# CI/CD PIPELINE FOR SPRING PROJECTS WITH JENKINS AND DOCKER HUB

Name: Atharva Kothawade Date: 01/08/2024

2

# PRESENTATION OVERVIEW

* Introduction to Spring Projects
* Overview of CI/CD
* Introduction to Jenkins
* Setting Up Jenkins for Spring Projects
* Building and Testing the Spring Project
* Introduction to Docker
* Dockerizing the Spring Application
* Pushing Docker Images to Docker Hub

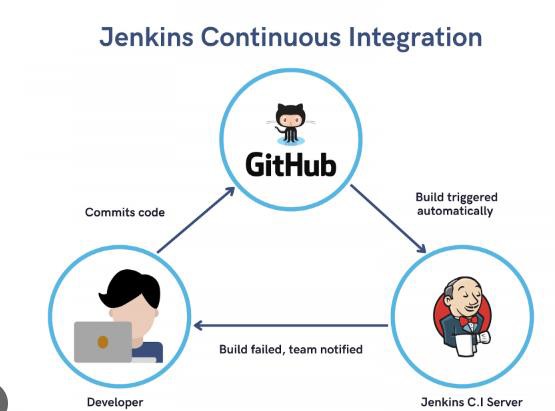
# INTRODUCTION TO SPRING PROJECTS 3



* A powerful framework for building Java-based enterprise applications.
* Provides support for dependency injection, transaction management, and more.
* Simplifies Spring application development
* with auto-configuration and
* embedded servers.

## Typical Spring Project Structure:

* + Main application class
  + Controllers, Services, Repositories
  + Configuration files



# OVERVIEW OF CI/CD JENKINS

4

* **Jenkins** is a popular tool used for automating the software development process. It helps manage and automate tasks like building, testing, and deploying code. Think of it as a robot that helps developers by handling repetitive tasks and making sure code changes are tested and delivered smoothly.
* Continuous Integration (CI): Developers frequently add their code to a shared repository. Automated tests are run to make sure the new code doesn’t break anything. This helps catch problems early.
* Continuous Delivery (CD): After code is integrated and tested, it’s automatically prepared for release. This means it can be deployed to users quickly and reliably.

# SETTING UP JENKINS FOR SPRING 5

PROJECTS

* **Title:** Configuring Jenkins for Spring CI/CD **Content:**

### Installation:

* + Download Jenkins from the official site and follow the installation instructions for your OS.
  + Set up Jenkins on your server or local machine.

### Creating a Jenkins Job:

* + **Freestyle Project:** For simple configurations, where you can specify build steps, test results, and deployment.
  + **Pipeline Project:** For complex workflows, using a Jenkinsfile to define the CI/CD pipeline in code.

### Configuration:

* + **Source Code Repository:** Connect Jenkins to your Git repository.
  + **Build Triggers:** Set up triggers like commits, pull requests, or scheduled builds.

6

# BUILDING AND TESTING THE SPRING PROJECT

**Build Configuration:** (Building with Jenkins)

Use Maven or Gradle to compile and package your Spring application. Configure build steps in Jenkins:

For Maven: mvn clean install For Gradle: gradle build

**Testing:**

Unit Tests: Ensure individual components work correctly.

Integration Tests: Verify that components work together as expected.

Configure Jenkins to run tests and report results, using plugins to display test results and code coverage

7



# INTRODUCTION TO DOCKER

Docker lets you build, test, and deploy applications quickly

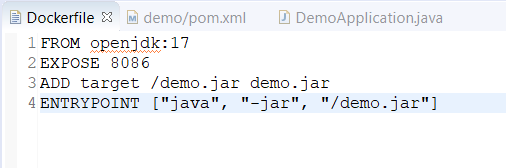
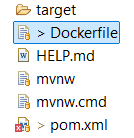
Using Docker, you can quickly deploy and scale applications into any environment and know your code will run. Running Docker on AWS provides developers and admins a highly reliable, low-cost way to build, ship, and run distributed applications at any scale.

