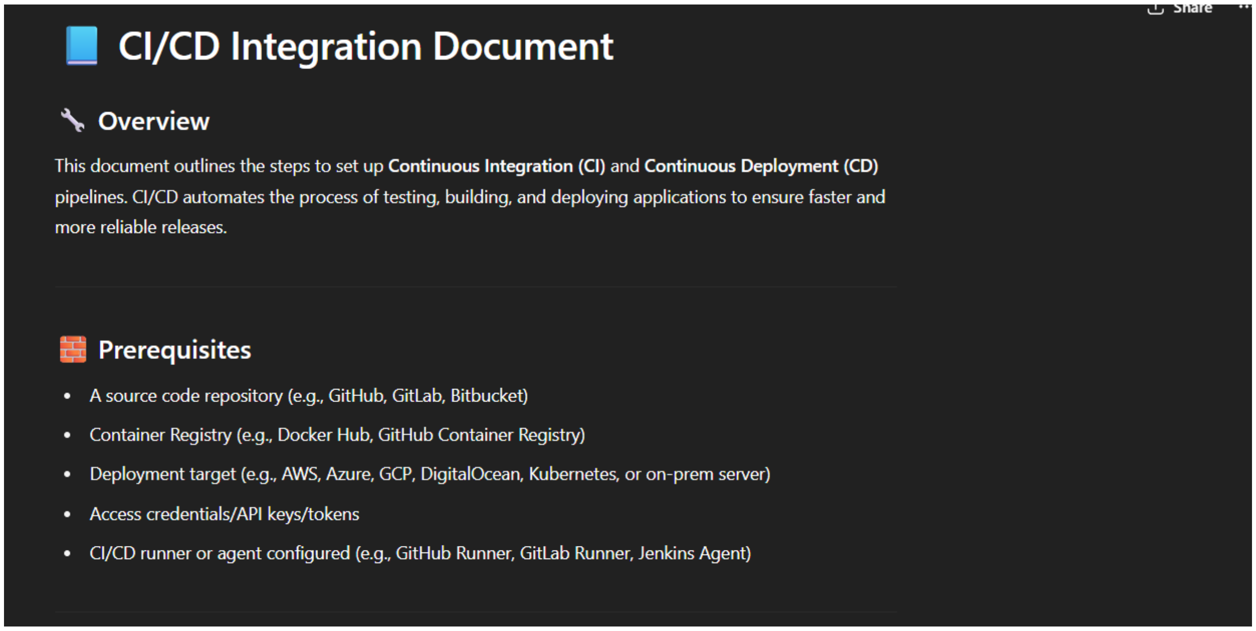
Cicdtools-githubaction



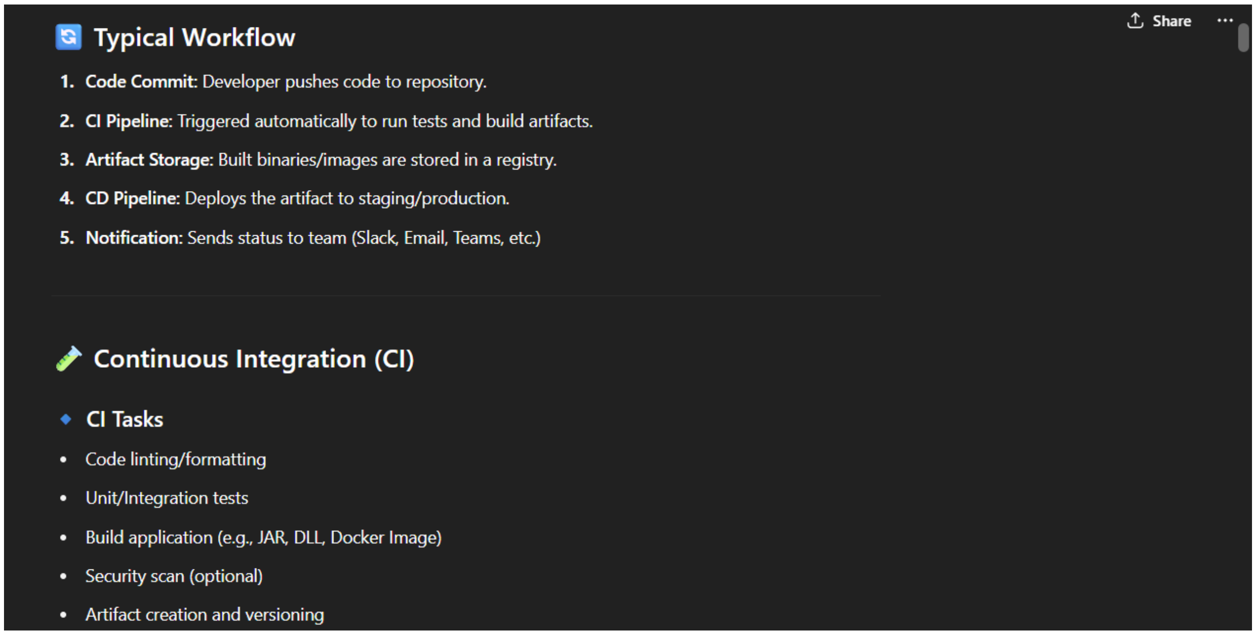
📘 CI/CD Integration Document

🔧 Overview

This document outlines the steps to set up **Continuous Integration (CI)** and **Continuous Deployment (CD)** pipelines. CI/CD automates the process of testing, building, and deploying applications to ensure faster and more reliable releases.

