INTRODUCTION

About Me

Maksym Dovbeshko

01

Served as first Ukrainian volunteer in Greenland and Hong Kong.

02

Climbed Mont Blanc and several other summits.

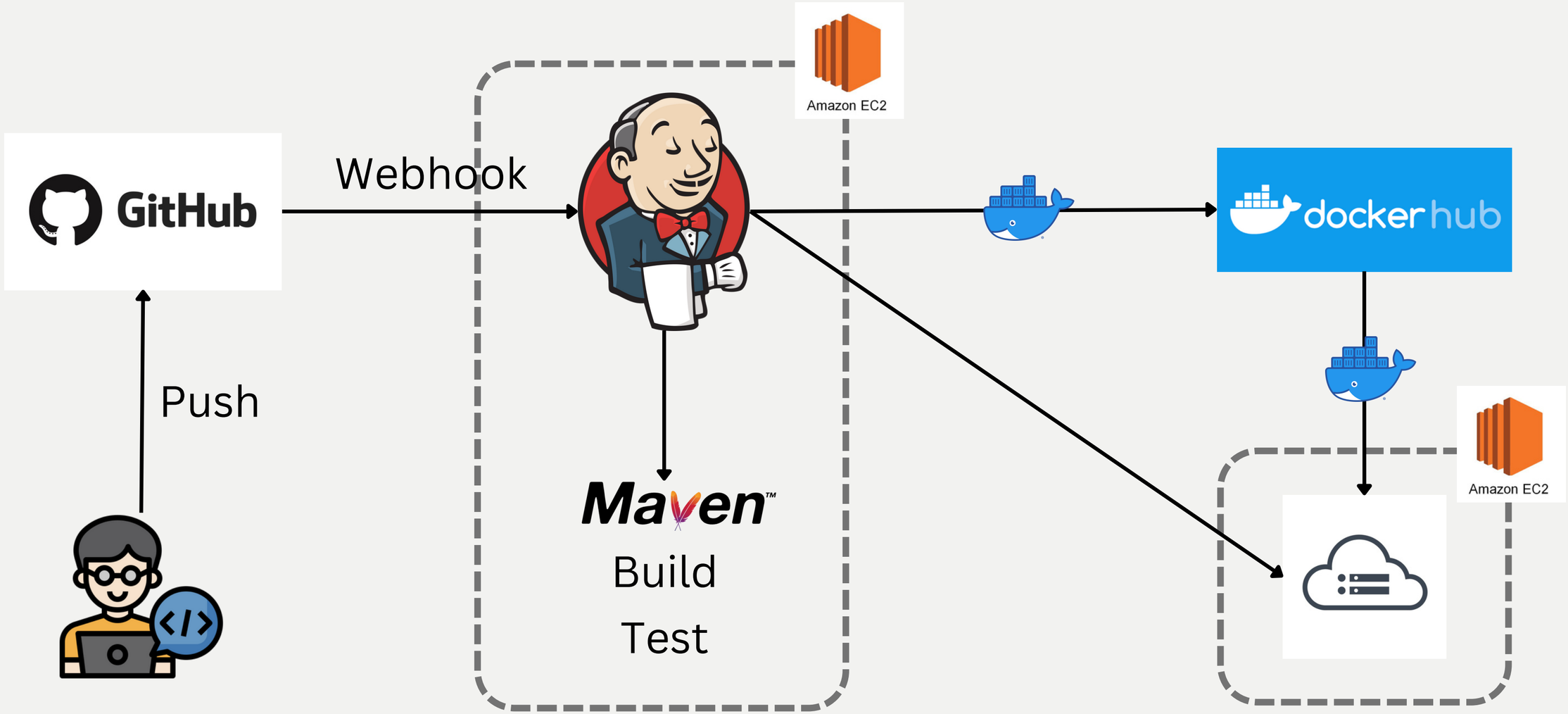
03

Was a co-organizer of a music festival in Belgium.

I am a highly motivated and experienced professional with a diverse background in construction project management, retail and wholesale trade management, and digital marketing. My leadership skills, project management experience, and the ability for optimizing processes have enabled me to improve efficiency and productivity wherever I go. Recently, I completed training in DevOps essentials and IT fundamentals, and I am now excited to transition into a junior DevOps role at EPAM!

CI/CD Pipeline Using Git, GitHub, Docker, Maven and Jenkins

PROJECT



PROJECT

Goal

The goal of my CI/CD Pipeline is to automate the software development lifecycle and ensure faster, more efficient, and higher quality delivery of the Java application.

This is achieved through a series of automated stages that include receiving the application code from GitHub, building and testing it, creating a Docker image, publishing the image to Docker Hub, and deploying the image to the deployment server.

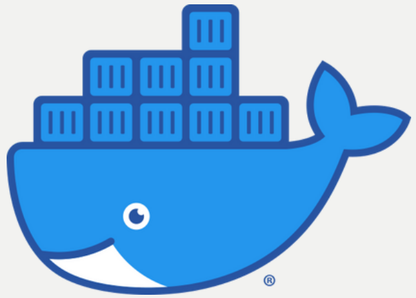
The ultimate goal of this process is to make the application development and deployment process more streamlined, reduce errors and ensure that the application can be deployed quickly, reliably and with minimal manual intervention.