## Dockerfile:

8

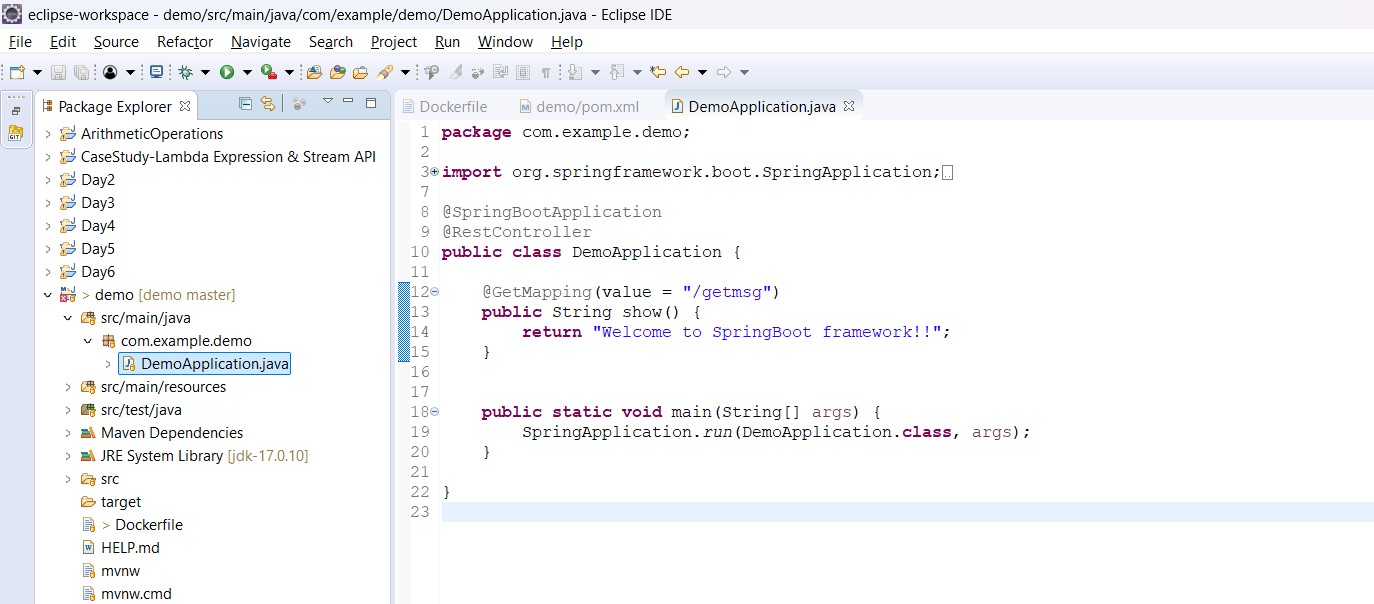
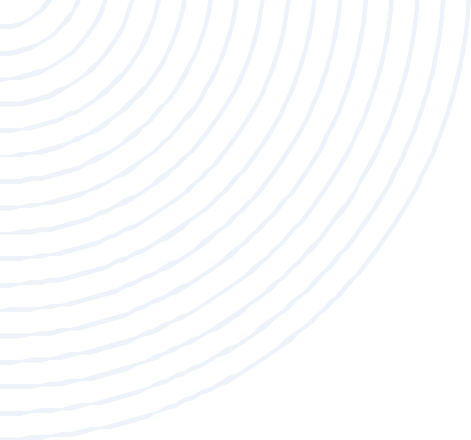
# DOCKERIZING THE SPRING APPLICATION

Base Image: Start with an official Java base image (e.g., openjdk). Add Application: Copy your Spring Boot JAR file into the image. Expose Port: Define the port the application will run on.

