Project 3

Flask-Based Student Registration Web Application deployed using Jenkins

Batch

MCA 7 Feb 2025 &
DevOps 2 Jun 2025

Name

Suraj Molke

Name of the Mentor

Ravindra Bagle Sir
Swati Zampal Ma’am

Fortune Cloud Technologies

2nd Floor, Shirodkar House, Opposite To Amit Cafe, Congress House Road, Near Pune Municipal Corporation, Shivajinagar, Pune - 411005.

smolke9@gmail.com Suraj Molke Smolke9

Title: Flask-Based Student Registration Web Application deployed using Jenkins

Objective:

Develop and deploy a Flask-based web application that allows users to register students by submitting their details via a web form. The application should store the submitted data in a backend and provide the ability to retrieve or display this information when needed.

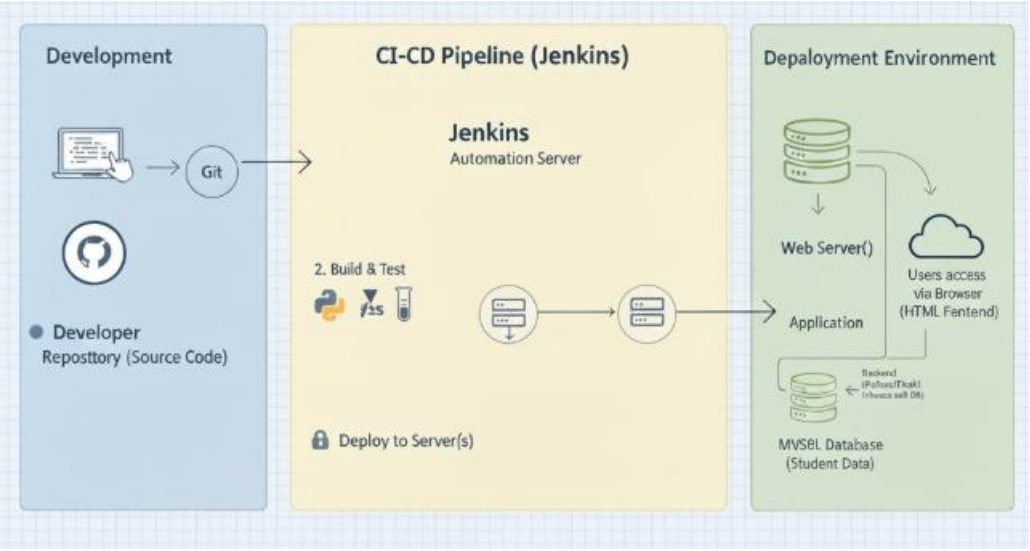
Project Overview:

This project involves designing a simple web application that mimics a student registration system. The goal is to gain hands-on experience in web development, form handling, backend logic, and data storage using Python Flask and standard frontend technologies.

Technology Stack:

- ✓ Frontend: HTML, CSS
- ✓ Backend: Python (Flask)
- ✓ Database: MySQL
- ✓ Version Control: Git & GitHub
- ✓ Deployment (Optional): EC2

Architectural diagram



Step .1: Install Python and Dependencies

1. Install Python 3: Ensure that Python 3 is installed on your system.

• Flask==2.3.2

• mysql-connector-python

```
Flask==2.3.2
mysql-connector-python
```

Step 2: Set up MySQL Database

1. Install and start MySQL Server.

2. Create a database named studentdb.

3. Create students table with fields:

• id (primary key, auto-increment)

• name, email, phone, course, address, contact

```
CREATE TABLE students (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    email VARCHAR(100),
    phone VARCHAR(20),
    course VARCHAR(100),
    address VARCHAR(200),
    contact VARCHAR(100)
);
```

Step 3: Install Jenkins

1. Install Jenkins server.

2. Install Git Plugin via Manage Jenkins → Manage Plugins.

- Manage Jenkins → Manage Plugins → Available → Git Plugin
- Git
- Pipeline
- SSH Agent
- Ansible (optional)
- Python plugin

