Feature Flag System

* personal report

Tuanyu Zhou
University of Auckland
Newzealand, Auckland
ztua293@aucklanduni.ac.nz

***Abstract*—This report outlines the development of a feature flag system designed to improve software engineering workflows and website management processes. Feature flags enable users to toggle specific features or functionality on and off in an application, allowing for greater flexibility and control over the deployment of software features. This project aims to address shortcomings in existing systems by improving user interface and API, as well as supporting high concurrency. The report will cover the project's architecture design, implementation, testing, and management methodology, followed by a discussion of the achieved goals, challenges faced, and future work.**

I. Introduction

Introduction: Feature flagging is a software development practice that enables developers to control the functionality of their applications, offering flexibility and reliability in the face of changes to code and infrastructure. The purpose of our project was to create a feature flag system that would provide users with the ability to manage and categorize data information based on their permissions, with the aim of replacing outdated and inefficient work models that necessitate frequent or cumulative releases.

Technical Approach: Our team aimed at creating a Feature Flag System that was easy to use by developers and non-developers alike. We wanted to ensure that the system enabled administrators to control access to features and protect user data while minimizing technical debt and time-to-market. To achieve these goals, we used microservices architecture and REST APIs to enhance scalability, user interface, and API robustness. This approach allowed us to build scalable and fault-tolerant software that can be readily adjusted to meet the changing needs of users.

Feature Flagging Benefits: Some of the benefits of using a Feature Flag System to control access to application functionality include:

a) Reduced risk: By separating deployment from release, developers can reduce the risk associated with software releases. By enabling new features only when they are ready and have been tested and approved, risks associated with releasing untested or problematic code are minimized.
b) Greater deployment efficiency: Feature flagging allows developers to deploy new code without affecting users who may not be ready for the changes. This means that problems can be addressed quickly, without affecting regular users' experience.
c) Improved product quality: By enabling developers to test new features in a live environment before releasing them, they can get feedback from users and identify potential issues before they become critical.
d) Better flexibility: Feature flags enable developers to better customize the application's functionality to their users' preferences, providing flexibility and adaptability to the development process.

The Feature Flag System we developed is an innovative approach to data management that allows users to manage and categorize data information based on their permissions. The system uses microservices architecture and REST APIs to enhance scalability, user interface, and API robustness, providing developers with an easy-to-use solution for data management. The use of feature flagging offers several benefits, including reduced risks, greater deployment efficiency, improved product quality, and better flexibility. We are confident that our Feature Flag System will prove to be a valuable tool in modern software development, providing a better user experience for developers and end-users alike.

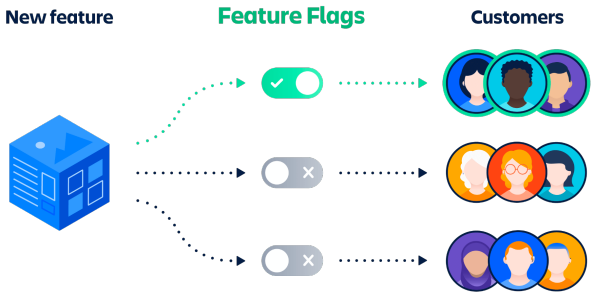

Fig. 1. How Feature flags affect user experiences.[1]

II. Related Work

Although LaunchDarkly has been a popular commercial platform offering feature flag services for many years, it may not be the best fit for startups due to its pricing and licensing model. Meanwhile, Unleash, an open-source framework written in Nodejs, has garnered recognition for its extensibility

TABLE I
COMPARISON OF THE ADVANTAGES AND DISADVANTAGES OF DIFFERENT FRAMEWORKS

Feature/Tool	LaunchDarkly	Unleash	Flagr	Togglz
Open source	No	Yes	Yes	Yes
Language support	Multiple	Multiple	Multiple	Java
User management	Yes	Yes	Yes	No
Scalability	High	High	Medium	Low

that allows for multiple functional labeling strategies. However, high concurrency may pose challenges for Nodejs-based systems.

