GitHub CI/CD Workflow Setup Documentation

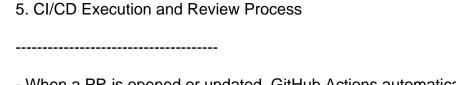
This document outlines the steps taken to set up a GitHub-based CI/CD workflow for a React project with pull request validation and collaboration.

Repository Setup and Task Assignment
- Created a GitHub repository and initialized it with the React project.
- Tasks were managed via GitHub Issues.
- Each feature/task is developed in a separate feature branch created from main.
2. Branching and PR Workflow
- Developers create feature branches using:
git checkout -b feature/feature-name
- After completing the task, code is pushed to the feature branch and a pull request (PR) is opened
to the main branch.
3. Collaborator Access and Review Rules
- Collaborators were added under GitHub repository Settings > Collaborators.
- Branch protection rules were added for main:
- Require pull request reviews before merging.
- Require status checks to pass before merging.
- Block direct pushes to main.

4. GitHub Actions CI Pipeline Setup

Created a workflow file .github/workflows/ci.yml with the following configuration:
name: CI Pipeline
on:
pull_request:
branches:
- main
jobs:
build:
runs-on: ubuntu-latest
steps:
- name: Checkout code
uses: actions/checkout@v2
- name: Set up Node.js
uses: actions/setup-node@v2
with:
node-version: '14'
- name: Install dependencies
run: npm install
- name: Run tests
run: npm test

- This pipeline installs dependencies and runs tests on every PR targeting the main branch.



- When a PR is opened or updated, GitHub Actions automatically runs the workflow.
- The "Checks" tab in the PR shows pipeline results.
- Collaborators review the PR and click Approve.
- After approval and successful checks, the PR can be merged.
- Optionally, delete the feature branch post-merge.

Conclusion

This setup ensures that only reviewed and tested code reaches the main branch. It enforces code quality, review discipline, and automated verification using GitHub Actions.