

Configuration Manual

MSc Practicum Part 2
MSc Cybersecurity

Praveen Derenda Seetharam
Student ID: 23174501

School of Computing
National College of Ireland

Supervisor: Vikas Sahni

National College of Ireland
MSc Project Submission Sheet



School of Computing

Praveen Derenda Seetharam

Student Name:
23174501
Student ID:
MSc Cybersecurity 2024-2025
Programme: **Year:**
MSc Practicum Part 2
Module:
Vikas Sahni
Supervisor:
Submission Date: 12/12/2024
Project Title: Developing a Framework for Integrating Security
Testing into the CI/CD Pipeline using Automation.
.....
849 6
Word Count: **Page Count:**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Praveen D S
11/12/2024
Date:

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Praveen Derenda Seetharam

Student ID: 23174501

1 Introduction

This manual gives a step-by-step guide to configure and set up the OWASP Juice Shop web-application and use the secure CI/CD pipeline framework developed for integrating security testing into the CI/CD process. The project uses the following tools to ensure a secure development lifecycle:

- **GitHub Actions** -- <https://github.com/features/actions>
- **CodeQL** -- <https://codeql.github.com/>
- **Snyk** -- <https://snyk.io/>
- **OWASP ZAP** -- <https://www.zaproxy.org/>
- **Docker** -- <https://www.docker.com/>

2 System Configuration

- **Operating System:** macOS Sequoia
- **Hardware Configuration:**
 - Chip: M3
 - Memory: 16 GB
 - Storage: 500 GB
- **Software Configuration:**
 - **Docker:** Version 27.2.0, build 3ab4256
 - **GitHub:** git version 2.44.0
 - **Node.js:** Version v23.1.0
 - **OWASP ZAP:** Version 2.15.0

3 Installation and Setup

OWASP Juice shop Setup:

To install and configure the OWASP Juice shop application on the local machine, the following are the requirements.

Prerequisites:

1. Install node.js¹ in your machine from the official website.

Installations:

The following are the steps for ease installation and deployment on localhost.

Step 1: Clone the git repository² into your machine from 1994dsp OWASP repository.

Step 2: Go into the cloned folder with `cd juice-shop`.

Step 3: Run `npm install` (must be done before first start or when you change the source code)

Step 4: Run `npm start`

Step 5: Browse to `http://localhost:3000`

¹ <https://nodejs.org/en/download/package-manager>

² <https://github.com/1994dsp/Owasp>

Installing Dependencies

Docker Installation:

Download Docker³ desktop from Official Website and follow the installation instructions for your OS.