🧱 Prerequisites

- A source code repository (e.g., GitHub, GitLab, Bitbucket)
- Container Registry (e.g., Docker Hub, GitHub Container Registry)
- Deployment target (e.g., AWS, Azure, GCP, DigitalOcean, Kubernetes, or on-prem server)
- Access credentials/API keys/tokens
- CI/CD runner or agent configured (e.g., GitHub Runner, GitLab Runner, Jenkins Agent)



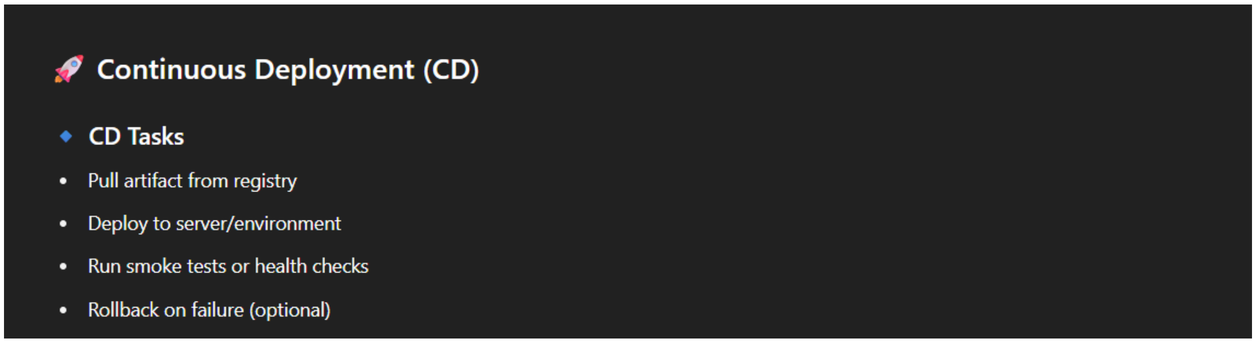
🔄 Typical Workflow

1. **Code Commit:** Developer pushes code to repository.
2. **CI Pipeline:** Triggered automatically to run tests and build artifacts.
3. **Artifact Storage:** Built binaries/images are stored in a registry.
4. **CD Pipeline:** Deploys the artifact to staging/production.
5. **Notification:** Sends status to team (Slack, Email, Teams, etc.)

🧪 Continuous Integration (CI)

🔹 CI Tasks

- Code linting/formatting
- Unit/Integration tests
- Build application (e.g., JAR, DLL, Docker Image)
- Security scan (optional)
- Artifact creation and versioning

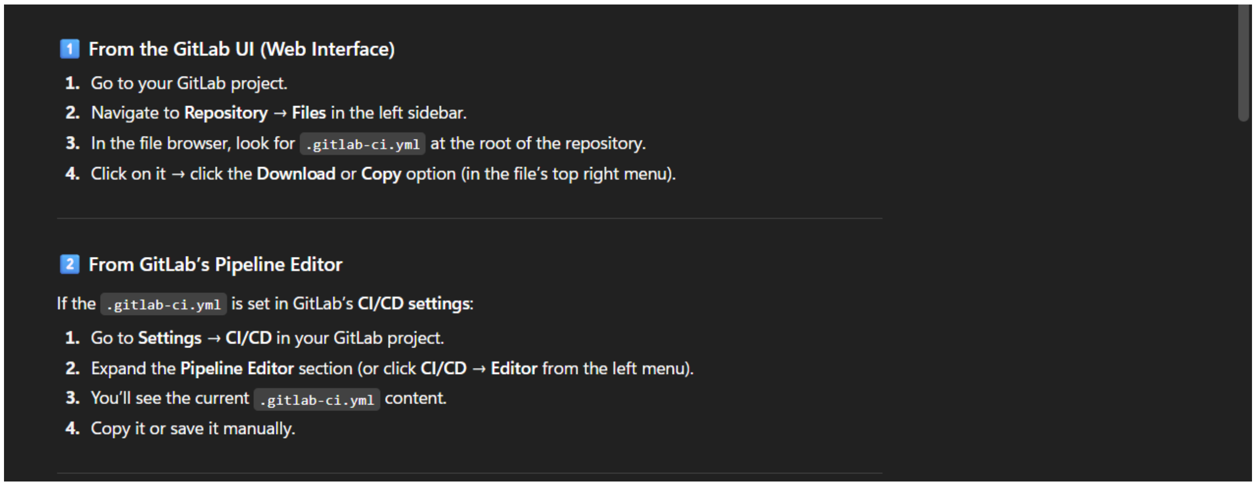


🚀 Continuous Deployment (CD)

🔹 CD Tasks

- Pull artifact from registry
- Deploy to server/environment
- Run smoke tests or health checks
- Rollback on failure (optional)

Yaml file



1️⃣ From the GitLab UI (Web Interface)

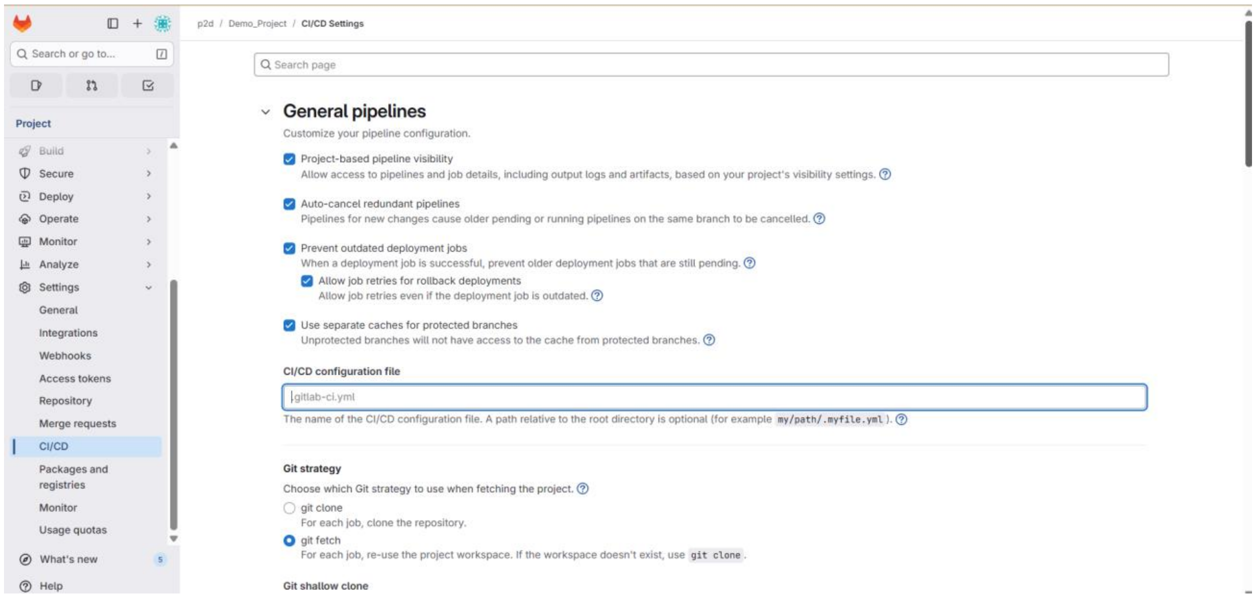
1. Go to your GitLab project.
2. Navigate to **Repository** → **Files** in the left sidebar.
3. In the file browser, look for `.gitlab-ci.yml` at the root of the repository.
4. Click on it → click the **Download** or **Copy** option (in the file's top right menu).

