DIT 637 Smart and Secure Systems
TT04 Experiencing DevSecOps - A Mobile App with GitHub Actions and OWASP ZAP

06/20/2024 Developed by Clark Ngo
07/03/2024 Reviewed by Sam Chung
School of Technology & Computing (STC) @ City University of Seattle (CityU)

Key Concepts and Tools for Cloud-based Full Stack Continuous Integration & Deployment (CI/CD)

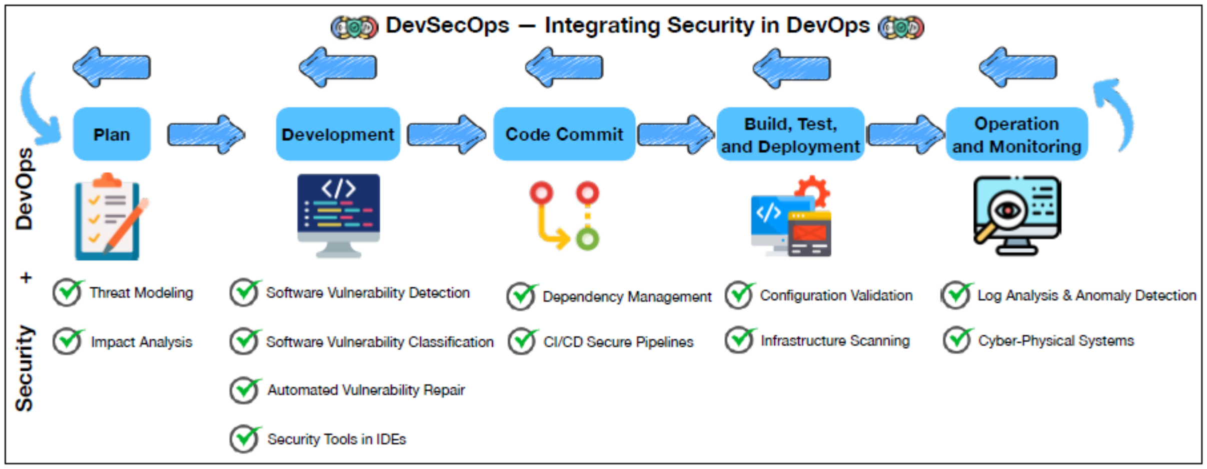


Fig. 1. The overview of the DevOps workflow and the identified security tasks relevant under each step in the DevOps process.

Continuous Integration (CI) and Deployment (CD)

GitHub Actions:
To automate the testing and deployment process. When code is pushed to the repository, GitHub Actions can automatically run tests and deploy the application to Amazon EC2, saving you time and effort. It can also run a security scanner like ZAP.

- https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions#overview

Open Worldwide Application Security Project (OWASP) Zed Attack Proxy (ZAP):
An open-source web application security scanner maintained by OWASP. It is used to find security vulnerabilities in web applications during development and testing phases.

- https://github.com/marketplace/actions/zap-baseline-scan

YAML ("YAML Ain't Markup Language" **or** "YAML ain't markup language"):

- https://yaml.org/
- A **YML file** is a text document that contains data formatted using **YAML**
- **YAML** is a **human-readable data serialization language** commonly used for writing **configuration files**.
- **Comparison with Other Formats:**
    - **XML**, which stands for **Extensible Markup Language**, is a generalized markup language. It offers a structured yet flexible syntax and defined document schema. However, XML can be verbose and harder to read.

  - **JSON** (JavaScript Object Notation) is a language-independent format inspired by JavaScript. It's concise and widely used but lacks some features like comments and multiline strings.
- **YAML** strikes a balance between readability and structure. It's often used for configuration files due to its intuitive syntax and human-friendly format.
  - **Basic Syntax:**
  - **Indentation**: YAML uses indentation (spaces or tabs) to define nested structures.
  - **Scalars**: Represent simple values (strings, numbers, Booleans).
  - **Collections**: Include lists (arrays) and dictionaries (key-value pairs).
  - **Comments**: Start with # and provide context or explanations.
  - **Explicit Data Types**: Specify data types (e.g., strings, integers) explicitly.

Example: Movie Search Application

To practice and demonstrate the capabilities of React Native and Express in creating real-world applications, like searching for movies on Netflix.

Uploading one image files to your GitHub Repository generated from GitHub Classroom

1. The screenshot of your 'Scan ZAP Website – Zap Scan' as '*first*_*last*_zap_scan.png' by using your first and last name.

1) User Case

As a movie enthusiast using a **mobile device**, I want to search and browse a list of movies with details such as title, genre, and year, so that I can easily find information about movies I am interested in **while on the go**.
Note: the user story didn't change. The question then is why we need MongoDB Atlas, ExpressJS, GitHub Actions, and Amazon EC2.

Why Do We Need a Backend and Database?

1. **Data Management:**
   - **Database:** Storing and managing large amounts of movie data (titles, genres, years, etc.) efficiently. A database like MongoDB Atlas is ideal for handling such documents and providing quick access to them.
   - **Backend:** Facilitating the retrieval, manipulation, and updating of data from the database. The backend (using ExpressJS) acts as an intermediary that processes requests from the frontend, queries the database, and sends the relevant data back to the frontend.
2. **Business Logic:**
   - **Backend:** Encapsulating the business logic required to process data and perform operations such as filtering movies by genre, searching for specific titles, and handling user preferences. It separates the logic from the frontend, making the application more modular and easier to maintain.

Why Do We Need Cloud Deployment?

1. **Scalability:**
   - **Cloud Deployment (e.g., Amazon EC2):** Allows the application to scale up or down based on demand. As more users search and browse movies, the cloud infrastructure can handle increased traffic without performance issues.
2. **Availability and Reliability:**
   - **Cloud Deployment:** Ensures that the application is highly available and reliable. Cloud providers offer features such as load balancing, automated backups, and disaster recovery, which help keep the application running smoothly.
3. **Global Access:**
   - **Cloud Deployment:** Provides global access to the application. Users from different parts of the world can access the service with low latency, ensuring a seamless experience.

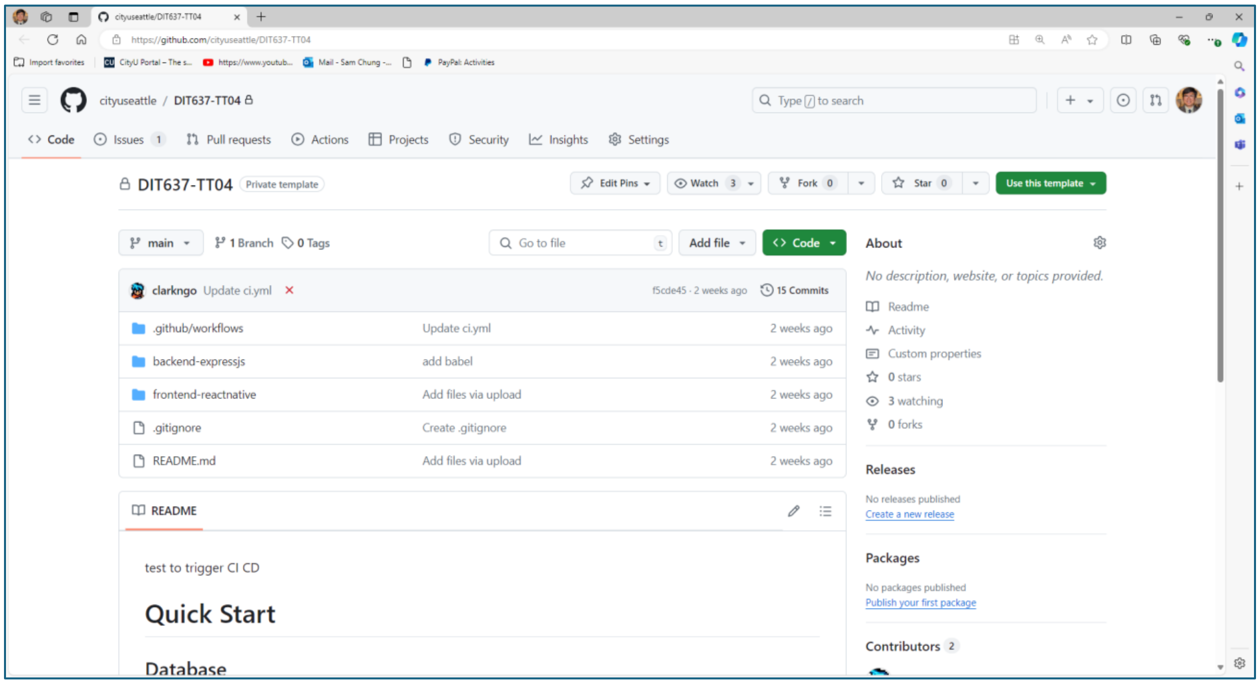
Why Do We Need Full-Stack Development?

