National College of Ireland

Configuration Manual

MSc Research Project
Cloud Computing

Gaurav S. Chinchane
Student ID: x21234191

School of Computing
National College of Ireland

Supervisor: Prof. Aqeel Kazmi

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Gaurav S. Chinchane
Student ID:	x21234191
Programme:	Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Prof. Aqeel Kazmi
Submission Due Date:	14/12/2023
Project Title:	Configuration Manual
Word Count:	1405
Page Count:	10

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:	13th December 2023

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

Gaurav S. Chinchane
x21234191

1 Introduction

For integrating Site Reliability Engineering (SRE) concepts into DevSecOps, consult the Configuration Manual. The goal of this handbook, which is the result of substantial study, is to offer clear-cut recommendations for early vulnerability identification and correction in the dynamic field of software development. We address the dynamic problems of security threats by integrating SRE with DevSecOps processes and providing useful configurations and best practices. As an invaluable tool, this document enables teams to strengthen their software against vulnerabilities in cloud-based environments and promotes a continuous security and improvement culture.

A web application built using Django, the Todo App helps users efficiently manage chores. Its two main functions are as follows:

Admin: The administrator can see the total number of users who have signed up for the application. They also have access to resources for analyzing the vulnerabilities on the website.

User: A user's to-do list can contain additional tasks. A task entry's title, details, and completion checkbox are all included. Users can delete tasks and search for particular tasks.

System Requirements
Data Analytics: Requires Python 3.8, Azure CLI.

Todo App:
Operating System: Windows, macOS, or Linux
Python: Version 3.7 or newer
Web Browser: Latest versions of Chrome, Firefox, Safari, or Edge
Database: SQLite (for development) or PostgreSQL (for production)
RAM: Minimum 2 GB (4 GB recommended)
Disk Space: Minimum 100 MB for installation.

2 Setting Up the Environment

2.1 Cloud Hosting Setup

- Login to your aws account using url [1] and Launch an AWS EC2 Instance

[1] AWS Login `https://eu-west-1.console.aws.amazon.com/`

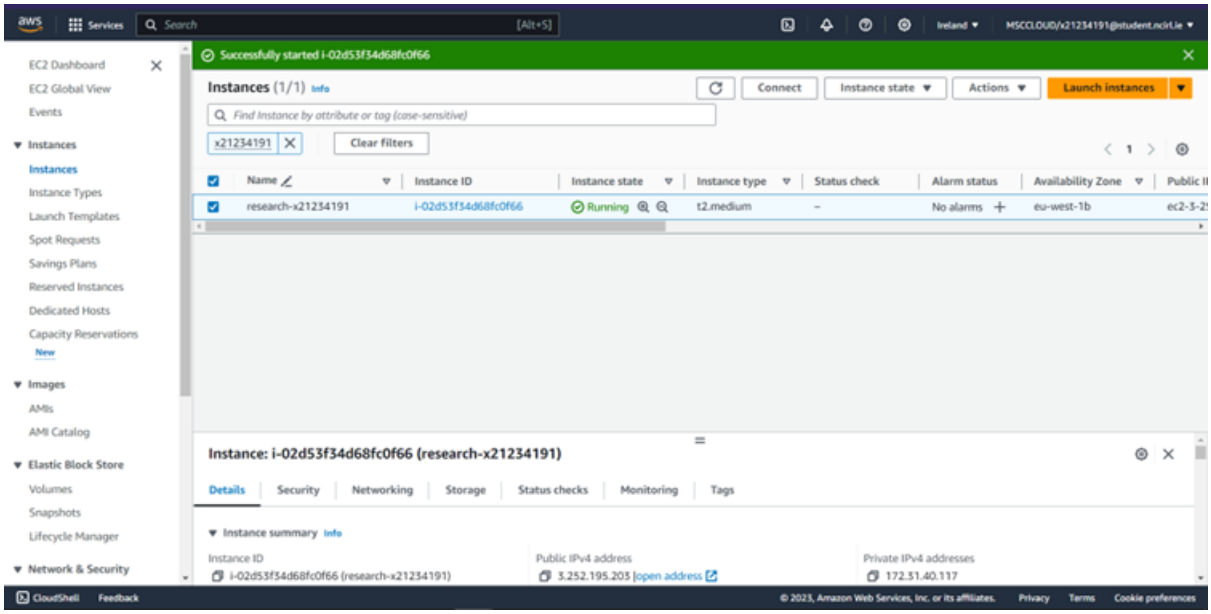

Figure 1: EC2 Instance Launch Page

- Access the Instance via SSH:
  - Use an SSH client: Secure Shell (SSH) clients like PuTTY are used to remotely access the EC2 instance.
  - Connect using the public IP and private key: The public IP allows to find our instance on the internet, and the private key is used to securely access it

