

Brandon Cunningham
CS 472-1001
February 12th, 2024

Report - Continuous Integration

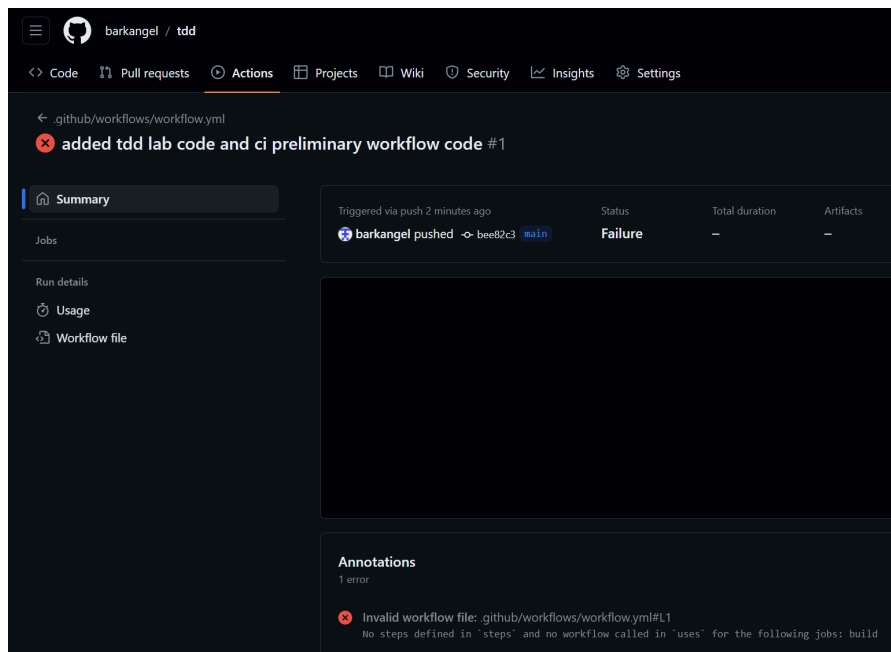
Fork repository: <https://github.com/barkangel/tdd>

Task 1: Adding workflow github folder and creating new repository

- Workflow preliminary code

```
workflow.yml 2 x
.github > workflows > workflow.yml
1  name: "CI workflow"
2
3  on:
4    push:
5      branches:
6        - main
7    pull_request:
8      branches:
9        - main
10
11 jobs:
12   build:
13     runs-on: ubuntu-latest
14     container: python:3.9-slim
15     steps:
16
```

- Error proof: (repo name at top)



Task 2:

- Updating workflow.yml to include new jobs:

```
workflow.yml  counter.py
.github > workflows > workflow.yml
1  name: "CI workflow"
2
3  on:
4    push:
5      branches:
6        - main
7    pull_request:
8      branches:
9        - main
10
11  jobs:
12    build:
13      runs-on: ubuntu-latest
14      container: python:3.9-slim
15      steps:
16        - name: Checkout
17          uses: actions/checkout@v3
18        - name: Install dependencies
19          run: |
20            python -m pip install --upgrade pip
21            pip install -r requirements.txt
22        - name: Lint with flake8
23          run: |
24            flake8 src --count --select=E9,F63,F7,F82 --show-source --statistics
25            flake8 src --count --max-complexity=10 --max-line-length=127 --statistics
26        - name: Nosetests
27          run: |
28            nosetests -v --with-spec --spec-color --with-coverage --cover-package=app
```

- Running jobs on Github actions from workflow.yml (repo name at top)

The screenshot shows the GitHub Actions interface for the repository 'barkangel / tdd'. The left sidebar contains navigation links: Code, Pull requests, Actions (selected), Projects, Wiki, Security, Insights, and Settings. Under the 'Actions' tab, there are links for 'All workflows', 'CI workflow', 'Management', 'Caches', and 'Runners'. The main area is titled 'All workflows' and shows a list of workflow runs. The first run is a successful build job (CI workflow #6) pushed by 'barkangel' on the 'main' branch, completed 1 minute ago. The second run is a failed job (flake8 spacing issue fixed on counter.py) pushed by 'barkangel' on the 'main' branch, completed 2 minutes ago. A 'Help us improve GitHub Actions' banner is visible at the top of the workflow runs list.

Event	Status	Branch	Actor
✓	Success	main	barkangel
✗	Failure	main	barkangel

- Successful builds: (repo name at top)

The screenshot shows the GitHub Actions interface for the repository 'barkangel / tdd'. The 'Actions' tab is selected, displaying a workflow named 'CI workflow' with a green checkmark and '#6'. The 'Summary' section on the left lists 'Jobs' with 'build' as the active job, and 'Run details' with 'Usage' and 'Workflow file'. The main panel shows the 'build' job status as 'succeeded 2 minutes ago in 16s'. A list of steps follows, each with a green checkmark: 'Set up job', 'Initialize containers', 'Checkout', 'Install dependencies', 'Lint with flake8', 'Nosetests', 'Post Checkout', 'Stop containers', and 'Complete job'.

- Run unit tests with nose, which I named “Nosetests”:

This screenshot provides a detailed view of the 'Nosetests' step within the 'build' job. The job is marked as 'succeeded 3 minutes ago in 16s'. The 'Nosetests' step is expanded, showing its execution logs. The logs include a list of tests: 'Counter tests' with sub-items 'It should create a counter', 'It should delete a counter', 'It should return an error for duplicates', 'read a counter', and 'update a counter'. A warning from coverage is also visible: 'Module app was never imported. (module-not-imported)'. A table summarizes the test results, showing 5 tests passed with 100% coverage. The step concludes with 'Ran 5 tests in 0.137s' and 'OK'.

Name	Stmts	Miss	Cover	Missing
src/counter.py	28	0	100%	
src/status.py	6	0	100%	
TOTAL	34	0	100%	

- Adding delete cases

Creating `test_delete_a_counter(self)` in `test_counter.py`:

```
def test_delete_a_counter(self):  
    """It should delete a counter"""  
    result = self.client.post('/counters/toBeDel')  
    self.assertEqual(result.status_code, status.HTTP_201_CREATED)  
  
    delResult = self.client.delete('/counters/toBeDel')  
    self.assertEqual(delResult.status_code, status.HTTP_204_NO_CONTENT)
```

We are now in **RED** stage, because our new test case fails.

Creating `delete_counter(name)` in `counter.py`:

```
@app.route('/counters/<name>', methods=['DELETE'])  
def delete_counter(name):  
    app.logger.info(f"Request to delete counter: {name}")  
    if name in COUNTERS:  
        COUNTERS.pop(name)  
        return {name: name}, status.HTTP_204_NO_CONTENT
```

We are now in **GREEN** stage, because we have written minimum amount of code to pass test.