**Expree Eat Web DevOps Report**

SUBMITTED TO MIT SCHOOL OF COMPUTING

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE

**BACHELOR OF INFORMATION TECHNOLOGY**

**(Information Technology/-Data Analytics)**

**BY**

Pooja Mule MITU23BTITD007

Saraswati Shinde MITU23BTITD011

**Under The Guidance of**

Prof. Ashvini Jadhav

**DEPARTMENT OF INFORMATION TECHNOLOGY**

**MIT School of Computing**

**MIT Art, Design and Technology University Rajbaug Campus, Loni-Kalbhor, Pune 412201**

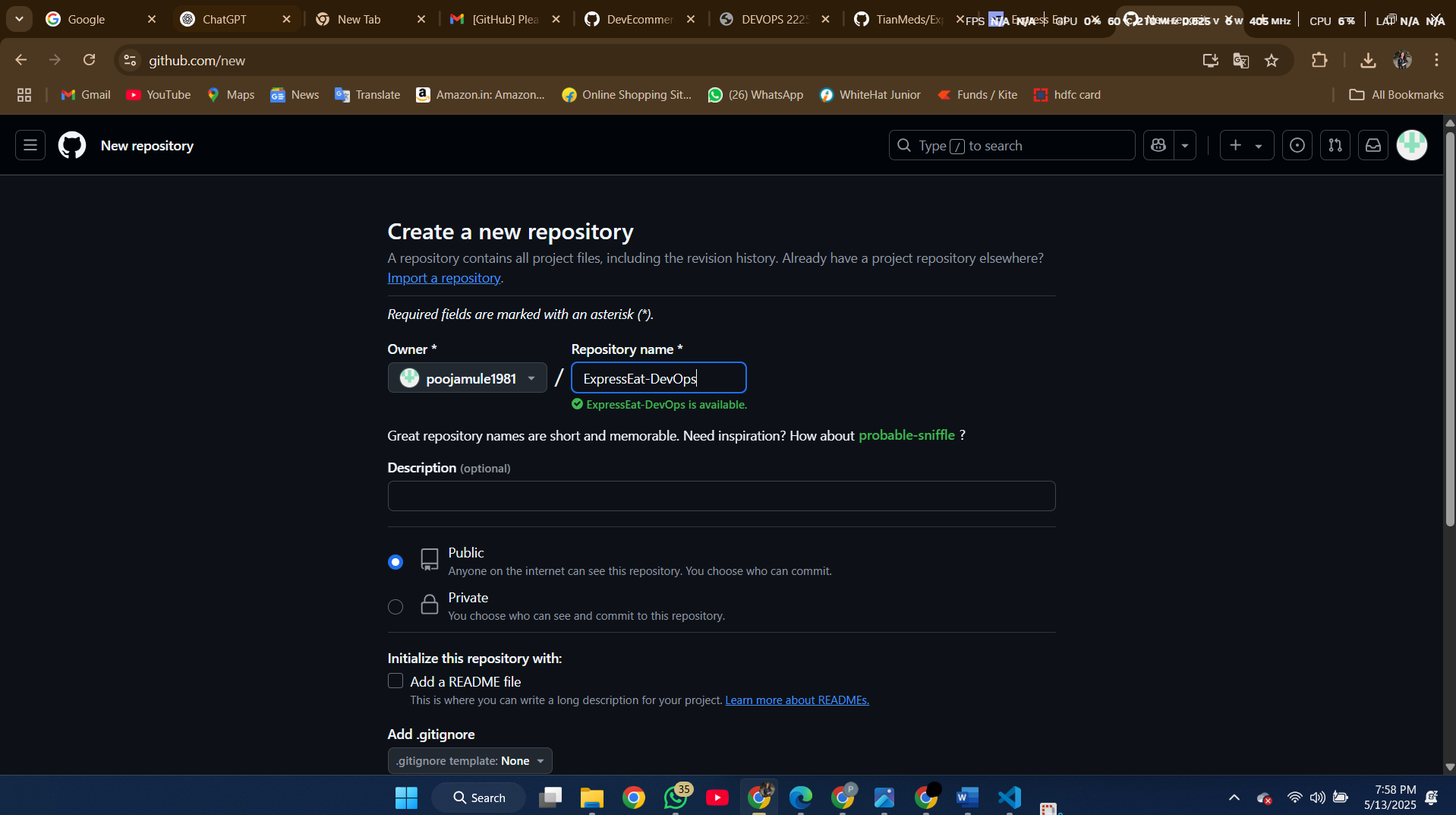
**2025-26**

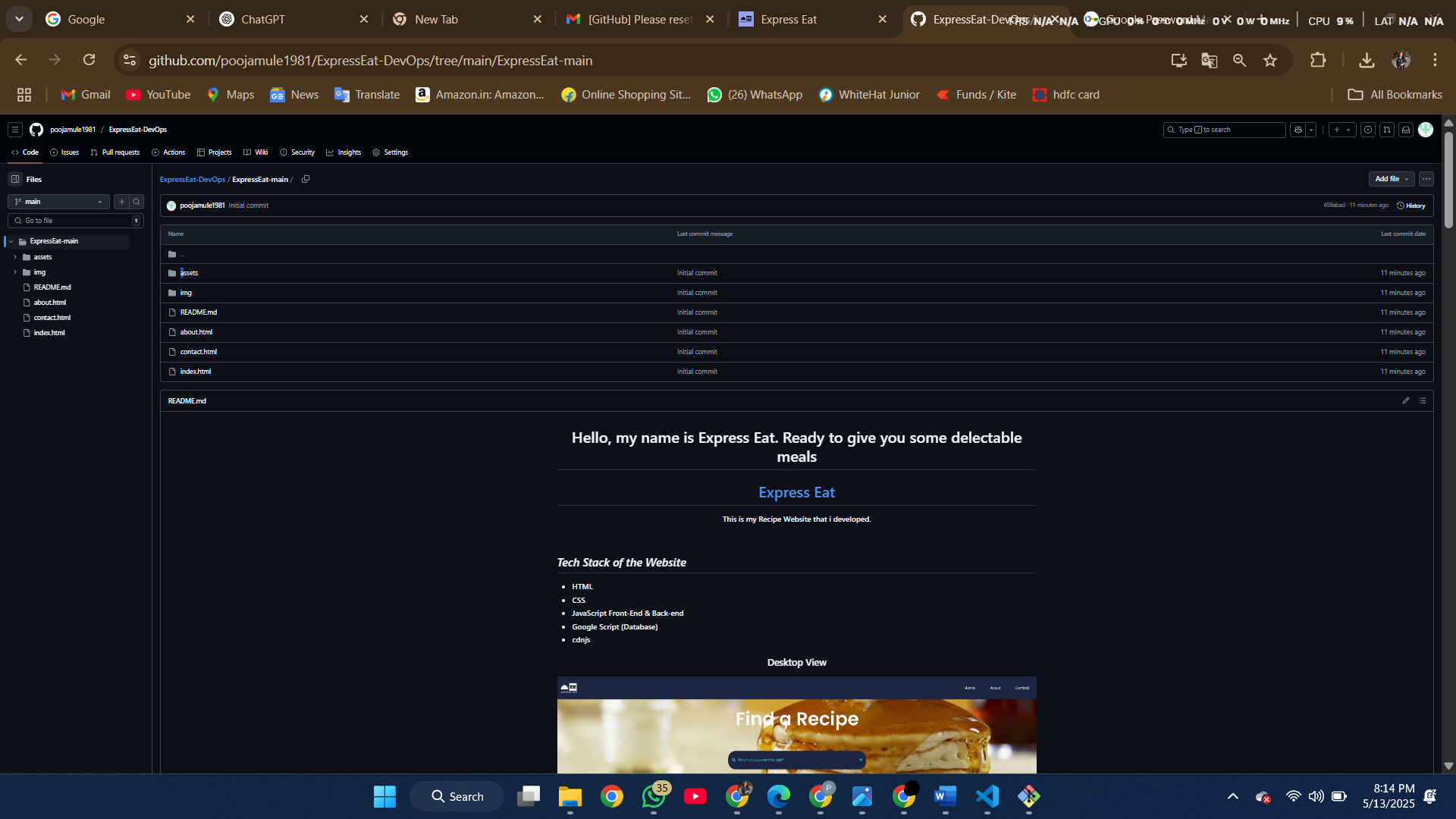
**Expree Eat Web DevOps Report**

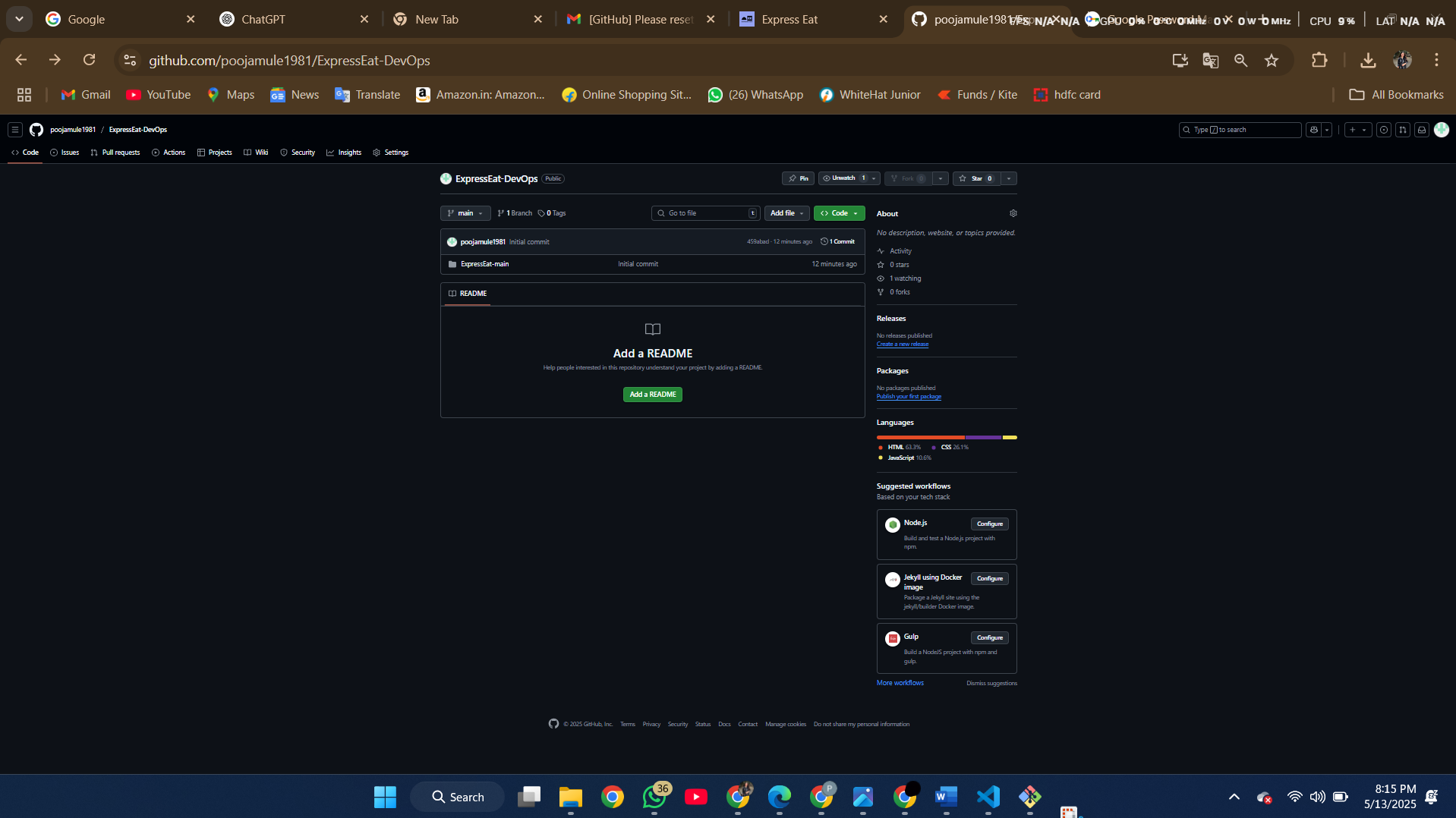
**Git, GitHub, and GitHub Desktop**

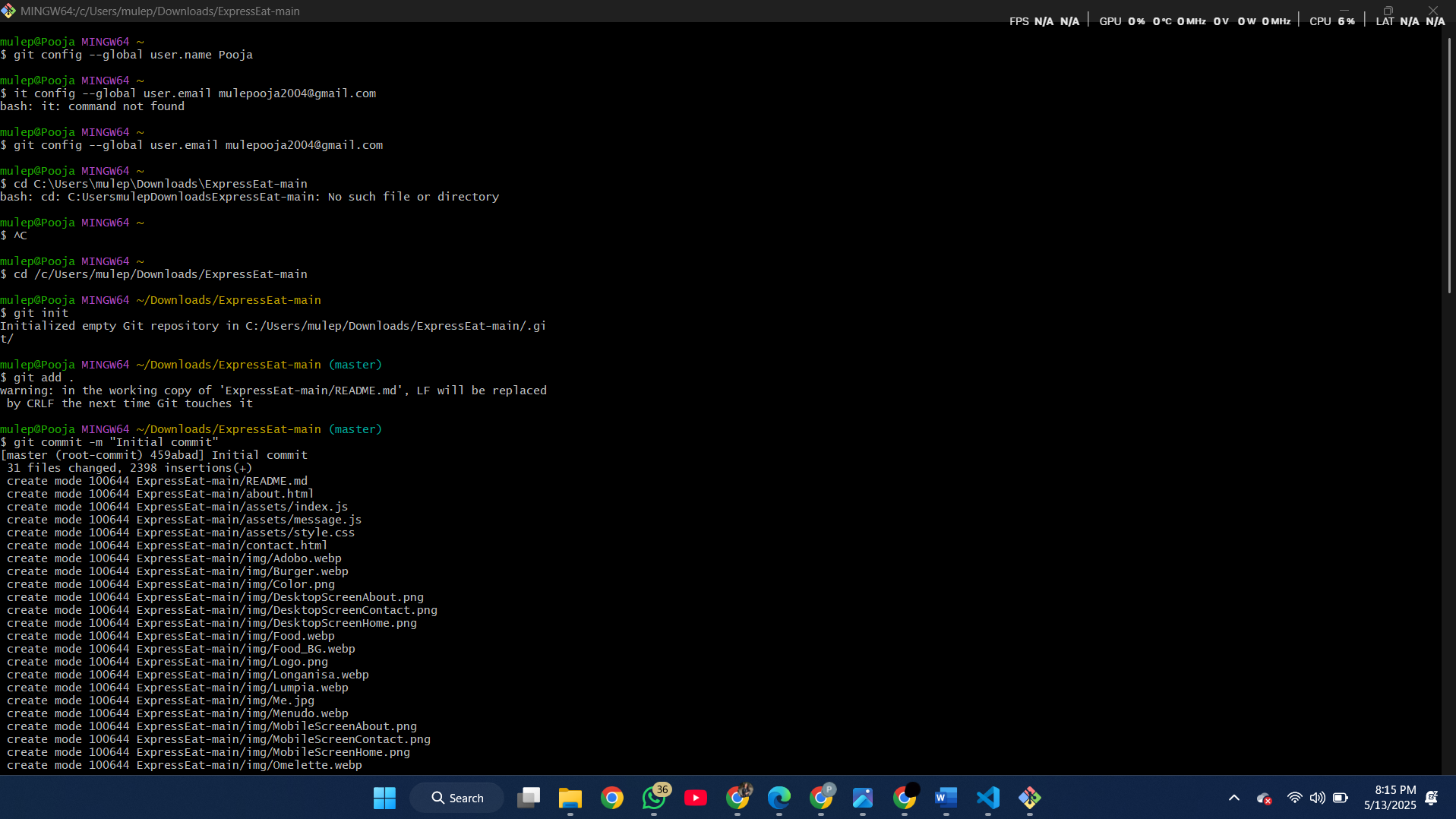
Version control is essential for any modern project. Git is a distributed version control system that tracks code changes over time[datanovia.com](https://www.datanovia.com/learn/programming/tools-and-ides/version-control-with-git-and-github.html#:~:text=Git%20is%20a%20distributed%20version,into%20your%20main%20project%20seamlessly). GitHub is a cloud-based platform built around Git: it hosts remote repositories and adds collaboration features (issues, pull requests, etc.)[datanovia.com](https://www.datanovia.com/learn/programming/tools-and-ides/version-control-with-git-and-github.html#:~:text=Git%20is%20a%20distributed%20version,into%20your%20main%20project%20seamlessly). GitHub Desktop is a free GUI application for Windows/Mac that lets you work with GitHub repos without command-line