Web development of MyNewsScan Platform for the Creation of Sustainable User Community

Yuhang Tang

Master of Science

Computer Science

School of Informatics

University of Edinburgh

2024

Abstract

This study aims to develop online communities through the MyNewsScan news aggregator platform to address the problem of information filter bubbles and the spread of fake news. The project focuses on fixing previous bugs, adding new site features, strengthening user engagement mechanisms, building a community-driven content ecosystem, and ensuring sustainability. By introducing a reward system, enhancing user interaction on the platform and building a strong moderation system, it aims to enhance the diversity and authenticity of news content, thereby promoting democratic values and critical thinking. In technical realisation, the dissertation discusses how to improve the performance and security of the website through the application of new technology and the technical reconstruction of the old code and system design from the perspective of back-end. Moreover, the project achieves sustainability goals by referring to design patterns, using cloud platforms, container technologies such as Docker, and maintaining a good documentation. By achieving these goals, the project is expected to have a positive impact on the way the public accesses information and promote the greater social good.

Research Ethics Approval

This project obtained approval from the Informatics Research Ethics committee.
Ethics application number: 135752
Date when approval was obtained: 2024-06-05
The participants' information sheet and a consent form are included in the appendix.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Yuhang Tang*)

Acknowledgements

In Edinburgh's embrace, my journey found its start,
A city rich with knowledge, a haven for the mind.
This thesis now complete, a work close to my heart,
With gratitude I pen these lines, my thanks to bind.

First of all, I would like to thank the city of Edinburgh and the University of Edinburgh. Studying for a master's degree and living here for one year is the most wonderful year in my life. It is the first time for me to leave my home and country alone and come to a place so far away to live and study alone. The city embraced me and made me feel like a second home in a place with a completely different language and culture. I've seen a lot, learned a lot and grown a lot here.

To professors wise, who guided me with care,
Your insight, like a lantern, lit my way.
In moments of confusion and despair,
Your words and wisdom brought the light of day.

And then I want to thank my mentor, Dr. Gedi Luksys and Dr. Robin Hill have given me this opportunity to work on this project. For the first time, I independently undertook a complete back-end project including database and operation and maintenance work, which was not a small challenge for me. But they trusted me very much and gave me a lot of freedom to play and design and reconstruct. I learned a lot from this project and made a lot of progress.

To friends beside me, through thick and thin,
Your laughter and your strength were always near.
In libraries long, and through the city's din,
Together we have grown, and conquered fear.

At the same time, I want to thank all the MyNewsScan team for their efforts this summer, especially Yichen Li, who independently undertook all the parts of front-end project, user experience and interface design, and cooperated with me to do a lot of front-end adjustment work.

I also want to thank my friends in Edinburgh, your friendship has been the most precious gift I have received this year. The time I spent with you will be the best memories of my life

To family afar, your love a constant beam,
Though miles apart, your support remained my dream.

Finally, I would like to thank my family, who are far away in China to give me all-round support and ensure that I successfully complete my master's degree.

Last of last, I want to thank myself. Thanks to my own efforts and perseverance in the face of difficulties, I have been able to come this far. I also believe that I will continue to work hard to move towards a brighter future.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

1.1.1 Background

The rapid evolution of the Internet and digital technologies has transformed the way news is produced, distributed, and consumed. This transformation has led to unprecedented access to information, but it has also brought significant disruption to the traditional media sector [13]. This disruption is particularly evident with the emergence of various platforms and aggregators like Google News and Facebook, leading to a notable shift in how consumers engage with news content. Rather than relying solely on individual news sources, consumers are now inclined toward accessing aggregated and personalised news offerings. In the United States, while conventional news outlets struggle to adapt to the digital landscape, news aggregators have become the preferred means of accessing news for most Americans [13].