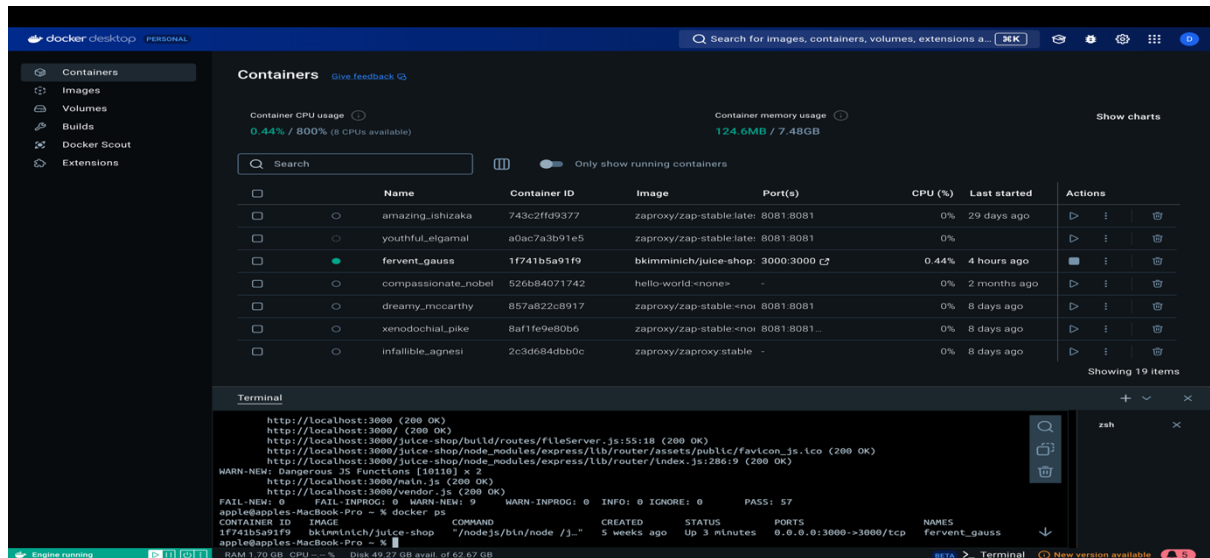


Figure 1: Docker Desktop for mac

4 Framework Configuration

GitHub Repository Configuration

1. Create a **GitHub Repository** and clone it to your local machine , here in this project I am using Visual Studio Code for the local development work.
2. Go to your project and Add a `.github/workflows/` directory to store CI/CD pipeline YAML files as shown in the below figure.

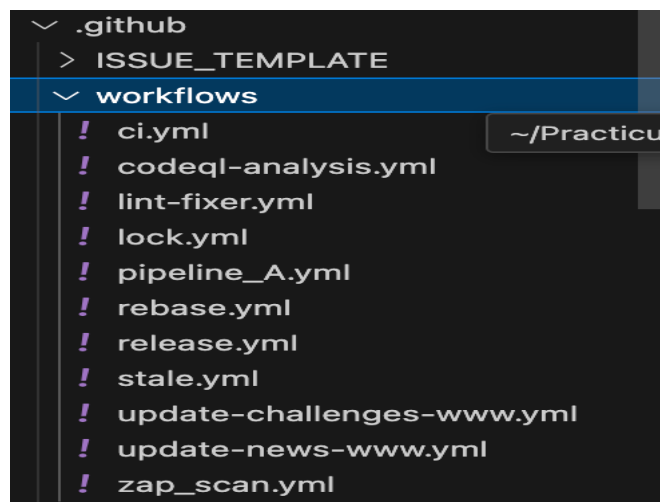


Figure 2:YML files under .github/workflows

³ <https://www.docker.com/>

GitHub Actions Workflow

1. Create a ci.yml file in .github/workflows/.
2. Add configurations for:
 - **Code Linting:** Using ESLint
 - **Static Analysis:** Using CodeQL
 - **Dependency Scanning:** Using Snyk

```
.github > workflows > ! lint-fixer.yml
1  name: "Let me lint:fix that for you"
2
3  on: [push]
4
5  jobs:
6    LMLFTFY:
7      runs-on: ubuntu-latest
8      steps:
9        - name: "Check out Git repository"
10         uses: actions/checkout@3df4ab11eba7bda6032a0b82a6bb43b11571feac #v4.0.0
11        - name: "Use Node.js 18"
12         uses: actions/setup-node@5e21ff4d9bc1a8cf6de233a3057d20ec6b3fb69d #v3.8.1
13         with:
14           node-version: 18
15        - name: "Install CLI tools"
16         run: npm install -g @angular/cli
17        - name: "Install application"
18         run: |
19           npm install --ignore-scripts
20           cd frontend
21           npm install --ignore-scripts --legacy-peer-deps
22        - name: "Fix everything which can be fixed"
23         run: 'npm run lint:fix'
24        - uses: stefanzweifel/git-auto-commit-action@3ea6ae190baf489ba007f7c92608f33ce20ef04a #v4.16.0
25         with:
26           commit_message: "Auto-fix linting issues"
27           branch: ${GITHUB_HEAD_REF}
28           commit_options: '--signoff'
29           commit_user_name: JuiceShopBot
30           commit_user_email: 61591748+JuiceShopBot@users.noreply.github.com
31           commit_author: JuiceShopBot <61591748+JuiceShopBot@users.noreply.github.com>
32
```