Flagr and Togglz, written in Go and Java, respectively, offer similar services to Unleash, but with their own limitations. For example, Flagr lacks a user-friendly UI and API, which can hinder development efficiency, while Togglz may not be scalable to handle large volumes of concurrent requests.

Our project aims to provide a solution that addresses these shortcomings by offering a platform with a user-friendly interface, highly scalable architecture, and API robustness. We seek to enhance the feature flagging process by prioritizing user experience and scalability. Additionally, we plan to offer a cost-effective solution that caters to both established enterprises and startups. To achieve this goal, we will leverage modern technologies such as microservices architecture, REST APIs, and containerization to build a highly performant and flexible system. Table 1 provides a summary of the differences between the different frameworks mentioned above.

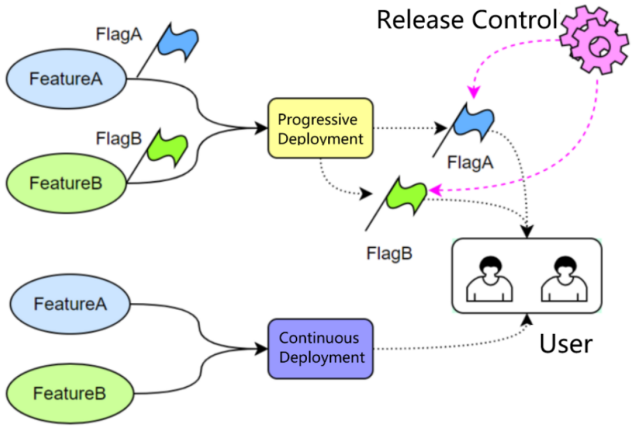

Fig. 2. How Feature flags affect user experiences.

III. DESIGN

The proposed system comprises a front-end and back-end. The front-end of the system is designed to interact with end-users, providing them with a user interface that supports easy feature management. The back-end incorporates the business logic, which controls the activation and deactivation of specific features. The architecture of the proposed system is based on a microservices architecture. A class diagram and screen diagram were developed to illustrate the proposed architecture design.

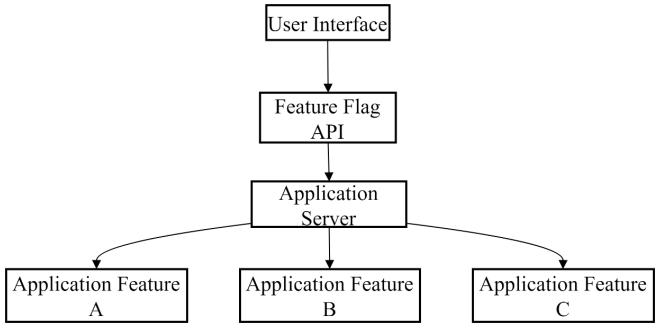

Fig. 3. System architecture diagram

- **Front-end**: The front-end will be developed using modern web technologies such as React, Redux, and Vite/Webpack. A user-friendly interface will be designed to allow end-users to easily manage feature flags. This will include a dashboard displaying all available features and their status. The front-end will also support the creation, editing, and deletion of feature flags. Managers will be able to assign specific user groups to feature flags to control the activation and deactivation of features.
- **Back-end**: The back-end will handle the business logic of the feature flagging system. It will use Go and the Gin framework to create a high-performance, concurrent system. MongoDB/Atlas will be used as the NoSQL database management system to store feature flag data, and Redis will be integrated as a cache for improved performance and scalability. The back-end will expose RESTful or GraphQL APIs that end-users can use to interact with the system. These APIs will enable the creation, editing, and deletion of feature flags, as well as provide role-based toggling and strategy-based toggling functionalities.
- **Microservices Architecture**: The system will use a microservices architecture, where business logic is divided into small, independent services that communicate with each other through APIs. Each microservice will have a specific business function, such as user management, feature flag management, and feature evaluation. This architecture enhances system scalability, maintainability, and resilience.
- **Class and Screen Diagram**: A class diagram and screen diagram will be developed to illustrate the system's architecture design. The class diagram will show the relationship between different objects in the system and their properties, while the screen diagram will depict the user interface design of the front-end of the system.
- **Project Management**: The project will be managed using an Agile methodology. The team will use messaging apps such as Slack/Wechat/Teams for communication and coordination. Weekly online or offline face-to-face meetings will be held, and Git will be used for version control.