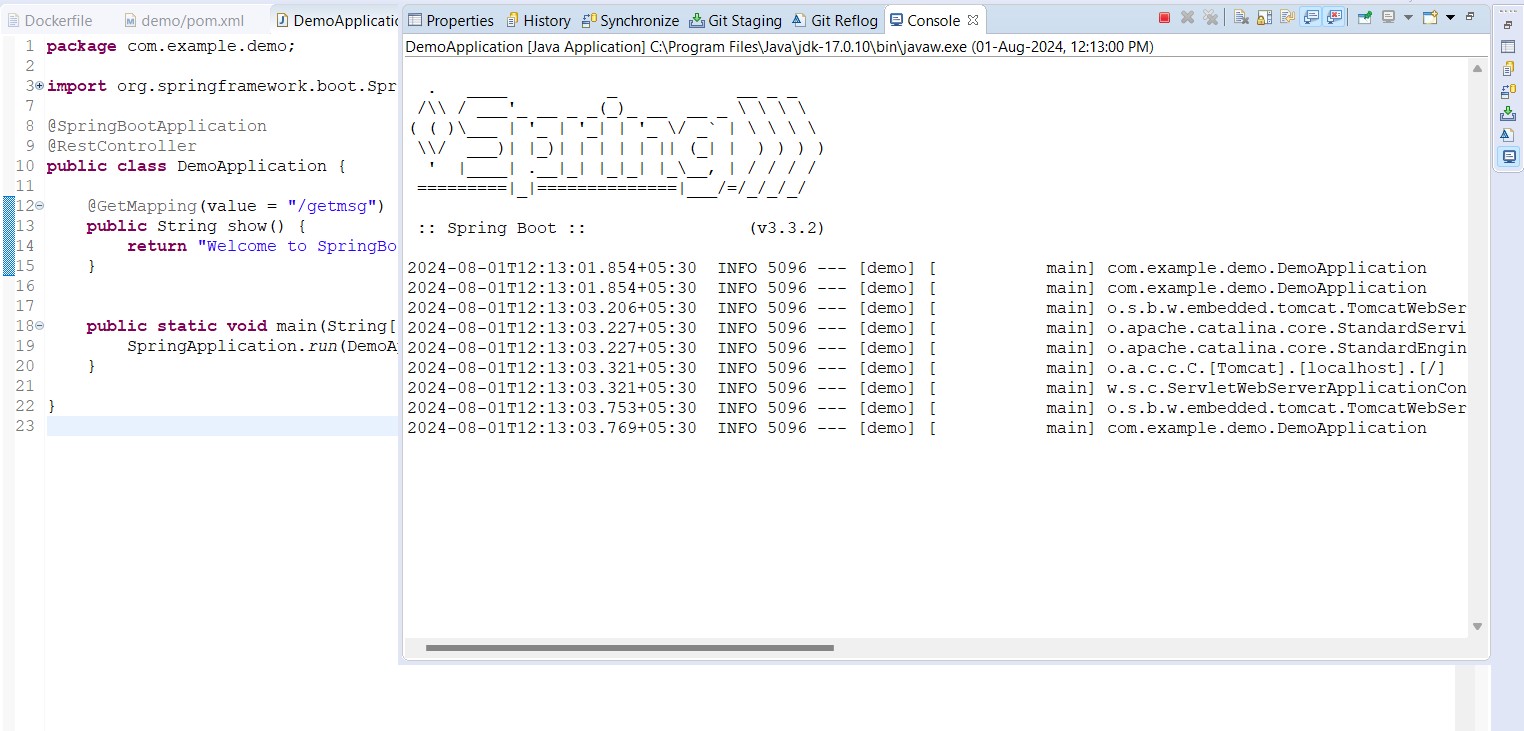


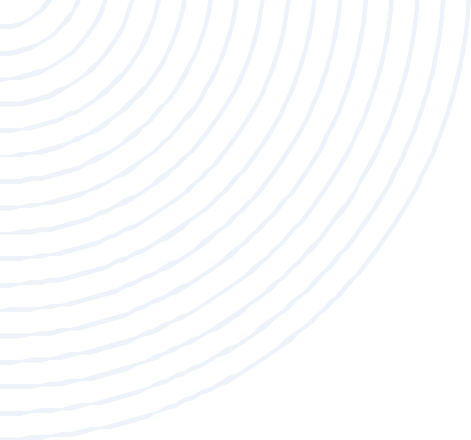
Command to Run: Specify the command to run your Spring Boot application