1. **Limited Data Handling:**
   - React Native alone cannot efficiently handle large datasets or complex queries. It is primarily a frontend framework meant for building user interfaces, not for managing or processing data.
2. **No Data Persistence:**
   - Without a backend and database, data would be stored locally on the device, making it inaccessible if the user switches devices or uninstalls the app. A cloud database ensures that data is persistent and accessible from anywhere.
3. **Security Concerns:**
   - Storing and processing data directly on the client-side exposes it to security risks. A backend server ensures data is processed securely and can implement necessary security protocols.
4. **Separation of Concerns:**

- Maintaining a clear separation between frontend and backend simplifies development and maintenance. It allows each part to be developed, tested, and scaled independently.

2) GitHub Actions with OWASP ZAP

Open the project in GitHub.



clarkngo Update ci.yml — f5cde45 · 2 weeks ago — 15 Commits

.github/workflows	Update ci.yml	2 weeks ago
backend-expressjs	add babel	2 weeks ago
frontend-reactnative	Add files via upload	2 weeks ago
.gitignore	Create .gitignore	2 weeks ago
README.md	Add files via upload	2 weeks ago

README

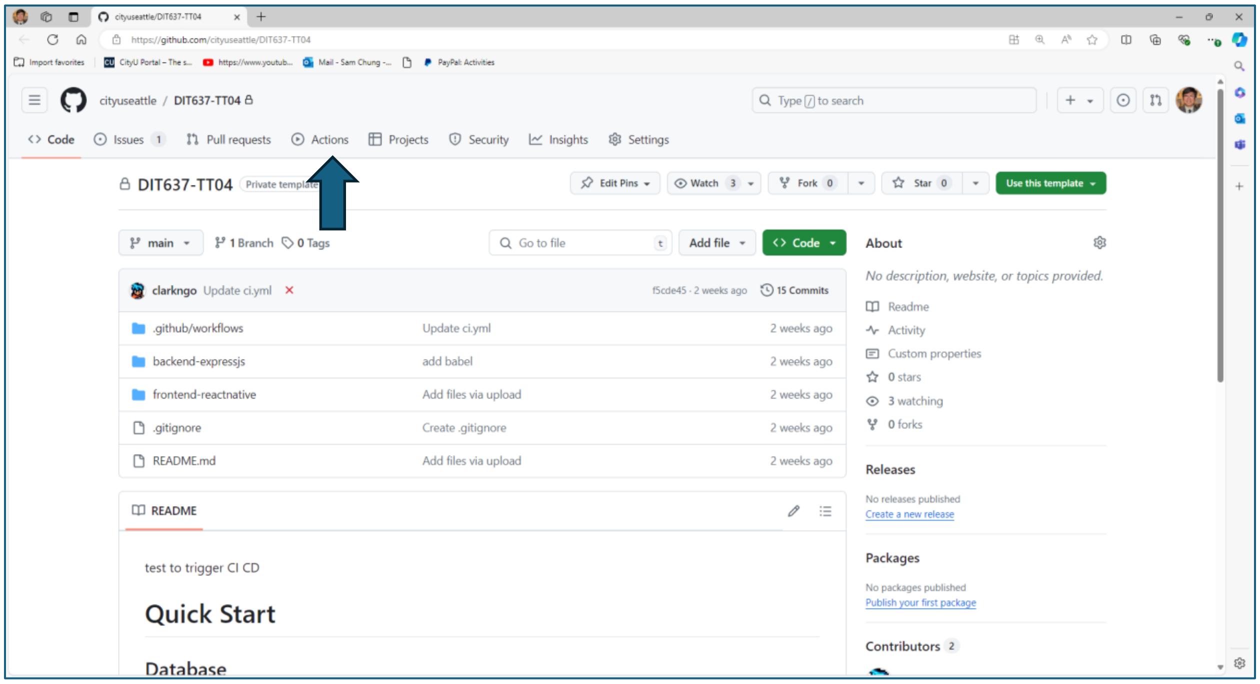
test to trigger CI CD

Quick Start

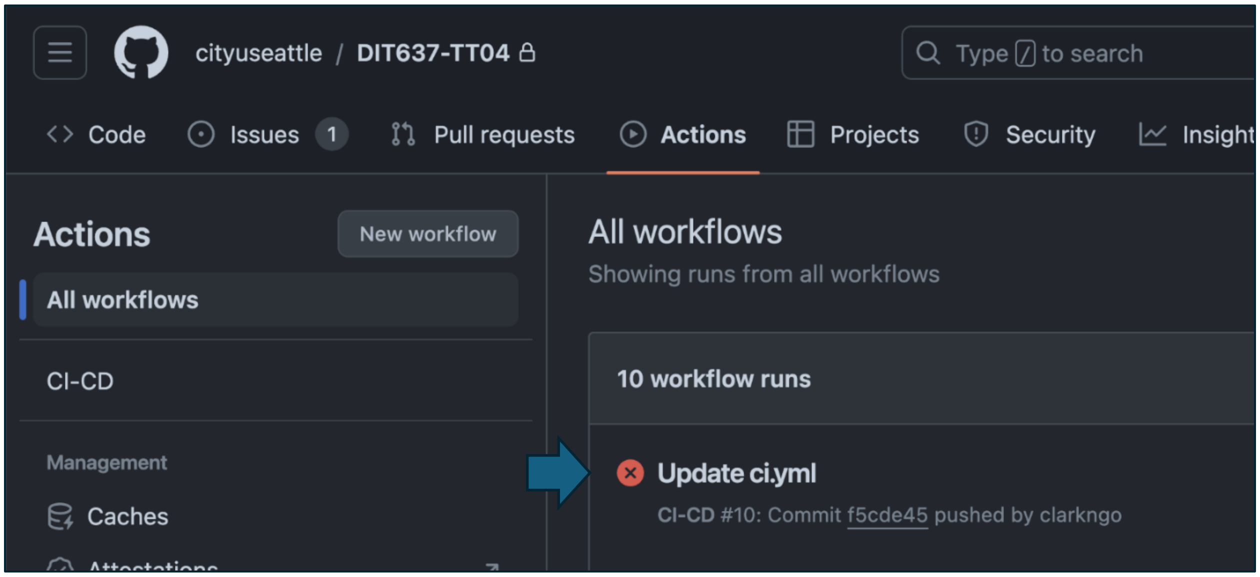
Database

Running ZAP

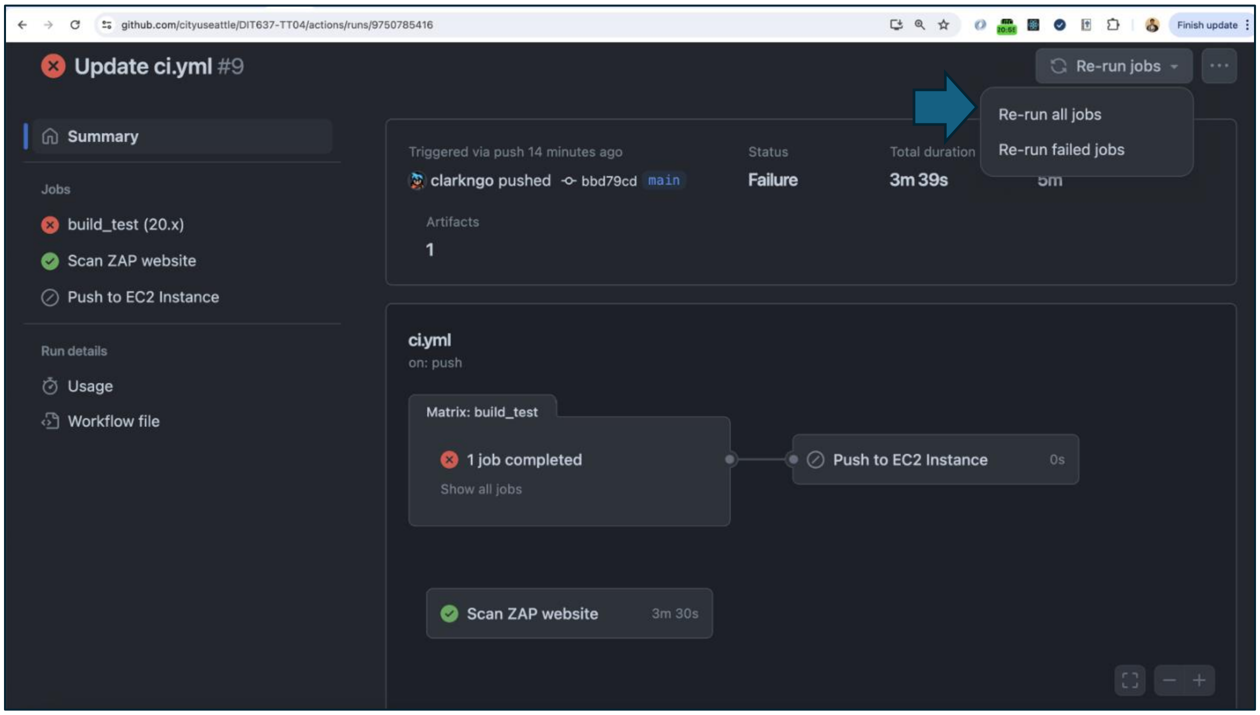
1. Go to Actions Tab.