1.1.2 Problem Statement

The problem of filter bubbles, a term popularised by [18], refers to algorithmic biases in online platforms that create echo chambers, isolating users from contrasting viewpoints. Similarly, the spread of fake news, which gained considerable attention during the 2016 U.S. Presidential elections, represents a critical threat to informed public discourse [1]. Both phenomena undermine the democratic process, manipulate public opinion, and exacerbate polarisation. Some argue that personalised algorithms used by social networks and aggregators to select what to show individual users can decrease the diversity of information presented [16]. Concerns have been raised that these algorithms could limit people's exposure to information that offers varying perspectives and shared knowledge [4].

1.1.3 Importance

The importance of tackling these issues cannot be overstated. As societies become increasingly reliant on digital news sources, the need for mechanisms that promote diverse perspectives and factual content becomes paramount. Balancing content customisation with the promotion of diverse viewpoints is a noteworthy topic of discussion. Previous research has indicated that various news consumption can mitigate the effects of filter bubbles and misinformation [3].

1.1.4 Project Description

The project we undertake is to develop a sophisticated platform and real-world environment, the MyNewsScan news aggregation platform, which will allow for a wide range of experiments and analyses that can address these intertwined challenges. The MyNewsScan project, a research project launched by Dr. Hill and Dr. Luksys' team, aims to address this problem. At the beginning, the team developed a prototype website using PHP technology and aggregated a certain number of articles. In this prototype site, they collected feedback and comments from volunteers in different countries after reading articles on it. This feedback is collected through questionnaires and is used to analyse how different attributes of the articles readers read (such as topic, length, and popularity) affect their choices. After the initial phase, the project migrated the site to SpringBoot + Vue technology because of the obsolescence of PHP technology and the inability to find the next developer and maintainer. But for technical reasons, there have been so many bugs in the site that it cannot be put into use. Therefore, the project first needs to make bug fixes so that the website can be used normally. Thereafter, the project aims to enhance existing features and introduce the idea of an online community. This concept encourages users to become not just consumers of content, but also producers of content on the platform. Users can access content and also propose and annotate articles within the site. In addition, the user's behaviour will be spontaneous and guided by the manager to produce a community culture, influence other users in the community, and thus establish a good community ecology. In addition, the project is being used to support Dr. Robin Hill's Eye Movement Lab experiments on monitoring eye movement focus and emotion recognition while browsing web news content.

1.2 Aims and Objectives

The primary aim of the project is to develop a community-driven news aggregator platform back-end that prioritizes user engagement and ease of use, addressing the challenges of content moderation and user interaction. The platform will focus on improving the MyNewsScan system back-end through comprehensive moderation, user feedback incentivised, and robust community features.

To achieve the above goal, the project will pursue the following objectives.

1. **Bugs Fixing**: As mentioned earlier, the project's website has been only a work-in-progress, with a lot of major bugs, and at the beginning of the project almost completely unusable. Therefore, the primary objective is to resolve all urgent bugs and follow the specification in subsequent development to avoid bugs. Developers and other team members also need to continuously test the site and report bugs to ensure the stability and reliability of the site.

2. **Implement Comprehensive Article Moderation**: Develop a complete article moderation system to ensure the health and quality of content. This system will involve the collaborative efforts of human moderators (eventually also platform algorithms) to assess and provide feedback on articles and associated questions. The goal is to create a reliable and credible news source by mitigating the spread of misinformation and promoting diverse viewpoints. Implement effective moderation and content management strategies to maintain a safe, respectful, and constructive environment. Protect users from inappropriate content and harassment, ensuring that the platform remains a trusted space for news consumption and discussion.

3. **Encourage Active User Involvement**: Enhance user engagement through a user point system that incentivizes feedback and participation. Users will be rewarded for providing feedback after reading articles and for submitting articles they've discovered. This system aims to foster an active and involved user community, where feedback is promptly integrated to improve the overall user experience. At the same time, establish a separate discussion board, establish a sound group, posting, and commenting mechanism, to further encourage users to participate in the discussion and express their views.

4. **Develop a Community Forum**: Establish a comment section beneath each article, including associated questions, to promote content-driven discussions.

This section will support hierarchical nesting of comments to facilitate in-depth exploration of specific points without cluttering the discussion space. The aim is to build a community where users can engage in meaningful conversations directly related to the content.

