PWC - BlogPipe

External Design Document

Team Zenith

Date	Version
Feb 1, 2023	1.0

Table of Contents

1 - Top Level Business Requirements

PriceWaterhouseCooper(PWC) wants to pursue code quality by automating tests into version control systems. This project aims to construct a blog application that offers essential features such as authentication, commenting, creating and deleting blog posts. This serves the purpose of demonstrating the progress of the CI/CD pipeline.

The scope of the project includes a blog platform, a CI/CD solution with automated unit tests, and a web UI for displaying the CI/CD progression status. Any additional features will not be included unless mutually agreed upon by the project sponsor and development team.

The project is constrained by the limited cloud and CI/CD experience of the development team and the short time span for project design compared to a formal business project. The project is expected to be completed within 4 months, and it is assumed that all group members will be present and efficient and that all necessary resources and technologies will be available. The project will be considered complete if it includes all essential features, and successful if it achieves additional stretch goals.

2 - Top Level System Requirements

This project consists of three components:

Blog Platform: A blog platform that will allow anyone to read blog posts. This blog platform will allow registered users to create, modify, and delete blog posts. In addition to the previously mentioned capabilities, users may also comment on other blog posts.

CI/CD Pipeline: CI/CD pipeline will take care of carrying out automated unit tests and code pushes to the version control system (when code is pushed on dev, prod, and qa branches). The pipeline should also support multiple developers pushing code simultaneously. The pipeline should be built such that it may be easily repurposed for some other project.

Web UI for CI/CD Progress

A web UI needs to be created to display the CI/CD progress. This web UI is intended to be accessed by the developer team only.

3 - Use Cases

1 - Register a User Account	
Description	A user can register an account with the blog platform using Google OAuth or another method.
Primary Actor	Blog user
Precondition	User must have an email address (or google account if only Google OAuth is used)
Postconditions	Users will be registered with the blog platform. They can now create, update, and delete their posts.
Main Scenario	1. The user will signup with a signup form 2. Users will fill in their credentials (username, email, and password). 3. Once the user has successfully signed up, they will be informed about it and then redirected to the home page.
Alternate Scenarios	1. An unregistered user tries to create a post/comment on a post. 2. Users will be redirected to a signup / sign-in page to allow them to register with the blog platform. 3. Users will fill in their credentials (username, email, and password). 4. Once the user has successfully signed up, they will be informed about it and then redirected to the home page.

2 - Create a Post	
Description	Contributors would like to create a post to be viewed by readers.
Primary Actor	Blog User (Contributor)
Precondition	The blog user has to be signed in and has to be approved for ‘Contributor’ status.
Postconditions	The created post will be displayed to all readers and be open to comments from readers.
Main Scenario	When the contributor makes a post, it will be displayed with the creator’s user ID and avatar.
Alternate Scenarios	N/A

3 - Create a Comment	
Description	Blog users would like to comment on posts that interest them
Primary Actor	Blog User
Precondition	The blog user has an account and is logged in
Postconditions	A new comment has been created and it will be displayed under the post in chronological order of posting with the user account ID and avatar.
Main Scenario	When the blog user makes a comment, the comment will be displayed under the post.
Alternate Scenarios	N/A

4 - Read a Post	
Description	A blog visitor wants to read a post
Primary Actor	Anyone who visits the blog
Precondition	None
Postconditions	Blog visitors are able to read the blog posts.
Main Scenario	Blog visitors visit the blog website and click on the blog post's headline that they want to read further. They can navigate the website through links in the Navbar.
Alternate Scenarios	Blog visitors view a blog post by directly requesting it through the URL.

