**SDG6 Mobile App – DevOps Implementation Documentation**

**Overview**

* This document outlines the DevOps workflow implementation for the SDG6 Mobile App, focusing on structured development, CI/CD automation, security practices, deployment strategies, and observability.

**Phase 2: Development Workflow (20%)**

**1. GitHub Issues**

Each major task is tracked using GitHub Issues:

| **Task** |
| --- |
| * Create login screen |
| * Firebase auth setup |
| * Daily water usage UI |
| * Firebase Realtime DB integration |
|  |

**2. GitHub Project Board**

* **Board Name:** SDG6 DevOps Board
* **Columns:** To Do, In Progress, Done
* Issues are linked to cards and moved as work progresses.

**3. Git Branching Workflow**

# Create development branch

git checkout -b dev

git push origin dev

# Feature branches (example)

git checkout -b feature/login-screen

**4. Pull Request Template (“.github/PULL\_REQUEST\_TEMPLATE.md”)**

## 📝 Summary

Briefly explain what your PR does.

## ✅ Checklist

- [ ] Feature implemented

- [ ] Code builds without errors

- [ ] Tests added or updated

- [ ] Related docs updated

**5. Contribution Guide (“CONTRIBUTING.md”)**

# 🚀 Contributing Guide

## Git Workflow

- Base branch: `dev`

- Feature branches: `feature/<name>`

- PRs must target `dev`

## Commit Message Convention

- feat: new features

- fix: bug fixes

- docs: documentation changes

**6. GitHub Actions CI (“.github/workflows/android.yml”)**

name: Android CI

on:

push:

branches: [dev]

pull\_request:

branches: [dev]

jobs:

build:

runs-on: ubuntu-latest

steps:

- uses: actions/checkout@v3

- uses: actions/setup-java@v3

with:

java-version: '11'

- name: Build

run: ./gradlew build

**7. README.md Highlights**

# 📱 SDG6 Mobile App

Track daily water usage and promote water conservation awareness.

## 🛠️ Built With

- Java (Android)

- Firebase Realtime DB

- GitHub Actions for CI/CD

**8. Progress Tracking**

Cards on the Project Board are continuously updated. Issues are closed upon completion.

**Phase 3: CI/CD & Quality Assurance**

**9. Matrix Testing (Skipped)**

**10. Android Instrumented Test**

**File:** “src/androidTest/java/com/your/package/ExampleInstrumentedTest.java”

@RunWith (AndroidJUnit4.class)

public class ExampleInstrumentedTest {

@Test

public void useAppContext () {

Context appContext = InstrumentationRegistry.getInstrumentation(). getTargetContext ();

assertEquals ("com.your.package.name", appContext.getPackageName());

}

}

Automatically triggered in CI on push/PR to dev.

**11. Security Scans**

**A. Dependabot Alerts**

* Enabled via GitHub → Settings → Security → Code security and analysis → Enable Dependabot alerts.

**B. CodeQL Static Analysis (.github/workflows/codeql-analysis.yml)**

name: "CodeQL"

on:

push:

branches: [dev]

pull\_request:

branches: [dev]

schedule:

- cron: '0 0 \* \* 0'

jobs:

analyze:

runs-on: ubuntu-latest

permissions:

security-events: write

strategy:

matrix:

language: ['java']

steps:

- uses: actions/checkout@v3

- uses: github/codeql-action/init@v2

with:

languages: ${{ matrix.language}}

- uses: github/codeql-action/autobuild@v2

- uses: github/codeql-action/analyze@v2

**12. Artifact Management (Optional)**

APK artifacts upload is available on request.

**Phase 4: Production Deployment & Observability**

**13. Release Strategy**

git tag -a v1.0.0 -m "Initial Release"

git push origin v1.0.0

* GitHub → Releases → Draft New Release → Tag: v1.0.0

**14. Observability**

**A. Crash Reporting**

* Integrated **Firebase Crashlytics** via Firebase SDK.

**B. Usage Analytics**

* Integrated **Google Analytics for Firebase**.

**C. Log Monitoring**

* Used Logcat during QA cycles.

**15. Synthetic Monitoring**

Skipped (mobile app only).

**16. Project Documentation (Divio Framework)**

Create /docs/ folder and add:

**Tutorials:**

docs/tutorials/installing-app.md

**How-to Guides:**

docs/how-to/report-usage.md

**Reference:**

docs/reference/firebase-schema.md

**17. OpenAPI Documentation (If backend APIs used)**

Can be generated on request.

**18. Architecture Decision Records (ADRs)**

**Directory:** docs/architecture/

Files:

* firebase.md – Rationale for using Firebase.
* java-vs-kotlin.md – Technology decision record.
* ci-cd-choice.md – Why GitHub Actions for CI/CD.

**19. Incident Response Playbook**

**File:** “docs/incident-playbook.md”

# Incident Response Playbook

## Common Issues

- Firebase data not syncing

## Steps to Resolve

1. Check network connectivity

2. Check Firebase console

3. Redeploy if needed

## Contacts

- DevOps Led: you@example.com

**Summary:**

|  |
| --- |
| * GitHub Project & Issues |
| * Branching & CI |
| * Tests & Security Scans |
| * Release Strategy |
| * Observability & Monitoring |
| * Documentation (Divio + ADR) |