5. **Enhance Usability and Accessibility**: Utilize modern web technologies and user experience design principles to create a platform that meets the diverse needs of the user community. Emphasise usability, accessibility, and user engagement to ensure a seamless and intuitive user experience.

6. **Ensure User Privacy and Website Security**: By updating modern web technology, adding security components, etc., to ensure that the user's personal information is not leaked, to ensure that the user's account security is not illegally infringed, so that users can use the website with confidence.

7. **Standards and Sustainability**: Since this project is currently the dissertation of the taught master's program, all related codes must follow the code specification and develop documentation in order to ensure the sustainability of the project. This ensures that the team that takes over the project can more easily understand the project and carry out subsequent development and maintenance.

By achieving these objectives, the project aims to transform MyNewsScan into a vibrant and engaging community platform that not only aggregates news but also fosters user interaction and trust.

1.3 Completion Status

The project went through two iterations over a full project development cycle of about three months.

1. **First Iteration**:

    - **Bug Fixing**: In the first iteration, the most urgent task is to fix the major bugs left behind before, so that the website can normally let users perform basic functions such as registration, login, submit articles, browse articles.
    - **Reasonable, user-friendly moderation system**: The original moderation system prototype was improved to stage moderation to reduce user burden.

- **System Refactor**: Change the project code to a canonical form as much as possible and upgrade the version of the language, framework, and related dependencies to avoid exposing vulnerabilities in the project.
- **Standards and Sustainability**: A part of the code documentation was written. And the project's deployment was changed to Docker Compose to simplify deployment [21].

2. **Second Iteration**:

    - **Usability and Accessibility**: Continuously optimize various instances of web logic to enhance ease of use.
    - **Community Forum**: New development of a user community forum including group, post, comment, like and other functions.
    - **Security**: Ensure the security of your website by deploying Spring Security, JSON Web Token (JWT), Cross-origine resource sharing configuration, and other technologies, and through periodic backup to ensure that user's data will not be lost [5, 7, 9].

Overall, the project has made great progress in this phase of development, making it from an unusable standstill to a viable restart, and adding a lot of new features to bring it closer to a mature web system.

In addition, there are still some issues that need to be optimized by the next developer, such as the stability of user-submitted articles, the richness of community features, and more advanced security.

Chapter 2

Literature Review

2.1 Background

The phenomena of filter bubbles and misinformation have received significant academic attention in recent years. The concept of ”filter bubbles,” introduced by Pariser and Eli [18], underscores the risk that personalised algorithms pose by potentially isolating users from diverse perspectives. This concept has set the stage for a deeper understanding of how algorithmic curation can impact the consumption of news and information.

Allcott et al. [1], provided a detailed examination of the proliferation of fake news, particularly focusing on its implications for democratic processes. Their work highlighted the need for empirical research aimed at developing and testing interventions to counter these challenges. This foundational research has paved the way for subsequent studies exploring the intersection of misinformation and media consumption.

In an empirical study, Bakshy et al. [3], demonstrated that while algorithms contribute to ideological segregation, the choices individuals make in selecting their news sources play an even more crucial role. This insight suggests that platforms that facilitate more user-driven content selection and foster interaction can help mitigate the impacts of filter bubbles and misinformation.

Recent studies have continued to emphasise the critical nature of addressing misinformation, especially in the context of upcoming elections and significant public events. For example, Aslett et al.[2], explored how misinformation thrives in ’information voids,’ situations where the scarcity of reliable information can lead to a higher acceptance of false news. Their research found that even users trying to verify the facts can inadvertently lend credibility to misinformation due to the prevalence of low-quality sources in search engine results [2]. This issue is exacerbated by the rise of generative

AI technologies capable of producing realistic but fabricated multimedia content, posing new challenges for digital news platforms.

The role of community-driven platforms in promoting diverse news consumption has also been examined, particularly within social media contexts. Studies such as those by Gillani et al.[8], highlight the potential of online communities to cultivate a more inclusive and critically engaged news ecosystem.