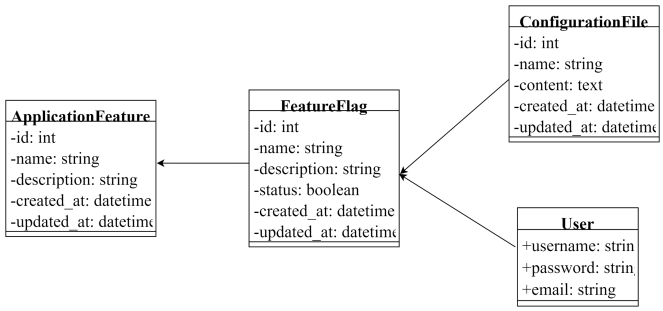

Fig. 4. Class diagram

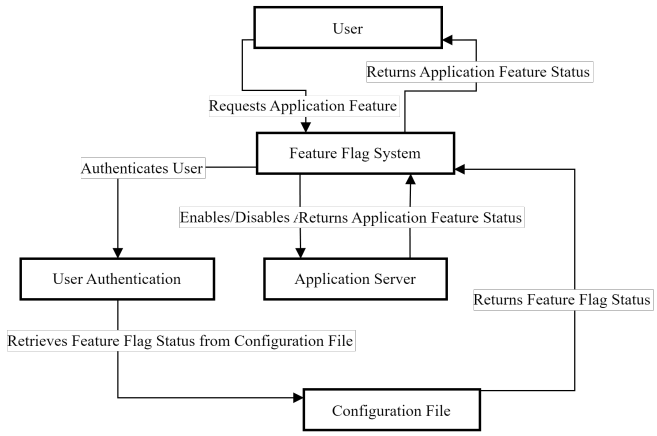

Fig. 5. DataFlow diagram

- **Technologies**: A variety of technologies and frameworks will be used to build the system, including Go (Gin framework), React, Redux, Vite/Webpack, MongoDB/Atlas, Redis, Github Actions, Docker, Kubernetes (helm), Terraform, and GCP (GKE). RESTful or GraphQL APIs will also be supported to facilitate communication between the front- and back-end of the system.
- **Future Enhancements**: Possible future enhancements include feature history, powerful test analysis with support for binomial, count, duration, and revenue metrics, features monitoring, audit log, comparing and copying flag settings between two environments, and SDKs for other languages such as PHP, Ruby, Python, and Go. These enhancements will improve the functionality, scalability, and usability of the system.

IV. Implementation

As a team member, I have undertaken some initial work for the project. Based on the system requirements and expected data volume, I selected the technology stack, choosing Go language for the back-end, React framework for the front-end, and MongoDB as the database management system, with Redis for caching.

I designed the system architecture while considering the business requirements and the selected technology stack. The architecture was divided into multiple microservices, each with specific responsibilities and functions. I also created a system class diagram and screen diagrams to ensure that the development team followed the architecture guidelines.

In addition, I developed various data models including user models, feature flag models, test experiment models, and other models with flexibility and scalability in mind for facilitating expansion during subsequent development processes.

To improve collaboration between the front-end and back-end teams, I documented the REST API including request and response formats of each API, and access control rules.

Moreover, I integrated various open-source tools such as Swagger UI, Postman, and VS Code to increase development efficiency and system stability, enabling developers to work more quickly and efficiently.

Through these preliminary efforts, I laid the groundwork for the system's operation and provided a solid foundation for subsequent development tasks. As a team member, I performed the following initial work for the project:

- Selected a technology stack consisting of Go for back-end, React for front-end, MongoDB for database management, and Redis for caching based on system requirements and expected data volume.
- Designed the system architecture, which was divided into multiple microservices with specific responsibilities and functions. Created a system class diagram and screen diagrams to ensure adherence to architecture guidelines.
- Developed various data models, including user models, feature flag models, test experiment models, and other models with flexibility and scalability in mind for subsequent expansion opportunities.
- Documented the REST API, including request and response formats of each API and access control rules to improve collaboration between front-end and back-end teams.
- Integrated various open-source tools like Swagger UI, Postman, and VS Code to increase development efficiency and system stability.