Git[docs.github.com](https://docs.github.com/en/desktop/overview/getting-started-with-github-desktop#:~:text=GitHub%20Desktop%20is%20a%20free%2C,For%20more%20information%2C%20see). It simplifies operations like cloning, committing, and pushing changes via a graphical interface[docs.github.com](https://docs.github.com/en/desktop/overview/getting-started-with-github-desktop#:~:text=GitHub%20Desktop%20is%20a%20free%2C,For%20more%20information%2C%20see). Using Git and GitHub together enables team collaboration, code history, and code reviews (via pull requests).

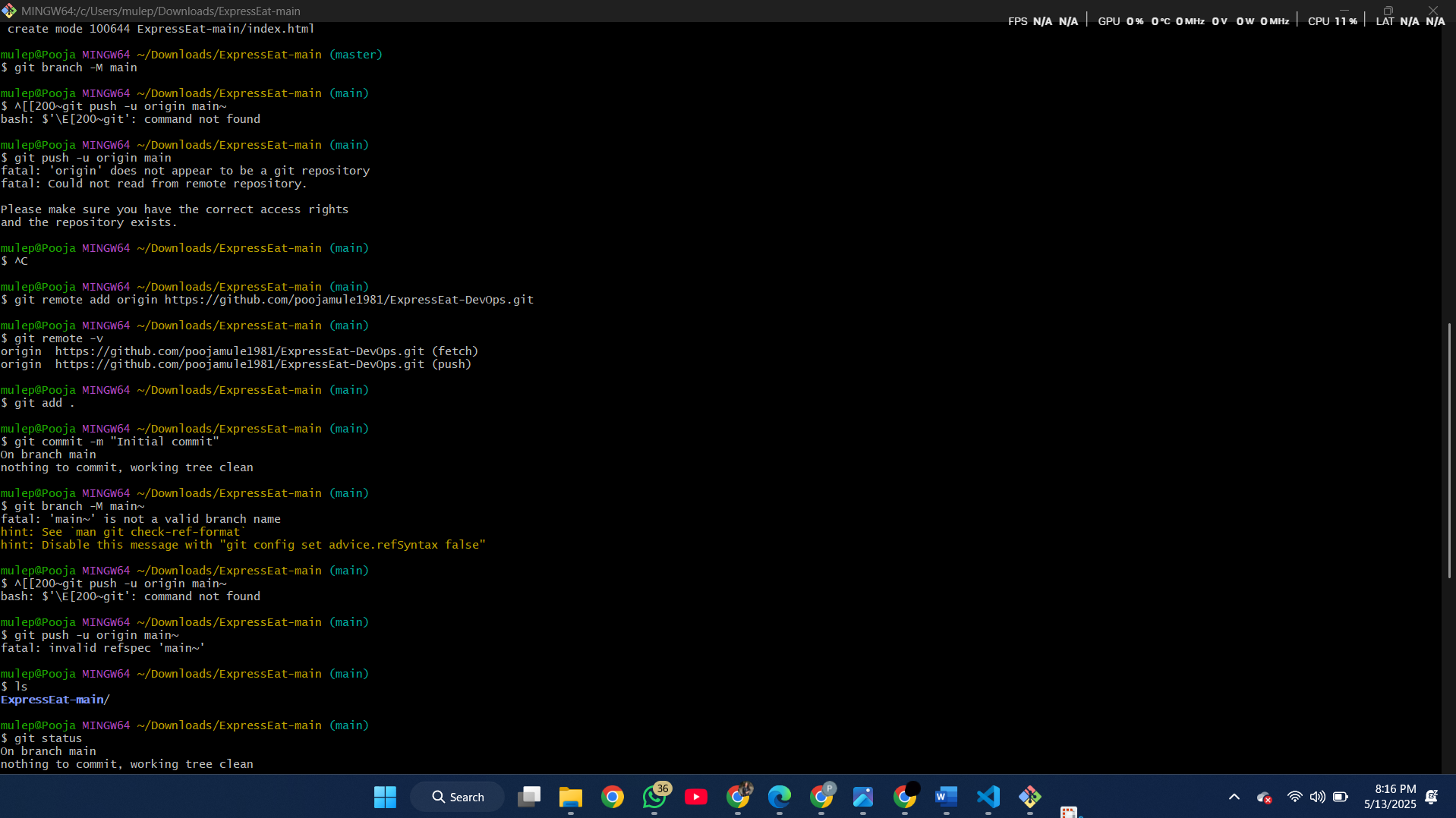
* *Key points:* GitHub stores the code; GitHub Desktop provides an easy UI for common Git tasks[docs.github.com](https://docs.github.com/en/desktop/overview/getting-started-with-github-desktop#:~:text=GitHub%20Desktop%20is%20a%20free%2C,For%20more%20information%2C%20see)[datanovia.com](https://www.datanovia.com/learn/programming/tools-and-ides/version-control-with-git-and-github.html#:~:text=GitHub%20is%20a%20web,especially%20when%20working%20with%20teams).
* *Benefits:* Enables branching, merging, and version history with visual tools (for beginners or quick workflows)[datanovia.com](https://www.datanovia.com/learn/programming/tools-and-ides/version-control-with-git-and-github.html#:~:text=Git%20is%20a%20distributed%20version,into%20your%20main%20project%20seamlessly)[docs.github.com](https://docs.github.com/en/desktop/overview/getting-started-with-github-desktop#:~:text=GitHub%20Desktop%20is%20a%20free%2C,For%20more%20information%2C%20see).  
  *(Screenshot placeholder: GitHub Desktop showing a cloned repository.)*