2️⃣ From GitLab's Pipeline Editor

If the `.gitlab-ci.yml` is set in GitLab's **CI/CD settings**:

1. Go to **Settings** → **CI/CD** in your GitLab project.
2. Expand the **Pipeline Editor** section (or click **CI/CD** → **Editor** from the left menu).
3. You'll see the current `.gitlab-ci.yml` content.
4. Copy it or save it manually.

Common configuration yaml file



General pipelines

Customize your pipeline configuration.

- [✓] Project-based pipeline visibility
  Allow access to pipelines and job details, including output logs and artifacts, based on your project's visibility settings.
- [✓] Auto-cancel redundant pipelines
  Pipelines for new changes cause older pending or running pipelines on the same branch to be cancelled.
- [✓] Prevent outdated deployment jobs
  When a deployment job is successful, prevent older deployment jobs that are still pending.
  - [✓] Allow job retries for rollback deployments
    Allow job retries even if the deployment job is outdated.
- [✓] Use separate caches for protected branches
  Unprotected branches will not have access to the cache from protected branches.

CI/CD configuration file

.gitlab-ci.yml

The name of the CI/CD configuration file. A path relative to the root directory is optional (for example `my/path/.myfile.yml`).

Git strategy

Choose which Git strategy to use when fetching the project.

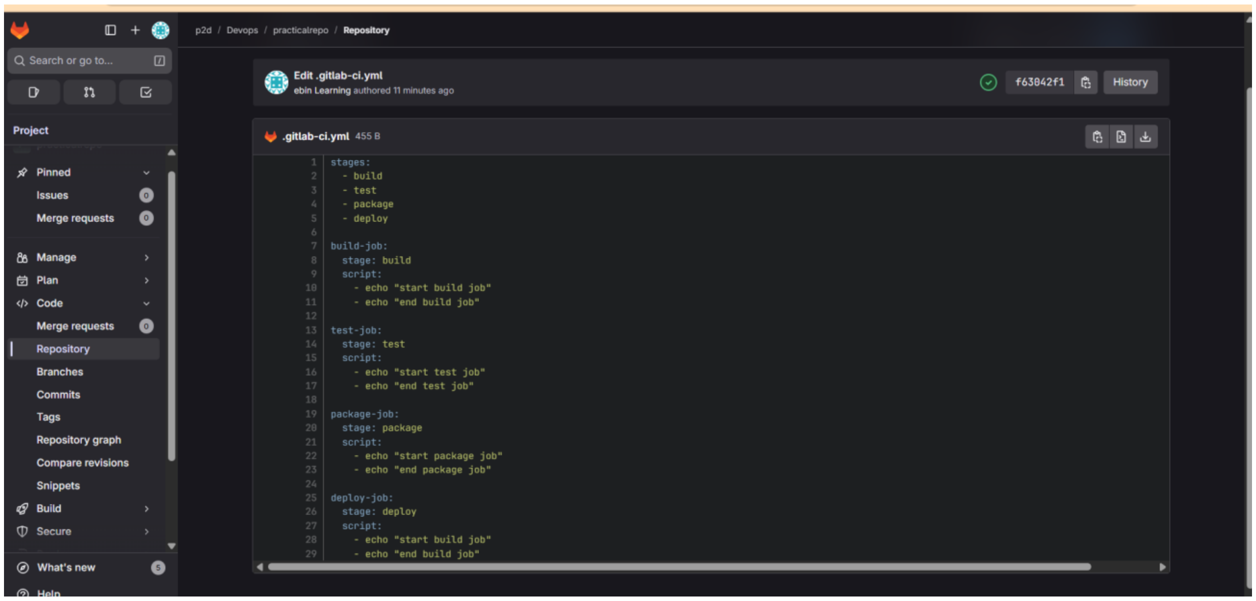
- ○ git clone
  For each job, clone the repository.
- [✓ ● git fetch]
  For each job, re-use the project workspace. If the workspace doesn't exist, use `git clone`.

Git shallow clone

First I am going to create sample yml file

Filename: .gitlab-ci.yml

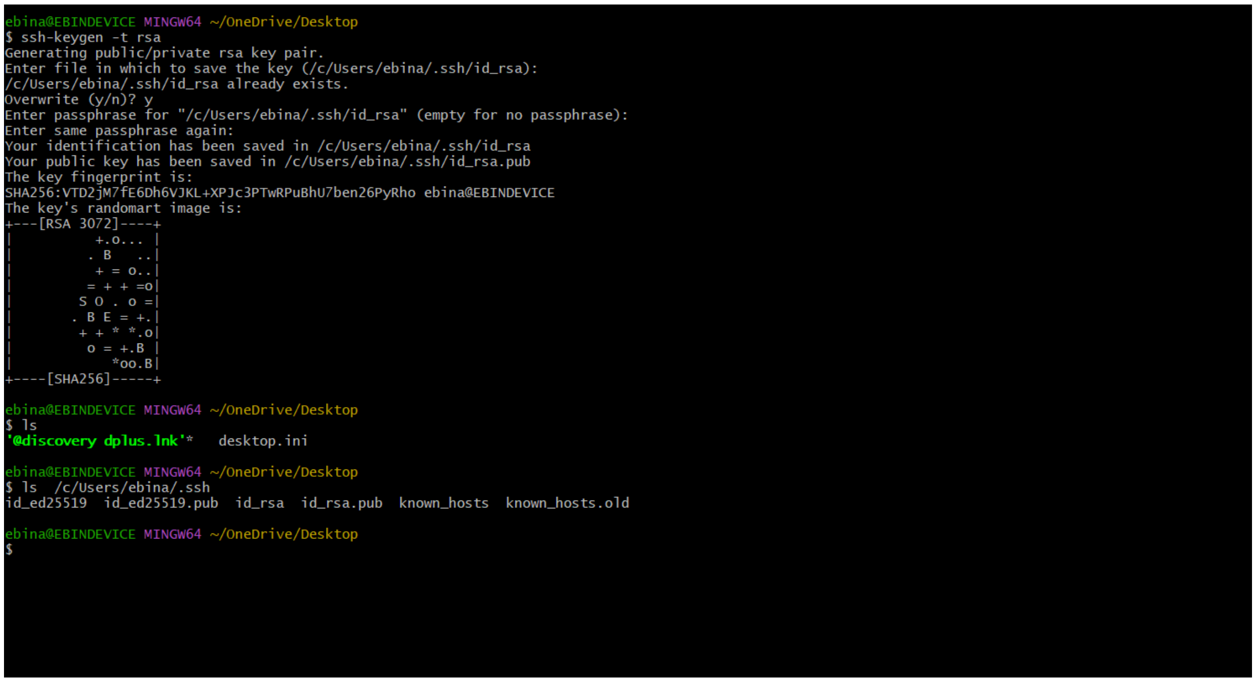
4 type of stages: build, test,package,deploy



```yaml
stages:
  - build
  - test
  - package
  - deploy

build-job:
  stage: build
  script:
    - echo "start build job"
    - echo "end build job"

test-job:
  stage: test
  script:
    - echo "start test job"
    - echo "end test job"

package-job:
  stage: package
  script:
    - echo "start package job"
    - echo "end package job"

deploy-job:
  stage: deploy
  script:
    - echo "start build job"
    - echo "end build job"
```