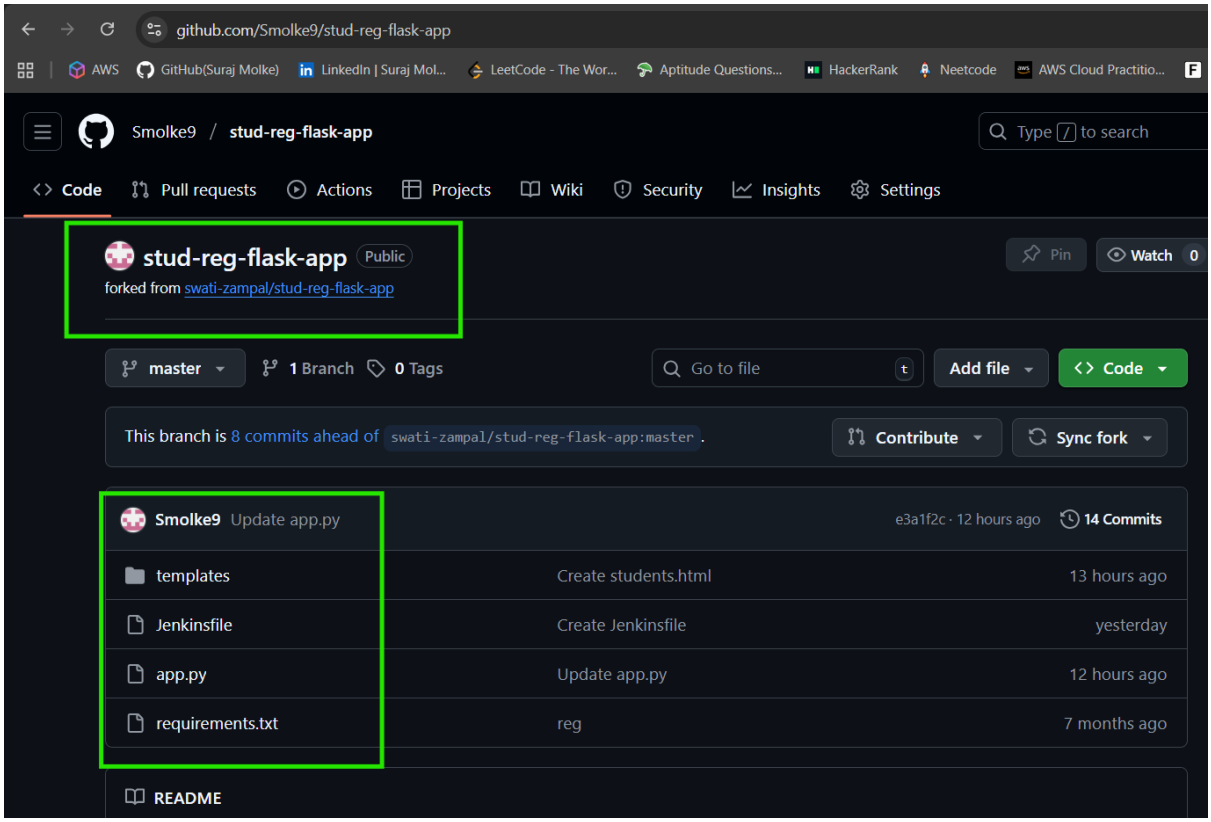
Step 4: Install socat

Install socat on Jenkins agent using:

sudo apt-get install socat

Step 5: Application File Structure From Forked Via https://github.com/swati-zampal/stud-reg-flask-app.git To My Repo https://github.com/Smolke9/stud-reg-flask-app.git

- app.py – Main Flask application logic
- requirements.txt – Python dependencies
- templates/
  - register.html
  - students.html
- Jenkinsfile – CI/CD pipeline stages



Step 6: Application Code Development

- Uses mysql.connector for database connectivity.
- '/' route displays registration form.
- '/students' displays stored student data.
- Runs on host=0.0.0.0 port=5000.