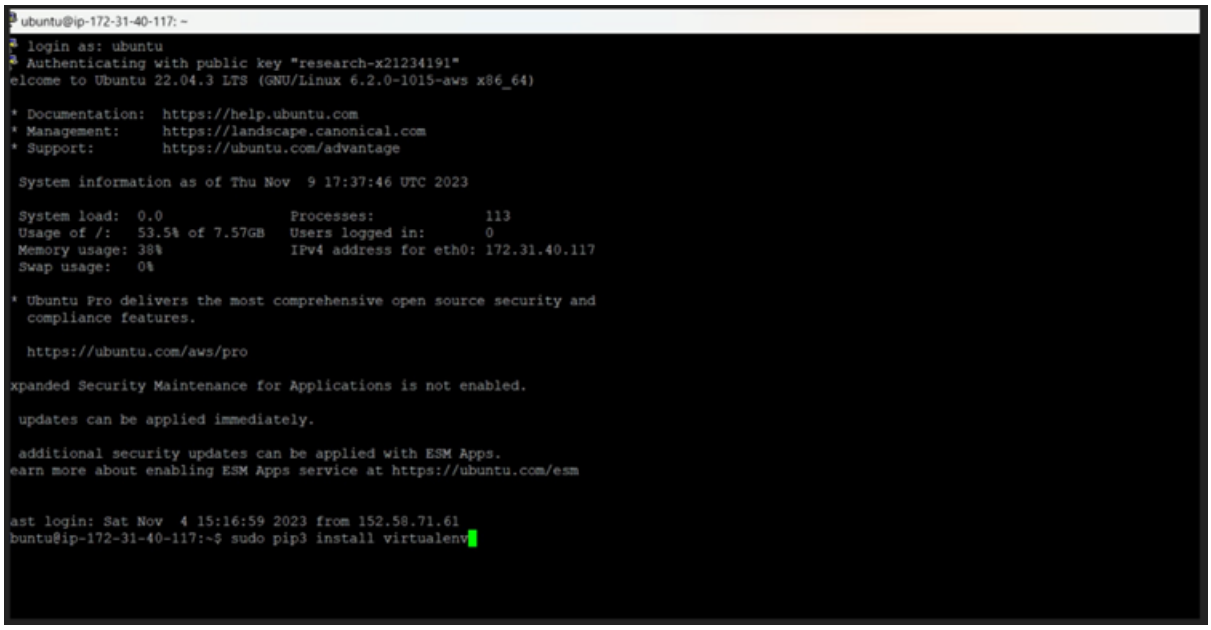

Figure 2: Connected using PUTTY

- Update the necessary Package Repository by using 1st command from Figure 5

- After the update if some packages need to Install New Packages use the 2nd command from Figure 5

```
sudo apt update
```

```
sudo apt install python3-pip python3-dev nginx
```

Figure 3: Commands to use

2.2 Python Installation

- Visit the official Python website: url [2]
- Download the installer for your operating system.
- Run the installer and follow the on-screen instructions. Ensure to select the option to add Python to your system path. Refer Figure 5

```
mkdir ~/projectdir
cd ~/projectdir
```

```
virtualenv env
```

Figure 4: Commands to use

2.3 Django Installation

- Open a terminal or command prompt.
- Install Django using pip: "pip install Django"
- Install with pip: 'Django' is a high-level Python Web framework, and 'Gunicorn' is a Python WSGI HTTP server for UNIX. Refer Figure 5

```
pip install django gunicorn
```

Figure 5: Command to use

[2]Python Download `https://www.python.org/downloads/release/python-3121/`

2.4 Additional Dependencies

- Django REST Framework: For building APIs (if needed): "pip install djangorest-framework"
- Database Drivers: Depending on your database choice, e.g., pip install psycopg2 for PostgreSQL.

3 Data Storage and Management

3.1 Database Setup for Todo App

- SQLite: Ideal for development due to its simplicity and ease of setup.
- PostgreSQL: For production env due to its robustness and scalability.