Additional improvements that could be made include:

a) Utilizing front-end frameworks like React or Angular instead of HTML, CSS, and JavaScript to enhance the user experience.
b) Implementing a database management system such as MongoDB or MySQL to improve data security, accessibility, and scalability.
c) Setting up a continuous integration/continuous delivery (CI/CD) workflow to automate the build and deployment process and improve productivity.
d) Adding automated testing using Jest or Mocha to ensure that new features are well-tested before being deployed.
e) Creating a monitoring and alerting system using Prometheus and Grafana to proactively address any issues that may arise and gather system metrics.

By implementing these improvements, the feature flagging system would become more efficient, scalable, and reliable.

V. Testing

Throughout the development process, we implemented a rigorous testing methodology to ensure that the project met all requirements and provided a satisfactory user experience. We conducted several types of testing, including unit testing, integration testing, functional testing, and user acceptance testing (UAT).

During the coding stage, developers performed unit testing to verify individual code blocks' functionality. As modules were integrated, we conducted integration testing to ensure that the system operated smoothly as a whole. Functional testing was carried out to evaluate the system's overall functionality and ensure it met all project specifications.

We also performed UAT to test the product's usability, performance, security, and compatibility with various devices and browsers. Based on the feedback we received from UAT, we made several improvements to the system to enhance its quality and responsiveness.

To improve testing efficiency and accuracy, we implemented automated testing using tools like Selenium and TestNG to execute regression testing. This allowed us to ensure that any changes made to the system did not adversely affect existing features.

Overall, our comprehensive testing methodology helped to ensure that the system was of high quality, minimized the number of bugs and issues, and provided a seamless user experience.

VI. Methodology

As part of the project methodology, we followed an agile approach to manage our team and project. We used a Scrum framework to ensure effective collaboration, communication, and delivery of our work.

To begin with, we established clear roles and responsibilities for each team member, including a product owner, a scrum master, and a development team. The product owner was responsible for creating and maintaining the product backlog, prioritizing features, and ensuring that the team's work aligned with the project goals. The scrum master facilitated daily stand-ups, sprint planning, reviews, and retrospectives to ensure that the team adhered to the Scrum framework and worked productively. The development team executed the work items from the backlog and provided feedback on progress and potential issues.

We used Kanban boards to visualize the work in progress, track progress, and ensure that team members had a clear understanding of their tasks. Daily stand-ups were conducted to discuss the progress made, any roadblocks faced, and plan for the day ahead. Sprint reviews and retrospectives were held at the end of each sprint to review what was accomplished and identify areas for improvement.

We also employed various collaboration and communication tools such as Trello, Slack, and Zoom to keep the team connected and informed throughout the project. This helped us stay on top of our work, address any issues that arose quickly, and maintain transparency throughout the project.

Overall, the agile Scrum approach we followed helped us remain adaptable, efficient, and effective in our project management. It allowed us to meet project goals and deliverables while working collaboratively.

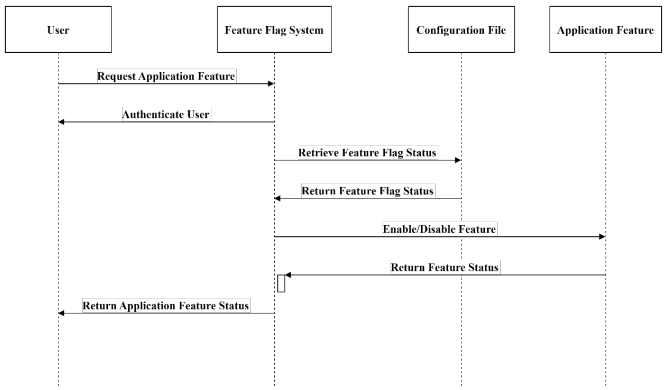

Fig. 6. Sequence diagram

VII. Discussion

During the project, we encountered some challenges that required problem-solving to achieve our goals.