5 - Edit a Post	
Description	The contributor of a post is allowed to edit it.
Primary Actor	Blog User (Contributor)
Precondition	The blog user has to be signed in and has to be approved for 'Contributor' status.
Postconditions	The contributor will now see their edited post instead of the one they had before.
Main Scenario	The contributor can easily edit their post using an 'edit' button which

	will open their current post in a text box. They can then publish their new post.
Alternate Scenarios	N/A

6 - Delete a Post	
Description	Blog users would want to delete their posts in some cases
Primary Actor	Blog User (Creator of the blog post)
Precondition	The user has created a post and is classified as a contributor
Postconditions	The post and the comments under the post will be deleted, the blog posts list will no longer show this post.
Main Scenario	When the blog user deletes a post, the post will be deleted and the user will go back to the previous page (blog post list).
Alternate Scenarios	N/A

7 - Receive Slackbot Notification	
Description	The developer team receives a notification on their slack channel when the result from CI/CD pipeline is ready.
Primary Actor	Developer
Precondition	Any changes in the dev, qa, or prod environment are in progress via CI/CD pipeline and the pipeline finished execution with the result ready. There is an available slack bot and slack channel
Postconditions	Developers will receive an automated notification in the slack channel from the slackbot.
Main Scenario	After the CI/CD pipeline generated the result, slackbot will post a notification in the slack channel. This allows teams to be informed of the outcome of the pipeline and take any necessary actions.
Alternate Scenarios	N/A

8 - View CI/CD Dashboard	
Description	The developer wants to monitor the progress of their code change as it moves through the pipeline.
Primary Actor	Developer
Precondition	The developer can access the CI/CD dashboard which displays the status of the pipeline runs of each branch.
Postconditions	The developer uses the dashboard to verify that their code changes are meeting the requirements.
Main Scenario	Any troubleshooting needed can be easily viewed on this CI/CD dashboard and addressed promptly.
Alternate Scenarios	N/A

9 - Check-In code to Branch	
Description	Developers check in their code to the branch, which will trigger different CI/CD pipelines based on branch type (eg. non-featured branch or featured branch). Generate feedback based on the outcome.
Primary Actor	Developer
Precondition	Developer is logged in the system and has an available pipeline. The developer has code that needs to be tested.
Postconditions	The result will be shown on CI/CD platform UI. Different labels will be set depending on the result of each CI/CD stage. (green for pass, red for fail)
Main Scenario	As a developer, I want to push my code to my own branch which will trigger the CI/CD pipeline, so that I can ensure that my code is built, tested, and deployed in a consistent and automated manner. If the build fails, I want to be immediately notified of the failure and the reason for it, so that I can fix any issues and re-run the pipeline.
Alternate Scenarios	N/A

10 - Deploy code to production if needed	
Description	For the non-featured branch, if the pushed code passes all the tests, the code will be auto-deployed.
Primary Actor	Developer
Precondition	Code under test passes all the pipeline stages and is ready to be deployed. Have an available deployment environment.
Postconditions	the code will be automatically deployed to production
Main Scenario	As a developer, I want my code to be automatically deployed to production if it passes all tests in the CI/CD pipeline for a non-feature branch. This will allow for a streamlined and efficient process for delivering code changes to production, providing quick and seamless updates to the live environment.
Alternate Scenarios	If the code is pushed during a specified maintenance window or code freeze period, it will not be auto-deployed even if it passes all tests in the pipeline. This is to ensure that production updates do not interfere with critical maintenance or upgrade activities.

4 - Non-Functional Requirements

Security Needs
- SSL/TLS is required for encrypted connection
- Data in the database should be secure
    - Firestore has customizable security and data validation rules to protect data

Capacity Needs
- CI/CD pipeline needs to be able to handle 6 developers deploying simultaneously

5 - Requirement Traceability matrix based on Use Cases

REQ no.	Reference	Requirement description	Type	Priority	MVP(Y/N)
1	US - 1	Register User Account	F	2	Y
2	US - 2	Create a Post	F	1	Y
3	US - 3	Create a Comment	F	2	Y
4	US - 4	Read a Post	F	1	Y
5	US - 5	Edit a Post	F	1	Y
6	US - 6	Delete a Post	F	1	Y
7	US - 7	Receive Slackbot Notification	F	3	Y
8	US - 8	View CI/CD Dashboard	F	2	Y
9	US - 9	Check-in code to Branch	F	2	Y
10	US - 10	Deploy code to production	F	1	Y
11	NA	TLS/SSL encryption	N	3	Y
12	NA	Support 6 developers deploying simultaneously	N	3	N

Key:

US - User Story
F - Functional
N - Non-functional

1 - Highest Priority
2 - Medium Priority
3 - Lowest Priority