3.2 Database Configuration

- Edit the settings.py file in a ToDo App Django project.
- Under DATABASES, configure your database settings. For example, for Sqlite 3 Refer Figure 20

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
```

Figure 6: Portion to update in file

4 Data Analysis Tools Configuration

4.1 Setting Up Admin and User Roles

- Admin Role:
  - Create a superuser using command "python manage.py createsuperuser"
  - Assign privileges in the admin dashboard.
- User Role:
  - Implement user registration and login functionalities.
  - Develop the UI for adding, searching, and deleting tasks.
- Security Settings
  - Implement user authentication (e.g., using Django's built-in authentication system).
  - Set up permissions and role-based access control.

5 Setting Up Jenkins for Continuous Integration

5.1 Install Jenkins

Following official documentation for installation and setup used URL [3]

Jenkins is an open-source automation server which enables developers to build, test, and deploy their applications.

5.2 Start Jenkins Service

Using the command "systemctl" - Starts the Jenkins service and sets it to launch on boot.

5.3 Access Jenkins Dashboard

To Open in a browser use url [4]. Jenkins has a web interface for easy management.
For Login log into the dashboard Use the initial admin and password from the Jenkins setup.

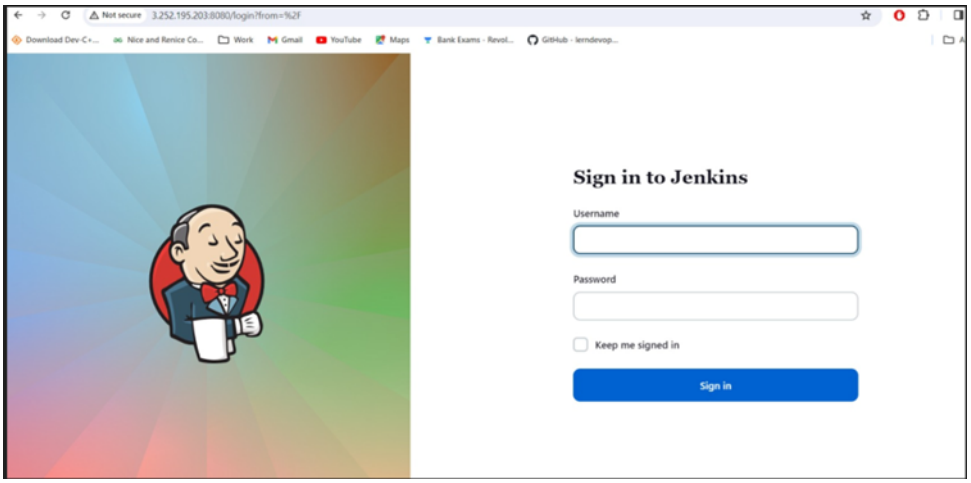

Figure 7: Jenkins Dashboard

5.4 Create a New Jenkins Pipeline

- Click on "New Item": Sets up a new job or pipeline in Jenkins.
- Configure the pipeline: This involves setting up source code management, build triggers, and build steps.
- Linking GitHub with Jenkins and Pipeline to Trigger for execution
- When changes are pushed to the GitHub repository, after committing the code, the Jenkins pipeline will automatically trigger.

[3] Jenkins Installation `https://www.jenkins.io/doc/book/installing/windows/`
[4] Jenkins Login `http://3.252.195.203:8080/`

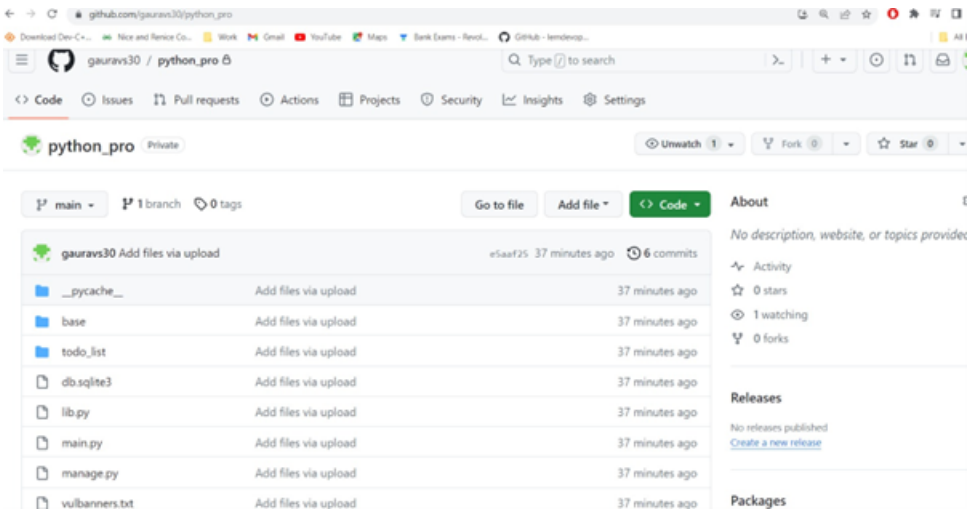

Figure 8: GitHub Folder

5.5 Run and Test the Pipeline

- Build Now: This will execute the pipeline you've set up.
- Console Output: Here, see the logs of the build process to troubleshoot any issue

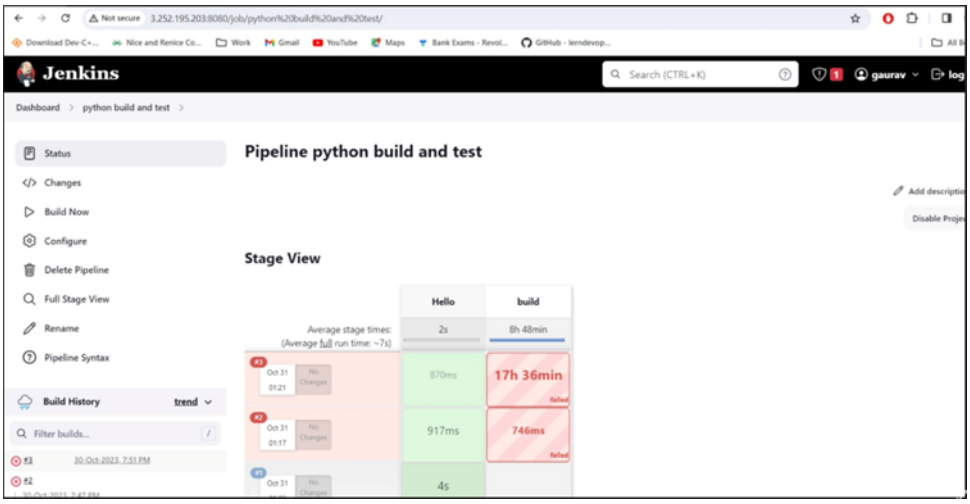

Figure 9: Pipeline execution status

6 Testing and Quality Assurance

6.1 Application

- Successful execution of an application is done. To access the Application url[5]
- Register your account in the system to make use of an application. If you already registered in a system then you can log in directly.