2. Select the most recent workflow. If this is not available due to no initial workflow that was run skip this section and go to the step **triggering GitHub Actions**.

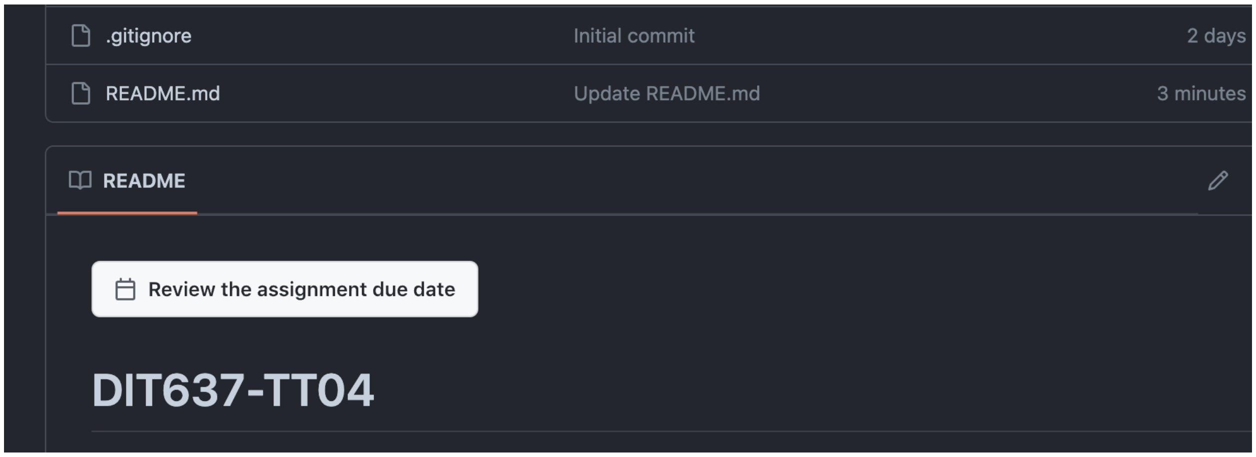


3. Click Re-run all jobs.



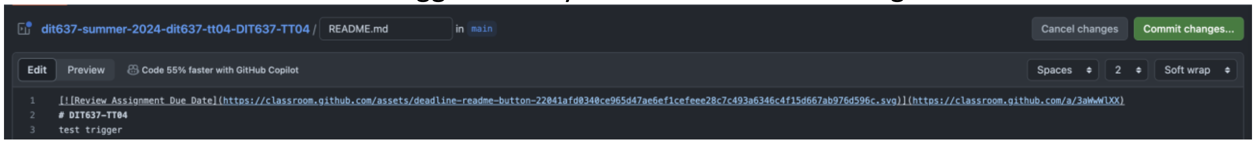
Triggering GitHub Actions

1. Edit your README.md to add any test.
   For example:



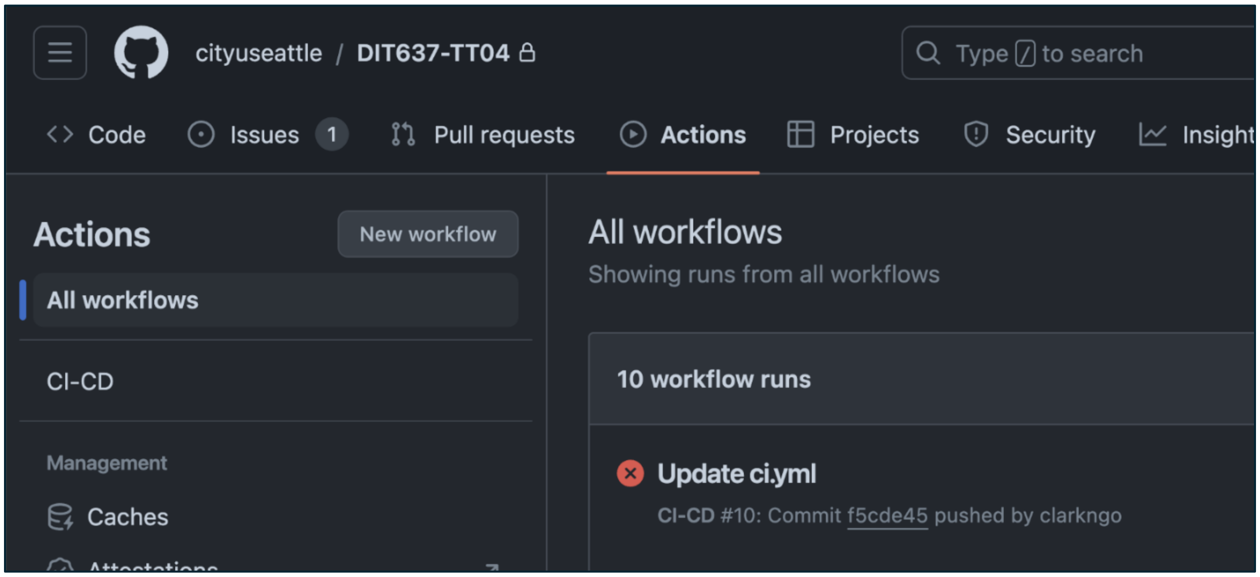
DIT637-TT04

2. Add the word “test trigger” or any word and Commit Changes

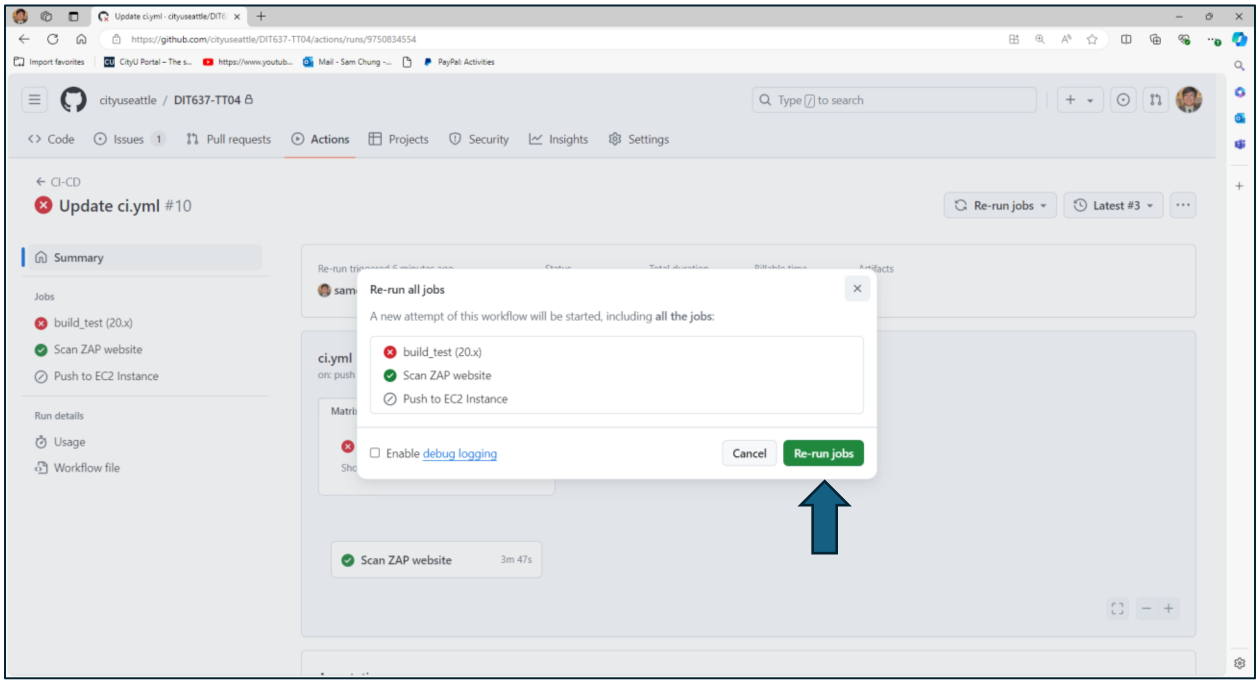

dit637-summer-2024-dit637-tt04-DIT637-TT04 / README.md in main
Cancel changes
Commit changes...
Edit
Preview
Code 55% faster with GitHub Copilot
Spaces
2
Soft wrap

```
[![Review Assignment Due Date](https://classroom.github.com/assets/deadline-readme-button-22041afd0340ce965d47ae6ef1cefeee28c7c493a6346c4f15d667ab976d596c.svg)](https://classroom.github.com/a/3aWwWlXX)
# DIT637-TT04
test trigger
```

