DevOps Report

Quantum TLC
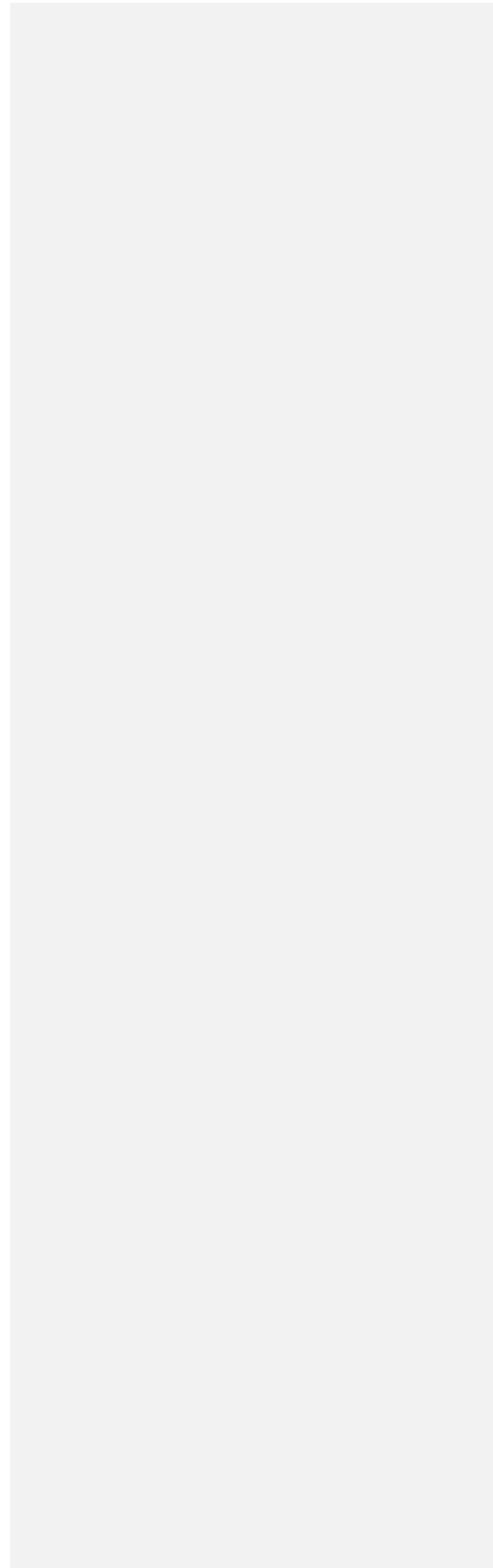
Table of Contents

Introduction

DevOps is a cross-functional systems engineering culture that aims at unifying systems development (Dev) and systems operations (Ops). A DevOps culture has an agile mindset. There are multiple common DevOps activities (DevOps, sd). In this document we will explain how the CI/CD pipeline will be setup and what DevOps activities we integrate in our DevOps cycle.

What is DevOps

DevOps combines development (Dev) and operations (Ops) to unite people, process and technology in application planning, development, delivery and operations. DevOps influences the application lifecycle throughout its planning, development, delivery and operations phases. A DevOps culture involves all roles in each phase to some extent. By using a CI/CD pipeline a lot of work can be automated. Continuous Integration helps with automating the merging and testing of code, this way bugs are caught early on. Continuous Delivery automates the deployment of the application to one or more testing and production environments. Version control is used to manage code in versions, it tracks change history to make code easy to review and recover. Using version control is fundamental in DevOps. DevOps also helps with monitoring the applications health (What is DevOps, 2023).

The benefits of DevOps are speed, changes to adapt to the market and customers needs can be made faster. This also means new features can be released quicker, by automating the steps from building to deploying. It also makes the application more reliable, by continuously testing each feature to make sure the application works as expected. Monitoring and logging help with staying informed of performance of the application in real time (What is DevOps, sd).
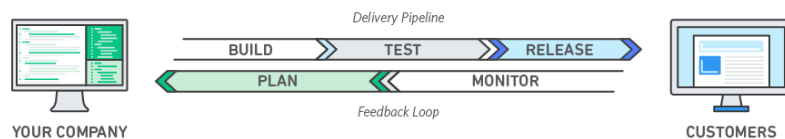
DevOps Best Practices

For DevOps, we can apply several methods to manage the project.

Agile Project Management and Workflow

With the use of methodologies of for example Scrum, the management of projects becomes far easier. The iterative approach with breaking down project tasks, while also having continuous feedback loops to adapt and change requirements and deliver value constantly. Furthermore, the clear workflow with phases of to-do, in-progress, code review, and done, enables the team to track progress effectively and respond to feedback.

CI/CD and Testing Automation

Integration continuous integration and continuous deployment (CD/CD) pipelines to automate several stages such as building, testing, analyzing, and deploying processing. Integrate this with early on testing in the development cycle, to identify and address issues sooner. Moreover, automated tests of different types including unit tests, integration tests, E2E tests, and performance tests will make sure the code is up to quality and reliable.

Tool Selection and Integration

Having tools for each stage in the DevOps lifecycle, with a focus on collaboration, automation, and monitoring capabilities. Selection of tools that are easy to integrate and communicate with between development. Also, version control systems like, Git, platforms like Jenkins, and monitoring tools like Prometheus to improve the smoothness of development and workflows to improve the software quality.