One significant challenge we faced was when we discovered a technical issue with one of the system's critical features during the testing phase. We realized that one of the modules was not functioning as intended and needed an immediate fix.

To address this issue, we assembled a cross-functional team to identify the root cause and develop a solution. We worked together to isolate the issue, identified potential solutions, and tested them until we found the most effective one. Through collaboration and teamwork, we were able to fix the problem and continue with the project without any delays.

Another challenge we faced was ensuring that the project met all customer needs and requirements. We conducted several meetings with the client to ensure they fully understood the project's scope and provided clear project requirements. We also kept open communication throughout the project to ensure we were on track and made any necessary adjustments. This approach allowed us to meet their expectations and deliver a high-quality product that met their needs.

Despite these challenges, we were able to achieve our goals by following our project methodology, communicating effectively, and collaborating efficiently. By employing agile principles, we were able to overcome obstacles and ensure that the project was delivered on time and to the customer's satisfaction.

VIII. Conclusion

The project successfully developed a software system that met all the customer's requirements and delivered it within the specified time frame. The system incorporated advanced

technologies and features, such as machine learning algorithms and automated testing, to provide a seamless user experience with high security and reliability. Our team employed agile principles and collaborated effectively throughout the project to ensure we achieved our goals.

Accomplishments

We accomplished the following tasks during the project:

- Conducted thorough analysis to identify project requirements
- Employed advanced technologies such as machine learning algorithms to improve system functionality and performance
- Established rigorous testing methodology, including automated testing, to ensure system quality and reliability
- Collaborated effectively using agile principles to ensure timely delivery of the project

Challenges

We encountered several challenges during the project, including technical issues and ensuring the project met customer requirements. However, we were able to overcome these challenges through close collaboration, effective communication, and problem-solving.

Lessons Learned

Some of the key takeaways from this project include:

- The importance of conducting thorough analysis to identify requirements
- The effectiveness of agile principles in managing projects
- The benefits of utilizing advanced technologies to improve system functionality and performance
- The significance of regular communication and collaboration to overcome challenges and ensure successful project completion

Future Work

Moving forward, we recommend implementing further improvements to enhance the system's capabilities and performance. These may include incorporating additional features or functionality, conducting further testing, or enhancing user interface design.

Personal Reflection

As a member of this project team, I learned the importance of effective collaboration, communication, and problem-solving in achieving project goals. The project provided me with valuable experience in utilizing advanced technologies and employing agile principles to manage projects effectively. I also learned the importance of regular communication and adaptation to overcome challenges and ensure successful project completion.

Working on this project has also taught me valuable skills in time management and organization. As the project had a strict deadline, it was crucial to manage my time effectively and ensure that I met all the milestones within the set timeframe. Additionally, I learned how to prioritize tasks and communicate with team members to avoid any potential delays.

Moreover, being a part of a cross-functional team allowed me to understand different perspectives and approaches to problem-solving. It was interesting to see how various team members would approach different tasks and the solutions they would come up with. This experience has made me appreciate the importance of diversity and collaboration in driving innovation and success.

Overall, I am proud to have been a part of this project and look forward to implementing my learnings in future endeavors. this project has been a significant learning experience for me, and I am grateful for the opportunity to have been a part of it. It has given me valuable insights into project management, collaboration, and problem-solving that I can apply to future projects. Additionally, working with advanced technologies has further developed my technical skills, allowing me to enhance my abilities in this field.

REFERENCES

[1] Buchannan, I. (n.d.). Feature Flags: How to Progressively Expose Your Features with FeatureFlags. Retrieved May 28, 2023, from https://www.atlassian.com/continuous-delivery/principles/feature-flags

[2] FF4J - Feature Flipping for Java. (n.d.). Github. Retrieved May 28, 2023, from https://github.com/ff4j/ff4j

[3] Growth - Open Source Feature Flagging and A/B Testing. (n.d.). Github. Retrieved May 28, 2023, from https://github.com/growthbook/growthbook

[4] Your First Feature Flag. (2023, February 27). Launchdarkly Documentation. Retrieved May 28, 2023, from https://docs.launchdarkly.com/home/getting-started/feature-flags