PWC - BlogPipe

Terms of Reference

Team Zenith
Date: 24th January, 2023.
Version 1.0

Introduction

This project aims to construct a blog application that offers a seamless user experience by incorporating essential features such as authentication, commenting, creating and deleting blog posts. Additionally, the project aims to further enhance the developers experience by including a live dashboard that shares multiple build statuses, managing cloud infrastructure using Terraform, and adding image attestation using Binary Authorization.

To ensure the quality of the final product, the project will implement a CI/CD pipeline using the Google Cloud platform and will be containerized. Communication within the team will be facilitated through the use of Slack. The project will go through development, quality assurance, and production stages to ensure the quality of the final product. Java will be used to construct backend and React JS/JavaScript will be used in frontend along with HTML and CSS.

The final product will showcase the blog application as well as the various stages of the CI/CD progression using a web-based user interface.

Project Introduction

Project Overview

Background Information

- The project aims to automate code testing and software deployment using CI/CD pipeline, in order to maintain high-quality and stable software.

Problem/Opportunity Statement

- Releasing features without automated testing carries a high risk of defects being found later on, therefore can result in significant time loss.

Goals and Objectives

Blog Platform

- The blog application will support account creation
- The blog application will support the creation and deletion of blog posts by registered users only
- Commenting on blog posts will only be allowed for registered users
- Minimum 80% coverage in unit tests

CI/CD Solution

- CI/CD solution should support the development team in checking in the code to the version control system and run automated unit tests on it.
- A web UI will be developed to display the CI/CD progression
- Unit test results should be shared in a Slack channel (via Slackbot)
- There should be three environments to deploy to:
  a. dev
  b. qa
  c. Prod
- Pushes to non-feature branches (dev, qa, prod) will deploy the solution to the appropriate environment.
  a. Pushes to feature branches should not cause a deploy (build + test the code only)
- Use of multiple build phases in the pipeline whenever possible.
- Software dependencies and binaries should be installed and built once respectively in the pipeline execution.

Non-functional Goals

- Solution should support concurrent usage of up-to 6 developers deploying solutions in the project.

Stretch Goals (in order of priority)

- Development of a live dashboard inside the UI to share multiple build status
- Management of cloud infrastructure using Terraform
- Image attestation using Binary Authorization

Deliverables

- A blog application with the functionality to read, create, update, and delete blog posts. The blog application must support user login and registration.
- A CI/CD solution using google cloud to automate deployment (after running automated unit tests) of any changes made on the version control for dev, qa, and prod environments.

Benefits

CI/CD: *faster and more robust release pipeline*

- Automation: Allows automated testing and deployment
- Faster releases: Increases rate of releases and easier to rollback changes if necessary
- Reduced risk/Quality Assurance: Automated testing and deployment decreases the risk of human error, thereby increasing the reliability of the software releases.

Google Cloud Platform: *cheaper, more secure*

- Easier collaboration
- High security
- High performance
- Cost-effective
- Global coverage

Docker: *PaaS technology that allows developers to deploy applications as containers that are:*

- Portable
- Independent/Self-sufficient
- Lightweight
- Isolated from the OS and other containers

Scope

In Scope

- Blog platform: A blog platform will be developed as per the technical requirements
- CI/CD solution: A CI/CD solution with automated unit tests, a web ui for displaying CI/CD progression status, and a slackbot will be developed.

Out-of-Scope

- Blog platform: Any additional features in the blog platform will not be added unless they are mutually agreed by the project sponsor and the development team.
- CI/CD solution: Any additional features in the CI/CD solution will not be added unless they are mutually agreed by the project sponsor and the development team.Project Constraints, Assumptions, Dependencies and Risks

Constraints

- Limited cloud and CI/CD experience of the development team which may result in certain tasks taking more time than expected.
- Due to limited time, the development team can meet and discuss project progress only on mutually agreed upon meeting times. It is hoped that the team would meet at least once a week.