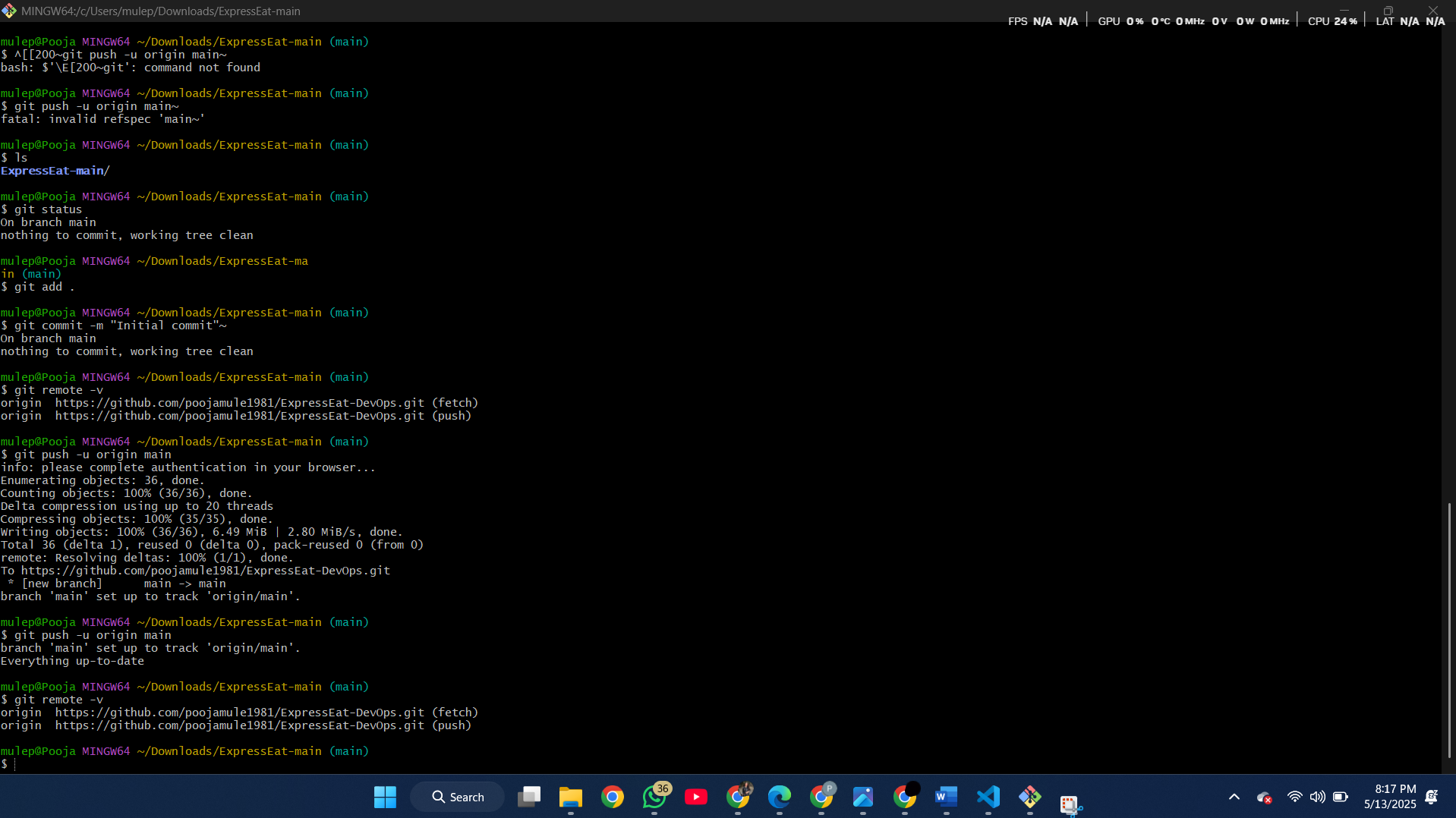




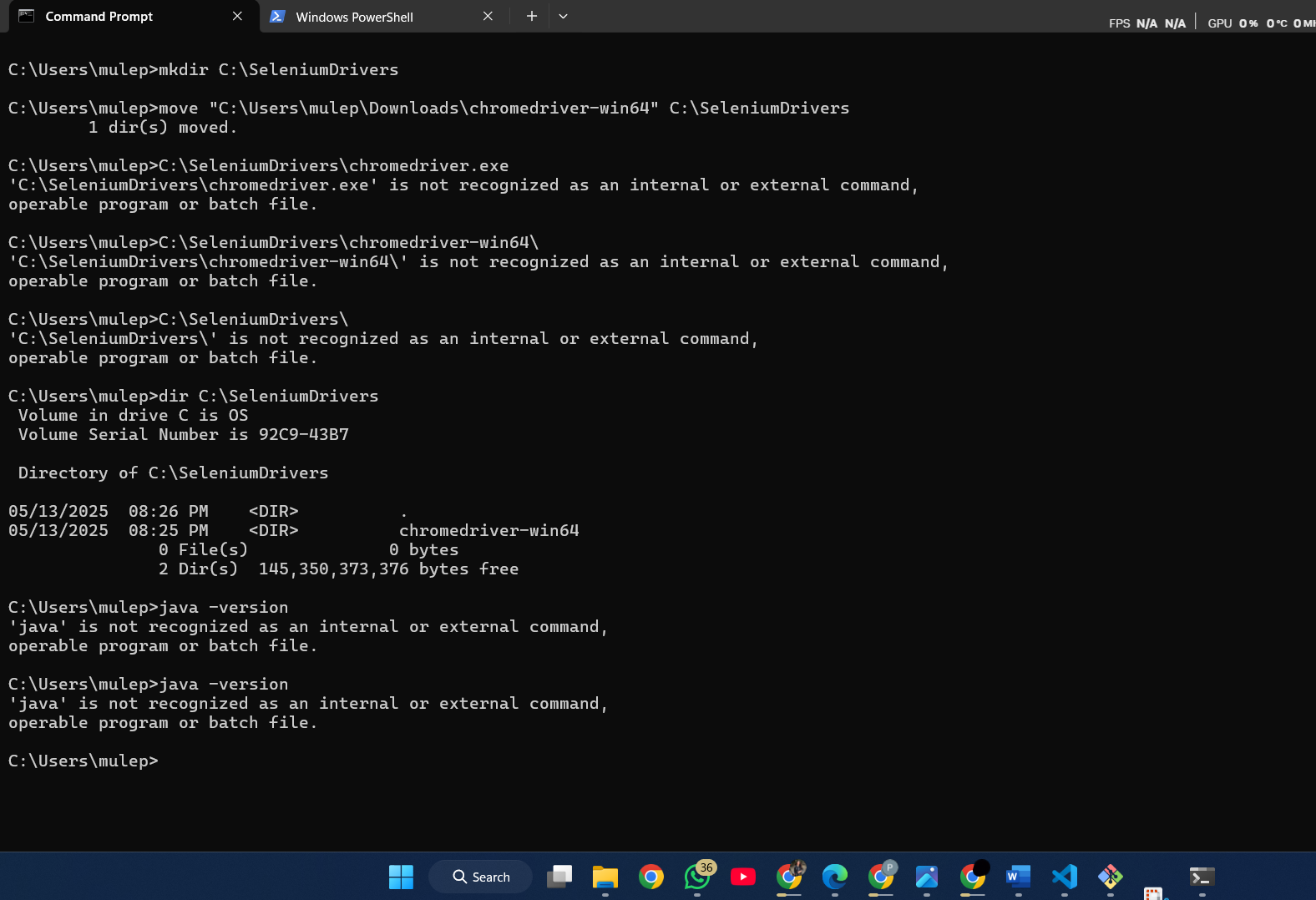












**Eclipse IDE**

Eclipse is a popular Java-integrated development environment (IDE) used for building applications[codejava.net](https://www.codejava.net/ides/eclipse/what-is-eclipse-ide-for-beginner#:~:text=Perhaps%2C%20Eclipse%20is%C2%A0%20the%20most,about%20this%20programmer%E2%80%99s%20favorite%20tool). It provides code editors with syntax highlighting, project management, debugging tools, and refactoring support[codejava.net](https://www.codejava.net/ides/eclipse/what-is-eclipse-ide-for-beginner#:~:text=Perhaps%2C%20Eclipse%20is%C2%A0%20the%20most,about%20this%20programmer%E2%80%99s%20favorite%20tool). For Java web projects (like Expree Eat Web), one typically uses the “Eclipse IDE for Java EE Developers”, which includes support for web technologies and Maven integration. Eclipse is free and open-source, and supports plugins for many languages and frameworks[codejava.net](https://www.codejava.net/ides/eclipse/what-is-eclipse-ide-for-beginner#:~:text=Eclipse%20is%20free%20and%20open,org). Developers use Eclipse to write, compile, and run code locally before any deployment.

* *Features:* Code editing, debugging, integrated testing, and project import (including Maven projects)[codejava.net](https://www.codejava.net/ides/eclipse/what-is-eclipse-ide-for-beginner#:~:text=Perhaps%2C%20Eclipse%20is%C2%A0%20the%20most,about%20this%20programmer%E2%80%99s%20favorite%20tool)[codejava.net](https://www.codejava.net/ides/eclipse/what-is-eclipse-ide-for-beginner#:~:text=Eclipse%20is%20free%20and%20open,org).
* *Advantage:* Provides a unified workspace so that setup (JDK, libraries, servers) is managed within the IDE.  
  *(Screenshot placeholder: Eclipse showing the Expree Eat project structure.)*

**Maven**

Maven is a widely used build automation and project management tool for Java projects[dev.to](https://dev.to/imsushant12/simplify-your-builds-a-quick-maven-guide-with-examples-2bp9#:~:text=Maven%20is%20a%20widely%20used,compilation%2C%20testing%2C%20packaging%2C%20and%20deployment). It uses a Project Object Model (POM) in an XML file (pom.xml) to define project structure, dependencies, plugins, and build lifecycle[dev.to](https://dev.to/imsushant12/simplify-your-builds-a-quick-maven-guide-with-examples-2bp9#:~:text=,test%2C%20package%2C%20install%2C%20and%20deploy)[dev.to](https://dev.to/imsushant12/simplify-your-builds-a-quick-maven-guide-with-examples-2bp9#:~:text=Maven%20works%20based%20on%20the,lifecycle%2C%20dependencies%2C%20and%20other%20aspects). Maven handles compiling code, running tests, and packaging artifacts in a standardized way. With Maven, you declare all library dependencies in pom.xml, and it automatically downloads required JARs from central repositories[dev.to](https://dev.to/imsushant12/simplify-your-builds-a-quick-maven-guide-with-examples-2bp9#:~:text=Maven%20works%20based%20on%20the,lifecycle%2C%20dependencies%2C%20and%20other%20aspects). It enforces a standard directory layout and build steps (validate, compile, test, package, etc.)[dev.to](https://dev.to/imsushant12/simplify-your-builds-a-quick-maven-guide-with-examples-2bp9#:~:text=,test%2C%20package%2C%20install%2C%20and%20deploy)[geeksforgeeks.org](https://www.geeksforgeeks.org/maven-build-automation/#:~:text=Maven%20is%20a%20powerful%20build,dependency%20management%2C%20and%20build%20execution).

* *Project setup:* A typical Expree Eat Web project will have a pom.xml specifying packaging (e.g. WAR), frameworks (e.g. Spring Boot Starter), and TestNG for testing.
* *Build lifecycle:* Maven phases (compile, test, package) ensure consistent builds with a single command (mvn package)[dev.to](https://dev.to/imsushant12/simplify-your-builds-a-quick-maven-guide-with-examples-2bp9#:~:text=,test%2C%20package%2C%20install%2C%20and%20deploy)[geeksforgeeks.org](https://www.geeksforgeeks.org/maven-build-automation/#:~:text=Maven%20is%20a%20powerful%20build,dependency%20management%2C%20and%20build%20execution).
* *Dependencies:* Example snippet (add this in pom.xml):

xml

Copy code

<dependencies>

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.4.0</version>

<scope>test</scope>

</dependency>

<!-- Add web frameworks or database drivers here -->

</dependencies>

*(Screenshot placeholder: Maven pom.xml file with dependencies.)*

**TestNG**

TestNG is an open-source testing framework for Java, designed to simplify writing and running tests[geeksforgeeks.org](https://www.geeksforgeeks.org/testng-tutorial/#:~:text=TestNG%20,with%20different%20sets%20of%20data). The “NG” stands for “Next Generation”, and it builds upon concepts from JUnit. TestNG allows annotation-based test methods, parameterized tests, and grouping of tests into suites[geeksforgeeks.org](https://www.geeksforgeeks.org/testng-tutorial/#:~:text=TestNG%20,with%20different%20sets%20of%20data). It is very flexible: tests can depend on each other, be grouped into categories, and support data-driven testing.

* *Usage:* In Expree Eat Web, test cases (e.g. for service methods or UI flows) are written as Java classes under src/test/java.
* *Features:* Annotate methods with @Test, define setup/teardown with @BeforeClass/@AfterClass, and use testng.xml to configure suites[geeksforgeeks.org](https://www.geeksforgeeks.org/testng-tutorial/#:~:text=TestNG%20,with%20different%20sets%20of%20data).
* *Example snippet:*

java

Copy code

import org.testng.annotations.Test;

public class LoginTest {

@Test

public void testUserLogin() {

// Example test steps for logging in to Expree Eat

}

}

*(Screenshot placeholder: Running TestNG tests in Eclipse or in console.)*

**Jenkins (CI Server)**

Jenkins is a popular open-source automation server used to implement continuous integration/continuous delivery (CI/CD) pipelines[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=Jenkins%20is%20an%20open,with%20over%20100%2C000%20installations%20worldwide). It can automatically build, test, and deploy applications when code changes are detected. Jenkins has a large plugin ecosystem to integrate with almost any toolchain[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=At%20its%20core%2C%20Jenkins%20provides,all%20types%20of%20development%20processes). In a DevOps workflow for Expree Eat Web, Jenkins would periodically (or via webhook) pull the latest GitHub code, run Maven builds and TestNG tests, and then package and deploy the application.