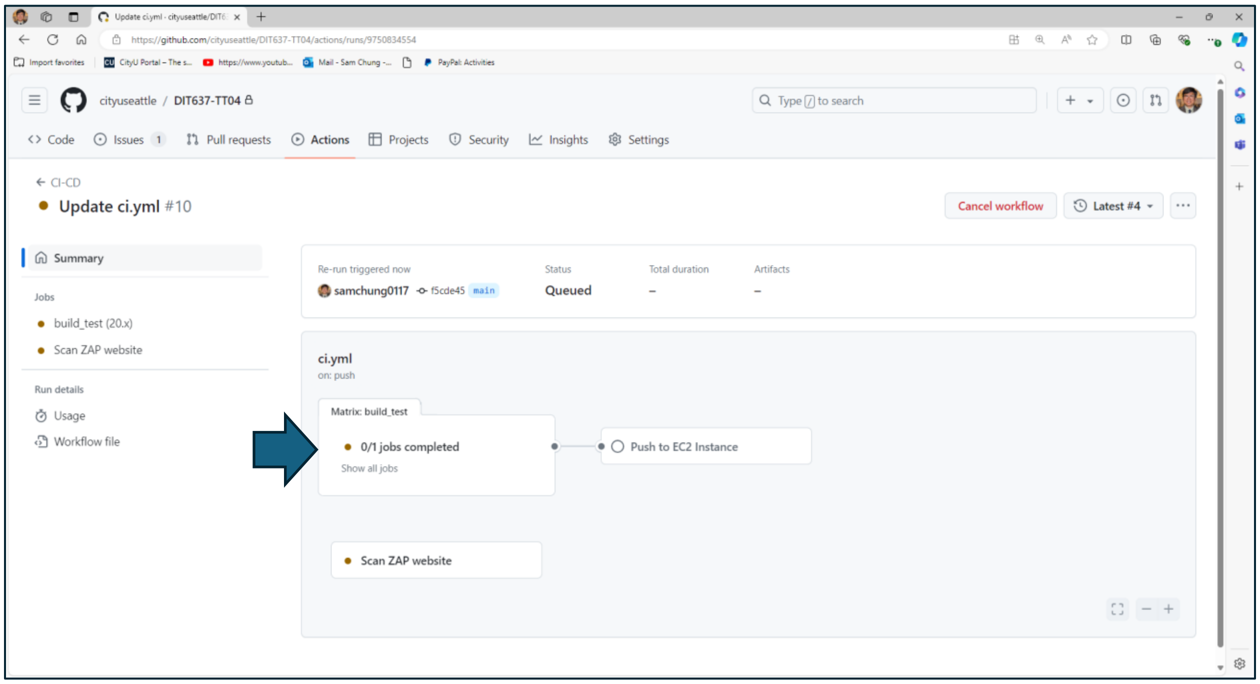
3. Check the Actions tab



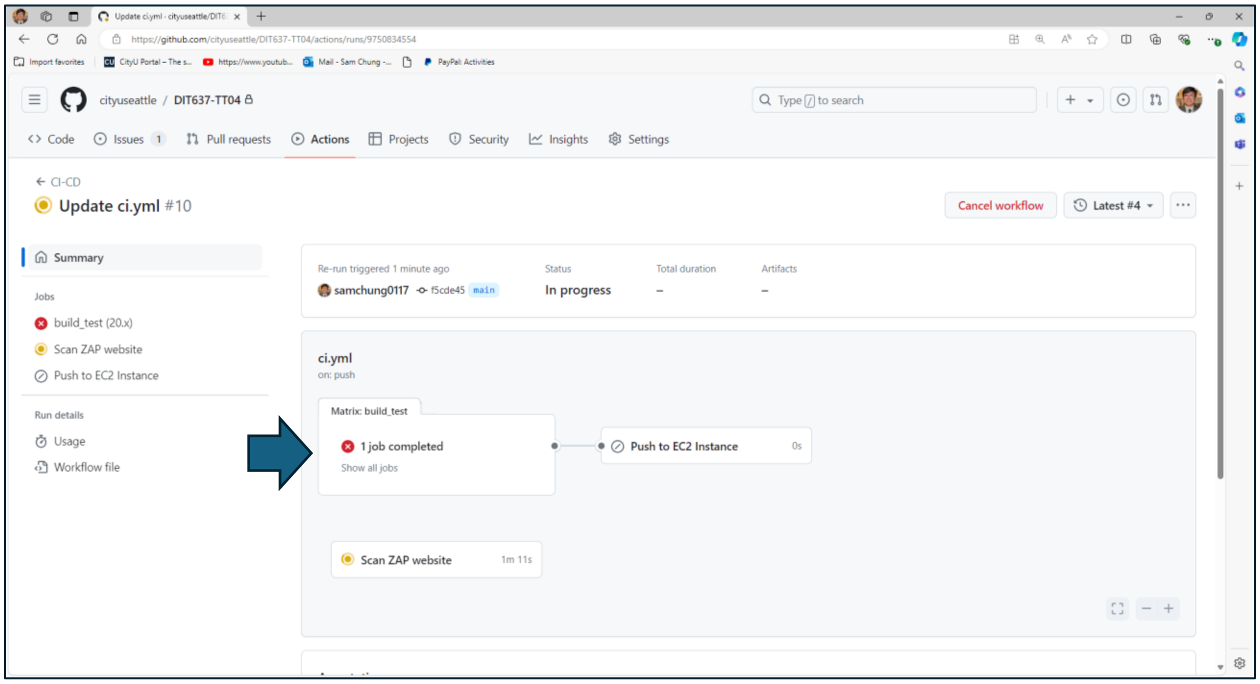
Update ci.yml
CI-CD #10: Commit f5cde45 pushed by clarkngo



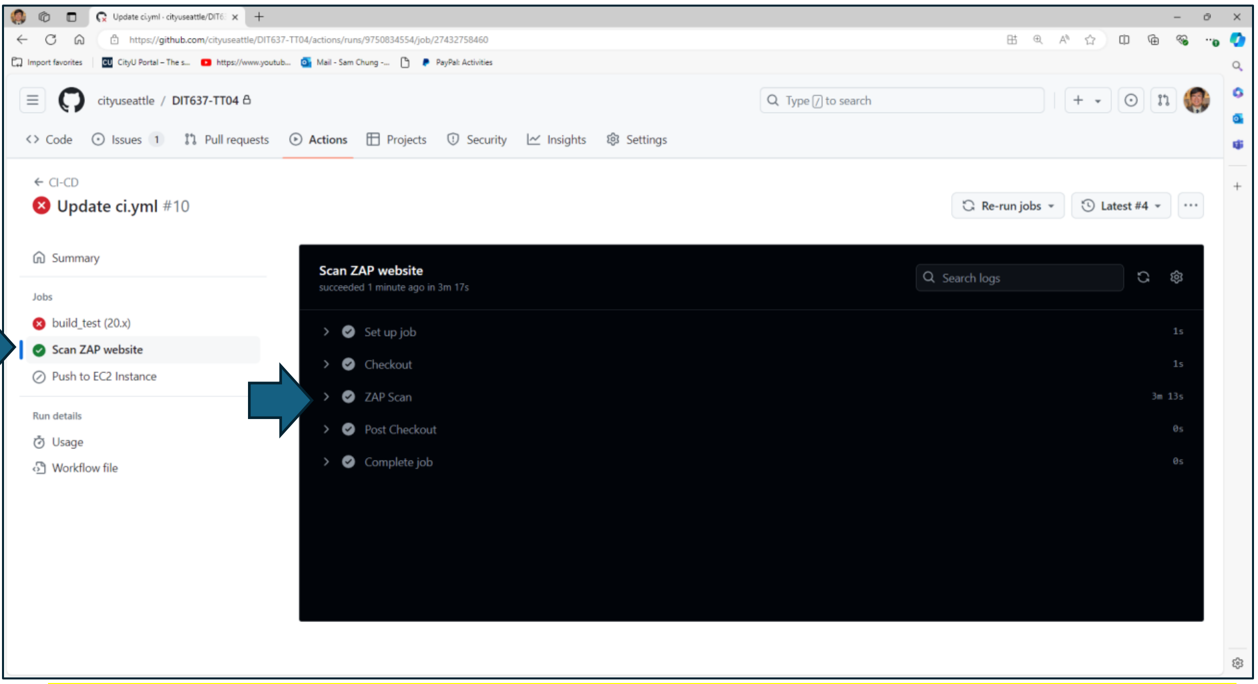
A new attempt of this workflow will be started, including **all the jobs**:



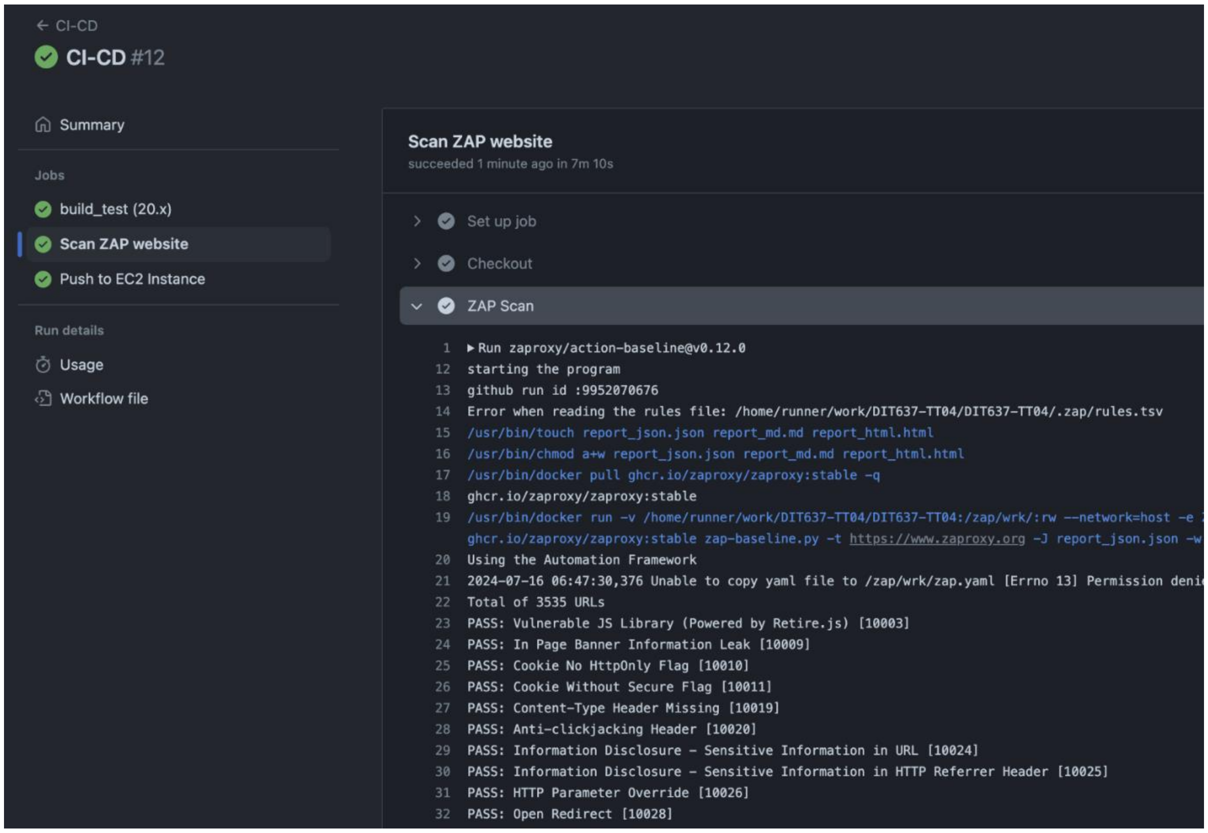
Re-run triggered now	Status	Total duration	Artifacts
samchung0117 f5cde45 main	Queued	–	–



4. Inspect the “Scan ZAP website” step.

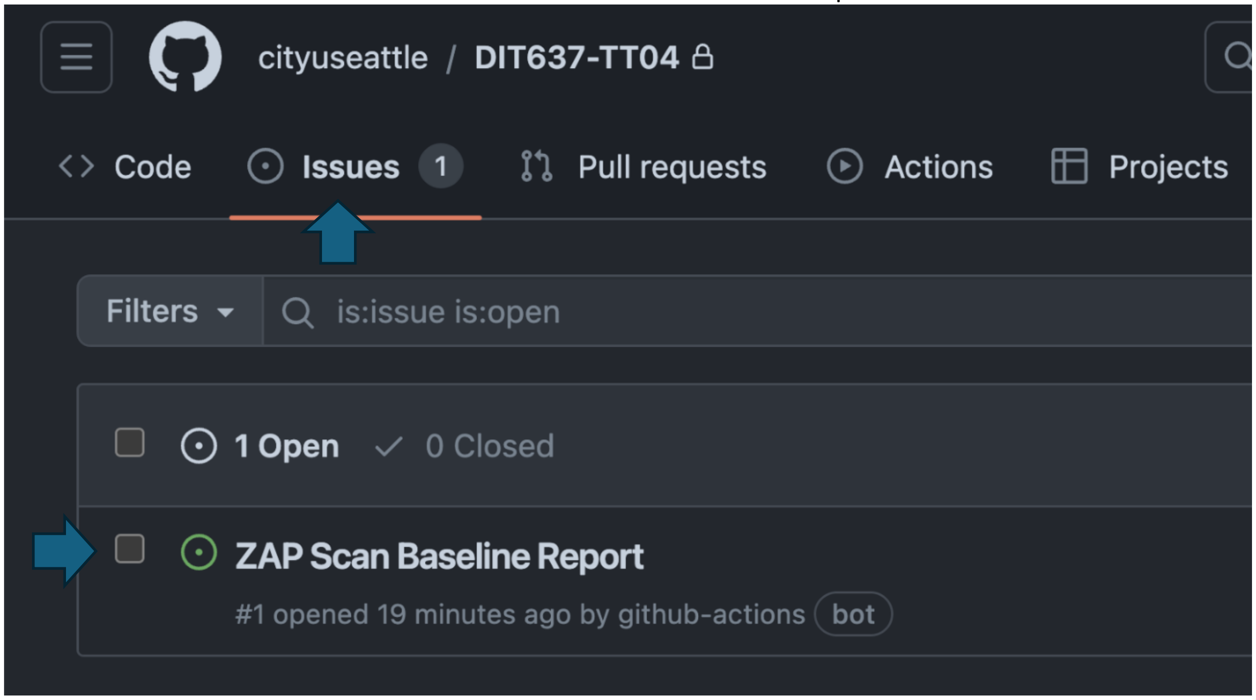


5. The screenshot of your ‘Scan ZAP Website – Zap Scan’ as ‘*first_last*_zap_scan.png’ by using your first and last name.



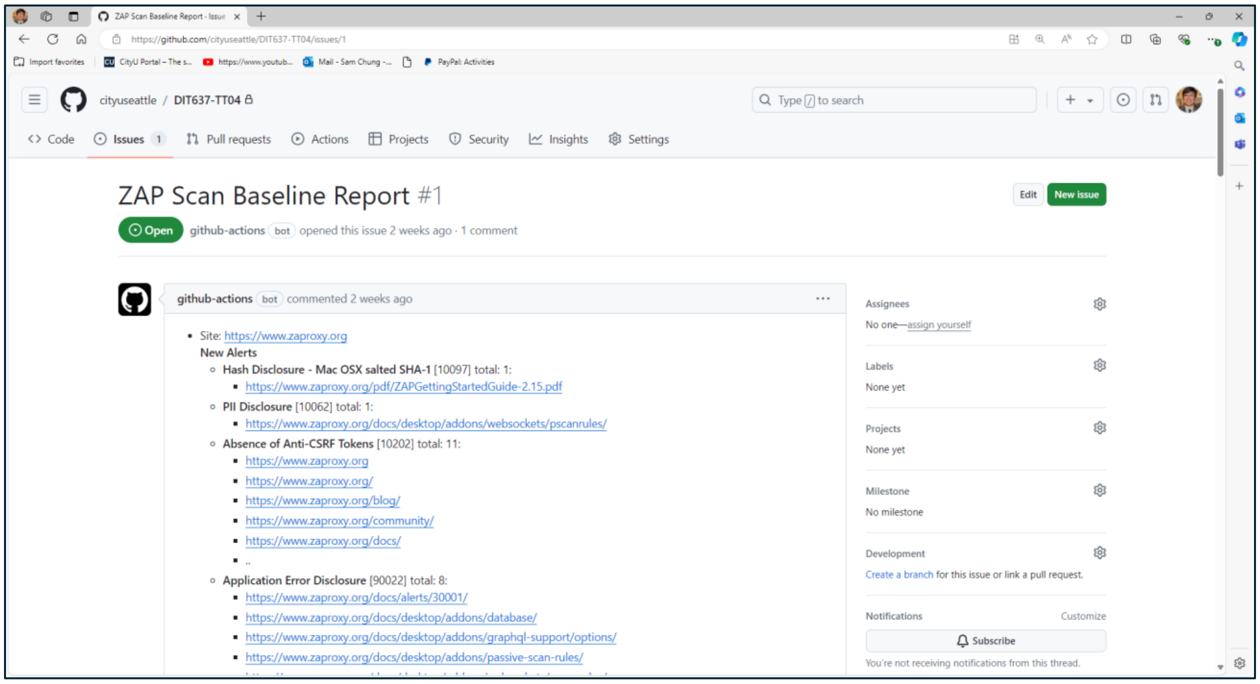
```
▶Run zaproxy/action-baseline@v0.12.0
starting the program
github run id :9952070676
Error when reading the rules file: /home/runner/work/DIT637-TT04/DIT637-TT04/.zap/rules.tsv
/usr/bin/touch report_json.json report_md.md report_html.html
/usr/bin/chmod a+w report_json.json report_md.md report_html.html
/usr/bin/docker pull ghcr.io/zaproxy/zaproxy:stable -q
ghcr.io/zaproxy/zaproxy:stable
/usr/bin/docker run -v /home/runner/work/DIT637-TT04/DIT637-TT04:/zap/wrk/:rw --network=host -e
ghcr.io/zaproxy/zaproxy:stable zap-baseline.py -t https://www.zaproxy.org -J report_json.json -w
Using the Automation Framework
2024-07-16 06:47:30,376 Unable to copy yaml file to /zap/wrk/zap.yaml [Errno 13] Permission deni
Total of 3535 URLs
PASS: Vulnerable JS Library (Powered by Retire.js) [10003]
PASS: In Page Banner Information Leak [10009]
PASS: Cookie No HttpOnly Flag [10010]
PASS: Cookie Without Secure Flag [10011]
PASS: Content-Type Header Missing [10019]
PASS: Anti-clickjacking Header [10020]
PASS: Information Disclosure - Sensitive Information in URL [10024]
PASS: Information Disclosure - Sensitive Information in HTTP Referrer Header [10025]
PASS: HTTP Parameter Override [10026]
PASS: Open Redirect [10028]
```