Moreover, the implementation of reward systems to incentivize user engagement and enhance diversity in news exposure has gained traction. Research by Shao et al. [22], aligns with this approach, advocating for mechanisms that reward user participation in news annotation and discussions. Such strategies resonate with our project's objectives to boost user interaction and ensure a broader range of perspectives within the MyNewsScan platform.

Building on these foundations, our project aims to integrate and expand upon these insights within the MyNewsScan platform. By incorporating community features and reward mechanisms, we aspire to contribute meaningfully to the ongoing discourse on effectively combating filter bubbles and misinformation in the digital era.

2.2 Technologies

2.2.1 Web Development Technology

Since the widespread adoption of the Internet in 1989, web technology has undergone significant evolution. Tim Berners-Lee developed Web 1.0 technology based on HTML (Hypertext Markup Language) in 1989 [29]. During the Web 1.0 era, users were limited to viewing static pages created by programmers, with no ability to interact or modify the content. Building on the foundational global network architecture of Web 1.0, web technology evolved from static to dynamic, marking the advent of Web 2.0.

PHP, a server-side scripting language, became widely used in web development, especially in early Web 2.0 forums [27]. Its open-source nature, ease of learning, and large developer community made it a popular choice. PHP is highly flexible and compatible with various web servers and operating systems [25]. However, PHP faces challenges in handling large volumes of data and is not as secure as other programming languages such as JSP [11].

Java Server Pages (JSP) allows developers to create dynamic web pages by integrating static content with dynamic data processing. JSP facilitates the maintenance and updating of web applications. Furthermore, portability of JSP ensures that it can be used with a variety of web servers and operating systems [11]. The shift to dynamic Web technology enabled interactive database integration, transforming Web systems from static document repositories to interactive, user-centred platforms. This shift paved the way for the development of social networks, blogs, and video-sharing sites that prioritize human interaction [11, 29].

However, JSP has several limitations, such as high complexity and cumbersome code, particularly for intricate pages. Its coupling of dynamic and static resources prevents true separation, necessitating frequent collaboration between front-end and back-end developers, which can decrease efficiency. This is where the Spring framework becomes significant—a more advanced, Java-based framework that has become central to modern Java enterprise development.

Spring is a Java Web Development framework, offers a comprehensive suite of tools and modules for building web applications, including Spring MVC for creating web controllers and views, Spring Boot for simplify configuration of Spring applications, and Spring Security for handling authentication and authorization [15]. Spring's robust and active developer community provides ample resources and support, enhancing its accessibility and usability.

The classic modern Java enterprise development framework is often referred to as SSM, which stands for Spring, Spring MVC, and Mybatis. Mybatis is a database persistence layer framework that simplifies Java database connectivity (JDBC), enabling developers to manage databases in Spring projects using straightforward Java methods or SQL statements with placeholders [15]. Compared to other Database connectivity framework such as Hibernate and JPA, Mybatis offers improved flexibility and database access efficiency. The synergy between Spring MVC and Mybatis enhances system development efficiency, reduces complexity, and strengthens security, stability, and robustness [15, 30].

By integrating Spring, Spring MVC, and Mybatis, developers can build a powerful and efficient web development framework. Spring provides a lightweight and flexible container for object management, while Spring MVC offers a robust and adaptable approach to building web applications. Mybatis complements these technologies by offering a streamlined and effective method for database interaction, which is crucial for many web applications. Collectively, these technologies form a potent and efficient framework for developing modern web applications that are maintainable and extendable [30].

However, the SSM framework has one notable drawback: As projects grow in size and complexity, the configuration can become highly cumbersome and intricate. To address this issue, Spring Boot, a module within the Spring framework, offers rapid application development (RAD) capabilities. Spring Boot simplifies configuration by leveraging aspect-orientated programming and other concepts to facilitate quick access to project information. One of its key advantages for developers is the minimal Spring configuration required to run applications, eliminating the need for mandatory XML configuration files [26].