PROJECT IMAGES

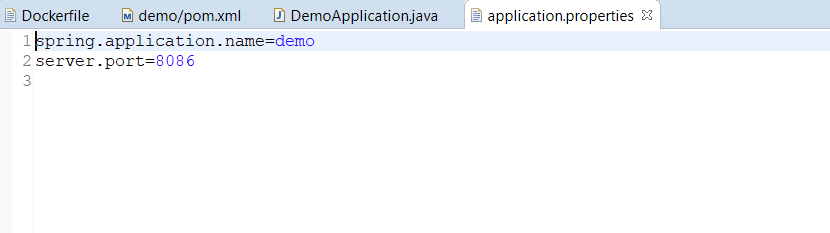
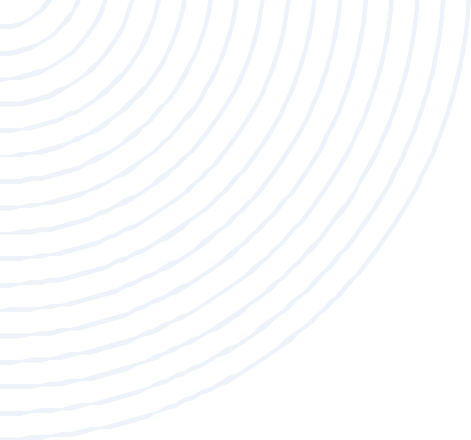