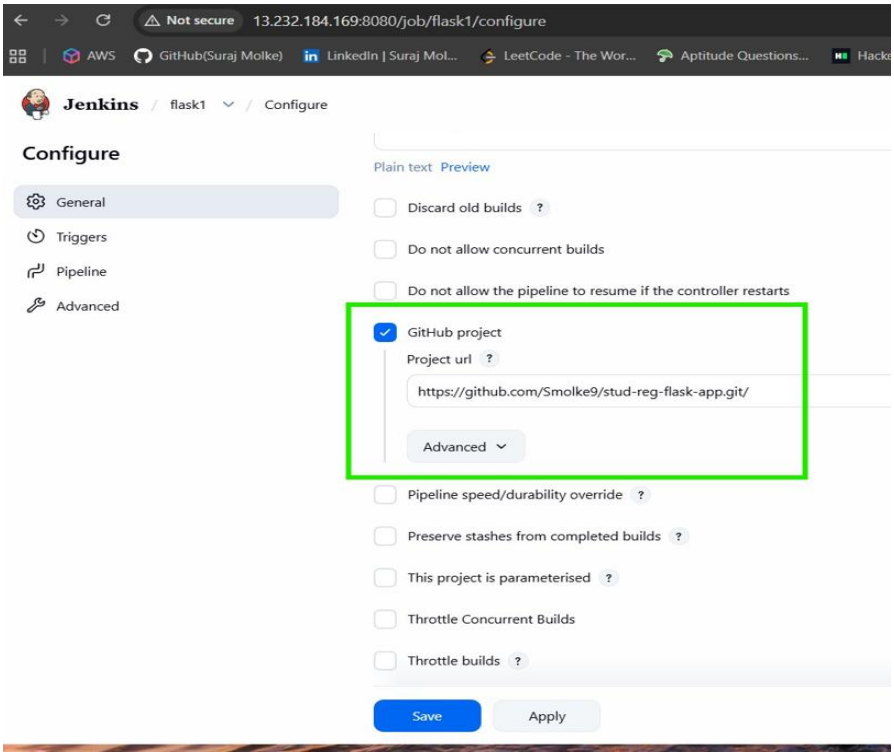
Step 7: Templates

• register.html – HTML form for user input.