Figure 3:Lint config file

Security Tools Integration

Static Code Analysis (CodeQL):

- Enable CodeQL in your repository's Security Settings.
- Add CodeQL to your ci.yml

```
.github > workflows > ! codeql-analysis.yml
1  name: "CodeQL Scan"
2
3  on:
4    push:
5    pull_request:
6
7  jobs:
8    analyze:
9      name: Analyze
10     runs-on: ubuntu-latest
11     permissions:
12       actions: read
13       contents: read
14       security-events: write
15     strategy:
16       fail-fast: false
17     matrix:
18       language: [ 'javascript-typescript' ]
19     steps:
20       - name: Checkout repository
21         uses: actions/checkout@3df4ab11eba7bda6032a0b82a6bb43b11571feac #v4.0.0
22       - name: Initialize CodeQL
23         uses: github/codeql-action/init@v2
24         with:
25           languages: ${GITHUB_TOKEN}
26           queries: security-extended
27           config: |
28             paths-ignore:
29               - 'data/static/codefixes'
30       - name: Autobuild
31         uses: github/codeql-action/autobuild@v2
32       - name: Perform CodeQL Analysis
33         uses: github/codeql-action/analyze@v2
```

Figure 4:CodeQL.yml file

Dependency Scanning (Snyk):

Add Snyk extension into your Visual studio Code

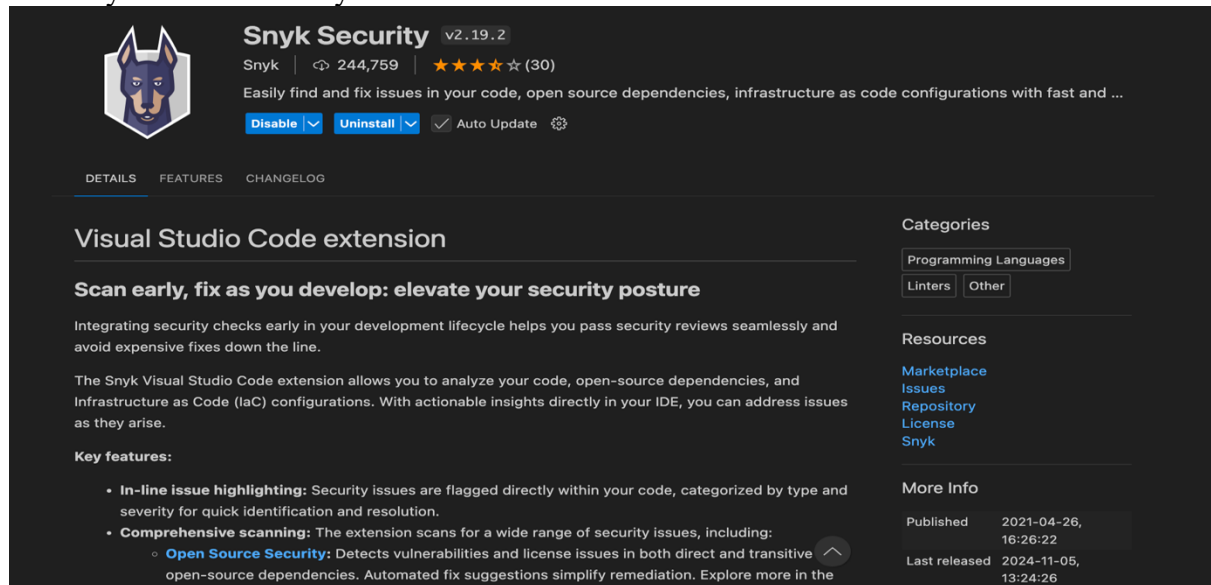


Figure 5:Snyk extension

Dynamic Security Testing (OWASP ZAP):