9

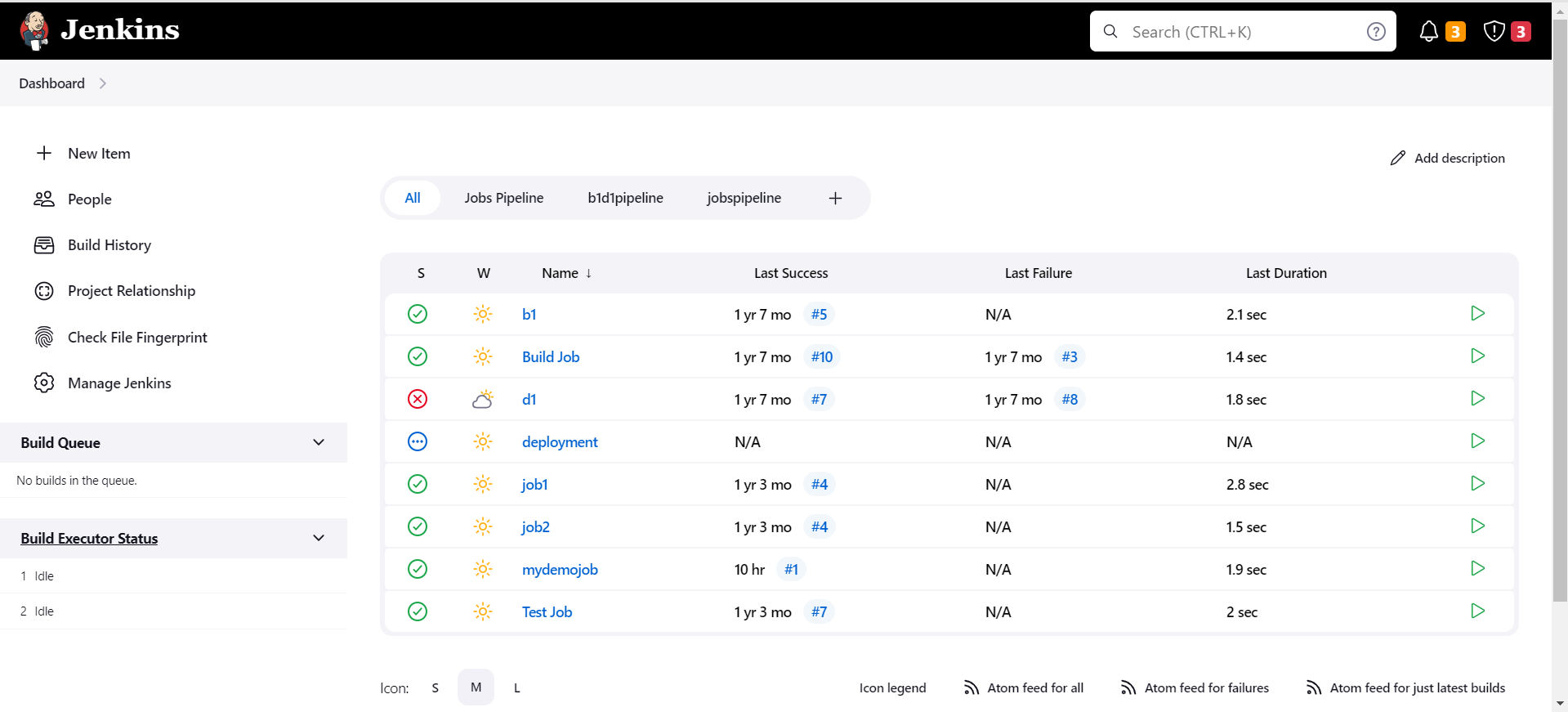
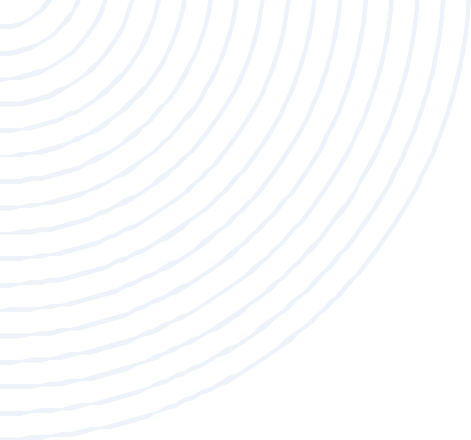