6. Head back to the Issues tab and Click on ZAP Scan Baseline Report.



ZAP Scan Baseline Report

#1 opened 19 minutes ago by github-actions bot



ZAP Scan Baseline Report #1

github-actions bot opened this issue 2 weeks ago · 1 comment

github-actions bot commented 2 weeks ago

- Site: https://www.zaproxy.org
  New Alerts
  - **Hash Disclosure - Mac OSX salted SHA-1** [10097] total: 1:
    - https://www.zaproxy.org/pdf/ZAPGettingStartedGuide-2.15.pdf
  - **PII Disclosure** [10062] total: 1:
    - https://www.zaproxy.org/docs/desktop/addons/websockets/pscanrules/
  - **Absence of Anti-CSRF Tokens** [10202] total: 11:
    - https://www.zaproxy.org
    - https://www.zaproxy.org/
    - https://www.zaproxy.org/blog/
    - https://www.zaproxy.org/community/
    - https://www.zaproxy.org/docs/
    - ..
  - **Application Error Disclosure** [90022] total: 8:
    - https://www.zaproxy.org/docs/alerts/30001/
    - https://www.zaproxy.org/docs/desktop/addons/database/
    - https://www.zaproxy.org/docs/desktop/addons/graphql-support/options/
    - https://www.zaproxy.org/docs/desktop/addons/passive-scan-rules/

Assignees
No one—assign yourself

Labels
None yet

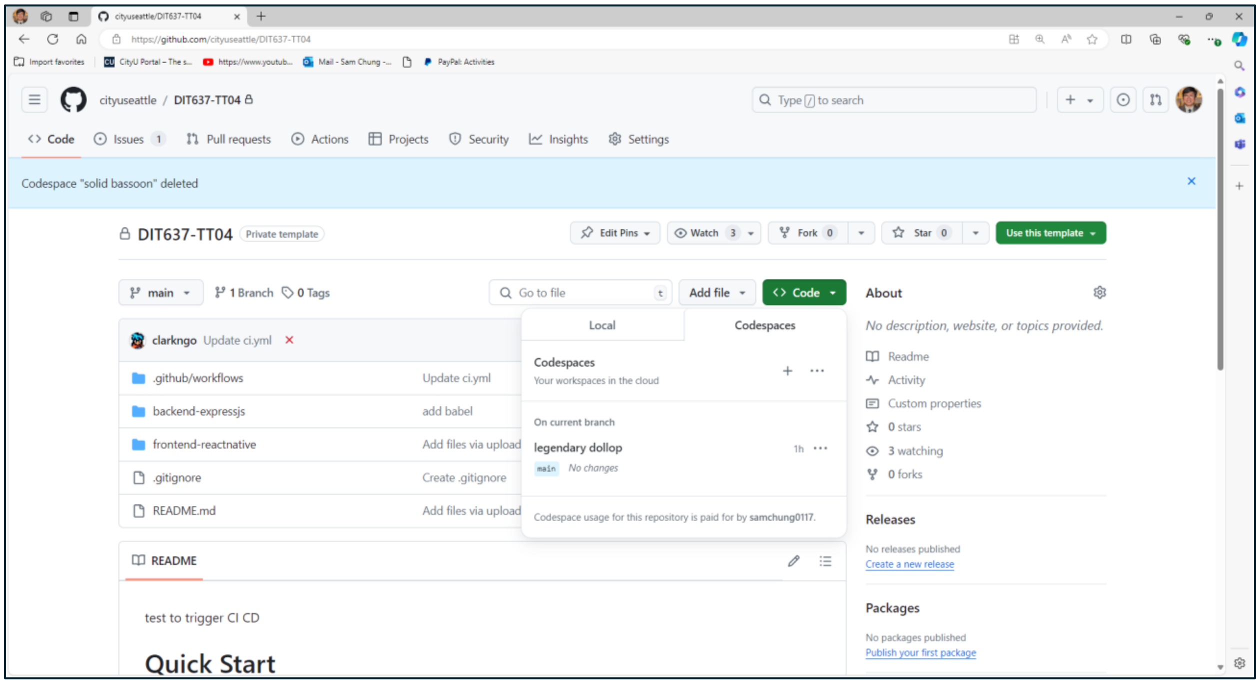
Projects
None yet

Milestone
No milestone

Development
Create a branch for this issue or link a pull request.

Notifications
Customize
Subscribe
You're not receiving notifications from this thread.

3) Breakdown of Workflow Trigger for GitHub Actions CI/CD



Codespace "solid bassoon" deleted

DIT637-TT04 Private template

clarkngo Update ci.yml

.github/workflows	Update ci.yml
backend-expressjs	add babel
frontend-reactnative	Add files via upload
.gitignore	Create .gitignore
README.md	Add files via upload

Codespaces
Your workspaces in the cloud

On current branch
legendary dollop 1h
main No changes

Codespace usage for this repository is paid for by samchung0117.

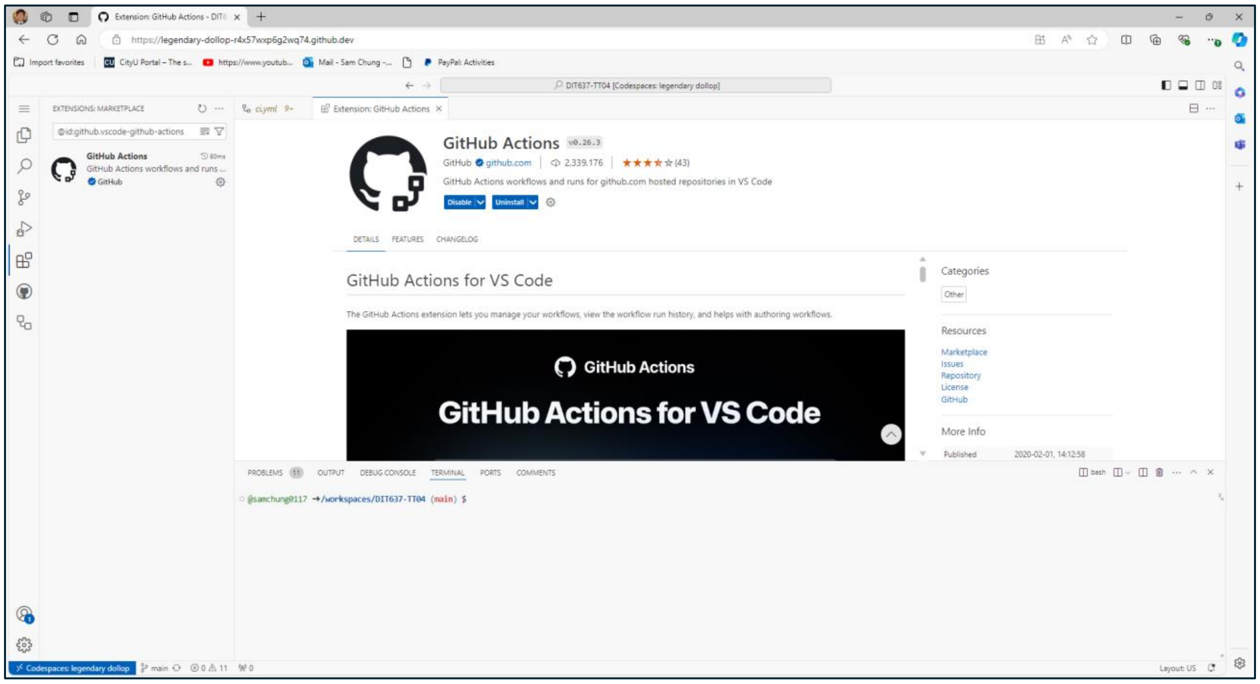
No description, website, or topics provided.