2.2.2 B/S Architecture and MVC

The B/S (Browser/Server) architecture is a web application design model that divides a system into a browser and a server. This architecture has gained popularity with the rise of the Internet and web-based applications because it allows users to access applications through a web browser without needing to install additional software on their devices[14]. The B/S architecture is a design concept, whereas in practice the SSM framework encompasses the technology used in application development, namely Spring, Spring MVC, and Mybatis, covering the three layers of the B/S architecture[30]:

1. **Presentation Layer (Client: Browser)**: This layer is responsible for displaying the application's user interface and handling user interaction. It is implemented using web front-end technologies that run in the user's web browser. The presentation layer communicates with the application layer by sending user input, requesting data, and receiving updates.

   In the practice of B/S architecture, the presentation layer is handled by Spring MVC (Model-View-Controller), which follows the MVC design pattern to separate the user interface, application logic, and data management concerns[30]. In Spring MVC, the controller handles user input and manages the flow of data between the model (application data) and the view (user interface). Processes incoming HTTP requests, maps them to appropriate application logic methods, and generates the corresponding HTTP responses. Typically, views are developed using JSP, Thymeleaf, FreeMarker, etc., and are responsible for rendering the user interface based on the data provided by the controller [12].

2. **Application Layer (Server Side)**: Also known as the business logic layer, this layer processes user requests, manages data, and enforces application rules and

policies. The application layer uses server-side technologies such as PHP, JSP, and Spring, as mentioned above. It communicates with the presentation layer to receive user input and update the user interface, while also interacting with the data layer to store, retrieve, and manipulate data. This layer handles most of the application's processing and decision-making and coordinates the overall operation of the application [26, 12].

The core Spring framework focuses on the application layer of the B/S architecture. Spring is a comprehensive framework that simplifies the development of Java applications by offering features such as dependency injection, inversion of control, and integration with various data access technologies. Spring's dependency injection feature allows for easy configuration and management of application components, promoting modularity and maintainability. Utilising Spring's features, developers can build applications with a clear separation of concerns, ensuring that business logic remains independent of the presentation and data layers [12].

3. **Data Layer (Server Side)**: This layer is responsible for storing, retrieving, and managing the application's data. It consists of databases, data storage systems, and data access components that enable applications to interact with data. The data layer is usually implemented using a database management system such as MySQL, PostgreSQL, Oracle, or SQL Server [26, 30].

   MyBatis is an open-source lightweight persistence framework that simplifies database access and data management in Java applications. It maps Java objects to SQL statements, enabling developers to work with relational databases without writing complex JDBC code. In the data layer of the B/S architecture, MyBatis is responsible for communicating with the database, executing SQL queries, and mapping the results to Java objects. It provides a flexible and efficient way to interact with the database, allowing developers to focus on the application logic without dealing with the details of access to the underlying data [30].

2.2.3 Separating Front-end and Back-end

In modern development practices, the front-end and the server are typically developed by different programmers. The front end constitutes the view layer in the MVC architecture. Given the need for complex functions such as web design, layout, and user interaction,

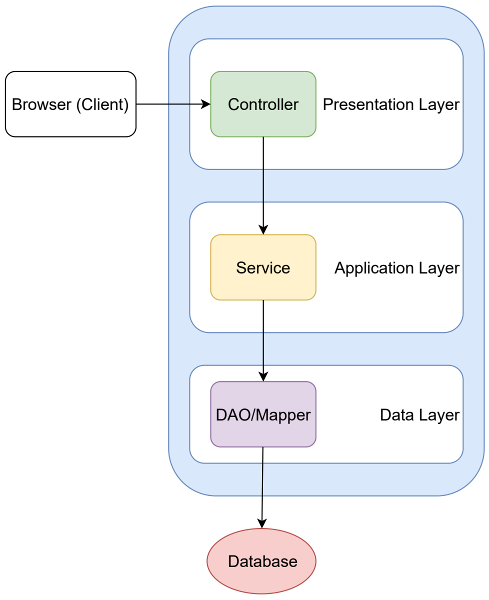


Figure 2.1: B/S Architecture with MVC Model

various frameworks and technologies have emerged, including Vue.js, React, and Angular. Vue.js, a JavaScript framework for building user interfaces, offers a cleaner and more flexible approach to building interactive web applications than other front-end frameworks. It follows a component-based development pattern, enabling developers to split the interface into reusable components[26, 19].