* *CI/CD Role:* Jenkins automates the pipeline stages (build, test, deploy) so developers get fast feedback on code changes[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=Jenkins%20is%20an%20open,with%20over%20100%2C000%20installations%20worldwide).
* *Plugins:* Use Git or GitHub plugins to connect to the code repo, Maven plugin to build, and perhaps Docker or AWS plugins for containerization/deployment.
* *Jenkinsfile:* A script (Jenkinsfile) defines the pipeline stages. For example:

groovy

Copy code

pipeline {

agent any

stages {

stage('Build') {

steps { sh 'mvn clean package' }

}

stage('Test') {

steps { sh 'mvn test' }

}

stage('Deploy') {

steps {

// e.g., build Docker image and push

sh 'docker build -t expree-eat-web .'

sh 'docker push your-docker-repo/expree-eat-web'

}

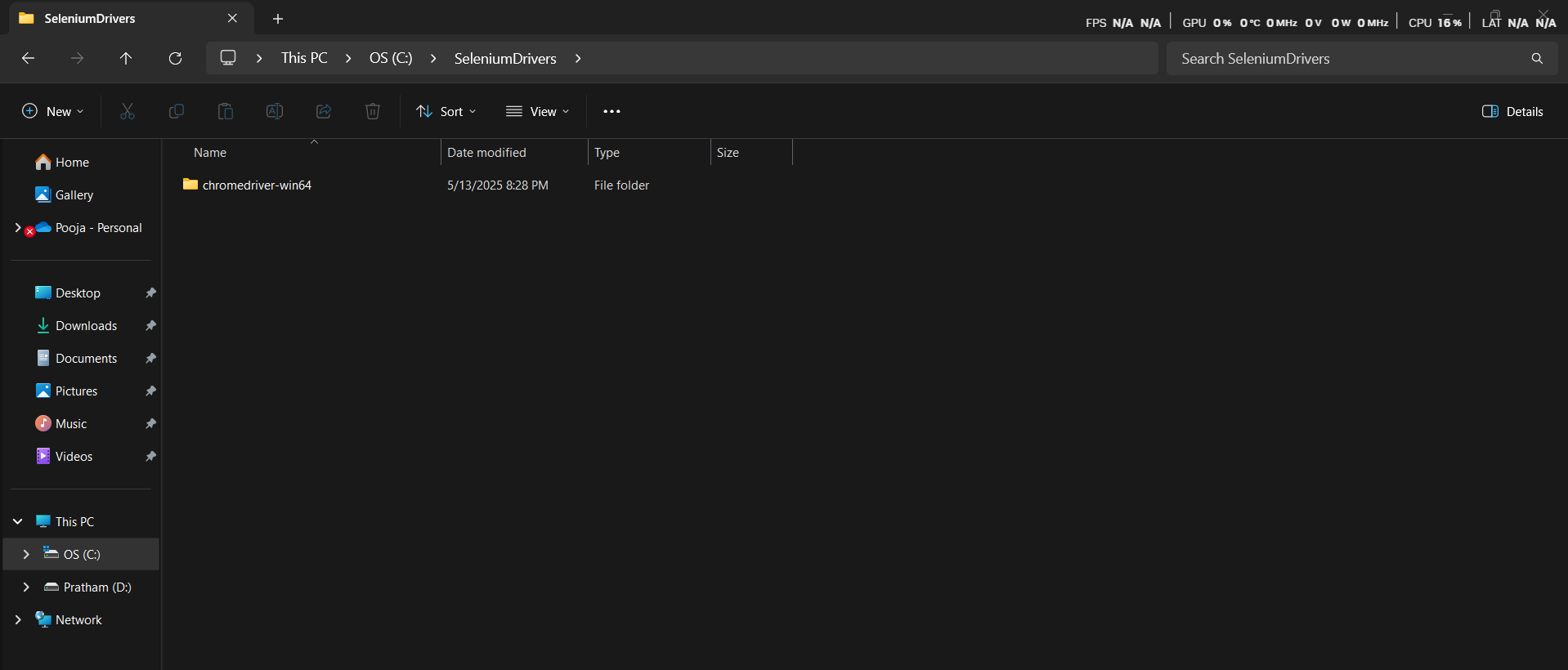
}

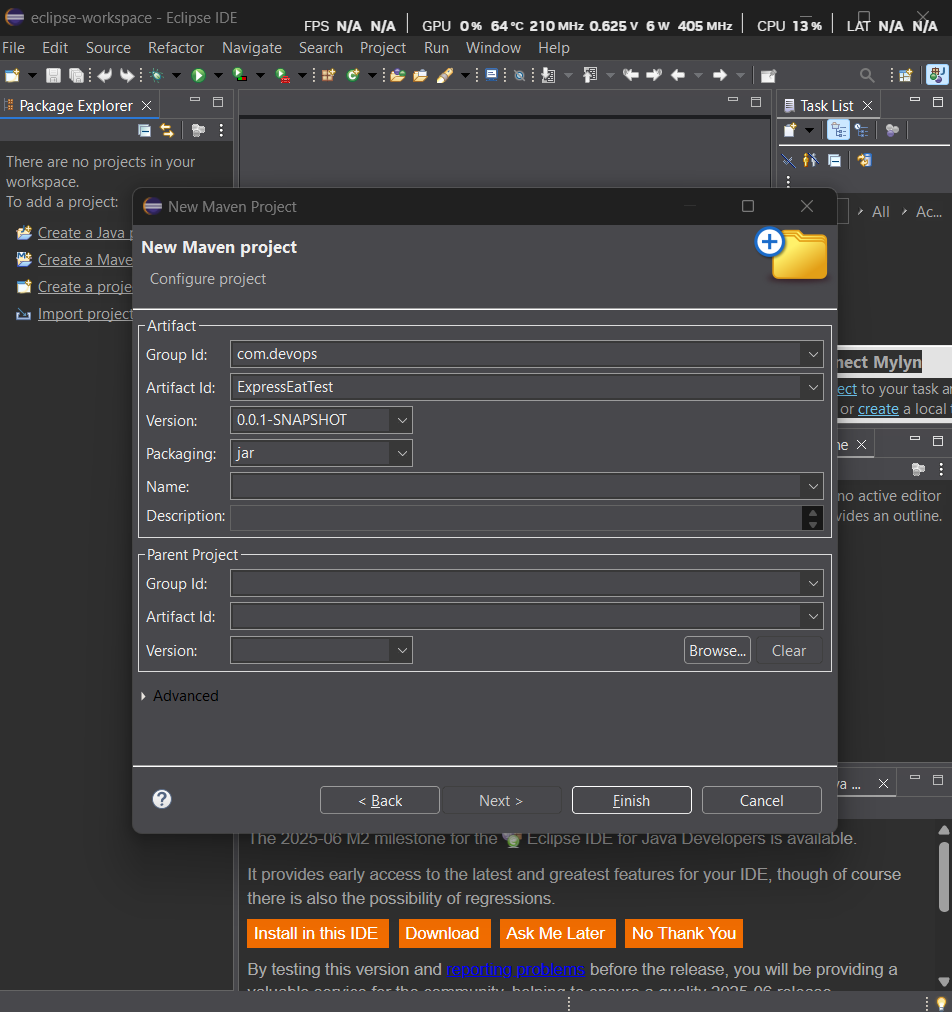
}

}

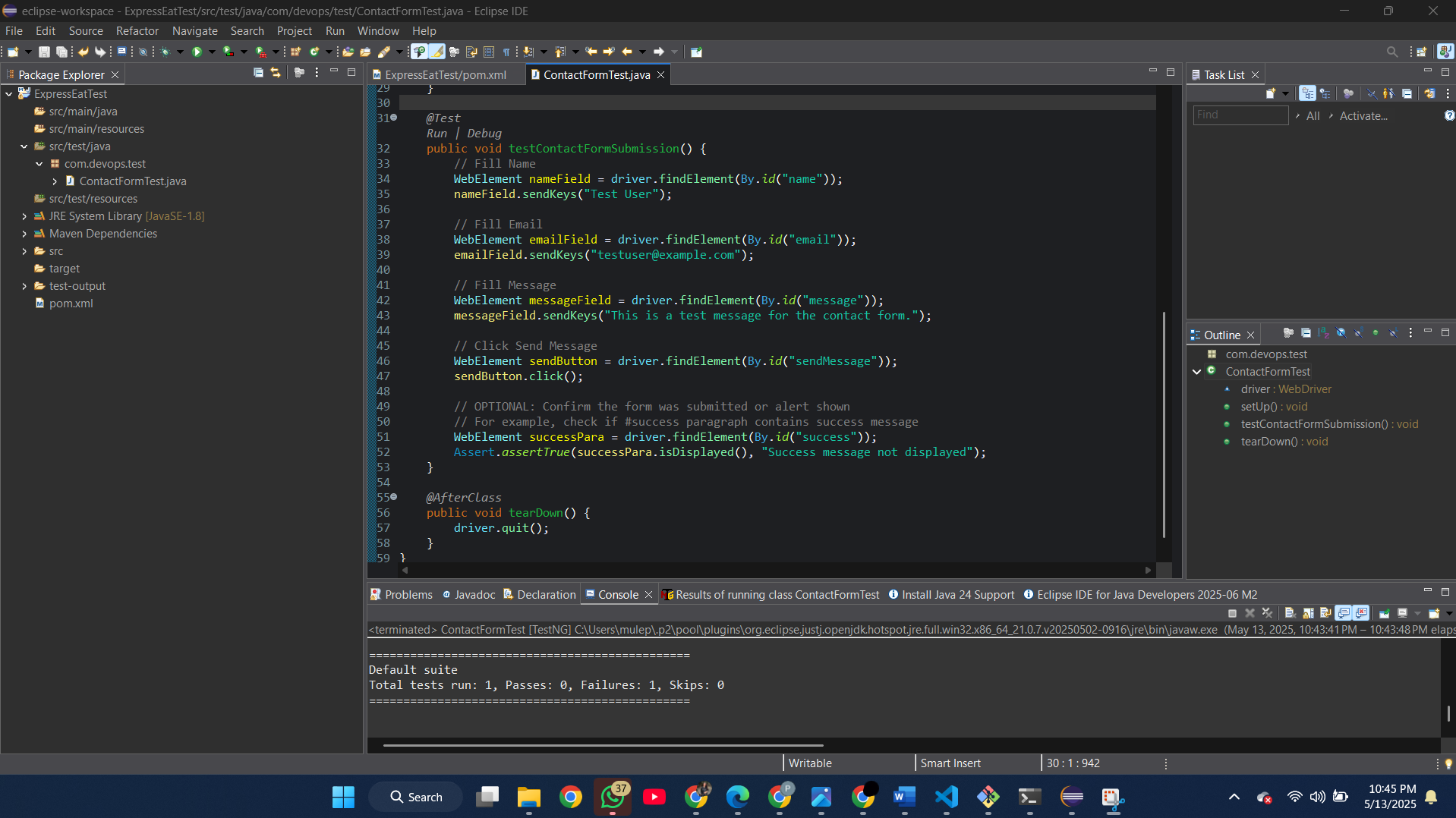
*(Screenshot placeholder: Jenkins pipeline run showing stages.)*