No releases published

No packages published

test to trigger CI CD

Quick Start


Extension: GitHub Actions - DIT6
https://legendary-dollop-r4x57wxp6g2wq74.github.dev
DIT637-TT04 [Codespaces: legendary dollop]
EXTENSIONS: MARKETPLACE
@id:github.vscode-github-actions
ci.yml
Extension: GitHub Actions
Disable
Uninstall
DETAILS
FEATURES
CHANGELOG
Categories
Other
Resources
Marketplace
Issues
Repository
License
GitHub
More Info
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
COMMENTS
Codespaces: legendary dollop
main
Layout: US

GitHub Actions v0.26.3
GitHub github.com | 2,339,176 | ★★★★☆ (43)
GitHub Actions workflows and runs for github.com hosted repositories in VS Code

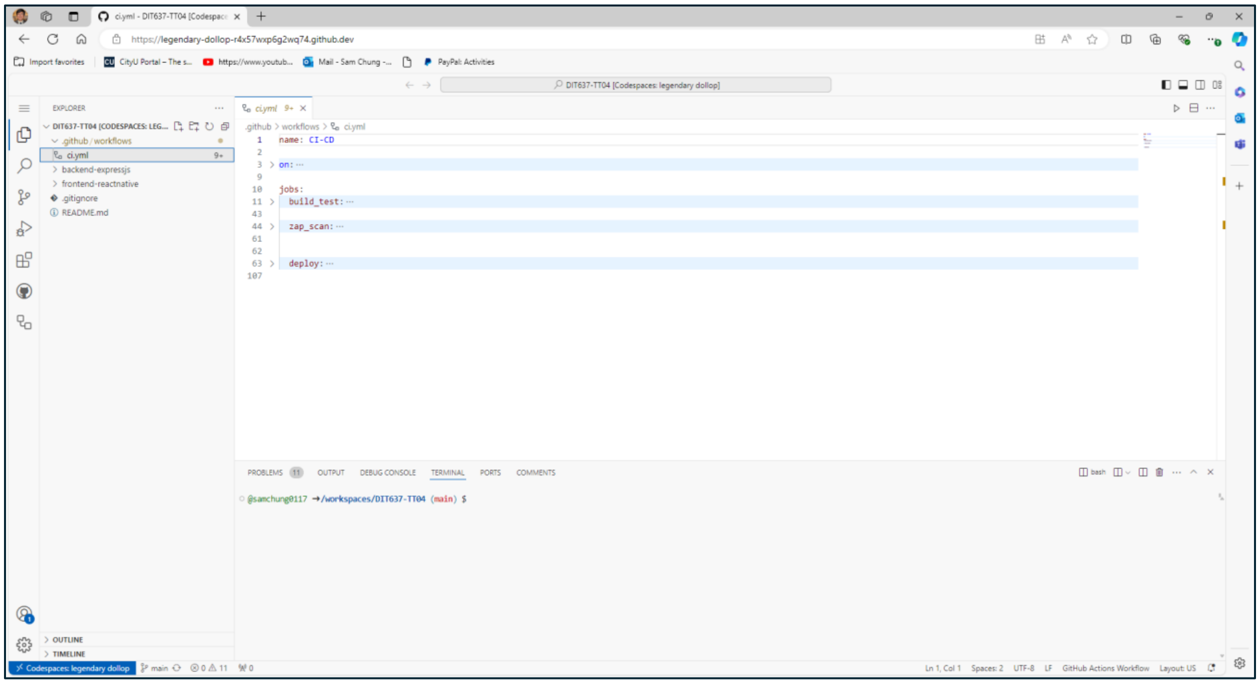
GitHub Actions for VS Code

The GitHub Actions extension lets you manage your workflows, view the workflow run history, and helps with authoring workflows.

GitHub Actions
GitHub Actions for VS Code

Published 2020-02-01, 14:12:58

```
@samchung0117 →/workspaces/DIT637-TT04 (main) $
```



```yaml
name: CI-CD

on: ···

jobs:
  build_test: ···

  zap_scan: ···


  deploy: ···
```

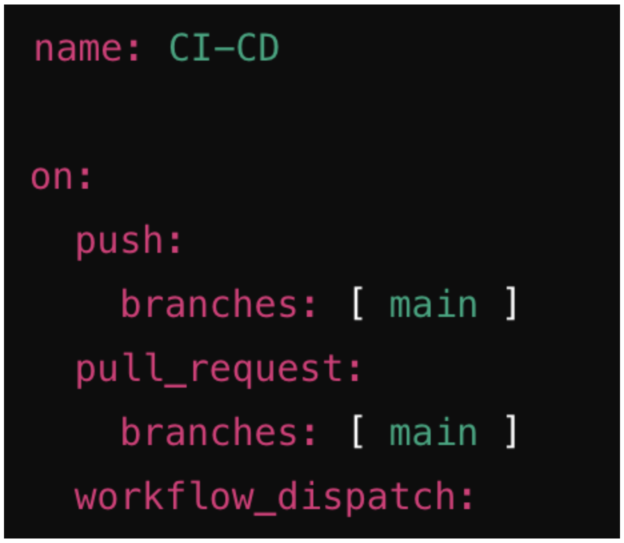
```
@samchung0117 →/workspaces/DIT637-TT04 (main) $
```

The GitHub Actions workflow defines a CI/CD pipeline for a Node.js (Express.js) application. It includes two jobs:

1. **build_test**: Checks out the code, sets up Node.js, installs dependencies, prints environment variables, and runs tests.
2. **deploy**: Checks out the code, creates a .env file, sets up SSH, deploys the code to an EC2 instance, installs dependencies, and starts the backend server on the EC2 instance.

The workflow is triggered on pushes and pull requests to the main branch, as well as manually.

Workflow Trigger



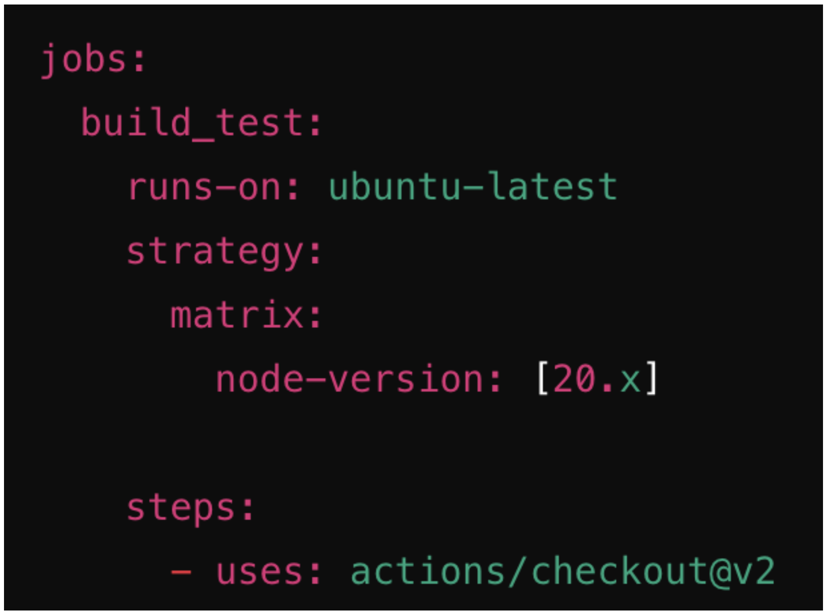
```yaml
name: CI-CD

on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]
  workflow_dispatch:
```

- **name**: The name of the workflow.
- **on**: Specifies the events that trigger the workflow.
    - **push**: Triggers the workflow on pushes to the main branch.

- **pull_request**: Triggers the workflow on pull requests targeting the main branch.
- **workflow_dispatch**: Allows manual triggering of the workflow.