Assumptions

- All resources/technologies needed to build the project will be available to team members and be in good working order.
- All team members will be present and contributing for the entirety of the project life cycle.
- All team members can complete tasks assigned to them within a specified time frame.

Risks and Assessments

Risk Description	Assessment
Low experience level: most members of the team have less than one year of professional software development experience.	Impact: low. The course has been actively equipping the group with new knowledge and plenty of support.
Lack of experience with certain technologies: there are some technologies and tools that have never been used by team members.	Impact: medium Everyone will eventually learn the necessary skills, but it may be more time-consuming to start.
Scalability: the final product should be able to handle a large number of users and traffic, if the project is not designed with scalability in mind it may become a risk.	Impact: low This project is putting more emphasis on building technical details other than scalability.
Dependency on external platforms: the project relies on various external platforms such as Google Cloud, Terraform, and Binary Authorization, which may introduce additional risks such as compatibility issues and platform downtime.	Impart: medium it may take some time to get it compiled and interaction between the platforms.

Project Approach and Acceptance Criteria

Project Approach

We will use the waterfall approach during phase 1, agile development methodology for other phases.

Phase 1 (documentation and team management) :

1. Define the project scope: Clearly define the project requirements and objectives, and create a project plan that outlines the overall approach, timeline, and deliverables. (Most of this is done in this document, others can be viewed in attached referenced documents)
2. Team management and task division: Since most of us are unfamiliar with sponsor proposed technologies (such as docker, google cloud, slack api). We assign learning tasks into smaller sub groups with different roles to make sure team members don't have to learn everything while still having the necessary skills to complete the project.
3. Define the product backlog: Create a product backlog that outlines the user stories and requirements for the project. Prioritize the items in the backlog based on their importance and dependencies.

Phase 2 (Implementation/CI):

1. Initialize the project, create repo on github and set up the environment.
2. Plan the sprint: At the start of each sprint, the team will review the product backlog and plan the work for the upcoming sprint. Tasks will be assigned to team members, and a sprint goal will be established.
3. Develop and test: During the sprint, the team will work on developing the functionality outlined in the sprint plan. The team will also perform testing and quality assurance to ensure that the functionality is working as expected.
4. Review and retrospect: At the end of each sprint, the team will review the work that has been completed, and reflect on what went well and what could be improved. Feedback will be collected and used to plan the next sprint.

Phase 3 (deployment/CD):

Deployment: Once the development and testing phase is completed, the team will deploy the blog application on a web server or hosting platform.

Most of the technical details are still to be decided, we will update once determined.

Estimated Schedule

Milestone or Key Activity	Start	Complete
Initiation	Jan 23	Jan 25
Requirements & design	Jan 23	Jan 31
Build – Core Node build	Feb 1	Feb 28
Pilot		
Implementation – Branch Visit		
Project Completion		April 1

Estimated Effort and Cost

We expect each group member works 10hr/week on this project

Project Governance

Project Team Responsibilities

Every person in this team is crucial and their efforts are directly related to the success of our project. Each team member should:

- Contribute to the overall project objective
- Document respective parts of the project
- Complete individual deliverables to the best of their ability
- Align with team members and have open lines of communication and collaboration

Project Role	Person Responsible
Project Sponsor	Evan Seabrook / Alvin Madar
Project Manager	Asem Ghaleb

Project Team Structure

Scrum Master: Andrew

Backend Devs: Andrew, Eric, Ruchir, Param

Frontend Devs: Anthony, Anusha, Saad

QA: Cheryl, Shawn

Project Communication Plan

2 meetings per week, each lasting about 1 hour. Meetings will be after class on Monday and Wednesday.

Additional meetings via Zoom can be added if needed.

The team uses Discord for communication and Google Docs for documentation.
The team will have weekly check-ins on Wednesday with their TA and Instructor to solve any issues relating to the project or to ask for help/advice.