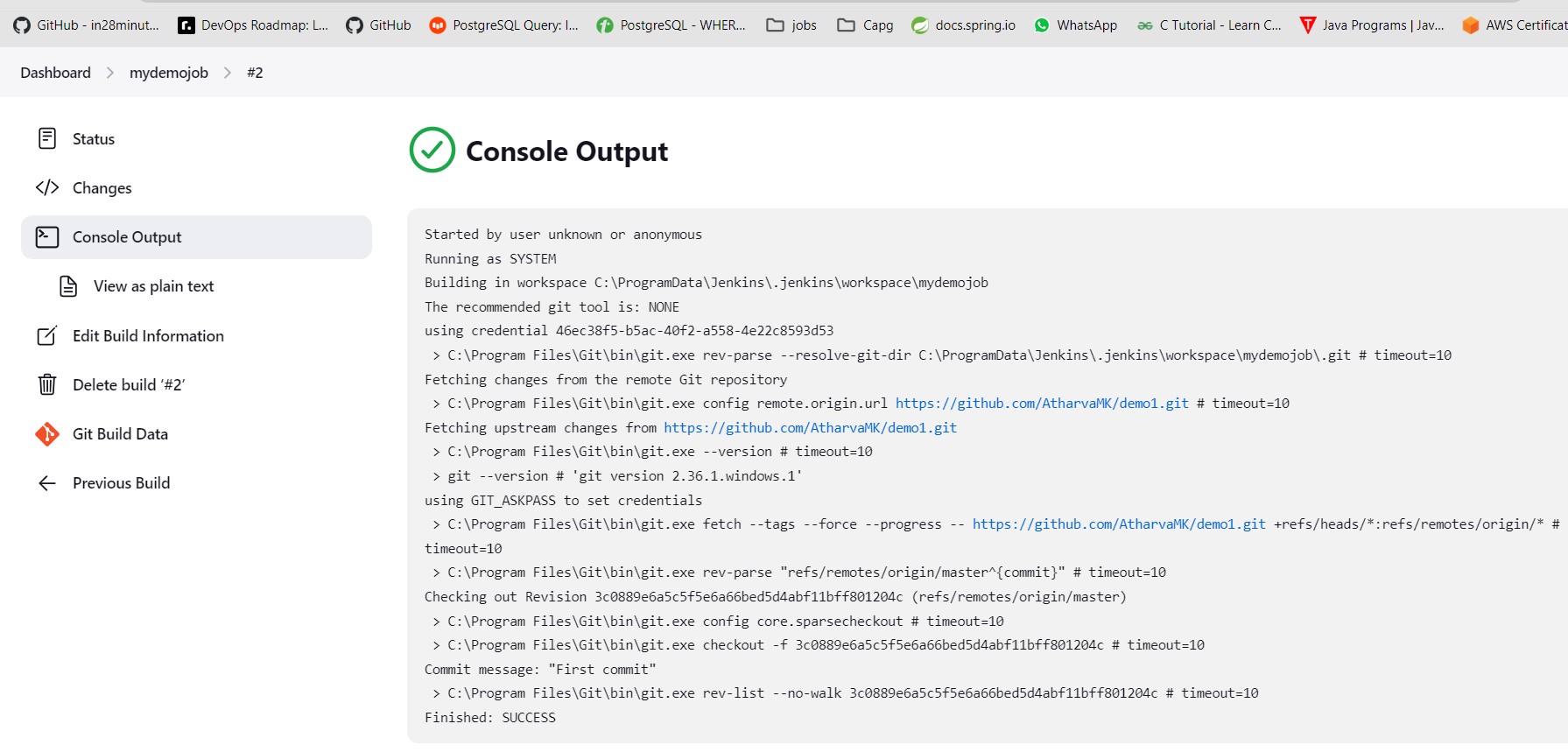
10

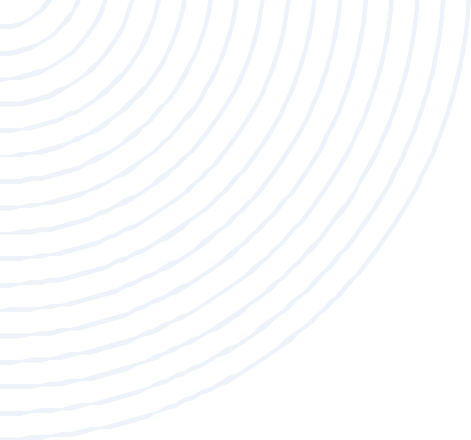
11



12

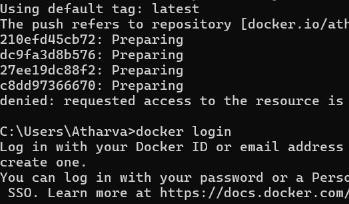
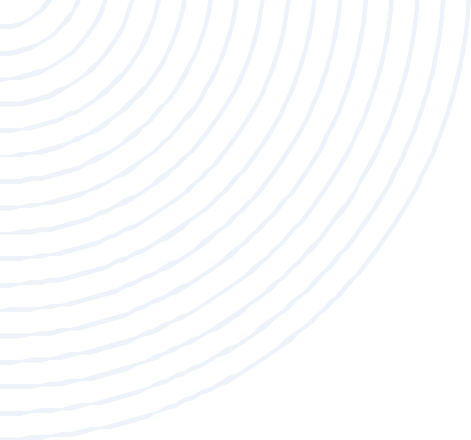