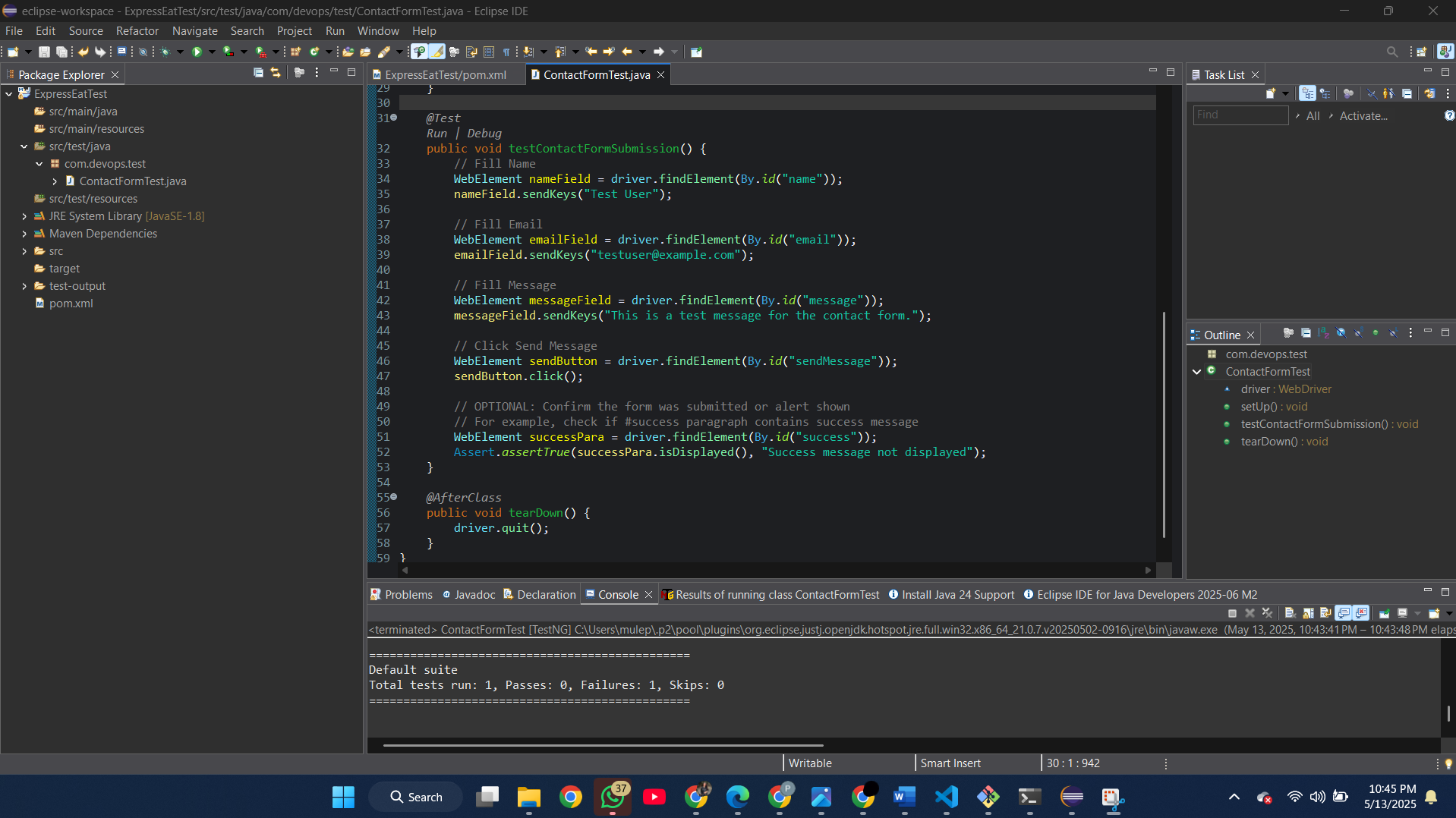


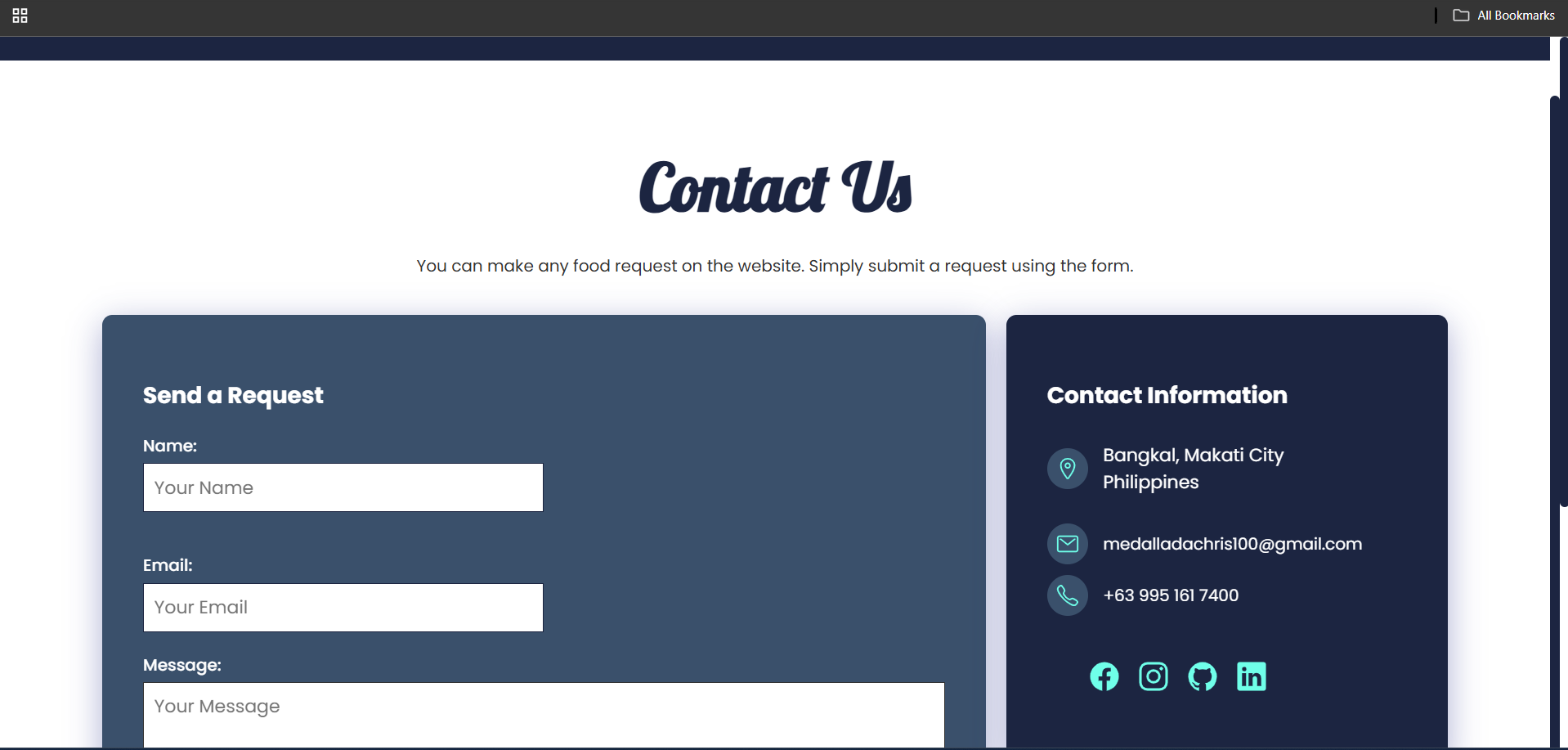


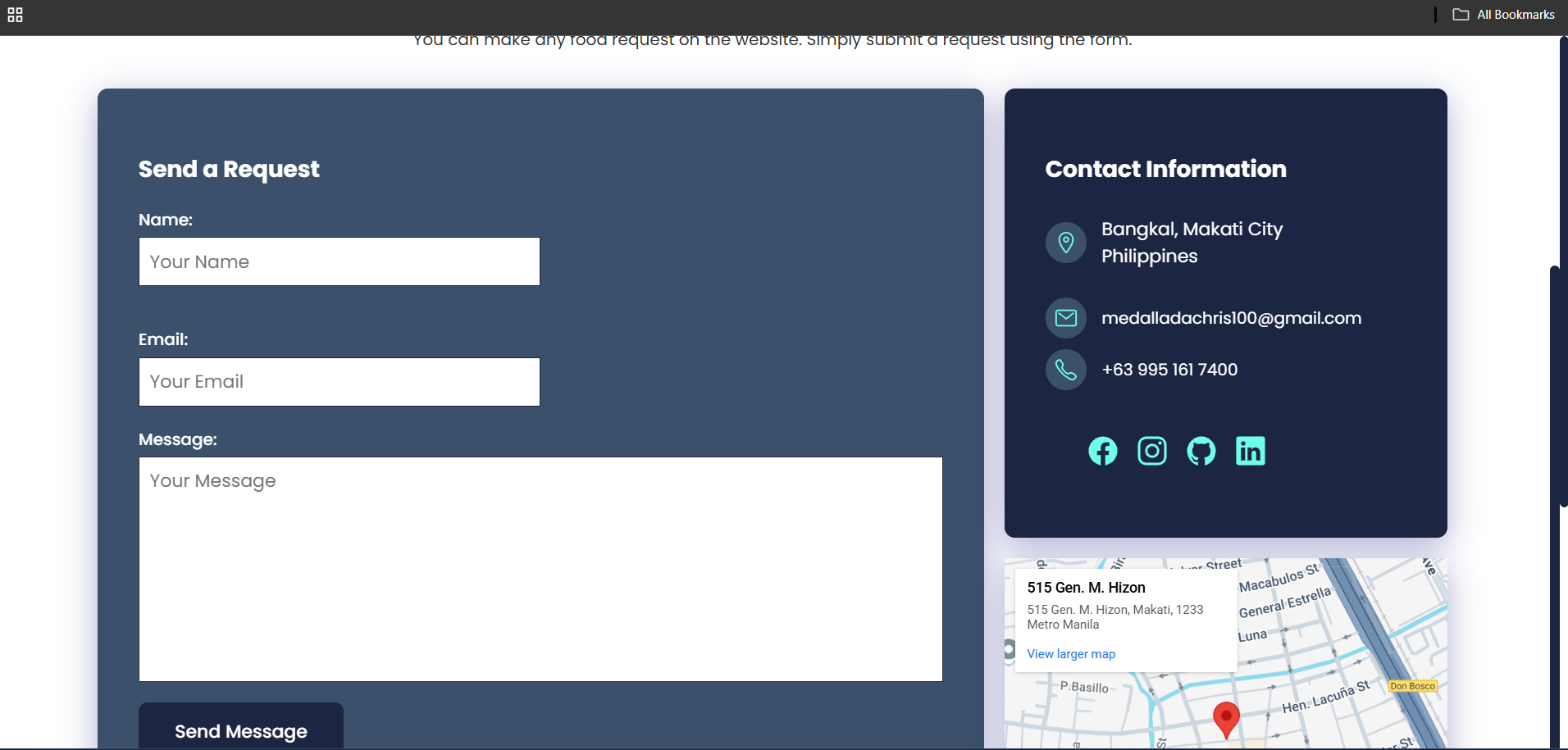




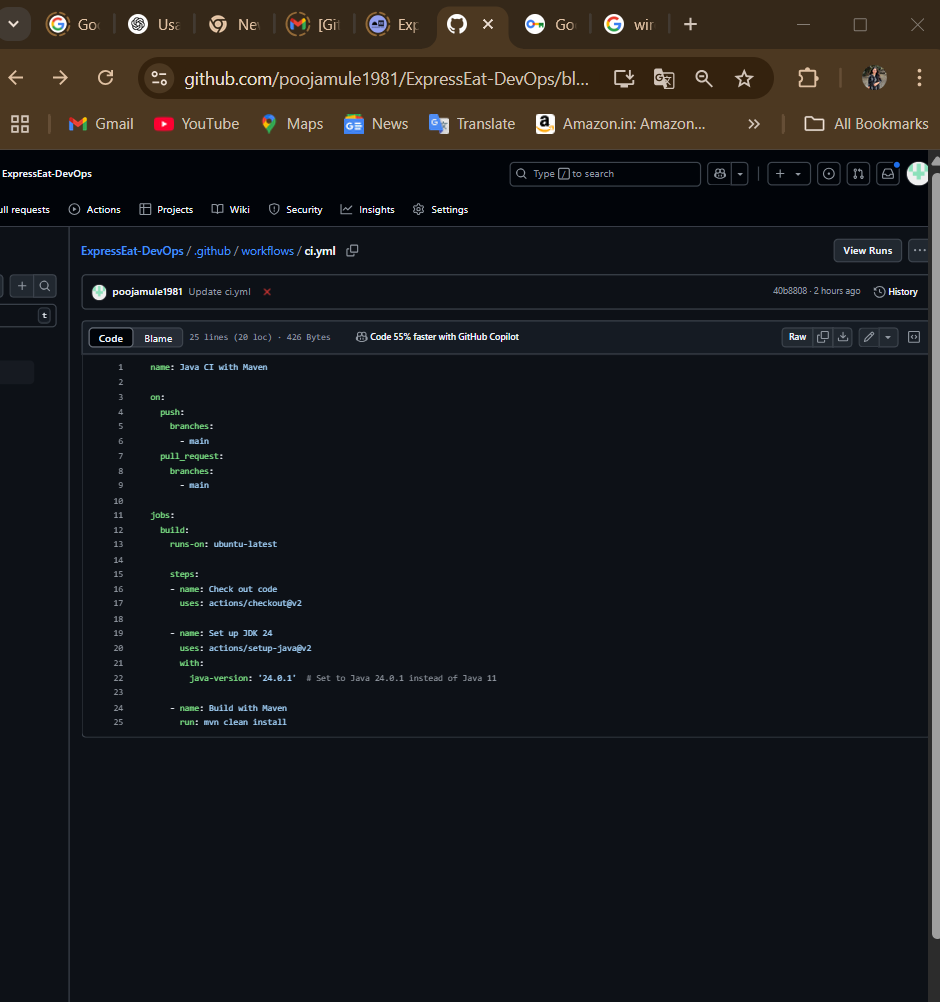












**Docker**

Docker is a containerization platform that packages an application and its dependencies into a lightweight, portable container[docker.com](https://www.docker.com/blog/docker-for-devops/#:~:text=What%20is%20Docker%3F). Docker containers ensure that Expree Eat Web runs the same way in development, testing, and production. Using Docker in CI/CD means Jenkins can build a Docker image after a successful build. Containers start quickly and use fewer resources than virtual machines[docker.com](https://www.docker.com/blog/docker-for-devops/#:~:text=Containers%20differ%20from%20virtual%20machines%2C,software%20development%20cycles%20and%20scalability).