Culture of Collaboration and Continuous Improvement

It is important to have a culture of collaboration that is transparent and is continuously improving across the team. It is important to share knowledge, receive and give feedback, and have overall open communication. To add on, the process is to continuously find improvements and seek a workflow that improves efficiency and effectiveness. This will lead to a more high-quality software product. (HALL, n.d.)

Furthermore, with DevOps, DevSecOps (Development, Security, and Operations) is where we think about security in each stage of the software development process. This is further explained in (Jacobs, 2024, Security Design) to show what stages there are. (What is DevSecOps?, n.d.)
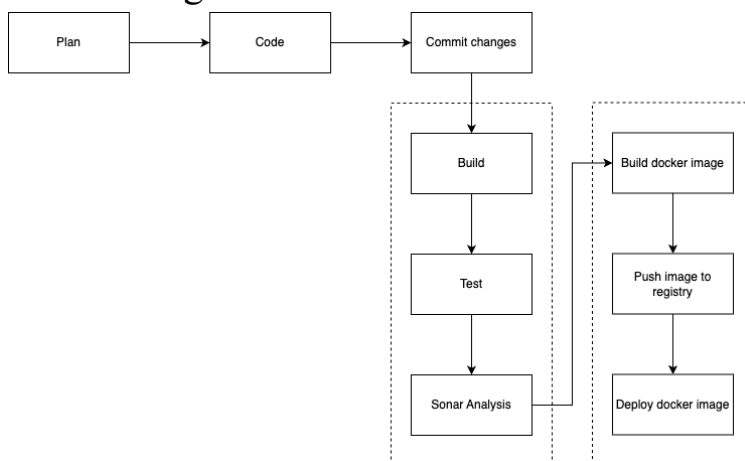
CI/CD Diagram

Figure 1: CI/CD Diagram

This is the CI/CD diagram. It starts with making a plan each sprint. In a meeting with the stakeholders the user stories that will be worked on during the sprint will be selected. The developers will write the code and commit the changes to Git. This will trigger the CI/CD pipeline, which will be explained below.

CI Pipeline

CI stands for Continuous Integration. The goal of continuous integration is to automatically build the application and run the tests, to discover errors and bugs quicker. This improves the productivity of the developers, because a lot of work is automated. It is important to commit changes often.

When new code is pushed to git the CI pipeline will be triggered. The first stage of the CI pipeline is building the application. It passes when a build is completed with no errors. The next stage is the test stage, this is where all the tests are automatically run. If any of the tests fail, the test stage also fails. The final stage in the CI pipeline is the sonar static code analysis. SonarQube is a code review tool that helps with delivering clean code. It scans the code for bugs, security risks and reveals the code coverage of the unit tests. This stage fails when the total test coverage is below 80% or if any bugs or vulnerabilities are detected in the code. Snyck will be used to check for vulnerabilities in the front end. The stage will fail when there are high vulnerability issues. After all the stages in the CI pipeline pass, the CD pipeline will be triggered.

> **Commented [JL1]:** Add Snyck, for security vulnerabilities of front end

CD Pipeline

CD stands for either Continuous Delivery or Continuous Deployment, depending on the chosen strategy. With continuous delivery the code is always ready for release but has to be manually approved before it is deployed. With continuous deployment the release process is automated once all the tests and stages of the pipeline pass. Continuous delivery is the best when extra control before releasing is important. When using the continuous deployment strategy, the quality of the automated tests should be considered. Continuous deployment is a good strategy for when new features are released on a daily or even hourly basis, when new features are released on a schedule continuous delivery is the better strategy (Continuous Delivery vs. Continuous Deployment: Where To Draw The Line?, sd). For the Quantum TLC tournament application we will be using continuous deployment, meaning the application will be automatically deployed.

When all the tests pass, the test coverage is above 80% and there are no vulnerabilities detected the CD pipeline will be triggered. The first step is building a docker image. The Docker image is then uploaded to a container registry, a storage and distribution system for Docker images, for example Docker Hub or Azure Container Registry. Finally, the Docker image will be deployed to Kubernetes. The Kubernetes cluster will be hosted on Azure.

Works Cited

Continuous Delivery vs. Continuous Deployment: Where To Draw The Line? (n.d.). From Katalon: https://katalon.com/resources-center/blog/continuous-delivery-vs-continuous-deployment#

DevOps. (n.d.). From TMAP: https://www.tmap.net/page/devops

HALL, T. (n.d.). *DevOps Best Practices*. From atlassian: https://www.atlassian.com/devops/what-is-devops/devops-best-practices

What is DevOps. (n.d.). From Amazons AWS: https://aws.amazon.com/devops/what-is-devops/

What is DevOps. (2023, 01 25). From Microsoft Learn: https://learn.microsoft.com/en-us/devops/what-is-devops

What is DevSecOps? (n.d.). From aws.amazon: https://aws.amazon.com/what-is/devsecops/#:~:text=building%20the%20software.-,What%20does%20DevSecOps%20stand%20for%3F,they%20are%20building%20software%20applications.