CI/CD for a Todo App

Semester Project Specification

Course: Continuous Integration (S4-CONINT)

Context and Motivation

You are working for an Austrian company that has an own product. The product is a Todo Application and consists at this moment of two services: a frontend which is a web application based on Vue.js and a backend which is a Node.js application based on express.

The goal is to introduce continuous integration in your company and use DevOps Principles to continuously deliver and deploy your application. In a best-case scenario multiple times a day.

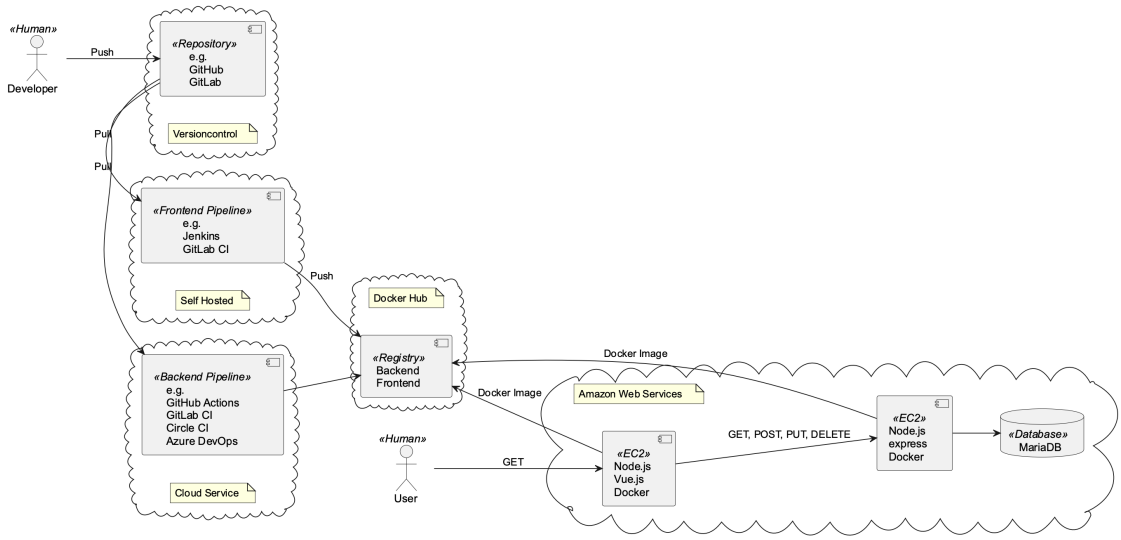
Your team has decided to use a cloud service and a self-hosted solution for a CI Server, to test and analyze both options, so that you can make a informed decision about which service you want to use.

Technical Specifications

The project should use the following technologies:

- Amazon Web Services
- RDS – for MariaDB
- EC2 with Docker Compose for the
  - Frontend
  - Backend
- Docker Hub
  - Docker Registry
- CI Server (one for each service)
  - One cloud Service (e.g., GitHub Actions, GitLab CI, Circle CI, Azure DevOps)
  - One self-hosted Service (e.g., Jenkins, GitLab CI)
- Code Quality Server (Self-Hosted)
  - SonarQube
- Feature Toggle - A/B Test Server
  - PostHog
- Each Pipeline needs to consist of at least 5 stages:
  - Linting Stage
    - Perform static code analysis with self-hosted SonarQube.
    - Perform security scan with Snyk
  - Testing Stage
    - Run your test suite

- Build Stage
  - Build your Docker image.
- Deliver Stage
  - Push your Docker image to Docker Hub
- Deploy Stage
  - Deploy your application on AWS.
  - Implement Blue/Green Deployment
- A failed stage will cause the stop of the pipeline and will inform the developers.
  - E.g., slack, email, ... (you choose the channel)
- A push to the "main" branch should run the first three stages.
  - The Docker Images are delivered to the Docker Hub registry.
- A push to the branch "deploy/production" runs all stages
  - The Todo Application is publicly available in the browser.



Evaluation

- You must use the projects provided by the Moodle course (frontend, backend). If you do not use this projects that would result in an immediate negative grade.
- Setup of two CI/CD Pipelines (one for the frontend, one for the backend)
  - 1 self-hosted in AWS (Jenkins, Gitlab CI)
  - 1 cloud-hosted (GitHub Actions, GitLab Ci, Circle CI, Azure DevOps)
- Setup of SonarQube in AWS

- Setup Snyk for Security Scans
- Setup PostHog for Feature Toggles and A/B Tests
- 5 Stages per pipeline
- Linting
  - Perform Static Code Analysis per Project (SonarQube, Snyk)
- Testing
  - Run at least 10 useful and real Tests per Project
  - You need to output the codecoverage
- Build
  - Build your Docker Image per Project
- Deliver stage
  - Push the Image to Docker Hub per Project
- Deploy Stage
  - Deploy your services to AWS
  - Run the Docker Image
  - Blue/Green Deployment
- Development of three features
  i. Feature: Allow users to register and login, so that multiple users can use your product, e.g., each user has its own Todos
  ii. Feature: Sort the Todos by date and unfinished Todos automatically go to the next day, finished Todos stay in the day when they were finished. The second feature is part of a bigger feature, make sure that this feature is behind a feature toggle, which can be turn on and off at runtime.
  iii. Feature: Create a register button in two different colors and two different headlines. The third feature should be used in an AB-Test. Split your users into two different groups and deliver two different register buttons and headlines to the end user.
- Create a Definition of Done for your team.
- Refine the Backlog and create User Stories for the features.
- Create Docker Files for your services (frontend, backend)

- Create a Documentation for your semester project where you document every part of your Project. Make sure to describe what, why you use something and how to use it.
  - Describe the
    - Applications + Docker Files
    - Version control
    - Infrastructure (AWS)
    - Docker Hub
    - CI Servers
    - Code Quality Server
    - Security Scan Server
    - Feature Toggle & A/B Test Server
    - Each Stage of the pipeline
    - A failing Stage and what happens after such an event
    - Features and the refined User Stories
    - Your A/B Testing, so that other developers can easily recreate a split test with another feature.
    - Your Feature Toggles, so that other developers can easily recreate a Feature Toggle with another feature.
    - Describe Your Blue/Green Deployment
  - Give an outlook on how and why you would start adding logging, monitoring.
  - The documentation must use the official FHTW Template