Add OWASP ZAP to your ci.yml:

```
- name: "Run Juice Shop Application"
  run: |
    docker run -d --name juice-shop-container -p 3000:3000 dsp1994/owasp:snapshot
```

5 Usage Instructions

1. Push Changes:

- Commit and push code to the repository:
git add .
git commit -m "Initial CI/CD pipeline setup"
git push origin main

2. Monitor Workflow:

- Navigate to the **Actions** tab in GitHub to view pipeline progress.

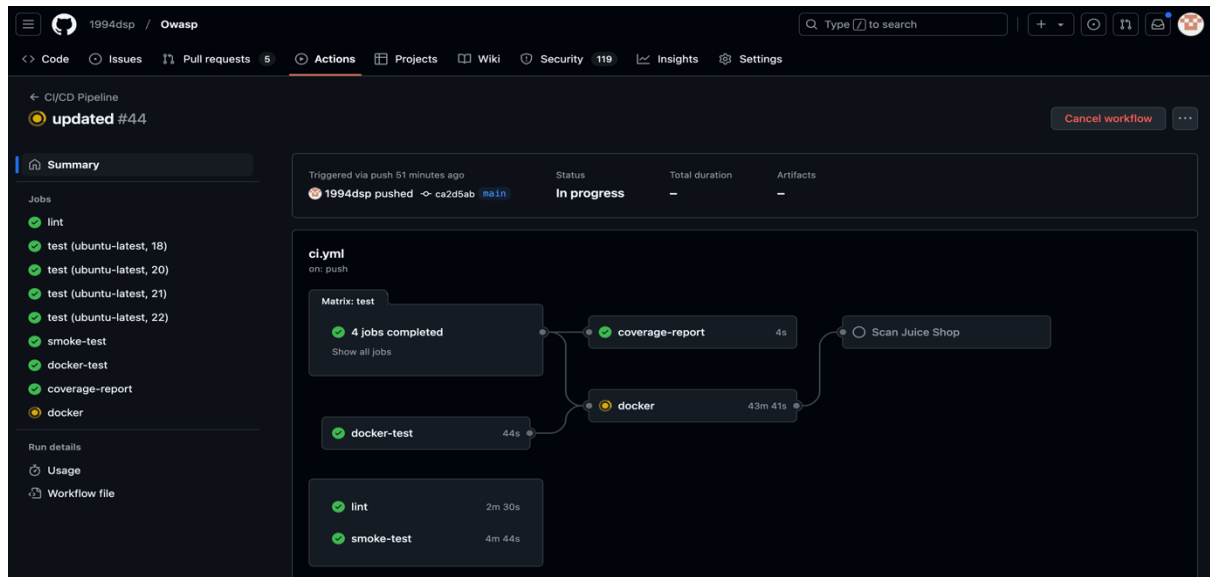


Figure 6: CICD workflow

3. Generate Reports:

- Review logs and reports generated by tools like CodeQL, Snyk, and OWASP ZAP.

6 Validation and Testing

1. Validate Static Analysis:

- Push insecure code (e.g., with SQL injection) and ensure CodeQL flags it.