Jobs Section

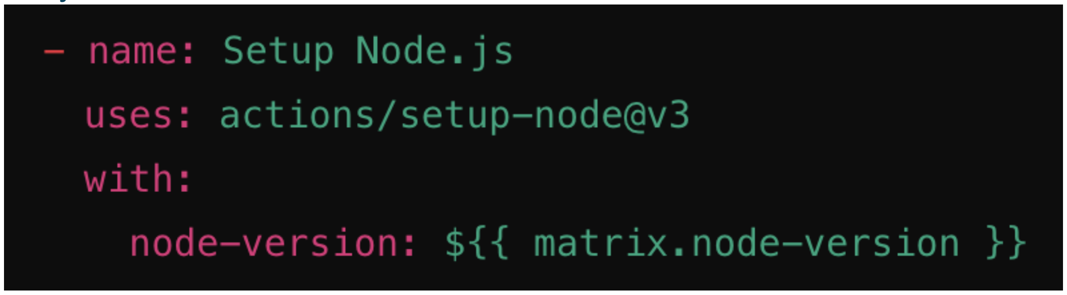
Build and Test Job



```yaml
jobs:
  build_test:
    runs-on: ubuntu-latest
    strategy:
      matrix:
        node-version: [20.x]

    steps:
      - uses: actions/checkout@v2
```

- **jobs**: Defines the jobs to be run in the workflow.
- **build_test**: The name of the job responsible for building and testing the application.
- **runs-on**: Specifies the type of runner to use, here it's ubuntu-latest.
- **strategy.matrix**: Allows running the job with different versions of Node.js (here it's version 20.x).
- **steps**: Defines the individual steps to be executed in the job.
  - **actions/checkout@v2**: Checks out the repository's code.

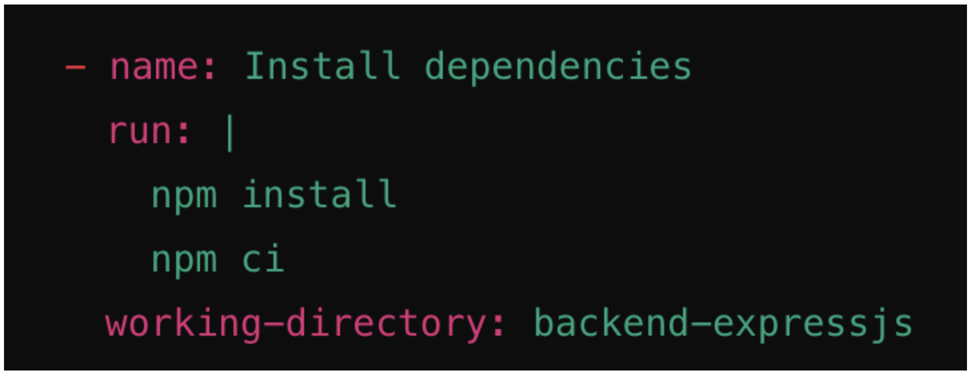
Setup Node.js



```yaml
- name: Setup Node.js
  uses: actions/setup-node@v3
  with:
    node-version: ${{ matrix.node-version }}
```

- **actions/setup-node@v3**: Sets up the specified version of Node.js.

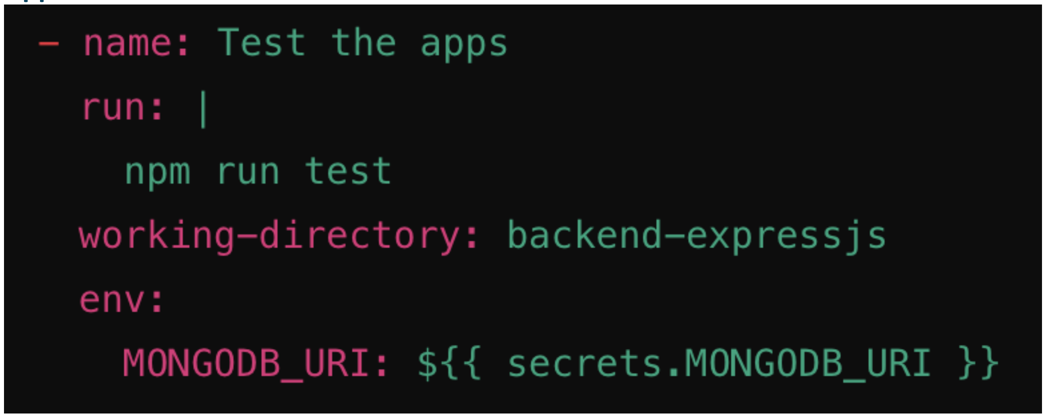
Install Dependencies



```
- name: Install dependencies
  run: |
    npm install
    npm ci
  working-directory: backend-expressjs
```

- **run**: Runs shell commands.
- **working-directory**: Specifies the directory where the commands should be executed.
    - **npm install**: Installs project dependencies.
    - **npm ci**: Cleans the node_modules folder and installs dependencies.

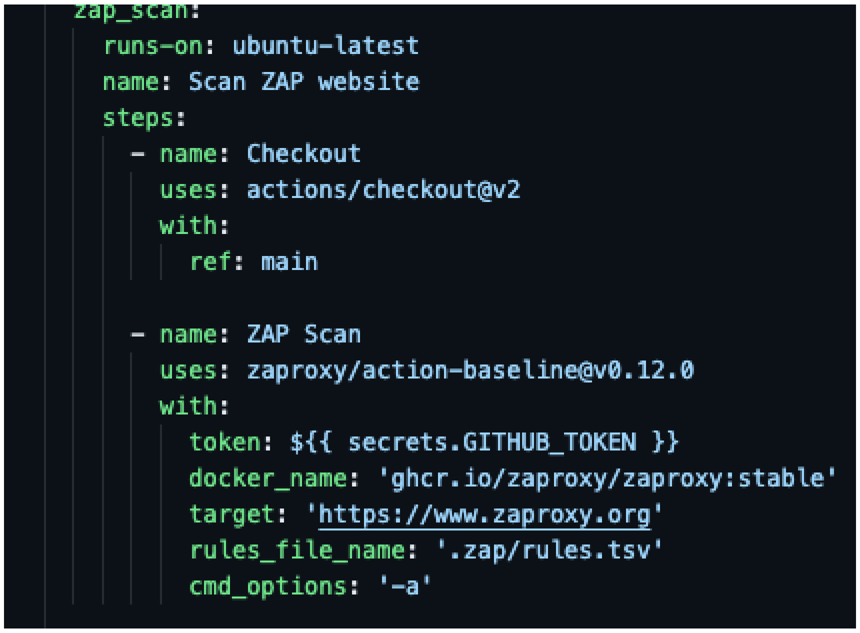
Test the Application



```
- name: Test the apps
  run: |
    npm run test
  working-directory: backend-expressjs
  env:
    MONGODB_URI: ${{ secrets.MONGODB_URI }}
```

- **npm run test**: Runs the test script defined in package.json.
- **env:** uses the MongoDB connection string in GitHub Action secrets named MONGODB_URI

Web App Scanner



```
zap_scan:
  runs-on: ubuntu-latest
  name: Scan ZAP website
  steps:
    - name: Checkout
      uses: actions/checkout@v2
      with:
        ref: main

    - name: ZAP Scan
      uses: zaproxy/action-baseline@v0.12.0
      with:
        token: ${{ secrets.GITHUB_TOKEN }}
        docker_name: 'ghcr.io/zaproxy/zaproxy:stable'
        target: 'https://www.zaproxy.org'
        rules_file_name: '.zap/rules.tsv'
        cmd_options: '-a'
```

- **zap_scan:** This is the name of the job within the GitHub Actions workflow.
- **runs-on: ubuntu-latest** Specifies the environment in which the job will run. In this case, it uses the latest version of Ubuntu.

Steps

- **Checkout Code**
  - **name: Checkout** This step is named "Checkout".
  - **uses: actions/checkout@v2** Uses the actions/checkout action version 2 to check out the code from the repository.
  - **with:** Specifies additional options for the checkout action.
    - **ref: main** Checks out the main branch of the repository.
- **ZAP Scan**
  - **name: ZAP Scan** This step is named "ZAP Scan".
  - **uses: zaproxy/action-baseline@v0.12.0** Uses the zaproxy/action-baseline action version 0.12.0 to perform a baseline scan.
  - **with:** Specifies additional options for the ZAP baseline scan.
    - **token: ${{ secrets.GITHUB_TOKEN }}** Passes the GitHub token to the action, which is necessary for authentication and accessing repository resources.
    - **docker_name: 'ghcr.io/zaproxy/zaproxy:stable'** Specifies the Docker image to use for running ZAP. In this case, it uses the stable version of ZAP from GitHub Container Registry.
    - **target: 'https://www.zaproxy.org'** Sets the target URL for the scan. Here, it is the ZAP website.
    - **rules_file_name: '.zap/rules.tsv'** Specifies a rules file that contains customized rules for the scan. The file is located in the .zap directory and is named rules.tsv.
    - **cmd_options: '-a'** Passes additional command-line options to ZAP. The -a option tells ZAP to run the active scan after the passive scan.

4) Pushing your work to GitHub

1. Go to Source Control on your GitHub codespaces and observe the pending changes.

2. Type the Message for your changes in the Message box on the top. For example,” **Submission for TT04 – Your Name**”

3. Click on the dropdown beside the commit button and select **Commit & Push** to update the changes to your repository main branch.

4. Select **Yes** when prompted.