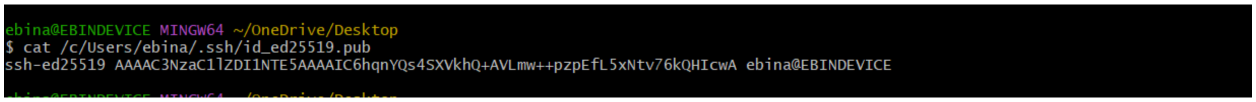
Create sshkey generation

Sshkey generation completed



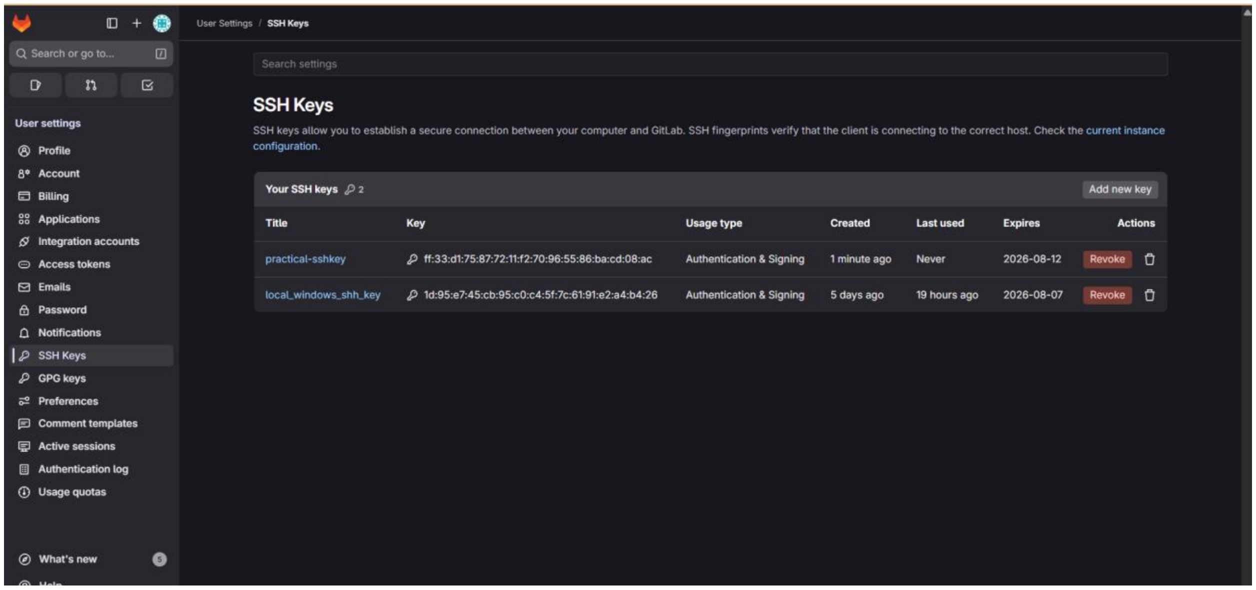
```
ebina@EBINDEVICE MINGW64 ~/OneDrive/Desktop
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/ebina/.ssh/id_rsa):
/c/Users/ebina/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase for "/c/Users/ebina/.ssh/id_rsa" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/ebina/.ssh/id_rsa
Your public key has been saved in /c/Users/ebina/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:VTD2jM7fE6Dh6VJKL+XPJc3PTwRPuBhU7ben26PyRho ebina@EBINDEVICE
The key's randomart image is:
+---[RSA 3072]----+
|          +.o... |
|         . B   ..|
|          + = o..|
|         = + + =o|
|        S O . o =|
|       . B E = +.|
|        + + * *.o|
|         o = +.B |
|            *oo.B|
+----[SHA256]-----+

ebina@EBINDEVICE MINGW64 ~/OneDrive/Desktop
$ ls
'@discovery dplus.lnk'*   desktop.ini

ebina@EBINDEVICE MINGW64 ~/OneDrive/Desktop
$ ls  /c/Users/ebina/.ssh
id_ed25519  id_ed25519.pub  id_rsa  id_rsa.pub  known_hosts  known_hosts.old

ebina@EBINDEVICE MINGW64 ~/OneDrive/Desktop
$
```

View public key



```
ebina@EBINDEVICE MINGW64 ~/OneDrive/Desktop
$ cat /c/Users/ebina/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIC6hqnYQs4SXVkhQ+AVLmw++pzpEfL5xNtv76kQHIcwA ebina@EBINDEVICE
```