* *Containerization:* You might include a Dockerfile in the repository. For example:

dockerfile

Copy code

FROM openjdk:11-jre

COPY target/expree-eat-web.jar /app.jar

ENTRYPOINT ["java","-jar","/app.jar"]

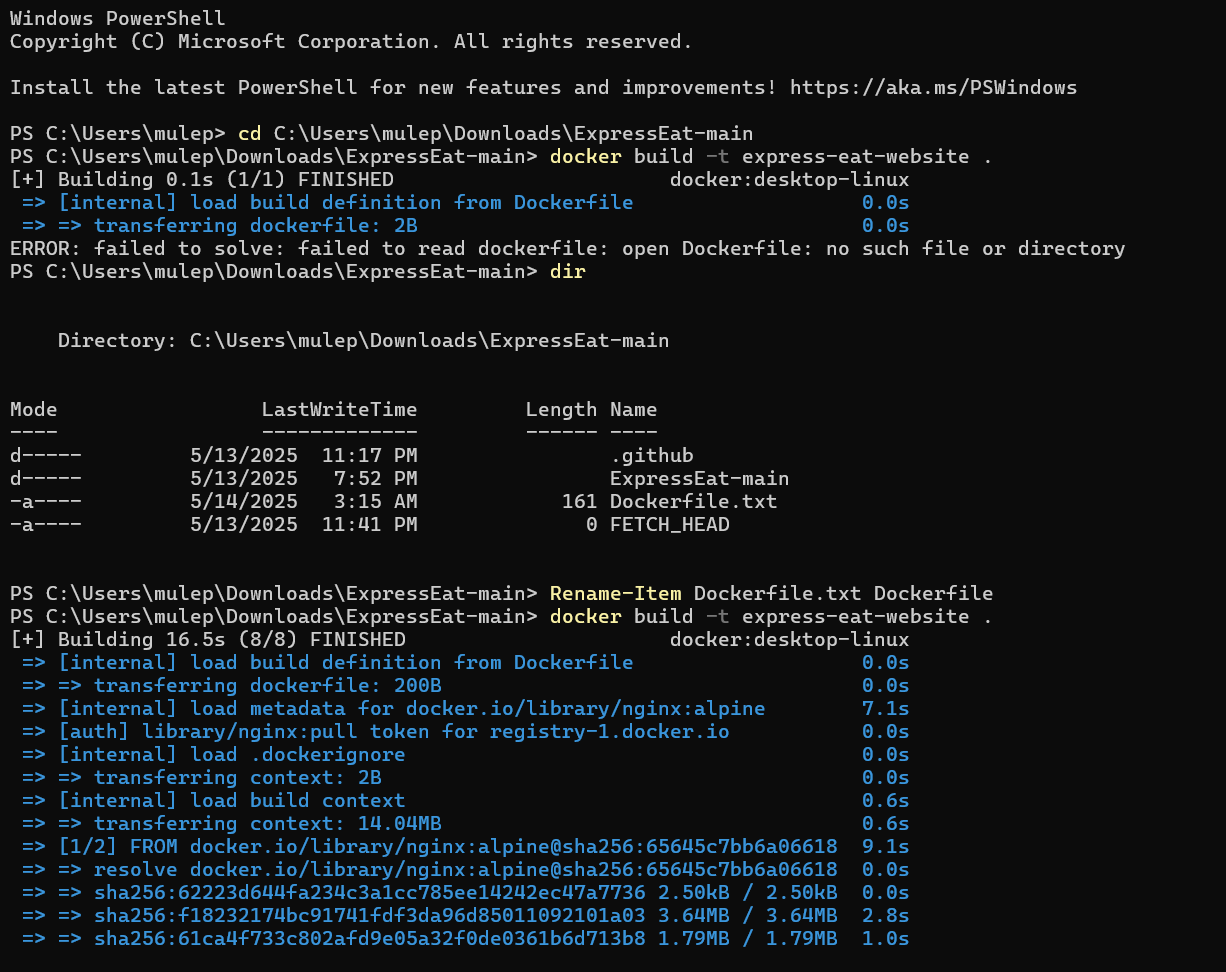
This creates an image with the compiled Java app.

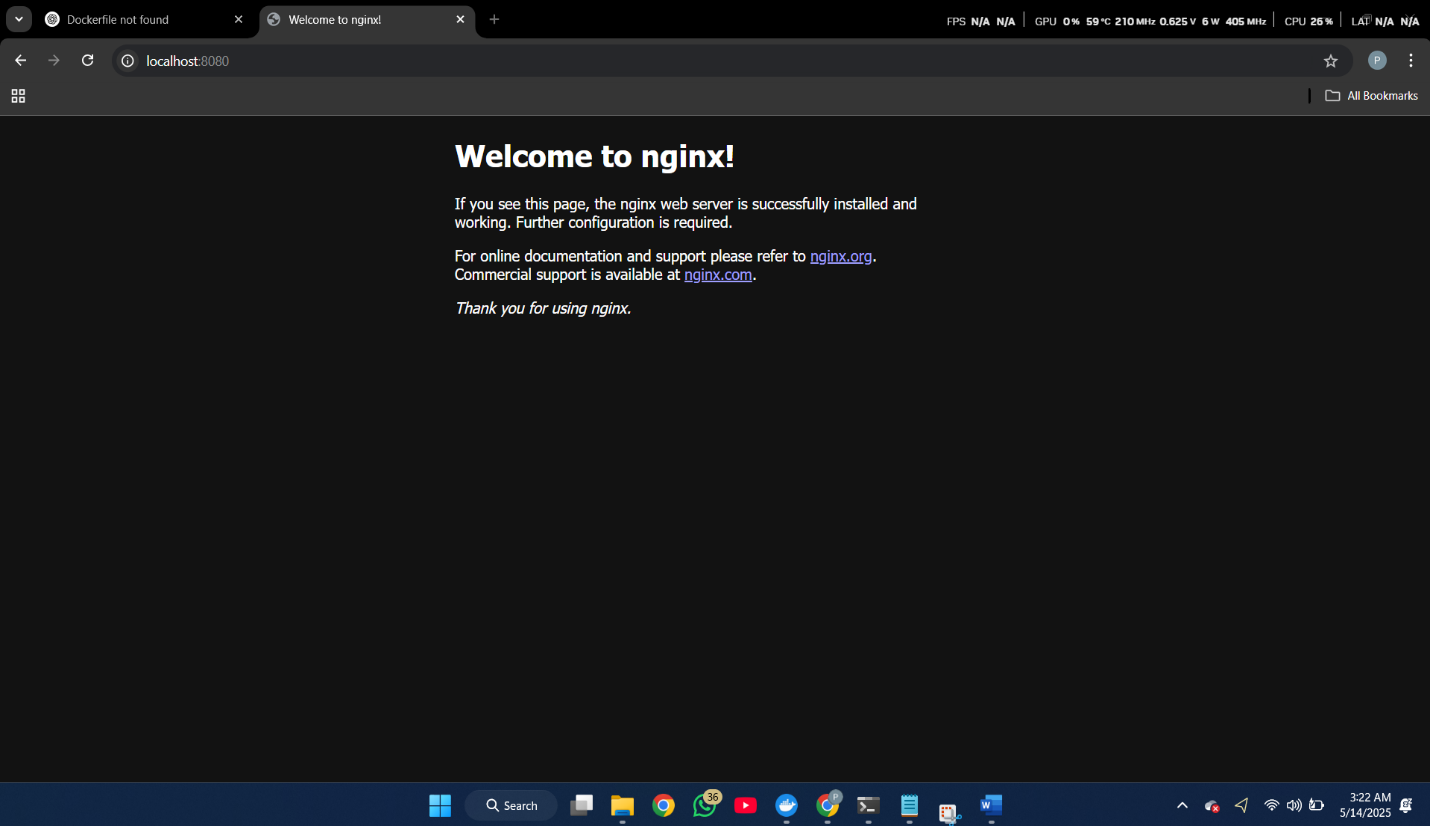
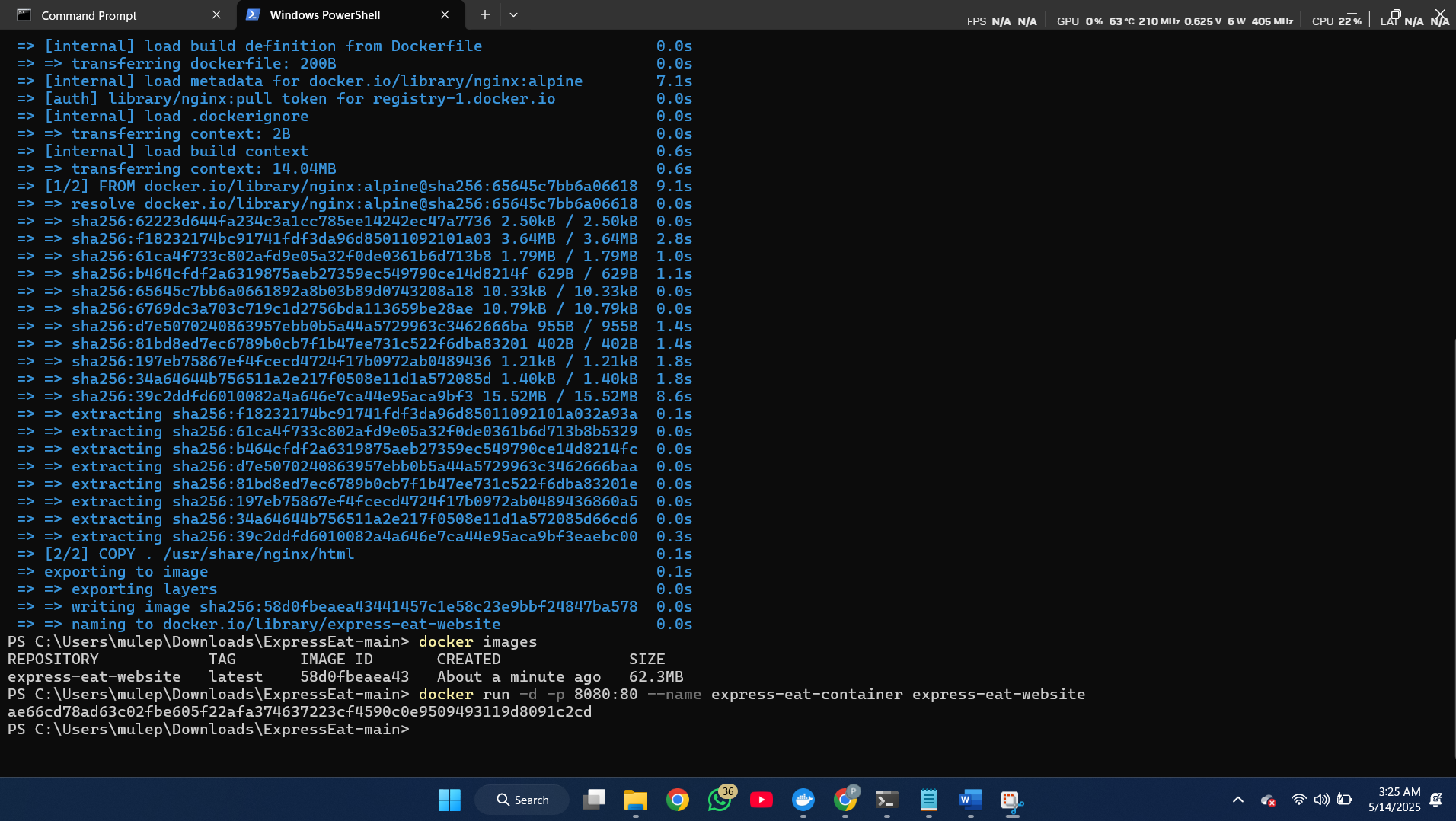
* *Benefits:* Docker images encapsulate everything needed (OS, libraries), giving consistency across environments[docker.com](https://www.docker.com/blog/docker-for-devops/#:~:text=The%20Docker%20platform%20enables%20developers,collaborative%20advantages%20over%20virtual%20machines).
* *Example use:* Jenkins can build and push the Docker image to Docker Hub or AWS ECR, ready for deployment.  
  *(Screenshot placeholder: Docker build or Docker Hub repository listing.)*

