National College of Ireland

Configuration Manual

MSc Research Project
MSCCYB

Pratham Deepak Shah
Student ID: 23275235

School of Computing
National College of Ireland

Supervisor: Evgeniia Jayasekera

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Pratham Deepak Shah
Student ID:	23275235
Programme:	MSc Cybersecurity **Year:** 2025
Module:	Practicum
Lecturer:	Evgeniia Jayasekera
Submission Due Date:	11th December 2025
Project Title:	A DevSecOps Framework for Automated Container Security in CI/CD Pipelines
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: .....

Date: 08/12/2025 .....

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
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.	□

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

Pratham Deepak Shah
23275235

1 Configuration Manual

This configuration manual specifies the technical configuration and operational steps required to replicate the security analysis conducted as part of the research project. All experiments were executed on a Windows 11 host; no virtual machines or WSL distributions were required. The manual is written to enable independent replication by researchers with access to equivalent hardware and software.

1.1 Scope

The manual documents the precise software versions, folder layout, and command sequences used to: (i) build the experimental container image, (ii) execute automated scans (Trivy, Dockle, Kube-linter), and (iii) collect and store machine-readable results for analysis.

2 System specifications

The hardware and operating system used for development and testing are summarised below.

- **Operating System:** Windows 11 Pro, Version 10.0.26200
- **Processor:** Intel® Core™ i5-8250U, 1.60 GHz (4 cores, 8 threads)
- **Installed RAM:** 16 GB
- **System Type:** 64-bit, x64 architecture
- **Virtualisation support:** Enabled (required for Docker Desktop)

3 Required software

Reproducibility requires installation of the following software components on the Windows host.

3.1 Windows components

- Windows 11 Pro (64-bit)
- Windows Terminal or PowerShell
- Docker Desktop for Windows (stable release, v27.x recommended)

3.2 Development tools

- Visual Studio Code (latest stable release) with the following extensions:
  - Docker (Microsoft)
  - YAML language support
  - Kubernetes Tools (optional)

  - Node.js extension pack
- Node.js (LTS) — required for the sample application prior to containerisation

3.3 Security tooling

- Trivy (v0.50+), installed as a Windows binary
- Dockle (v0.4.x), installed as a Windows binary
- Kube-linter, executed via the official container image (ghcr.io/stackrox/kube-linter:latest)

4 Directory structure

The project repository used during experimentation follows the layout below. The absolute path used in the original experiments is provided; substitute the local user directory as appropriate when reproducing the experiment.

`C:\Users\prath\OneDrive\Desktop\Masters\Sem 3\Automated_Container_Hardening_CI\`

```
Dockerfile
app\  (Node.js application source)
k8s\  (Kubernetes manifests)
   deployment.yaml
reports\
   trivy-output.json
   dockle-output.json
   kube-linter-output.json
run_me.ps1
```

5 Installation and setup

This section provides stepwise instructions to install required software and configure the host system.

5.1 Docker Desktop

1. Download the installer from `https://www.docker.com/products/docker-desktop`.
2. Install with default options and confirm Docker Desktop starts after reboot.
3. Verify the Docker CLI is operational:
   `docker --version`

5.2 Trivy (Windows binary)

1. Download the latest Trivy release ZIP from the project's GitHub releases.
2. Extract the binary to `C:\Tools\Trivy` (or a preferred tools directory).
3. Add the directory to the `PATH` environment variable.
4. Validate installation with `trivy --version`.

5.3 Dockle (Windows binary)

1. Download the Dockle release archive from GitHub.
2. Extract to `C:\Tools\Dockle`.
3. Add the Dockle folder to `PATH`.
4. Validate with `dockle --version`.

5.4 Kube-linter (containerised execution)

1. No host installation is required; the official container image is used.
2. Example invocation:
```
docker run --rm -v "%CD%:/work" -w /work ghcr.io/stackrox/kube-linter:latest \
  lint ./k8s --format json > reports/kube-linter-output.json
```

5.5 Node.js

1. Install the Node.js LTS release from `https://nodejs.org/`.
2. Confirm installation: `node -v` and `npm -v`.

6 VS Code configuration

Visual Studio Code was used as the primary development environment. The project folder was opened directly in VS Code and the integrated terminal (PowerShell) was used for command execution. The Docker, YAML and Node.js extensions were installed to facilitate editing and validation.

A PowerShell automation script (`run_me.ps1`) was placed in the project root to orchestrate image build and scanning steps.

7 Scanning and execution

All scanning tasks were automated using the PowerShell script `run_me.ps1`. The script executes the following sequence:

1. Build the test image:
```
docker build -t insecure-image -f Dockerfile .
```
2. Run Trivy and emit JSON-formatted output:
```
trivy image insecure-image --format json --output reports/trivy-output.json
```
3. Run Dockle and emit JSON-formatted output:
```
dockle -f json -o reports/dockle-output.json insecure-image
```
4. Run Kube-linter via the official container and capture JSON output:
```
docker run --rm -v "%CD%:/work" -w /work ghcr.io/stackrox/kube-linter:latest \
  lint ./k8s --format json > reports/kube-linter-output.json
```

8 Verification and expected output

The expected outputs are JSON files containing findings consistent with the intentionally introduced misconfigurations. Summary expectations:

- **Trivy:** Multiple medium and low-severity vulnerabilities associated with Node.js dependencies and base image layers.
- **Dockle:** Informational Dockerfile hygiene warnings.
- **Kube-linter:** Manifest-level misconfigurations (use of `latest` tag, missing CPU/memory requests, writable root filesystem).

9 Portability notes

To reproduce the environment on another machine:

1. Install Windows 11 and Docker Desktop.
2. Install VS Code and the required extensions.
3. Install Trivy and Dockle as described above.
4. Ensure Docker CLI operates correctly.
5. Copy the project directory and verify paths in `run_me.ps1` as required.

10 Summary

This manual provides a concise and reproducible specification for the experiments described in the MSc dissertation. The configuration balances simplicity and automation, enabling researchers to reproduce results and validate the DevSecOps workflow.