Next I am going to click profile -- > preference → sshkey



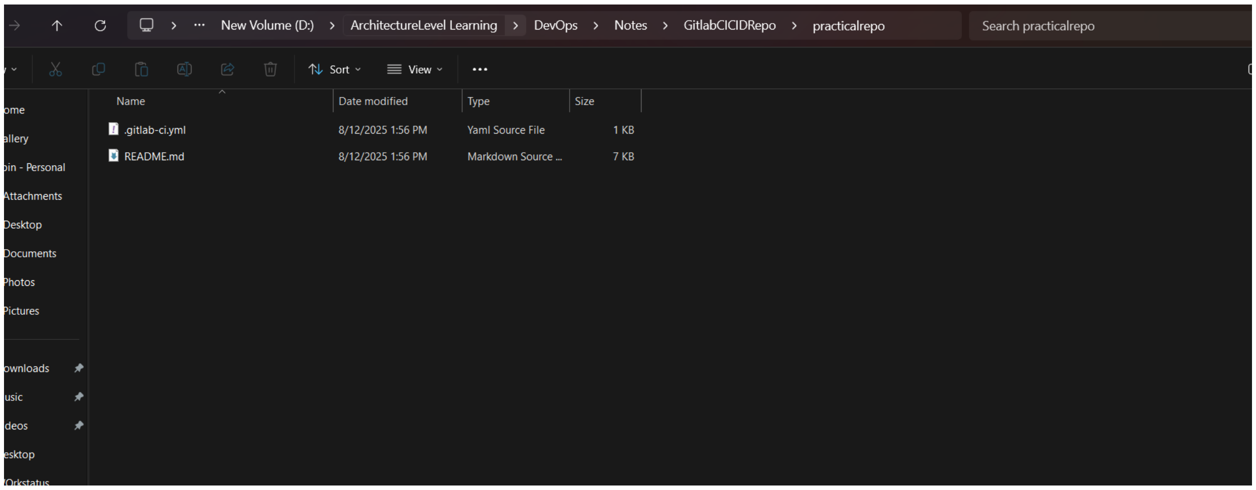
SSH Keys

SSH keys allow you to establish a secure connection between your computer and GitLab. SSH fingerprints verify that the client is connecting to the correct host. Check the current instance configuration.

Your SSH keys 2 · Add new key

Title	Key	Usage type	Created	Last used	Expires	Actions
practical-sshkey	ff:33:d1:75:87:72:11:f2:70:96:55:86:ba:cd:08:ac	Authentication & Signing	1 minute ago	Never	2026-08-12	Revoke
local_windows_shh_key	1d:95:e7:45:cb:95:c0:c4:5f:7c:61:91:e2:a4:b4:26	Authentication & Signing	5 days ago	19 hours ago	2026-08-07	Revoke

Now I am going to proceed clone with ssh



Name	Date modified	Type	Size
.gitlab-ci.yml	8/12/2025 1:56 PM	Yaml Source File	1 KB
README.md	8/12/2025 1:56 PM	Markdown Source ...	7 KB

Set global username and email

```
git config --global user.name "Your Name"

git config --global user.email your.email@example.com
```

verify setting

```
git config --global –list
```

httpstoken

---

click on Profile -→ Preference --→ AccessToken →Add new Token

⍰ **Global variables**

⍰ **Job-level variables**

⍰ **Secret/project variables**

⍰ **File variables**

⍰ **Branch-based logic**

⍰ **Environment-specific jobs**

```
# GLOBAL VARIABLE (available in all jobs)

variables:

  DEPLOY_ENV: staging

  DOCKER_IMAGE: node:18-alpine


stages:
```

```yaml
  - build
  - test
  - deploy

# JOB WITH GLOBAL VARIABLES
build-job:
  stage: build
  image: $DOCKER_IMAGE
  script:
    - echo "Building project in $DEPLOY_ENV environment"
    - npm install
    - npm run build

# JOB-LEVEL VARIABLE
test-job:
  stage: test
  variables:
    NODE_ENV: test
  script:
    - echo "Running tests in $NODE_ENV mode"
    - npm test

#  USING SECRET VARIABLES (set in GitLab UI)
deploy-job:
  stage: deploy
  script:
```

```
  - echo "Deploying to production..."
  - echo "Using secret token: $PROD_API_TOKEN"  # This value is masked
 only:
  - main  # deploy only from main branch
 environment:
  name: production
```

Example: Using a File Variable (for certificates)

Assume you set a GitLab variable like this in **UI → CI/CD → Variables**:

- Key: CERTIFICATE_FILE
- Type: **File**
- Value: paste the certificate contents
-

```
cert-job:
 stage: deploy
 script:
  - echo "Using cert from $CERTIFICATE_FILE"
  - cat "$CERTIFICATE_FILE" > /tmp/cert.pem
  - openssl verify /tmp/cert.pem
```

Branch-based Job Using $CI_COMMIT_BRANCH

```
conditional-job:
 stage: test
```

```
  script:
    - |
      if [ "$CI_COMMIT_BRANCH" = "main" ]; then
        echo "This is the main branch"
      else
        echo "This is not main branch"
      fi
```

Trigger a job only on tags

```
tagged-deploy:
  stage: deploy
  script:
    - echo "Deploying tagged release: $CI_COMMIT_TAG"
  only:
    - tags
```

💡 What is Maven?

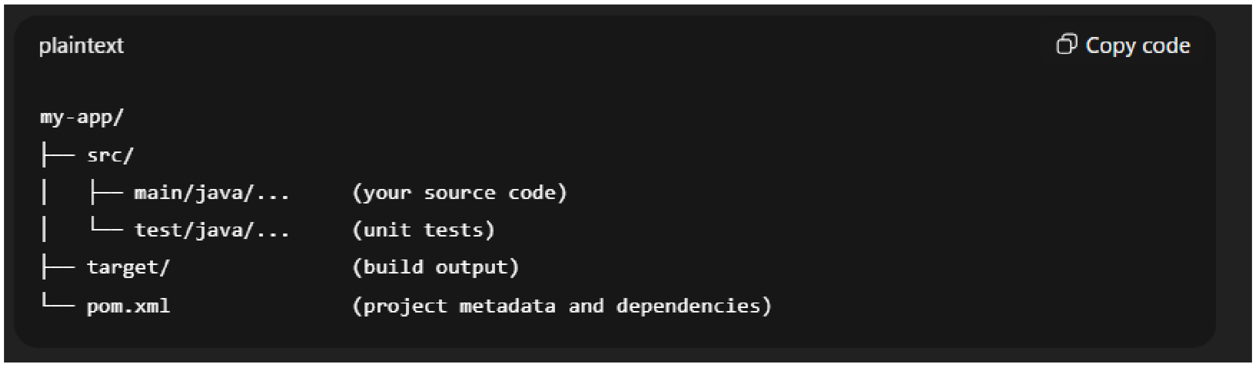
Maven is a build automation and dependency management tool used primarily for Java projects.

