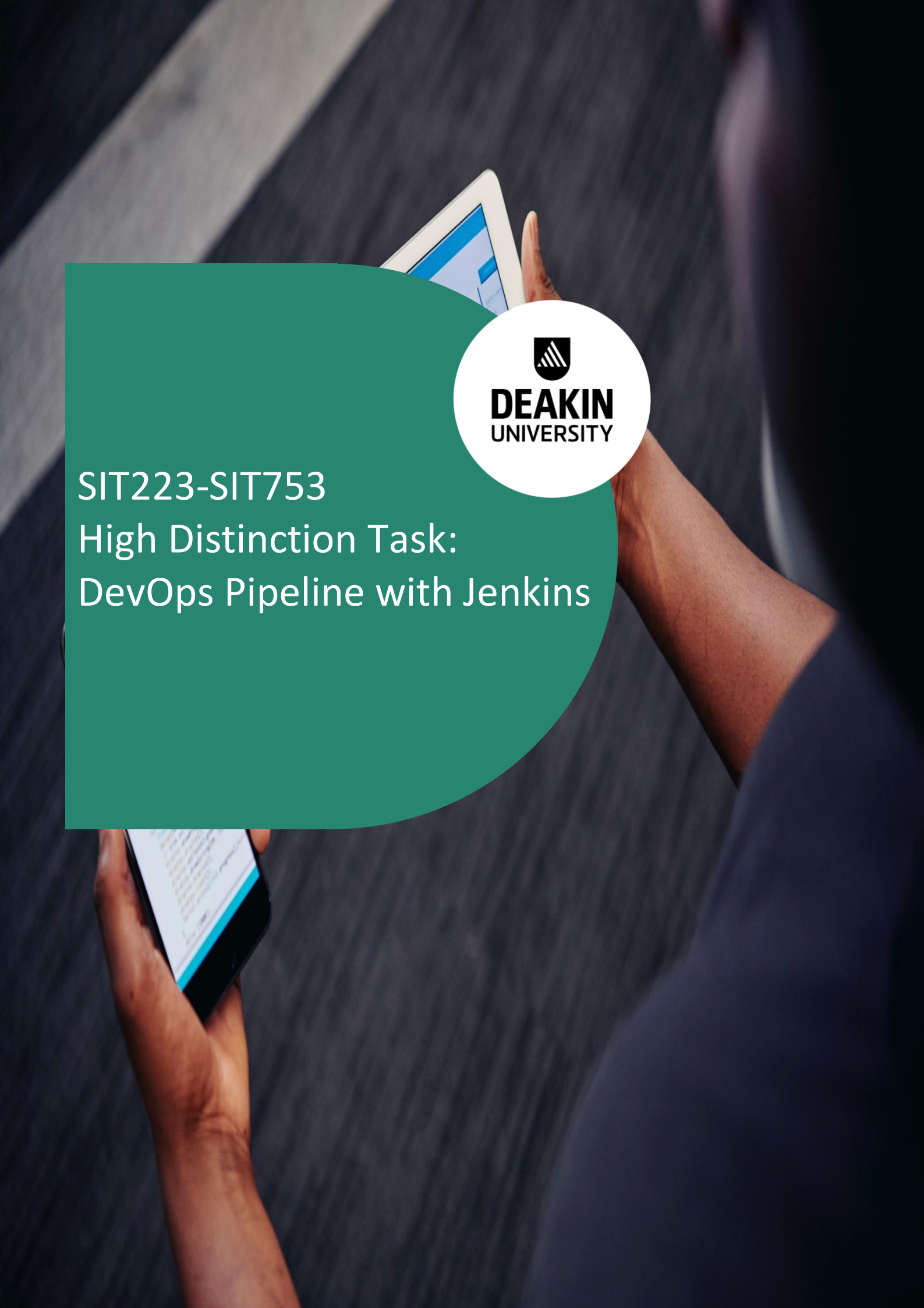

DEAKIN UNIVERSITY

SIT223-SIT753
High Distinction Task:
DevOps Pipeline with Jenkins

Overview

In this task, you will create a DevOps pipeline for your own project or code using Jenkins to ensure that your code is built, tested, and deployed in a reliable and consistent manner. The pipeline includes automated testing, code quality analysis, and deployment. Continuous delivery and deployment tools are included to ensure that changes are automatically deployed to test and production environments, and monitoring and alerting tools are included to ensure that any issues are quickly identified and addressed.

Assessment Instructions

To achieve a low HD grade, it is necessary to implement only three stages from steps 4-9. Successful implementation of these three required stages will demonstrate the completeness of pipeline functionality. However, developing more than three stages can lead to a high HD grade.

1. Choose a project of your choice. This could be your Capstone project or previous projects, for example, a web application or mobile application.

2. Create a Git repository for your project and push your code to the repository.

3. Create a Jenkins pipeline for your project with all stages: Build, Test, Deploy, and Release.

4. In the Build stage, configure Jenkins to build your code and create a build artifact. This could be a JAR file, Docker image, or any other artifact that can be used to deploy your application.

5. In the Test stage, configure Jenkins to run automated tests on your code. You can choose any testing framework of your choice, such as JUnit, Selenium, or Appium to test your application in the test environment to ensure that the deployment was successful, and the application is working correctly.

6. In the Code Quality Analysis stage, configure Jenkins to run code quality analysis on your code. You can choose a code quality analysis tool, such as SonarQube or CodeClimate, and configure it to automatically analyse your code for issues such as code duplication, code smells, and security vulnerabilities.

7. In the Deploy stage, configure Jenkins to deploy your application to a test environment, such as a staging server or a Docker container. You can use any deployment tool of your choice, such as Docker Compose or AWS Elastic Beanstalk.

8. In the Release stage, configure Jenkins to promote the application to a production environment. You can use any release management tool of your choice, such as Octopus Deploy or AWS CodeDeploy.

9. In the Monitoring and Alerting stage, configure Jenkins to monitor the application in production for any issues and alert the team if any issues arise. You can choose a monitoring and alerting tool, such as Datadog or New Relic, and configure it to monitor the application in production for any issues and alert the team if any issues arise.

Submission Guidelines

- Create a demo video that showcases your pipeline in action. The video should be no longer than 10 minutes and should demonstrate the following:
  - How to clone the repository and set up the pipeline in Jenkins.
  - How the pipeline works and how it progresses through each stage.
  - The final deployed application and any additional features you want to showcase.

- Submit a document that includes the following:

  - A link to the demo video
  - A brief description of your project and the technologies used.
  - A screenshot of your Jenkins pipeline.
  - A brief description of each stage of your pipeline.
  - A brief description of the testing framework used and the tests that were run.
  - A brief description of the deployment tool used and the test environment that was deployed to.
  - A brief description of the release management tool used and the production environment that was deployed to.