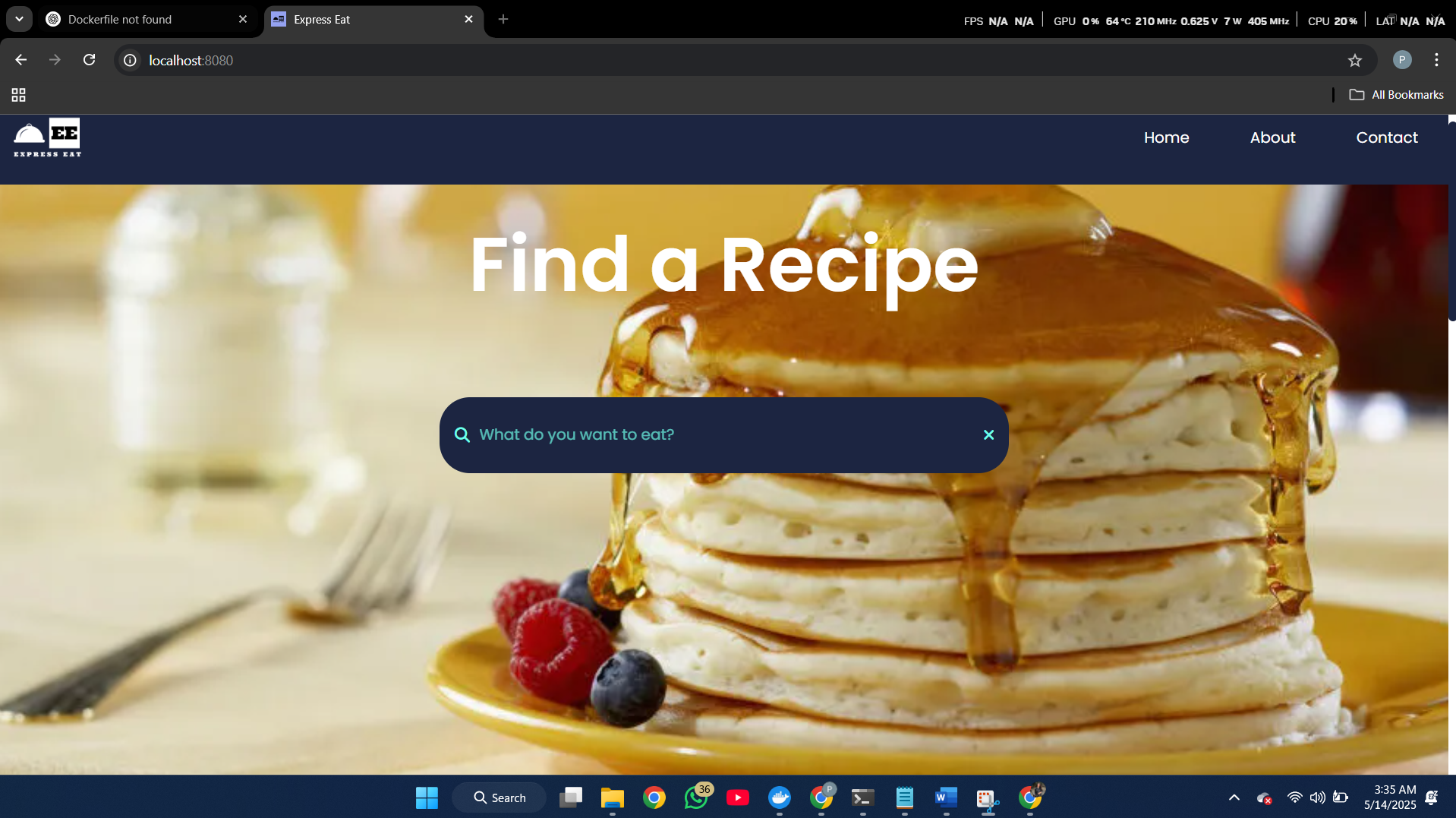
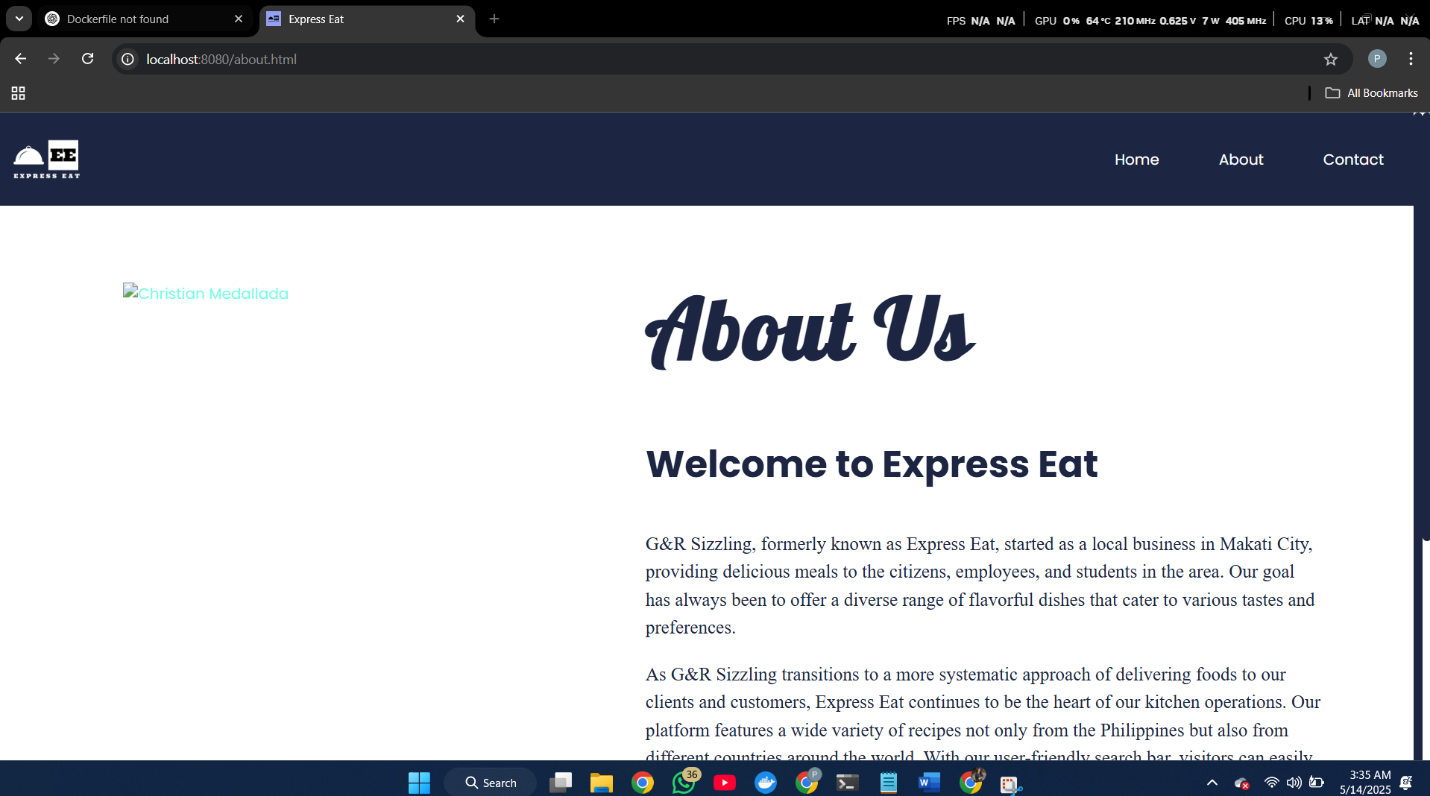
**AWS (Cloud Deployment)**

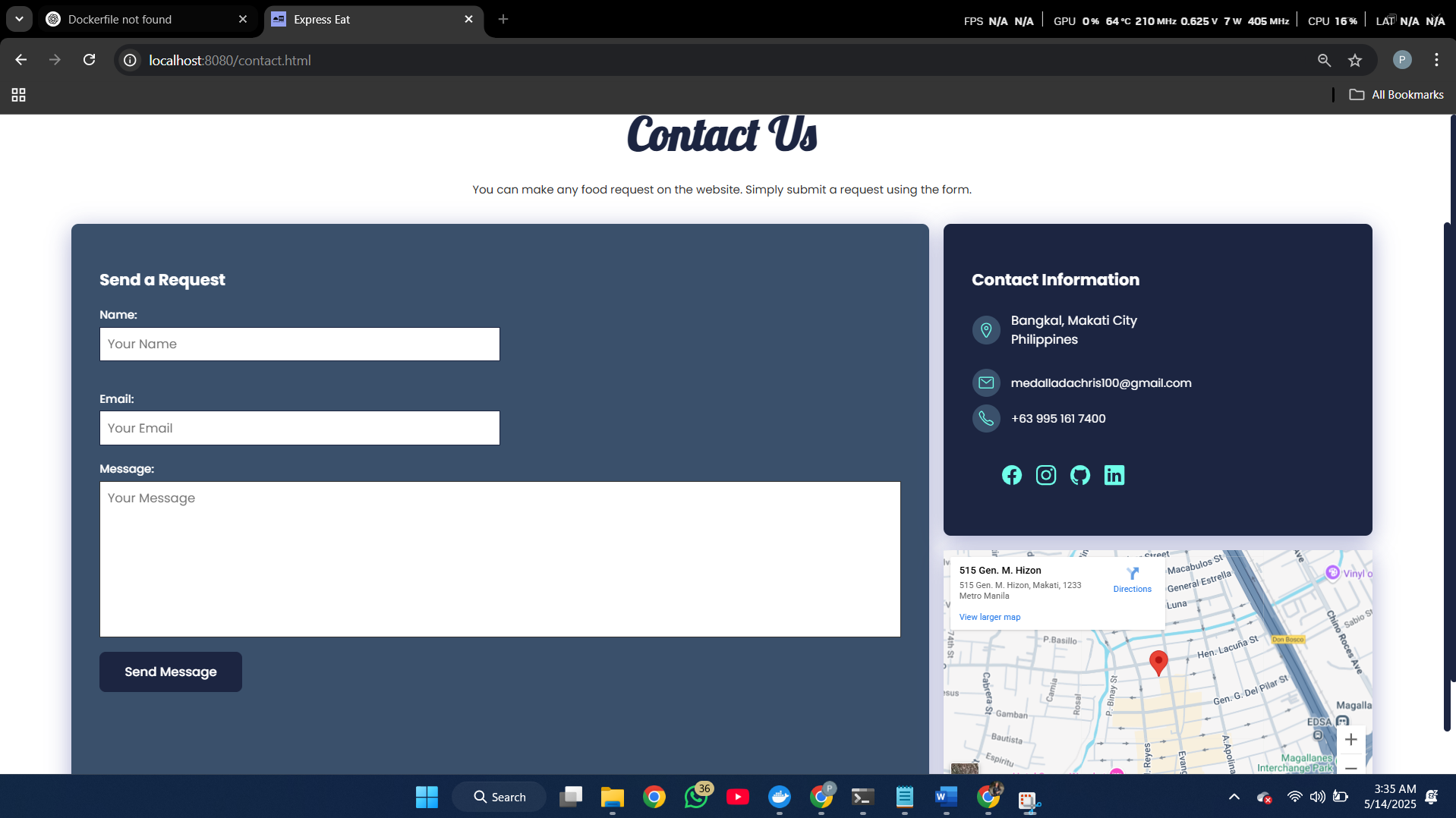
Amazon Web Services (AWS) provides managed DevOps services. For example, AWS CodePipeline orchestrates CI/CD workflows, CodeBuild compiles code, and CodeDeploy or ECS/Fargate can deploy containers to servers or clusters[geeksforgeeks.org](https://www.geeksforgeeks.org/how-to-build-a-ci-cd-pipeline-with-aws/#:~:text=,cloud%20for%20users%20to%20access). Using AWS means Expree Eat Web can be deployed on scalable infrastructure.