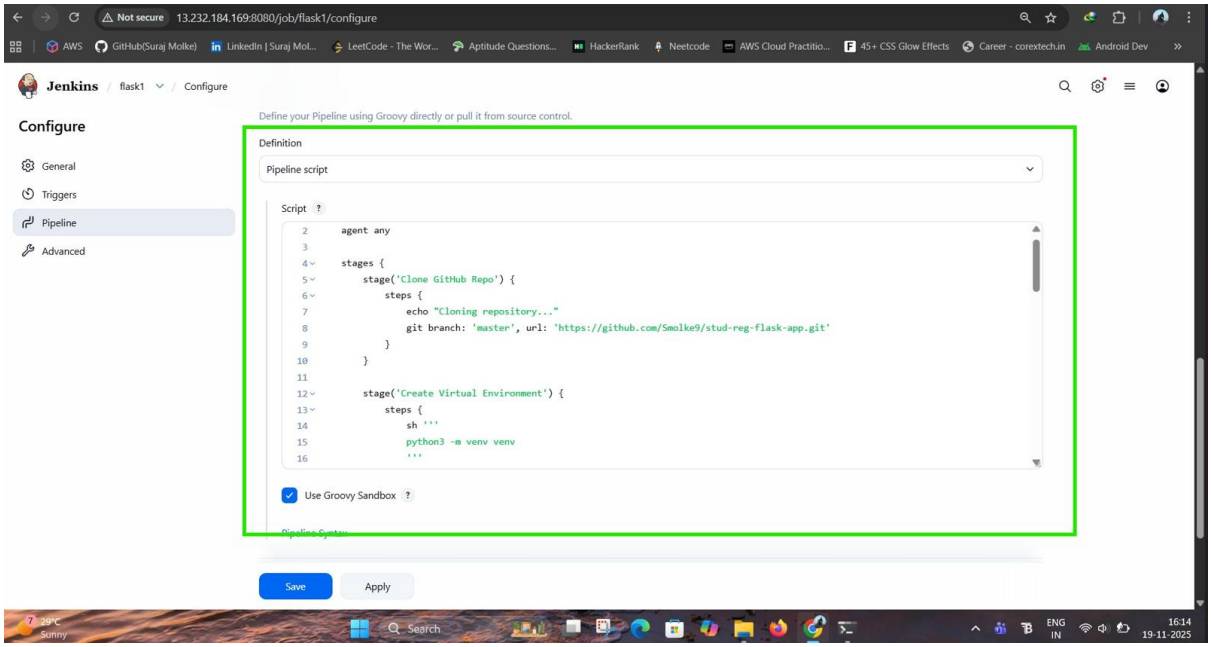
• students.html

Step 8: Jenkins Pipeline Stages

8.1 Clone GitHub Repo – pulls latest code.



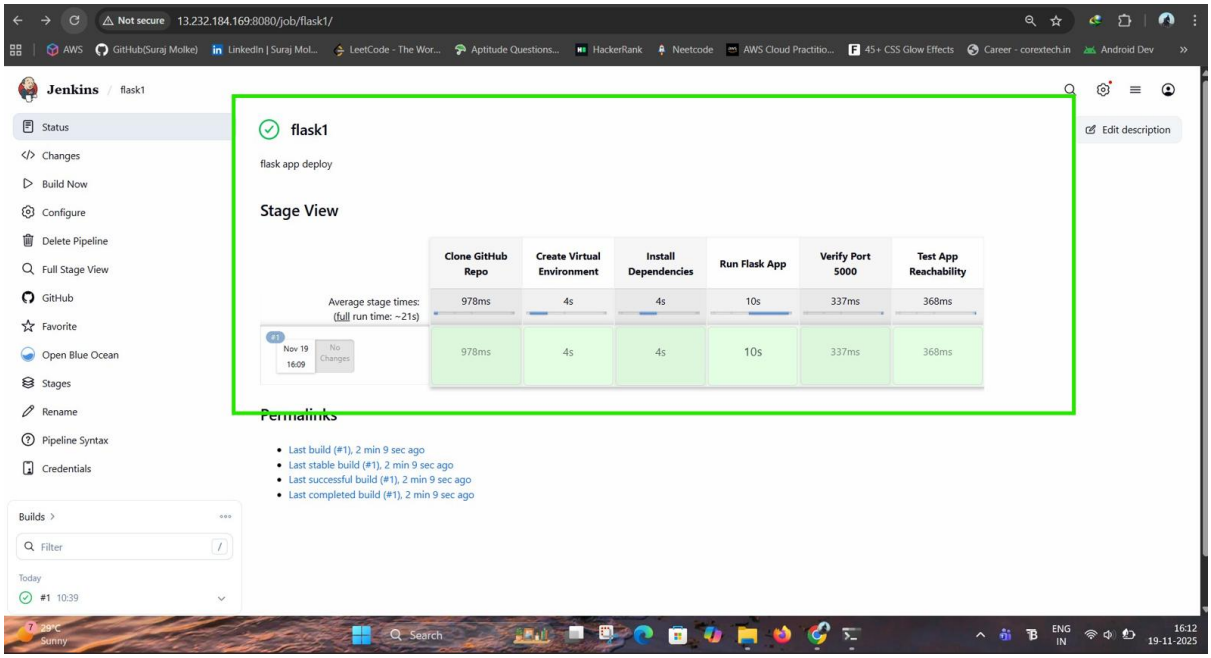
8.2 Create Virtual Environment – isolates dependencies.



8.3 Install Dependencies – Flask + MySQL connector.

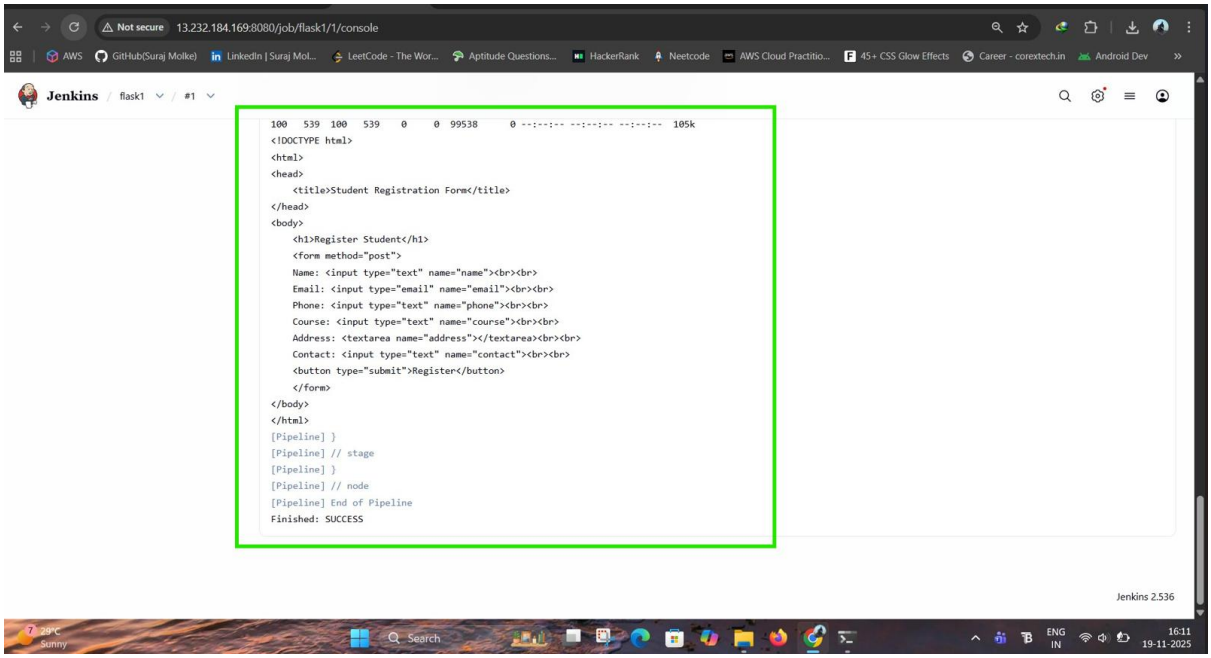
8.4 Run Flask App – starts on port 5000.

