**DEVOPS FINAL REPORT**

**Applying CI/CD for Software Development**

*Decentralized Application Front-End*

Instructor:

**Mr. Yongchang He**

Members:

**Hai Nam Nguyen – 000520322 –** [**nguyen0465@saskpolytech.ca**](mailto:nguyen0465@saskpolytech.ca)

**Cong Chi Tai Nguyen - 000516006 -** [**nguyen6169@saskpolytech.ca**](mailto:nguyen6169@saskpolytech.ca)

**Xuan Hieu Nguyen – 000518043 –** [**nguyen8191@saskpolytech.ca**](mailto:nguyen8191@saskpolytech.ca)

A logo with purple and grey stripes

Description automatically generated

**Table of Contents**

[**1.** **Problem Definition** 3](#_Toc163827287)

[**2.** **Design and Architecture:** 3](#_Toc163827288)

[**3.** **Implementation Stages** 6](#_Toc163827289)

[**4.** **Testing** 7](#_Toc163827290)

[**5.** **Conclusion** 8](#_Toc163827291)

# **Problem Definition**

This project aims to solve some common problems that have been happening while developing our front-end:

* **Versioning control**: it’s difficult to manage changes across teammates without a versioning control system. This thing will be a barrier for collaboration within our team in code discussion and development which needs to be centralized. In consequence, this can lead to conflicts, code override and difficult to track changes.
* **Build and deployment process**: Development cycles can be slowed down by build and deployment procedures that are manual or inefficient. By using different compilers can created different deployment packages leading to inconsistencies.
* **Deployment environment:** using different configuration and installing manually to create environment for deployment, testing or production can lead to unexpected errors and failure deployment.

1. **Design and Architecture:**

Our application will use modern technologies like Amazon Web Services (AWS) or DevOps, NextJS for the front-end. To buid it, the team will use AWS CodePipeline, CodeCommit, CodeBuild, and CodeDeploy to manage Continuous Integration and Continuous Delivery (CI/CD). This process is dedicated to the front-end application, which will be deployed to AWS Simple Storage Services (S3) and hosted as a static website.

A diagram of a software development process

Description automatically generated  
The front-end will be developed in NextJS which use ReactJS as the core library, combining with CSS Tailwind Framework, the simple of it look like the images below:

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

1. **Implementation Stages**

The implementation stages include several key steps. First, we create a strong foundation for our code by setting up a reliable CodeCommit repository. Then, we make things smoother by setting up continuous integration using CodeBuild. Next, we automate the deployment of our applications using CodeDeploy. We also take care of managing our project files using S3, a secure storage system. Finally, we consolidated all stages in a pipeline by using CodePipeline. These steps work together to make our development and operations run more smoothly. They help us automate tasks and create more efficient DevOps processes.

* Setting Up Version Control with AWS CodeCommit
  + Create a new AWS CodeCommit repository for the project.
  + Set up access controls and permissions for team members.
  + Migrate existing code repositories
* Building and Testing with AWS CodeBuild
  + Configure AWS CodeBuild to trigger builds automatically upon code changes.
  + Define build specifications and scripts.
* Deployment Automation with AWS CodeDeploy
  + Set up deployment groups and deployment configurations in AWS CodeDeploy.
  + Define deployment scripts and configurations.
  + Integrate CodeDeploy with CodeBuild for continuous deployment.
* Configuring AWS S3 Bucket for Storage
  + Set up an S3 bucket for storing application.
  + Configure access policies and permissions for secure storage.
  + Integrate CodeBuild and CodeDeploy with the S3 bucket for retrieval during deployment.
* Manage process with CodePipeline
  + Add build project, deployments and repositories into a pipeline.
  + Describe process in buildspec.yml.

1. **Testing**

Firstly, we made a change in repository and committed to the main branch.

A screenshot of a computer

Description automatically generated

Secondly, Pipeline triggered the changes and run pipeline automatically.

A screenshot of a computer

Description automatically generated

Thirdly, after completing, SNS sent notification email to my mailbox for pipeline’s status.

A screenshot of a computer

Description automatically generated

Code changes were deployed successfully from S3.

A screenshot of a computer

Description automatically generated

1. **Conclusion**

In summary, applying Continuous Integration and Continuous Deployment (CI/CD) in software deployment can reduce human mistakes, inconsistencies, and unexpected failures. CI/CD offers speed, efficiency, and high-quality delivery by automating tasks like code validation and deployment. It assures consistency and reliability in deployment environments, enabling collaboration among team members. For this reason, organizations can concentrate on business goals to reach customer satisfaction.