In the SSM framework, Vue.js can replace the view part of Spring MVC, working in conjunction with Spring MVC's controller to handle the user interface and interaction with the back end. Vue.js enables developers to create responsive, high-performance front-end applications and extend their functionality with the many plugins and tools available in the Vue.js ecosystem [26].

Additionally, Spring Boot has been mentioned for server-side development within the SSM framework. Spring Boot simplifies configuration by reducing the need for manual setup. It allows developers to quickly get applications up and running without dealing with extensive XML configuration files and boilerplate code[26, 19]. Spring Boot enables developers to focus on writing business logic without worrying about the complexity of framework configuration.

By integrating these technologies, developers can create modern web applications that are efficient, maintainable, and extendable, leveraging the strengths of each component within the B/S architecture.

2.3 Web Security Technology

Web security is an area that developers must focus on. Web security technologies are critical to protecting applications from a variety of threats, including data breaches, unauthorised access, and cyber attacks.

2.3.1 Secure Socket Layer (SSL) and Transport Layer Security (TLS)

SSL and its successor TLS are cryptographic protocols designed to provide secure communication over computer networks [17]. These protocols use encryption to protect the confidentiality and integrity of data transmitted between the client and the server. SSL/TLS is widely used to secure network traffic and is prefixed with https:// on the URL. SSL/TLS also supports authentication mechanisms that ensure data is sent to and received from trusted parties. Studies have shown that the implementation of SSL / TLS can significantly reduce the risk of data interception and man-in-the-middle attacks [17].

2.3.2 JSON Web Token (JWT)

JWT is an open standard (RFC 7519) for the transfer of information between parties in a compact and self-contained manner [10]. The main features of JWT are the following.

- Compactness: Because JWT is URL-safe and can be sent via URL, POST parameter, or HTTP header, it is suitable for scenarios such as single sign-on (SSO).
- Self-inclusion: JWT contains the necessary information to reduce the dependence on the server database and improve performance.

The JWT usually consists of three parts: the Header, the Payload, and the Signature. The header contains the token type and signature algorithm, the payload contains claims, such as user identity and permissions, and the signature is used to verify the authenticity and integrity of the token. Since JWT uses the HMAC(Hash-based message authentication code) or RSA algorithm for signature, it ensures that the token has not been tampered with [10].

2.3.3 Spring Security

Spring Security is a powerful Java security framework that provides comprehensive security services to protect applications based on the Spring Framework [24]. Its core

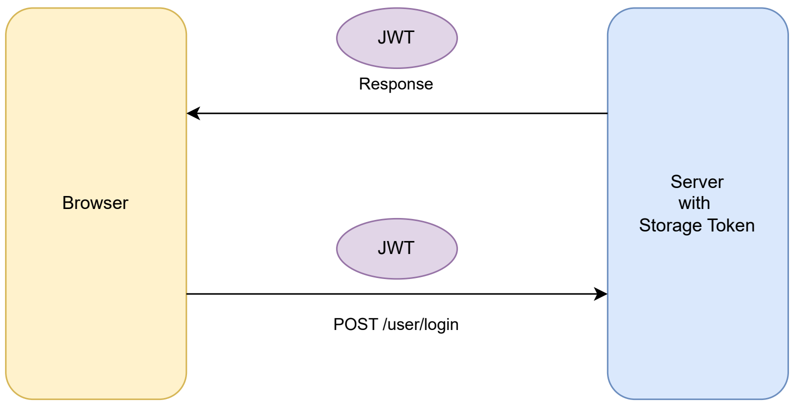


Figure 2.2: JWT Token Work Flow

capabilities include authentication, authorisation, security context management, and protection against common web attacks.