Key Features:

Feature	Description
Build Tool	Compiles Java code, runs tests, packages JAR/WAR files
Dependency Manager	Automatically downloads libraries (JARs) from a central repository
Project Management	Uses a standard folder structure and pom.xml file
Plugin Support	Supports plugins for compiling, testing, deploying, and generating reports

---

Maven Project Structure:



```
my-app/
├── src/
│   ├── main/java/...      (your source code)
│   └── test/java/...      (unit tests)
├── target/                (build output)
└── pom.xml                (project metadata and dependencies)
```

What is pom.xml?

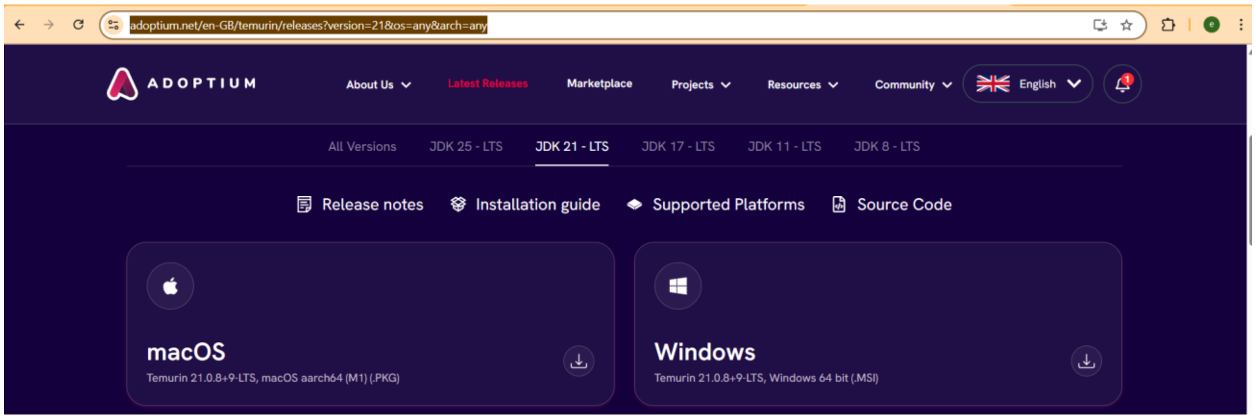
- Stands for **Project Object Model**
- It defines:
    - Project name, version, packaging
    - Dependencies (libraries your project needs)
    - Build plugins (e.g., compiler, test runner)

- ⚙️ **Common Maven Commands:**

Command	What it Does
`mvn compile`	Compiles the source code
`mvn test`	Runs unit tests
`mvn package`	Packages into `.jar` or `.war`
`mvn install`	Installs the package to local repo
`mvn clean`	Deletes the `target/` directory
`mvn dependency:tree`	Shows dependency hierarchy

Installtion first first got to:

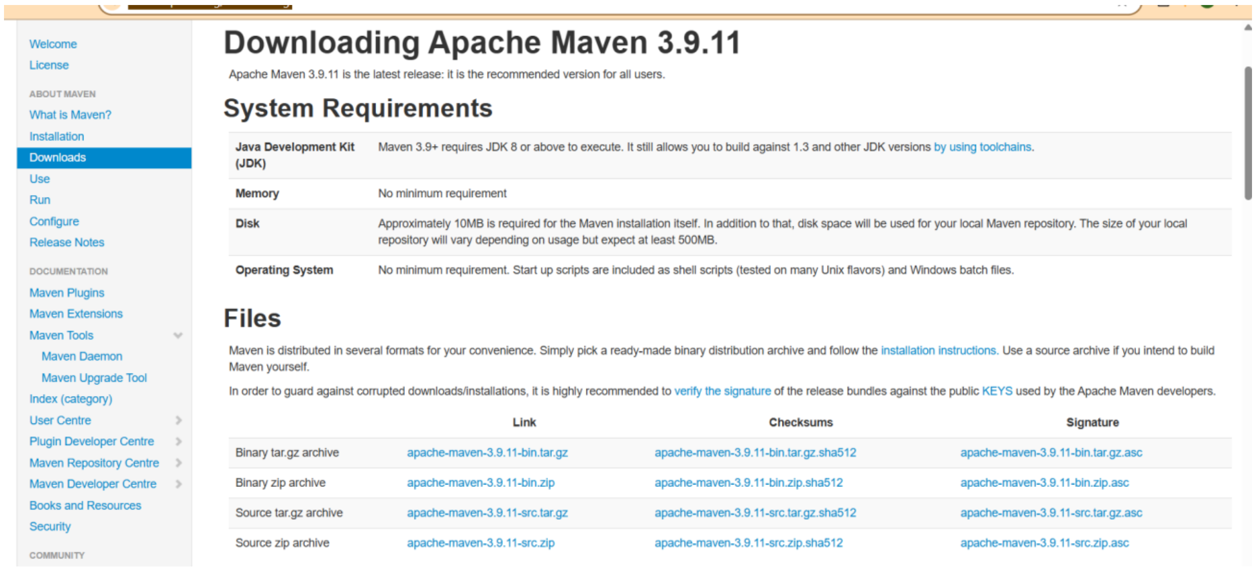
https://adoptium.net/en-GB/temurin/releases?version=21&os=any&arch=any



downlad msi installer

Install maven and extract folder add it into environment variable go to path add value

https://maven.apache.org/download.cgi



Downloading Apache Maven 3.9.11

Apache Maven 3.9.11 is the latest release: it is the recommended version for all users.