[5]Application Login `http://3.252.195.203:9000/`

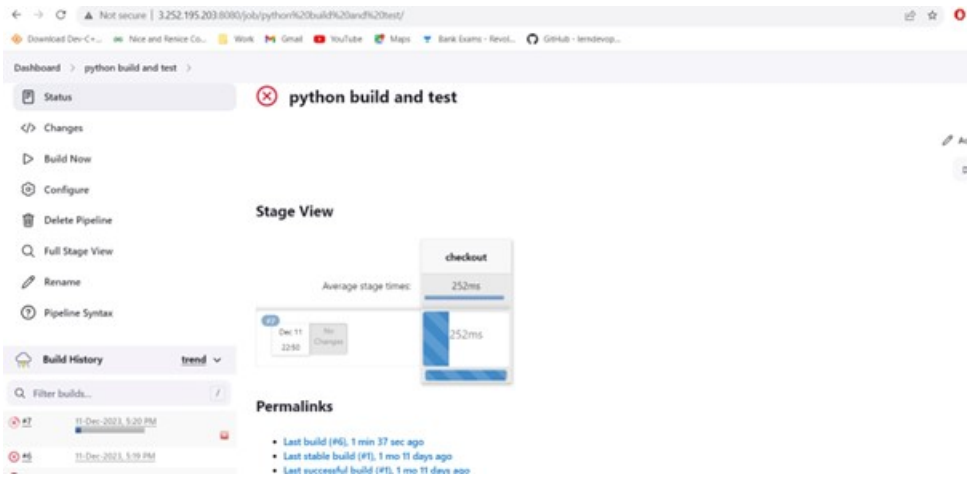

Figure 10: Successful CI/CD Pipeline

Note: Make sure the password length should be more than 8 characters and a combination of alphabet, numeric & special characters.

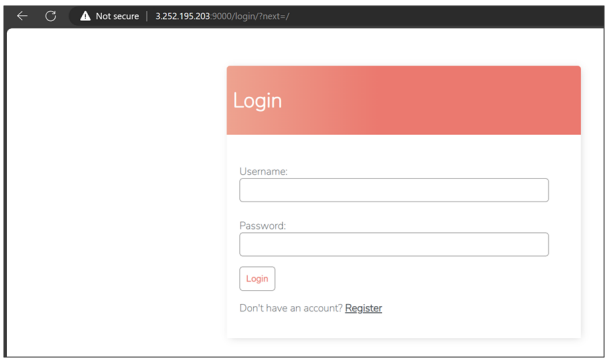

Figure 11: Application Landing page

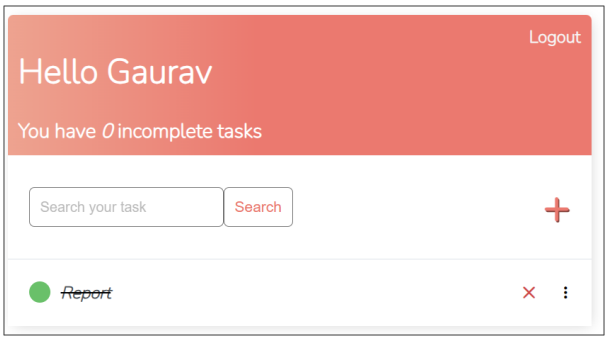

Figure 12: User Page

6.2 Python Scripting for Vulnerability Detection

- Place the files in the "Security Settings" section under "Application Configuration for Todo App". *CVE - download CVE list* (n.d.)

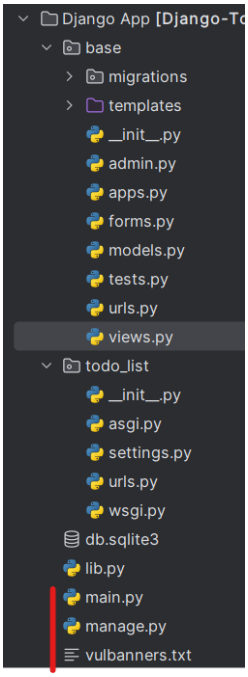

Figure 13: Application File Structure

- Change the path of the file according to the system location in the "Main.py" file
- Set the IP address and port number according to the requirement. Ref Figure 14

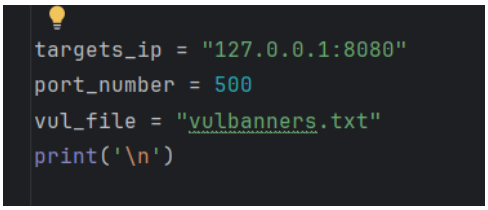

Figure 14: Setting up the values for testing

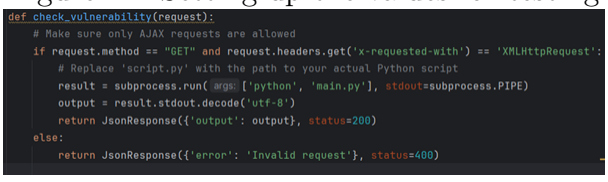

Figure 15: Code Function for vulnerability Check

- This should be detailed in the "main.py" section.
- Change the path of the "main.py" file according to the system location. The function needs be to updated in "view.py"