- Authentication: Spring Security supports multiple authentication mechanisms, including form login, HTTP basic authentication, OAuth2 and LDAP(Lightweight Directory Access Protocol). It can be integrated with various user storage (such as database, LDAP, memory) to provide flexible authentication methods [24].
- Authorisation: Spring Security controls user access to application resources by defining security policies. Role-based access control (RBAC) and permission expressions are common authorization methods [24].
- Security context management: The security context contains the security information of the current user, allowing the application to access the user identity and permission information throughout the request process [24].
- Protect against common Web attacks: Spring Security has built-in mechanisms to protect against XSS(Cross Site Scripting), CSRF(Cross-site request forgery), and session fixation attacks. For example, Spring Security automatically generates CSRF tokens and validates them on every request, effectively preventing CSRF attacks [24].

2.3.4 Combining JWT with Spring Security

Combining JWT with Spring Security can enhance the security of Web applications. Spring Security's authentication and authorisation mechanisms can be extended using

JWT to enable stateless distributed authentication. In this mode, after the user logs in, the server generates the JWT and returns it to the client. The client carries this JWT in subsequent requests, and the server identifies the user by verifying the JWT signature without storing session information on the server side. This stateless design improves the scalability and performance of the system [24].

2.4 Java Web Development Specification

In order to ensure the code quality, maintainability and system stability, it is very important to make and follow the development specification. A development specification is a set of standards and best practices that guide developers to write code. Good development specifications can improve the readability, maintainability, and reusability of code, reduce errors, and improve the efficiency of team collaboration [23]. For projects that use Spring Boot for Java backend development, developing and following development specifications is especially important because these frameworks provide a wealth of features and configuration options that require a unified standard to ensure project consistency and reliability [23].

- Project structure: Follow the standard Maven or Gradle project structure, clearly separating different modules and layers, such as Controller, Service, data access layer (Repository), etc. The control layer is used to write code related to API interface data interaction without writing business logic, the data access layer is used to connect with the database to obtain SQL query data, and the service layer is used to write business logic code.

- Dependency management: Use the dependency management provided by Spring Boot and try to use the versions recommended by Spring Boot to avoid version conflicts and compatibility problems. You should also avoid using dependent versions exposed with vulnerabilities.

- Log management: SLF4J and Logback are used to record logs at different log levels (such as INFO, DEBUG, and ERROR) to ensure that important information is documented.

- Exception handling: Unified exception handling, using Spring Boot's @ControllerAdvice annotation set to handle global exceptions and provide a consistent error response format. For security, you must ensure that error exceptions are not returned to the front end.

- RESTful API design: Following the principles of RESTful API design, HTTP verbs (GET, POST, PUT, DELETE) are used to represent different operations, and reasonable status codes are used to return results, ensuring the ease of use and consistency of the API.

Chapter 3

Previous Work and Design

3.1 Previous Work

When the original MyNewsScan project went live in 2018, the site used traditional PHP development methods to integrate and configure MySQL databases as shown in Figure 3.1. Article display, article filtering, user log-in and registration management, article-related survey, reward points according to browsing activities, and other core functions were implemented.

After that, the project underwent an update that included re-sizing the user interface and introducing new features that allowed users to post new articles. Users can submit articles they want to share and attach some questions related to the content to other visitors.

At this stage, although the site has only some basic functions, it can still run normally and stably. Therefore, the MyNewsScan website carried out preliminary development at this stage, accumulated a portion of the user and article database, and carried out several eye movement and emotion monitoring experiments.

However, due to the declining popularity of PHP and the need for newer extensions, the last upgrade rebuilt the code architecture and used a front-end back-end separation project architecture, using SpringBoot as the server-side framework for the project and Vue as the front-end framework.