8.5 Expose Port using socat – forwards 5000 → 5050.

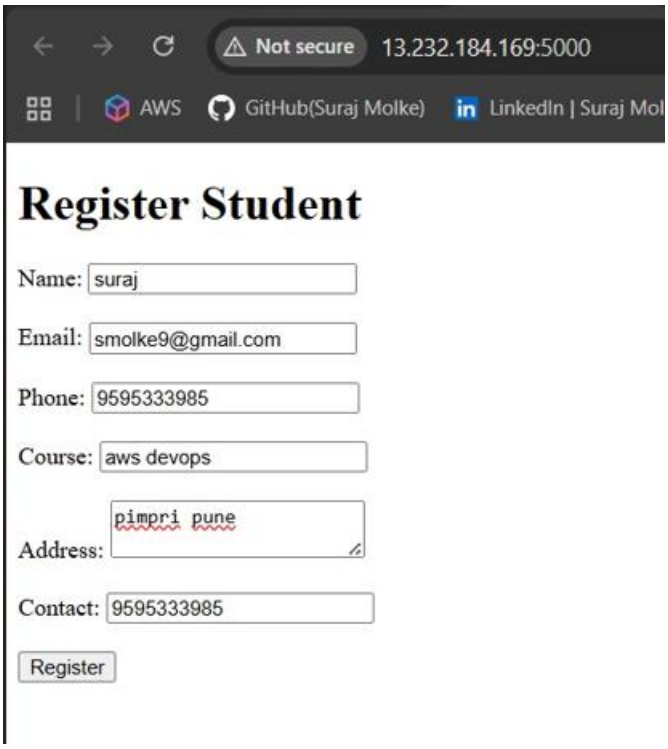


8.6 Verify Access – curl health check ensures app is running.

A curl command is executed to perform a final health check. A successful response confirms that the entire deployment chain, from the Flask app to the socat port forwarding, is working correctly



Step 9: On post 5000 page available



Finally Student data successfully registered in mysql database.

Conclusion

- **Separation of Concerns:** The frontend (HTML), backend (Flask), and database (MySQL) are distinct layers, making the application modular and easier to maintain.
- **Version Control Best Practices:** Using Git and GitHub ensures that all code changes are tracked, collaborated on effectively, and easily revertible. This is fundamental for team-based development and stable releases.
- **Automated CI/CD (DevOps):** Jenkins is at the heart of this project, automating the entire workflow from code commit to deployment. This significantly:
    - **Reduces Manual Errors:** Eliminates human mistakes in the build, test, and deployment phases.
    - **Increases Speed:** Speeds up the delivery of new features and bug fixes to users.
    - **Ensures Quality:** Automated tests (unit and integration) run with every change, catching issues early in the development cycle.
    - **Promotes Consistency:** Guarantees that deployments are consistent across environments.
- **Scalability & Maintainability:** A well-structured Flask application interacting with a robust MySQL database provides a solid foundation that can be scaled horizontally if needed and is relatively easy to debug and enhance.

In conclusion, this project provides a comprehensive and automated pipeline for building and deploying a Flask-based web application. It not only delivers a functional student registration system but also establishes a professional and efficient development and deployment process, making it a strong foundation for future enhancements and continuous delivery.