System Requirements

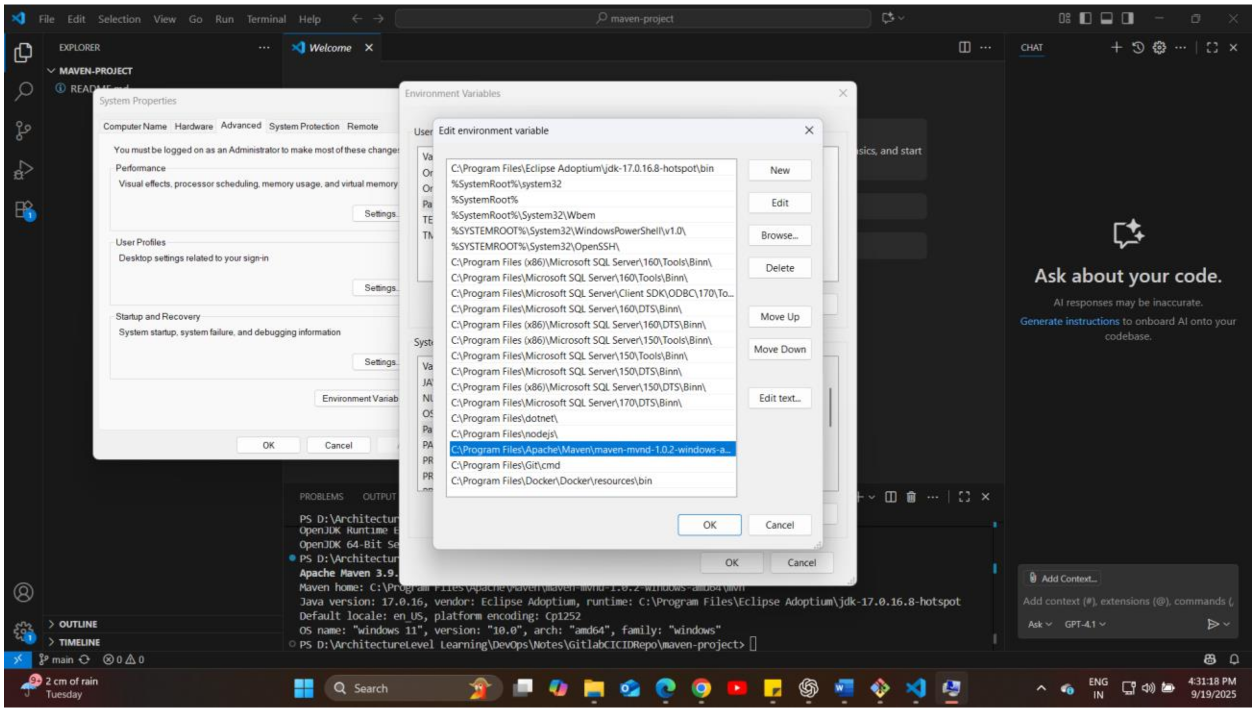
Java Development Kit (JDK)	Maven 3.9+ requires JDK 8 or above to execute. It still allows you to build against 1.3 and other JDK versions by using toolchains.
Memory	No minimum requirement
Disk	Approximately 10MB is required for the Maven installation itself. In addition to that, disk space will be used for your local Maven repository. The size of your local repository will vary depending on usage but expect at least 500MB.
Operating System	No minimum requirement. Start up scripts are included as shell scripts (tested on many Unix flavors) and Windows batch files.

Files

Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution archive and follow the installation instructions. Use a source archive if you intend to build Maven yourself.

In order to guard against corrupted downloads/installations, it is highly recommended to verify the signature of the release bundles against the public KEYS used by the Apache Maven developers.

	Link	Checksums	Signature
Binary tar.gz archive	apache-maven-3.9.11-bin.tar.gz	apache-maven-3.9.11-bin.tar.gz.sha512	apache-maven-3.9.11-bin.tar.gz.asc
Binary zip archive	apache-maven-3.9.11-bin.zip	apache-maven-3.9.11-bin.zip.sha512	apache-maven-3.9.11-bin.zip.asc
Source tar.gz archive	apache-maven-3.9.11-src.tar.gz	apache-maven-3.9.11-src.tar.gz.sha512	apache-maven-3.9.11-src.tar.gz.asc
Source zip archive	apache-maven-3.9.11-src.zip	apache-maven-3.9.11-src.zip.sha512	apache-maven-3.9.11-src.zip.asc



Edit environment variable

```
C:\Program Files\Eclipse Adoptium\jdk-17.0.16.8-hotspot\bin
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files (x86)\Microsoft SQL Server\160\Tools\Binn\
C:\Program Files\Microsoft SQL Server\160\Tools\Binn\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\To...
C:\Program Files\Microsoft SQL Server\160\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\160\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files\Microsoft SQL Server\170\DTS\Binn\
C:\Program Files\dotnet\
C:\Program Files\nodejs\
C:\Program Files\Apache\Maven\maven-mvnd-1.0.2-windows-a...
C:\Program Files\Git\cmd
C:\Program Files\Docker\Docker\resources\bin
```

System Properties: You must be logged on as an Administrator to make most of these changes. Performance: Visual effects, processor scheduling, memory usage, and virtual memory. User Profiles: Desktop settings related to your sign-in. Startup and Recovery: System startup, system failure, and debugging information.

```
PS D:\Architectur
OpenJDK Runtime E
OpenJDK 64-Bit Se
PS D:\Architectur
Apache Maven 3.9.
Maven home: C:\Program Files\Apache\Maven\maven-mvnd-1.0.2-windows-amd64\mvn
Java version: 17.0.16, vendor: Eclipse Adoptium, runtime: C:\Program Files\Eclipse Adoptium\jdk-17.0.16.8-hotspot
Default locale: en_US, platform encoding: Cp1252
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"
PS D:\ArchitectureLevel Learning\DevOps\Notes\GitlabCICIDRepo\maven-project>
```