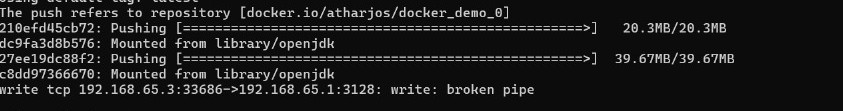


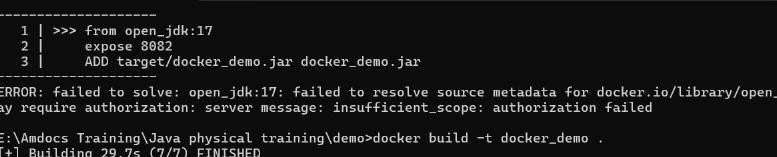
13

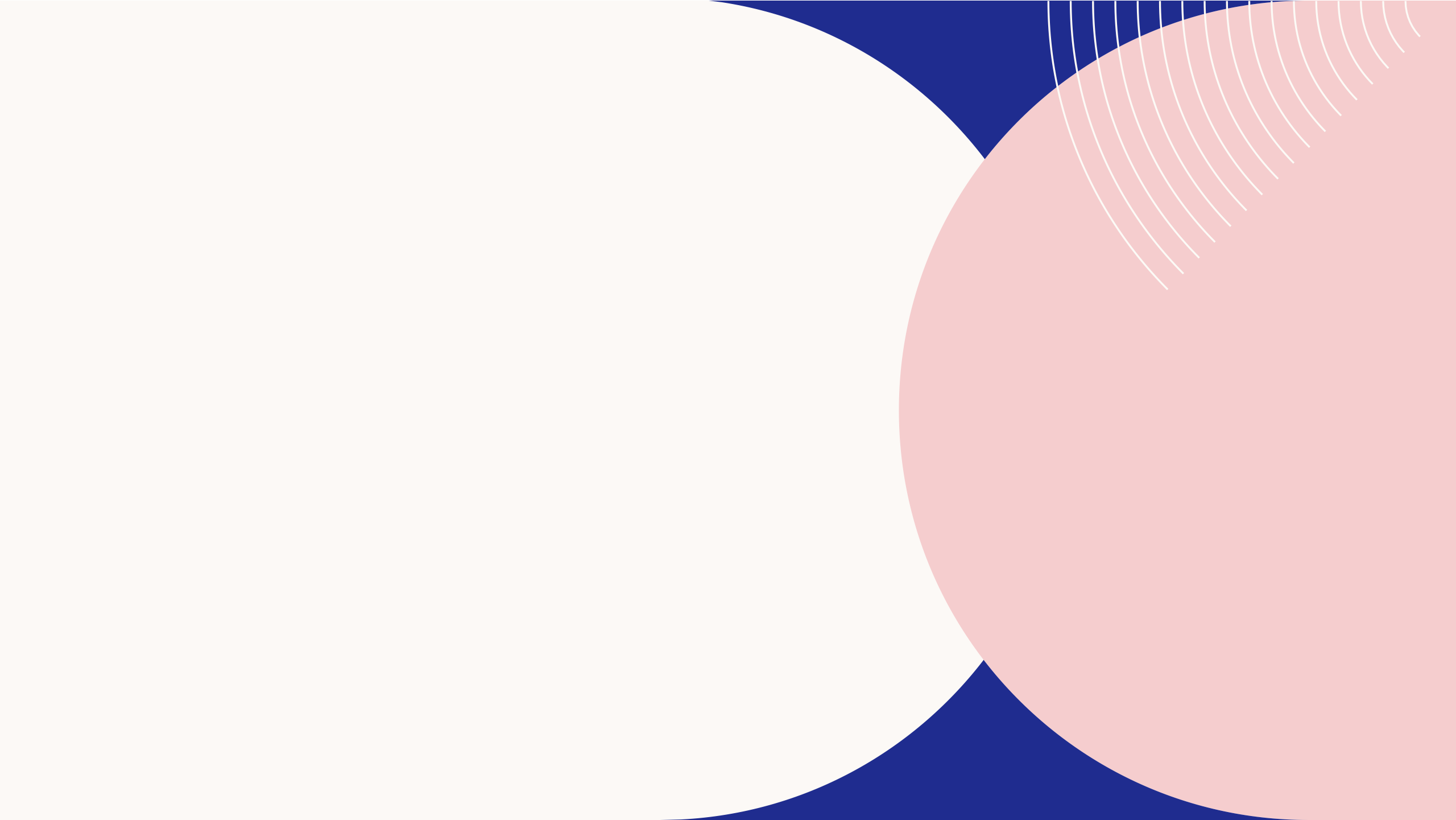


14









THANK YOU