Key Benefits

01

Collaborative development:

- Enables developers to work collaboratively on a codebase while ensuring seamless and continuous integration of changes.
- Reduces time and effort required to resolve integration conflicts.
- Promotes faster release cycles.



02

Consistent and portable development environment with Docker:

- Allows for the creation of a consistent and portable development environment.
- Can be replicated across different stages of the pipeline (development, testing, production).
- Ensures that code is tested in an environment as close as possible to production, reducing runtime issues.

03

Automation with Jenkins:

- Orchestrates the pipeline and automates tasks such as code building, testing, and deployment.
- Reduces manual effort required.
- Promotes consistency and reproducibility across different stages of the pipeline.

PROJECT

Key Benefits

04

Scalable and secure infrastructure with AWS:

- Enables the scaling of resources and deployment of the application.
- Provides secure and reliable infrastructure for the pipeline.
- Ensures application is available to end-users with minimal downtime.

05

Java app building and testing with Maven:

- Uses Maven to build and test the Java app.
- Promotes consistency in building and testing Java apps within the pipeline.

PROJECT

In Work

```
24    </dependencies>
25    <build>
26      <finalName>EPAM_DevOps_Final_Project</finalName>
27      <plugins>
28        <plugin>
29          <groupId>org.apache.maven.plugins</groupId>
30          <artifactId>maven-dependency-plugin</artifactId>
31          <version>2.3</version>
32          <executions>
33            <execution>
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
└ git push -u origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 316 bytes | 316.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/maximdove/java-groovy-docker.git
   2f27ff2..6dcca53  master -> master
branch 'master' set up to track 'origin/master'.
```

~/java-groovy-docker master

Jenkins

Search (⌘+K) | 1 | Maksym Dovbeshko | log out

Dashboard > maximdove_pipeline2 >

- Status
- Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- GitHub
- Rename
- Pipeline Syntax
- GitHub Hook Log

Build History trend

Pipeline maximdove_pipeline2

Add description

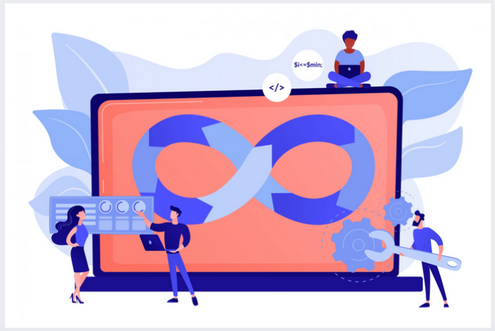
Disable Project

Stage View

	SCM Checkout	Build	Test	Build Docker Image	Publish Docker Image	Run Docker Image
Average stage times: (Average full run time: ~37s)	464ms	9s	13s	2s	3s	5s
#11 февр. 13 20:52 — 2 commits	399ms	8s	13s	2s	3s	5s
#10 февр. 13 20:21 — No Changes	335ms	8s	13s	2s	3s	10s

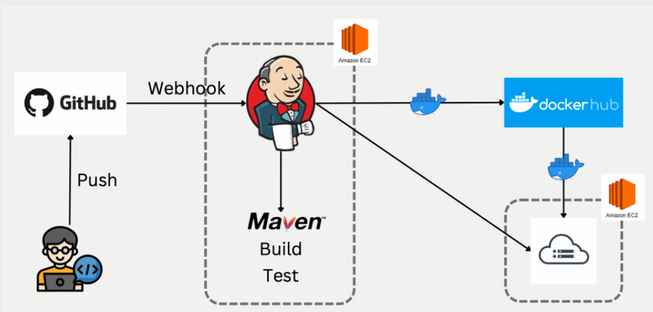
Небезопасно — 52.86.100.77:8082/EPAM_DevOps_Final_Project/#

CI/CD Pipeline Using Git, GitHub, Docker, Maven and Jenkins



Небезопасно — 52.86.100.77:8082/EPAM_DevOps_Final_Project/#

CI/CD Pipeline Using Git, GitHub, Docker, Maven and Jenkins



PROJECT

Summary

In summary, my CI/CD Pipeline is an effective tool for automating the software development process and ensuring faster and more efficient delivery of my Java application.

However, there is always room for improvement. One improvement I could make is to use Jenkins nodes on separate AWS instances for testing, which could be provisioned using Ansible. This would enable me to run multiple tests concurrently, improving the speed of the pipeline, reducing the time it takes to build and test the application, and making the testing process more scalable, efficient, and cost-effective.

By continuously improving my CI/CD Pipeline, I can ensure that it remains effective and relevant in the ever-changing landscape of software development.

Thank You

Thank you for your time and attention. I hope my CI/CD Pipeline presentation was informative and valuable. I remain committed to continuous improvement and innovation in software development.