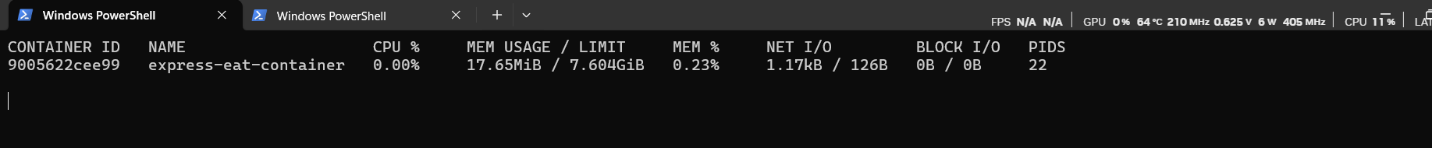
* *CI/CD Services:* AWS CodePipeline can automate the release pipeline, integrating with GitHub (source), CodeBuild (build/test), and CodeDeploy (deployment)[geeksforgeeks.org](https://www.geeksforgeeks.org/how-to-build-a-ci-cd-pipeline-with-aws/#:~:text=,cloud%20for%20users%20to%20access).
* *Infrastructure:* Components like EC2 (servers), ELB (load balancer), RDS (database), and S3 (static files) can be part of the architecture. These can be provisioned via AWS CloudFormation or Terraform (Infrastructure as Code) to keep environments consistent.
* *Example:* After a Jenkins or CodeBuild artifact is ready, AWS CodeDeploy could update EC2 instances, or an Elastic Beanstalk service could automatically deploy the new version.  
  *(Screenshot placeholder: AWS Console with a deployment or CodePipeline stages.)*











**tep-by-Step DevOps Workflow**

**1. Repository Creation and Cloning**

* **Create GitHub Repo:** In GitHub, create a new repository named ExpreeEatWeb. Add a README and .gitignore for Java.
* **Clone with GitHub Desktop:** Open GitHub Desktop, sign in to your GitHub account, then use File > Clone Repository and select the new ExpreeEatWeb repo. Choose a local directory to clone into.
* **Verify Local Git:** After cloning, GitHub Desktop shows the working copy. You can now commit, switch branches, and sync with origin visually[docs.github.com](https://docs.github.com/en/desktop/overview/getting-started-with-github-desktop#:~:text=GitHub%20Desktop%20is%20a%20free%2C,For%20more%20information%2C%20see).
* *Screenshot:* *(Insert GitHub Desktop window showing the cloned repository.)*

**2. Setting Up the Project in Eclipse**

* **Import as Maven Project:** In Eclipse, go to File > Import > Maven > Existing Maven Projects. Browse to the cloned repo directory and select the project. Eclipse will recognize pom.xml and set up the project.
* **Configure JDK and Server:** Ensure the project uses a compatible Java JDK (e.g. Java 11). If it’s a web app, configure a server runtime (Tomcat, Jetty, etc.) in Eclipse.
* **Project Structure:** Eclipse will create source folders (src/main/java, src/test/java) and include dependencies from Maven’s POM.
* *Screenshot:* *(Insert Eclipse IDE with the ExpreeEatWeb project open.)*

**3. Configuring Maven and Managing Dependencies**

* **Edit pom.xml:** Add necessary dependencies (e.g., Spring Boot Starter, database driver, TestNG) under <dependencies>. For example, to add TestNG:

xml

Copy code

<dependency>

<groupId>org.testng</groupId>

<artifactId>testng</artifactId>

<version>7.4.0</version>

<scope>test</scope>

</dependency>

* **Plugins and Packaging:** Ensure <packaging>jar</packaging> (or war for servlet containers). Add plugins if needed (e.g., Maven Compiler Plugin with source/target versions).
* **Dependency Management:** Once saved, Eclipse will download libraries to Maven Dependencies. All code editors will now resolve those imports.
* *Screenshot:* *(Insert view of pom.xml editing in Eclipse showing dependencies.)*

**4. Writing and Executing TestNG Tests**

* **Create Test Classes:** Under src/test/java, create a new TestNG class. Example: LoginTest.java with methods annotated @Test.
* **Sample Test Code:**

java

Copy code

import org.testng.Assert;

import org.testng.annotations.Test;

public class SampleTest {

@Test

public void testExample() {

// Simulate a test case

Assert.assertTrue(1 + 1 == 2);

}

}

* **Running Tests:** In Eclipse, right-click the test file and choose “Run As > TestNG Test”. Alternatively, run mvn test in the command line to execute all tests. Green checkmarks indicate passing tests.
* *Screenshot:* *(Insert screenshot of TestNG results or console output.)*

**5. Pushing Code to GitHub**

* **Commit Changes:** After initial setup (project code, tests, pom.xml), use GitHub Desktop or command line to git add and git commit your changes with a message like “Initial project setup”.
* **Push to Remote:** In GitHub Desktop, click Push origin to send commits to GitHub. Confirm on the GitHub website that the files appear in the repository.
* **Collaboration:** Developers can now clone or branch from this GitHub repo to start new features. Pull requests can be used for code review.
* *Screenshot:* *(Insert GitHub Desktop showing commit history and push status.)*

**6. CI/CD Pipeline and Deployment**

* **Set Up Jenkins Job:** On the Jenkins server, create a new Pipeline job for ExpreeEatWeb. Connect it to the GitHub repo (via Git SCM or webhook).
* **Pipeline Script:** Use the Jenkinsfile in the repo (as shown earlier) or configure stages manually. Each push to the main branch will trigger Jenkins.
* **Build Stage:** Jenkins runs mvn clean package to compile and package the app. If using Docker, it then builds the image (docker build) and pushes it to a registry.
* **Test Stage:** Jenkins runs mvn test. All TestNG tests must pass; Jenkins will mark the build as failed if any tests fail[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=Jenkins%20is%20an%20open,with%20over%20100%2C000%20installations%20worldwide).
* **Deploy Stage:** After a successful build, Jenkins deploys the app. This could mean:
  + Deploy to AWS (e.g. update an AWS Elastic Beanstalk environment or ECS service).
  + Or start Docker containers on a server.
  + *Example:* Using AWS CLI to push a new Docker image and update ECS:

sh

Copy code

aws ecr get-login-password | docker login --username AWS --password-stdin <account>.dkr.ecr.<region>.amazonaws.com

docker push <account>.dkr.ecr.<region>.amazonaws.com/expree-eat-web:latest

aws ecs update-service --cluster ExpreeCluster --service ExpreeService --force-new-deployment

* **Monitoring:** After deployment, Jenkins can send notifications (email/Slack) on success/failure. Logging and monitoring (CloudWatch, ELK, etc.) can be set up for runtime metrics.
* *Screenshot:* *(Insert Jenkins build console or AWS deployment activity.)*

**Architecture & Workflow Diagrams**

*(Insert diagrams illustrating the system and DevOps pipeline here.)*

* **System Architecture (Figure 1):** The Expree Eat Web app could consist of a front-end web application (e.g. Spring Boot service) behind an Elastic Load Balancer, connecting to a database (e.g. Amazon RDS). Static assets may be served from S3/CDN. All components would reside in a VPC with subnets and security groups as shown in a detailed architecture diagram.
* **DevOps Pipeline (Figure 2):** A CI/CD flow chart would show code being pushed to GitHub, triggering Jenkins or AWS CodePipeline. It would include stages: *Build → Test → Artifact Repository → Staging Deployment → Production Deployment*, with continuous feedback loops.

*(Screenshot placeholders: architecture diagram, CI/CD workflow diagram to be added.)*

**Summary of DevOps and CI/CD Benefits**

* **Automation:** Automated build/test/deploy pipelines reduce manual errors and ensure consistent deployments[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=Jenkins%20is%20an%20open,with%20over%20100%2C000%20installations%20worldwide)[github.com](https://github.com/resources/articles/devops/ci-cd#:~:text=,versus%20waiting%20for%20one%20release). Once set up, Jenkins and Docker handle repetitive tasks automatically.
* **Collaboration:** Version control (Git/GitHub) and CI enable multiple developers to work together easily. Frequent integration of small changes catches conflicts early[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=Jenkins%20is%20an%20open,with%20over%20100%2C000%20installations%20worldwide)[github.com](https://github.com/resources/articles/devops/ci-cd#:~:text=,Learn%20more%20about%20version%20control). Tools like GitHub pull requests and Jenkins build reports improve code quality.
* **Faster Delivery:** CI/CD shortens the release cycle. The State of DevOps Report notes teams with mature CI/CD deploy ~200× more often with lead time 106× faster than peers[github.com](https://github.com/resources/articles/devops/ci-cd#:~:text=The%20short%20answer%3A%20Speed,delivery%20pipeline%20enables%20much%20more).
* **Quality and Feedback:** Continuous testing (TestNG/Jenkins) means bugs are caught early. Automated test suites run on every commit, keeping the codebase stable[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=CI%2FCD%20means%20making%20small%20code,be%20tested%20and%20released%20faster)[github.com](https://github.com/resources/articles/devops/ci-cd#:~:text=,versus%20waiting%20for%20one%20release).
* **Scalability:** Using containers (Docker) and cloud infrastructure (AWS) allows quick scaling. Infrastructure-as-Code (CloudFormation/Terraform) ensures reproducibility.
* **Innovation Focus:** With routine work automated, developers can focus on new features and improvements instead of manual steps[github.com](https://github.com/resources/articles/devops/ci-cd#:~:text=,versus%20waiting%20for%20one%20release).

**Sources:** Official documentation and tutorials for GitHub/GitHub Desktop[docs.github.com](https://docs.github.com/en/desktop/overview/getting-started-with-github-desktop#:~:text=GitHub%20Desktop%20is%20a%20free%2C,For%20more%20information%2C%20see)[datanovia.com](https://www.datanovia.com/learn/programming/tools-and-ides/version-control-with-git-and-github.html#:~:text=GitHub%20is%20a%20web,especially%20when%20working%20with%20teams), Maven[dev.to](https://dev.to/imsushant12/simplify-your-builds-a-quick-maven-guide-with-examples-2bp9#:~:text=Maven%20is%20a%20widely%20used,compilation%2C%20testing%2C%20packaging%2C%20and%20deployment)[geeksforgeeks.org](https://www.geeksforgeeks.org/maven-build-automation/#:~:text=Maven%20is%20a%20powerful%20build,dependency%20management%2C%20and%20build%20execution), Eclipse[codejava.net](https://www.codejava.net/ides/eclipse/what-is-eclipse-ide-for-beginner#:~:text=Perhaps%2C%20Eclipse%20is%C2%A0%20the%20most,about%20this%20programmer%E2%80%99s%20favorite%20tool), TestNG[geeksforgeeks.org](https://www.geeksforgeeks.org/testng-tutorial/#:~:text=TestNG%20,with%20different%20sets%20of%20data), Jenkins[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=Jenkins%20is%20an%20open,with%20over%20100%2C000%20installations%20worldwide)[geeksforgeeks.org](https://www.geeksforgeeks.org/understanding-jenkins-ci-cd-pipeline-and-its-stages/#:~:text=At%20its%20core%2C%20Jenkins%20provides,all%20types%20of%20development%20processes), Docker[docker.com](https://www.docker.com/blog/docker-for-devops/#:~:text=What%20is%20Docker%3F)[docker.com](https://www.docker.com/blog/docker-for-devops/#:~:text=Containers%20differ%20from%20virtual%20machines%2C,software%20development%20cycles%20and%20scalability), and AWS CI/CD[geeksforgeeks.org](https://www.geeksforgeeks.org/how-to-build-a-ci-cd-pipeline-with-aws/#:~:text=,cloud%20for%20users%20to%20access). These sources explain the tools’ roles in DevOps and CI/CD.