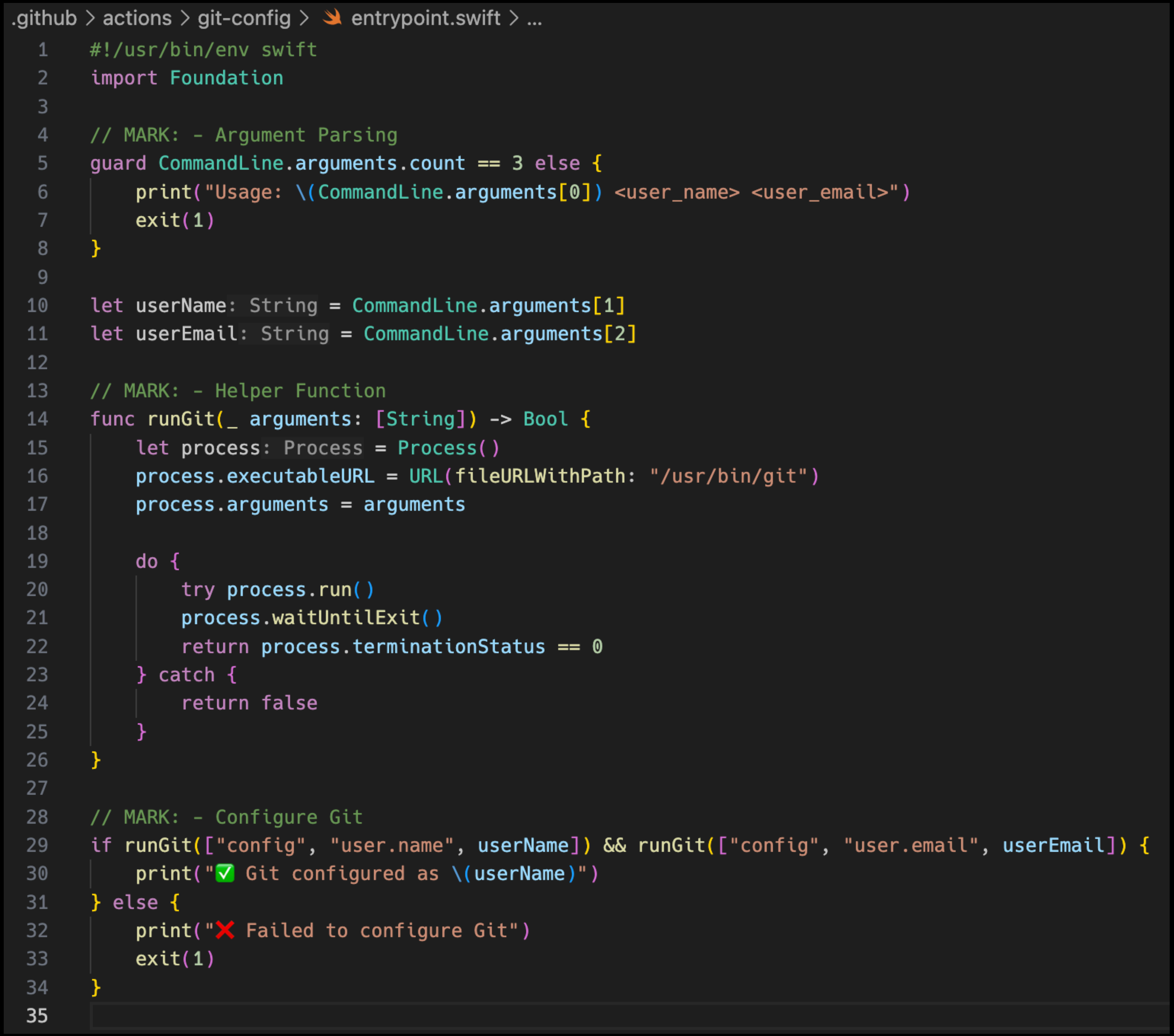
```yaml
name: 'Configure Git'
description: 'Configure Git user for GitHub Actions bot'
inputs:
  user_name:
    description: 'Git user name'
    required: false
    default: 'github-actions[bot]'
  user_email:
    description: 'Git user email'
    required: false
    default: 'github-actions[bot]@users.noreply.github.com'

runs:
  using: "composite"
  steps:
    - name: Configure Git
      shell: bash
      run: chmod +x ${{ github.action_path }}/entrypoint.swift && ${{ github.action_path }}/entrypoint.swift "${{ inputs.user_name }}" "${{ inputs.user_email }}"
```

# Core principle

## Swift GitHub Action

- with safe argument parsing
- Swift-world



```swift
#!/usr/bin/env swift
import Foundation

// MARK: - Argument Parsing
guard CommandLine.arguments.count == 3 else {
    print("Usage: \(CommandLine.arguments[0]) <user_name> <user_email>")
    exit(1)
}

let userName: String = CommandLine.arguments[1]
let userEmail: String = CommandLine.arguments[2]

// MARK: - Helper Function
func runGit(_ arguments: [String]) -> Bool {
    let process: Process = Process()
    process.executableURL = URL(fileURLWithPath: "/usr/bin/git")
    process.arguments = arguments

    do {
        try process.run()
        process.waitUntilExit()
        return process.terminationStatus == 0
    } catch {
        return false
    }
}

// MARK: - Configure Git
if runGit(["config", "user.name", userName]) && runGit(["config", "user.email", userEmail]) {
    print("✅ Git configured as \(userName)")
} else {
    print("❌ Failed to configure Git")
    exit(1)
}

```

# Deep dive

## Your choice

- api-tests
- calculate-build number
- create-release-branch
- deployment-notification
- docker-build
- docker-push
- extended-monitoring
- get-semantic-version
- get-version
- git-tag-push
- health-check
- set-app-version
- remote-scripts
- server-management (*)

# Demo

# Future

- spread the word
- beginners documentation
- optimise docker build time
- universal rollback
- ...
- open for other ideas

# Links

- GitHub: https://github.com/elmoritz
- https://github.com/elmoritz/swift-on-server-one-click-deployment
- linkedIn: https://www.linkedin.com/in/elmoritz/
- mastodon: https://mas.to/@elmoritz
- https://www.youtube.com/@arconsis (most of my videos)
- https://www.youtube.com/@elmoritz (no time for videos)

# Thank You

and contribute