Create hello world maven project

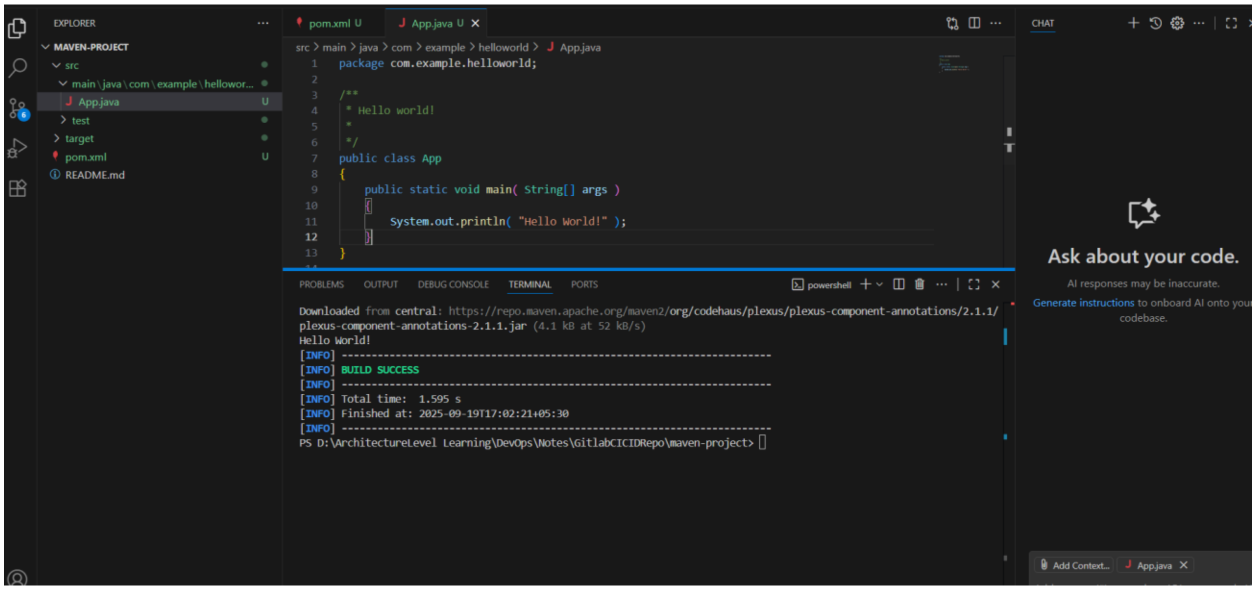
mvn archetype:generate -DgroupId=com.example.helloworld -DartifactId=helloworld -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

compile

mvn clean compile

run:

mvn exec:java



EXPLORER
MAVEN-PROJECT
src
main \ java \ com \ example \ hellowor...
App.java
test
target
pom.xml
README.md
pom.xml U
App.java U
src > main > java > com > example > helloworld > App.java
CHAT
Ask about your code.
AI responses may be inaccurate.
Generate instructions to onboard AI onto your codebase.
Add Context...
App.java

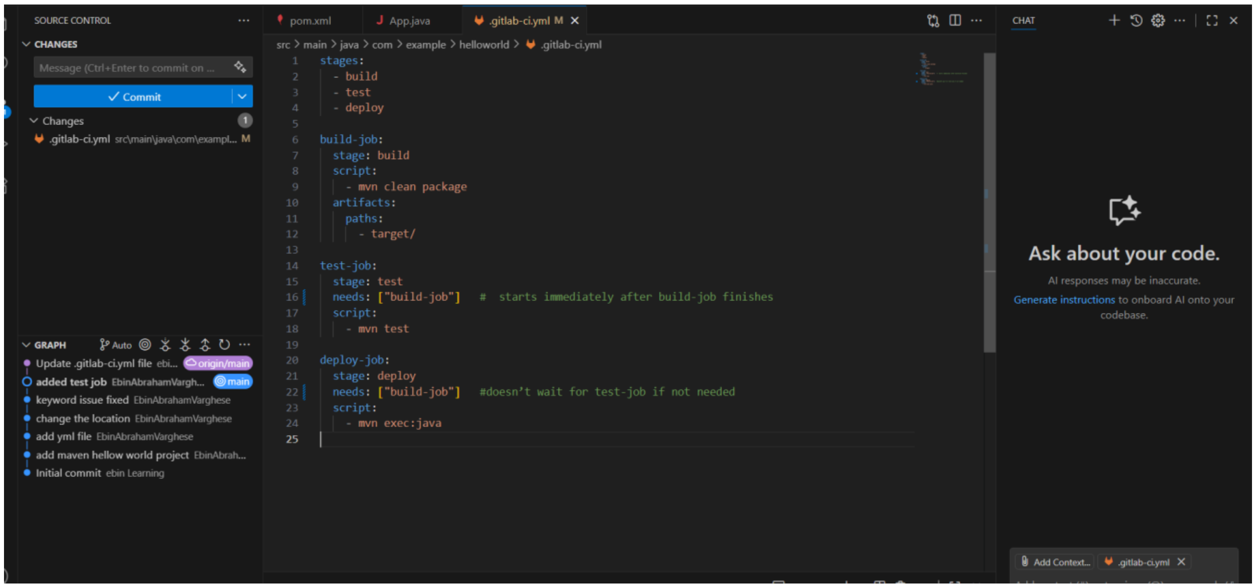
```java
package com.example.helloworld;

/**
 * Hello world!
 *
 */
public class App
{
    public static void main( String[] args )
    {
        System.out.println( "Hello World!" );
    }
}
```

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
powershell

```
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-component-annotations/2.1.1/
plexus-component-annotations-2.1.1.jar (4.1 kB at 52 kB/s)
Hello World!
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  1.595 s
[INFO] Finished at: 2025-09-19T17:02:21+05:30
[INFO] ------------------------------------------------------------------------
PS D:\ArchitectureLevel Learning\DevOps\Notes\GitlabCICIDRepo\maven-project>
```

Dockerimage availablility: https://hub.docker.com/_/maven/tags?name=latest

After project execution I am going to create yml file (.gitlab-ci.yml)



SOURCE CONTROL
CHANGES
Message (Ctrl+Enter to commit on ...
✓ Commit
Changes
.gitlab-ci.yml src\main\java\com\exampl... M
GRAPH
Auto
pom.xml
App.java
.gitlab-ci.yml M
src > main > java > com > example > helloworld > .gitlab-ci.yml
CHAT
Ask about your code.
AI responses may be inaccurate.
Generate instructions to onboard AI onto your codebase.
Add Context...
.gitlab-ci.yml

```
Update .gitlab-ci.yml file  ebi...  origin/main
added test job  EbinAbrahamVargh...  main
keyword issue fixed  EbinAbrahamVarghese
change the location  EbinAbrahamVarghese
add yml file  EbinAbrahamVarghese
add maven hellow world project  EbinAbrah...
Initial commit  ebin Learning
```

```yaml
stages:
  - build
  - test
  - deploy

build-job:
  stage: build
  script:
    - mvn clean package
  artifacts:
    paths:
      - target/

test-job:
  stage: test
  needs: ["build-job"]   #  starts immediately after build-job finishes
  script:
    - mvn test

deploy-job:
  stage: deploy
  needs: ["build-job"]   #doesn't wait for test-job if not needed
  script:
    - mvn exec:java

```