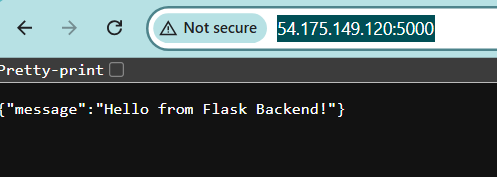
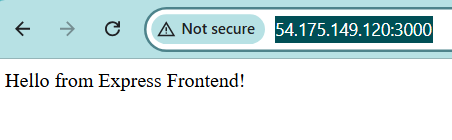
### **CI/CD Deployment Assignment**

You are required to deploy a **Flask backend** and an **Express frontend** on an **Amazon EC2 instance**. Additionally, implement a **CI/CD pipeline** using Jenkins to automate the deployment process.

#### **Part 1: Deploy Flask and Express on a Single EC2 Instance**

1. **Objective**:
   * Deploy both the Flask backend and the Express frontend on a **single Amazon EC2 instance**.
2. **Steps**:
   * **Provisioning the EC2 Instance**:
     + Launch an EC2 instance on AWS (you can use a free-tier eligible instance).
     + SSH into the instance and install the following dependencies:
       - Python for Flask.
       - Node.js for Express.
       - Git for pulling the application code.
   * **Application Setup**:
     + Clone the Flask and Express repositories onto the EC2 instance.
     + Install the required dependencies for both applications using pip and npm.
     + Configure both applications to run on different ports (e.g., Flask on port 5000 and Express on port 3000).
     + Start the applications using process managers like pm2 or systemd to ensure they remain active.
3. **Deliverables**:
   * A running EC2 instance with Flask and Express accessible via the instance's public IP.
   * A description or diagram of the deployed architecture





┌────────────┐

│ Browser │

└────┬───────┘

│

┌───────▼────────┐

│ EC2 Instance │

│ (Ubuntu) │

│ │

│ ┌──────────┐ │

│ │ Flask API│◄─┼── Port 5000

│ └──────────┘ │

│ ┌──────────┐ │

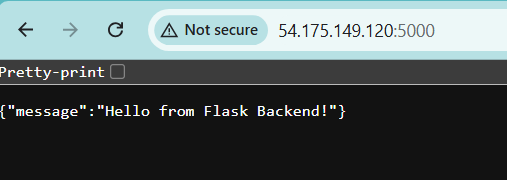
│ │ Express │◄─┼── Port 3000

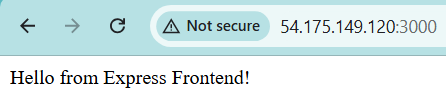
│ └──────────┘ │

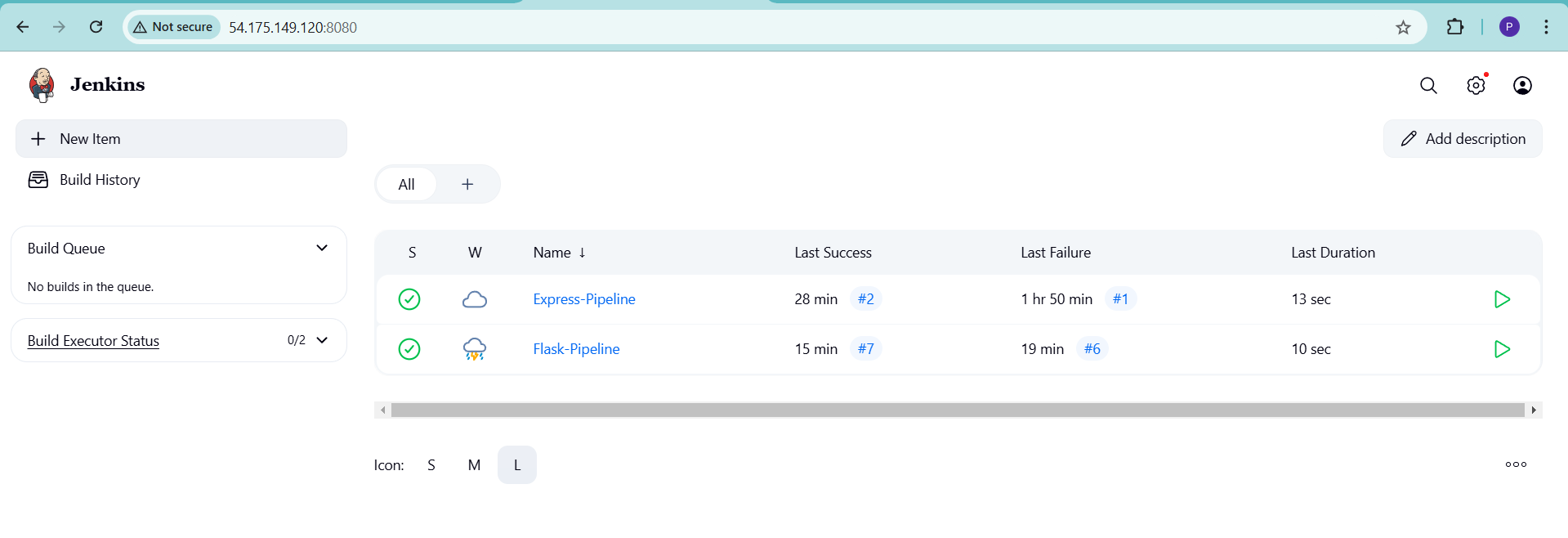
└────────────────┘

#### **: Implement CI/CD Pipeline Using Jenkins**

1. **Objective**:
   * Automate the deployment of Flask and Express applications using Jenkins.
2. **Steps**:
   * **Install Jenkins**:
     + Install Jenkins on the same EC2 instance or on a separate machine.
     + Configure Jenkins by installing essential plugins like Git, NodeJS, and Python.
   * **Set Up Jenkins Pipeline**:
     + Create two separate Jenkins pipelines for the Flask and Express applications.
     + **Pipeline Steps**:
       - Pull the latest code from the respective Git repositories.
       - Install dependencies for Flask (pip install -r requirements.txt) and Express (npm install).
       - Restart the applications using the process manager (e.g., pm2 restart <app>).
   * **Triggering the Pipeline**:
     + Set up a GitHub webhook to trigger the Jenkins pipeline on every push to the repositories.
   * **Optional Enhancements**:
     + Add testing stages to the pipeline for both applications.
     + Configure environment variables in Jenkins for managing sensitive data like API keys.
3. **Deliverables**:
   * A fully functional CI/CD pipeline that automates the deployment process for Flask and Express.
   * A Jenkins pipeline script (Jenkinsfile) for each application.
   * Evidence of the pipeline working (e.g., screenshots of successful builds and deployments).







### **Submission Requirements**

1. A link to the GitHub repositories for Flask and Express.
2. A document (README.md) describing the deployment process and CI/CD pipeline setup.
3. Screenshots of:
   * The running EC2 instance with Flask and Express accessible.
   * Jenkins pipeline execution logs showing successful deployment.