6.3 Cloud Watch Configuration

- Login to your aws account using url [6] and Launch an AWS Cloud Watch.
- Create a new dashboard using your EC2 instance where an application is hosted previously.

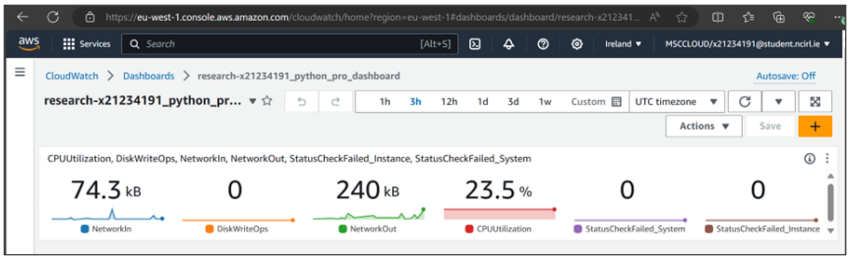

Figure 16: Cloud Watch used for Monitoring

6.4 Quality Checks for Data Analytics

- Running the Todo App, Start the Server
- Run python "manage.py runserver" to start the Django development server.

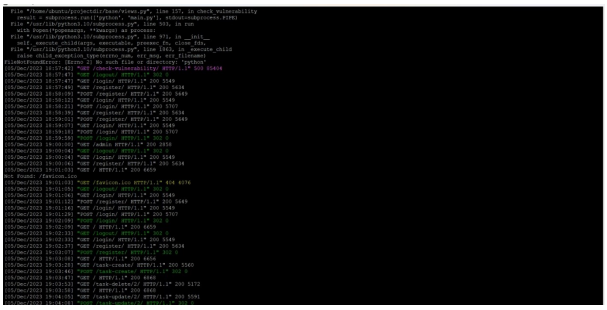

Figure 17: Running the application Server

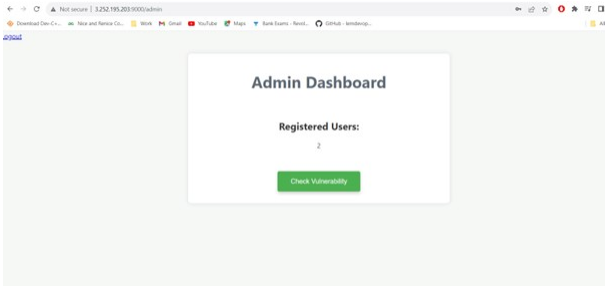

Figure 18: Running the application Server

[6] AWS Login `https://eu-west-1.console.aws.amazon.com/`

6.5 Comparison of systems for security and vulnerability

- For the 3 different pipelines executed for evaluation follow the below steps.
- Created 3 different repositories on Git-Hub.

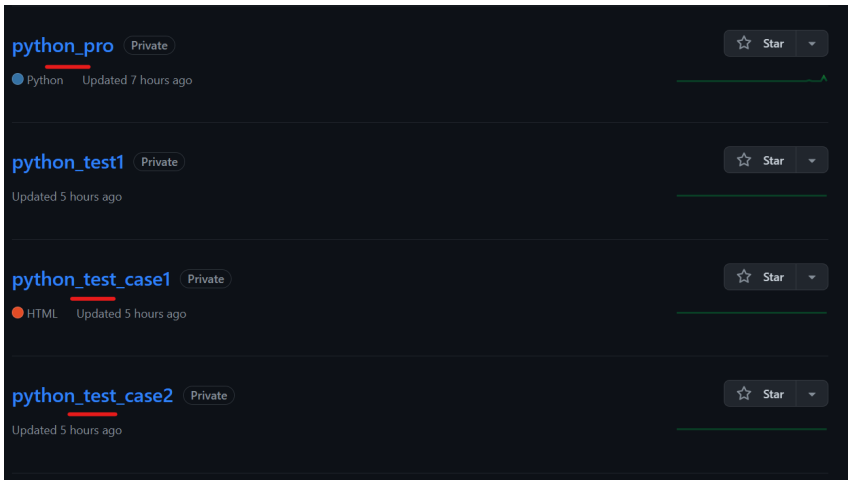

Figure 19: Git-Hub repos

- Created 3 different pipelines on Jenkins input as 3 repos from Git-Hub Refer to sections 5.4 and 5.5 for creating pipeline and execution.

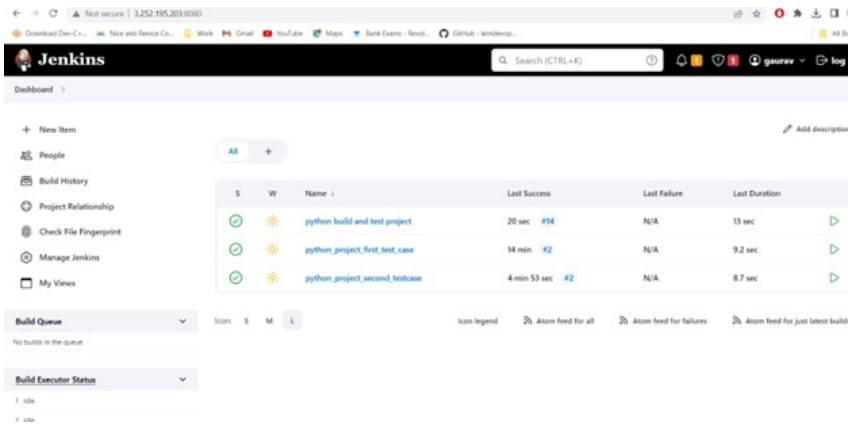

Figure 20: Executed Pipelines

References

CVE - download CVE list (n.d.). `https://cve.mitre.org/data/downloads/`. Accessed: 2023-12-13.