**Solution to Lab#03: Configuring Continuous Integration (CI) Pipelines**

**Objective**

In this lab, we will:

* Understand **Continuous Integration (CI)** and its benefits.
* Set up a **CI pipeline** using GitHub Actions, Jenkins, and GitLab CI/CD.
* Automate code testing and integration.
* Analyze pipeline execution results and troubleshoot errors.

**Exercise 1: Setting Up a CI Pipeline with GitHub Actions**

**Step 1: Create a GitHub Repository**

1. Go to [**GitHub**](https://github.com/) and log in.
2. Click on **New Repository**.
3. Provide a **repository name** (e.g., ci-demo), choose **public/private**, and click **Create Repository**.
4. Clone the repository to your local system:

git clone https://github.com/your-username/ci-demo.git

cd ci-demo

**Step 2: Set Up GitHub Actions Workflow**

1. Inside the repository, create a new folder .github/workflows/.
2. Inside this folder, create a new file ci-pipeline.yml.

**Step 3: Add the Following Workflow Configuration**

Copy and paste this into the ci-pipeline.yml file:

yaml

name: CI Pipeline

on: [push, pull\_request]

jobs:

build:

runs-on: ubuntu-latest

steps:

- name: Checkout Code

uses: actions/checkout@v2

- name: Set up Node.js

uses: actions/setup-node@v2

with:

node-version: '14'

- name: Install Dependencies

run: npm install

- name: Run Tests

run: npm test

**Step 4: Commit and Push the File**

git add .github/workflows/ci-pipeline.yml

git commit -m "Added CI pipeline workflow"

git push origin main

**Step 5: Observe Execution in GitHub Actions**

1. Go to your **GitHub Repository → Actions Tab**.
2. You will see the workflow running automatically on **push** or **pull requests**.
3. If it passes, your code is successfully tested!

**Exercise 2: Configuring a CI Pipeline in Jenkins**

**Step 1: Install Jenkins**

1. Download and install **Jenkins** from <https://www.jenkins.io/>.
2. Install required **plugins**:
   * Git Plugin
   * Pipeline Plugin
   * NodeJS Plugin

**Step 2: Create a New Jenkins Project**

1. Open **Jenkins Dashboard** → Click **New Item**.
2. Choose **Freestyle Project** or **Pipeline Project**, and name it ci-demo.
3. Click **OK**.

**Step 3: Configure the Git Repository**

1. Go to **Source Code Management → Git**.
2. Enter your **GitHub Repository URL**.
3. Add branch specification: \*/main.

**Step 4: Add Build Steps**

1. Under **Build Triggers**, select **GitHub hook trigger for GITScm polling**.
2. Under **Build Steps**, add:
   * **Clone Repository:** Automatically done by Jenkins.
   * **Install Dependencies:**

npm install

* + **Run Tests:**

npm test

1. Click **Save & Apply**.

**Step 5: Run the Pipeline**

1. Click **Build Now** to trigger a manual build.
2. Check the **Console Output** for success/failure messages.

**Exercise 3: Implementing CI with GitLab CI/CD**

**Step 1: Create a .gitlab-ci.yml File**

In your GitLab repository, create a new file **.gitlab-ci.yml** and add the following:

yaml

stages:

- test

- build

test:

stage: test

script:

- echo "Running tests..."

- npm install

- npm test

build:

stage: build

script:

- echo "Building the project..."

**Step 2: Commit and Push the File**

git add .gitlab-ci.yml

git commit -m "Added GitLab CI pipeline"

git push origin main

**Step 3: Check Pipeline Execution in GitLab**

1. Go to **GitLab Repository → CI/CD → Pipelines**.
2. Observe the pipeline execution status.

**Troubleshooting Tips**

* **Check repository structure**: Ensure package.json exists if using Node.js.
* **Fix syntax errors**: Validate YAML files using online tools like [YAML Lint](http://www.yamllint.com/).
* **Check logs**: Review logs under **GitHub Actions**, **Jenkins Console Output**, or **GitLab Pipelines** for errors.

**Conclusion**

In this lab, we successfully:  
✅ Set up **GitHub Actions** for automated CI.  
✅ Configured **Jenkins** to run CI pipelines.  
✅ Implemented **GitLab CI/CD** pipelines.  
✅ Automated **code testing and deployment** processes.

By mastering CI pipelines, you can ensure **faster, efficient, and reliable software development**!