Thanks for sharing the repo! The project you're referring to — https://github.com/nanuchi/my-project — is a simple **Java project built using Gradle**. You want to design a **CI/CD workflow** (probably via GitHub Actions) and score well across:

- ✅ **CI/CD Functionality (30 Marks)**
- ✅ **Workflow Structure & Clarity (10 Marks)**
- ✅ **Bonus Marks (10 Marks)** for best practices and robustness

---

🎯 Your Goal: Create a GitHub Actions Workflow for CI/CD

✅ Let's break this down:

🔧 Step 1: CI/CD Functionality (30 Marks)

Here's what you'll want your workflow to do:

✅ Continuous Integration

1. **Trigger on Push and Pull Requests**
2. **Set up Java environment**
3. **Grant permission to Gradle wrapper**
4. **Run ./gradlew build and run tests**

✅ Continuous Deployment

Since it's not a full deployment project (no Docker registry / cloud), you can simulate deployment (e.g., echo a message, or upload build artifacts).

🧱 Step 2: Workflow Structure and Clarity (10 Marks)

- Clean, readable YAML
- Logical job naming
- Modular steps with proper naming
- Well-indented and concise

✨ Step 3: Bonus Marks (10 Marks)

- Error handling (e.g., continue-on-error: false where needed)
- Caching Gradle dependencies
- Uploading test reports/artifacts

- Version pinning (actions/setup-java@v3)
- Secure and minimal

🧩 **Final Workflow File**

Create the workflow at:
.github/workflows/ci.yml

```
name: Java CI with Gradle


on:
  push:
    branches: [ "master" ]
  pull_request:
    branches: [ "master" ]


jobs:
  build:
    name: Build & Test
    runs-on: ubuntu-latest


    steps:
    - name: Checkout Repository
      uses: actions/checkout@v3


    - name: Set up JDK 11
      uses: actions/setup-java@v3
      with:
        distribution: 'temurin'
        java-version: '11'


    - name: Grant execute permission for gradlew
      run: chmod +x gradlew
```

```
- name: Cache Gradle packages
  uses: actions/cache@v3
  with:
    path: |
      ~/.gradle/caches
      ~/.gradle/wrapper
    key: gradle-${{ runner.os }}-${{ hashFiles('**/*.gradle*', '**/gradle-wrapper.properties') }}
    restore-keys: |
      gradle-${{ runner.os }}-

- name: Build with Gradle
  run: ./gradlew build --no-daemon

- name: Run Tests
  run: ./gradlew test --no-daemon

- name: Upload Test Results
  if: always()
  uses: actions/upload-artifact@v3
  with:
    name: test-results
    path: build/test-results/test

- name: Simulate Deployment
  if: success()
  run: echo "✅ Build successful. Deploying to staging (simulated)..."
```

---

1. **Build the Docker image**
2. **Login to Docker Hub (or another registry)**
3. **Push the image**

🔐 **First: Set up DockerHub Credentials in GitHub**

Go to your repo → **Settings > Secrets and Variables > Actions > New repository secret**, and add:

Name	**Value**
DOCKER_USERNAME	your-dockerhub-username
DOCKER_PASSWORD	your-dockerhub-password/token

✍️ **Updated Workflow with Docker Push**

Below is your enhanced workflow (.github/workflows/ci.yml) with Docker build and push:

```
name: Java CI with Gradle and Docker

on:
  push:
    branches: [ "master" ]
  pull_request:
    branches: [ "master" ]

jobs:
  build-and-deploy:
    name: Build, Test, Dockerize & Push
    runs-on: ubuntu-latest

    steps:
    - name: Checkout Repository
```

```yaml
  uses: actions/checkout@v3

- name: Set up JDK 11
  uses: actions/setup-java@v3
  with:
    distribution: 'temurin'
    java-version: '11'

- name: Grant execute permission for gradlew
  run: chmod +x gradlew

- name: Cache Gradle packages
  uses: actions/cache@v3
  with:
    path: |
      ~/.gradle/caches
      ~/.gradle/wrapper
    key: gradle-${{ runner.os }}-${{ hashFiles('**/*.gradle*', '**/gradle-wrapper.properties') }}
    restore-keys: |
      gradle-${{ runner.os }}-

- name: Build with Gradle
  run: ./gradlew build --no-daemon

- name: Run Tests
  run: ./gradlew test --no-daemon

- name: Upload Test Results
  if: always()
  uses: actions/upload-artifact@v3
  with:
    name: test-results
    path: build/test-results/test

- name: Log in to Docker Hub
  uses: docker/login-action@v3
  with:
    username: ${{ secrets.DOCKER_USERNAME }}
    password: ${{ secrets.DOCKER_PASSWORD }}

- name: Build Docker Image
  run: docker build -t ${{ secrets.DOCKER_USERNAME }}/my-project:latest .

- name: Push Docker Image
  run: docker push ${{ secrets.DOCKER_USERNAME }}/my-project:latest
```