<input type="checkbox"/>	<input type="info"/>	Hard-coded credentials Critical	main
#78 opened 2 months ago • Detected by CodeQL in lib/insecurity.ts :44			
<input type="checkbox"/>	<input type="info"/>	Hard-coded credentials Critical	main
#77 opened 2 months ago • Detected by CodeQL in lib/insecurity.ts :23			
<input type="checkbox"/>	<input type="info"/>	Hard-coded credentials Critical	main
#76 opened 2 months ago • Detected by CodeQL in lib/insecurity.ts :22			
<input type="checkbox"/>	<input type="info"/>	Server-side request forgery Critical	main
#29 opened 2 months ago • Detected by CodeQL in routes/profileImageUrlUpload.ts :22			
<input type="checkbox"/>	<input type="info"/>	Type confusion through parameter tampering Critical	main
#24 opened 2 months ago • Detected by CodeQL in routes/search.ts :22			
<input type="checkbox"/>	<input type="info"/>	Type confusion through parameter tampering Critical	main
#23 opened 2 months ago • Detected by CodeQL in lib/insecurity.ts :138			
<input type="checkbox"/>	<input type="info"/>	Code injection Critical	main
#10 opened 2 months ago • Detected by CodeQL in routes/showProductReviews.ts :34			
<input type="checkbox"/>	<input type="info"/>	Code injection Critical	main
#9 opened 2 months ago • Detected by CodeQL in routes/trackOrder.ts :17			
<input type="checkbox"/>	<input type="info"/>	Template Object Injection Critical	main
#3 opened 2 months ago • Detected by CodeQL in routes/dataErasure.ts :87			
<input type="checkbox"/>	<input type="info"/>	Template Object Injection Critical	main
#2 opened 2 months ago • Detected by CodeQL in routes/dataErasure.ts :72			
<input type="checkbox"/>	<input type="info"/>	Polynomial regular expression used on uncontrolled data High	main
#87 opened 2 months ago • Detected by CodeQL in routes/profileImageUrlUpload.ts :19			
<input type="checkbox"/>	<input type="info"/>	Database query built from user-controlled sources High	main
#72 opened 2 months ago • Detected by CodeQL in routes/search.ts :23			

Figure 7:CodeQL report

2. Run Dependency Scans:

- Use outdated libraries in `package.json` to validate Snyk scans.

▼

1994dsp/Owasp

8

C

51

H

58

M

221

L

...

Project

Imported

Tested

Issues ↓

package.json

an hour ago

an hour ago

5

C

22

H

27

M

3

L

...

frontend/package.json

an hour ago

an hour ago

3

C

4

H

2

M

0

L

...

Code analysis

an hour ago

an hour ago

0

C

25

H

29

M

218

L

...

Dockerfile

an hour ago

an hour ago

0

C

0

H

0

M

0

L

...

Figure 8:Snyk report

3. Test Runtime Security:

- Deploy OWASP Juice Shop and run OWASP ZAP against it.

ZAP Scanning Report		
Sites: http://preview.owasp-juice.shop https://preview.owasp-juice.shop		
Generated on Fri, 29 Nov 2024 00:39:35		
ZAP Version: 2.15.0		
ZAP by Checkmarx		
Summary of Alerts		
Risk Level	Number of Alerts	
High	0	
Medium	4	
Low	7	
Informational	7	
False Positives:	0	
Alerts		
Name	Risk Level	Number of Instances
COBS_Misconfiguration	Medium	12
Content_Security_Policy_(CSP)_Header_Not_Set	Medium	3
Cross-Domain_Misconfiguration	Medium	9
Proxy_Disclosure	Medium	13
Cross-Domain_JavaScript_Source_File_Inclusion	Low	4
Dangerous_JS_Functions	Low	2
Deprecated_Feature_Policy_Header_Set	Low	6
HTTPS_Content_Available_via_HTTP	Low	7
Permissions_Policy_Header_Not_Set	Low	1
Strict_Transport_Security_Header_Not_Set	Low	10
Timestamp_Disclosure_-_Unix	Low	11

Figure 9: ZAP report.

7 Conclusion

This manual equips developers with the knowledge to configure a secure CI/CD pipeline. By automating testing and integrating robust tools, this framework ensures efficient and secure software delivery.

References

Decan, A., Mens, T., Mazrae, P.R. and Golzadeh, M., 2022, October. On the Use of GitHub Actions in Software Development Repositories. In 2022 IEEE International Conference on Software Maintenance and Evolution (ICSME) (pp. 235-245). IEEE.