3.1.1 Problems Remain

However, while the project addressed some of the issues with previous versions of PHP and integrated many new features, there is no denying that the project still had many

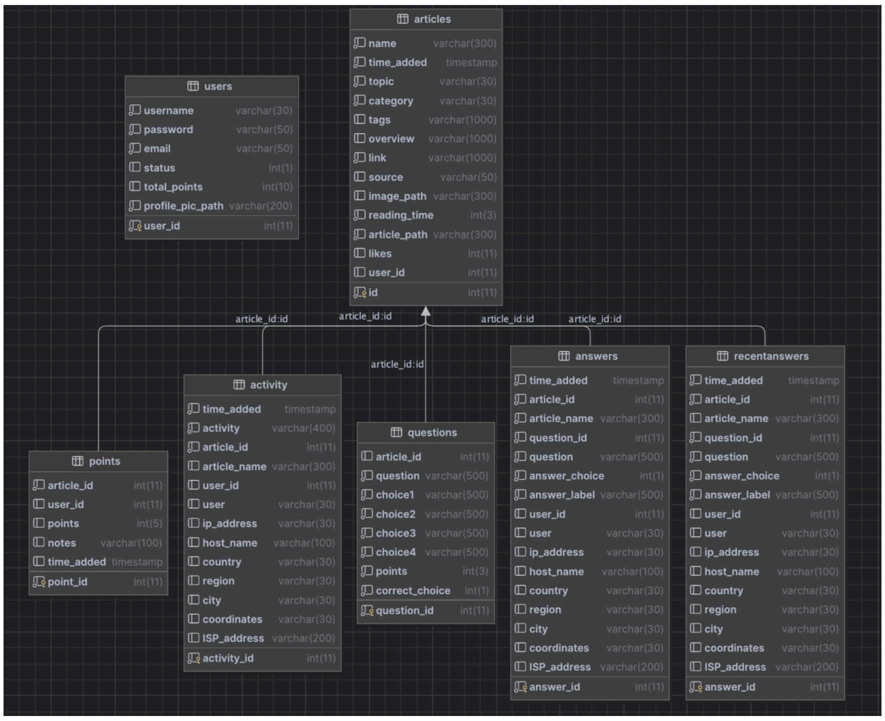


Figure 3.1: Previous database

issues that needed to be addressed urgently. Because the website completely refactored the code, only a small part of the basic functions were completed in the functional section, and a large number of bugs and defects were left behind. Therefore, the main direction of work after that has been to fix bugs and add features to make the site fully functional and working.

1. **Bugs**

   Before the author began to take over this project, more than 30 bugs had been reported by other project staff, and there were critical bugs such as failure to register and login, which caused the website to completely fail to run normally. After the actual operation and functional testing of the project, as well as the careful observation of the project code, we finally confirmed the actual existence of these bugs. Also, the project lacked many features and many features did not work properly, such as comments, avatar uploads, registration verification, and so on. In addition, there was a very serious problem that the project backend would return all user information, including unencrypted passwords, to the browser.

2. **Code Irregularities**

   There were many irregularities in the previous version of project. First, the wrong package reference, the previous programmer referenced the old ’sun.misc’, which caused the project to fail. Second, the project wrote the business logic in the controller layer, which was very irregular and logically confusing. Furthermore, the project referenced very old dependency packages that warned of vulnerabilities in the Maven package management framework.

3. **Lack of development documentation and automated testing**

   When the author first took over the project, there were no handover documents, project descriptions, or code comments to help the author understand the project code. In addition, there were no unit tests or automated tests.

3.2 Design

3.2.1 Project Refactor

In order to improve the maintainability and extensibility of the code, we refactor the project. Specifically, we include the following aspects:

1. **Project structure optimisation**: Follow the standard Maven project structure to separate different modules and layers. Even though this has been done formally in the previous work, the next step is to make the project actually follow this standard. For example, clear separation of control layer (Controller), Service layer (Service), data access layer (Repository), etc., to ensure that the code structure is clear.

2. **Dependency management**: Use the Spring Boot dependency management function to ensure the consistency of dependency packages’ version and avoid version conflicts and compatibility problems. Upgrade dependency packages with security vulnerabilities in a timely manner.

3. **Log management**: Integrates SLF4J and Logback to set different log levels (such as INFO, DEBUG, ERROR) to ensure that important information is recorded.

4. **Exception handling**: Use Spring Boot’s @ControllerAdvice annotation to uniformly handle global exceptions, provide a consistent error response format, and ensure that sensitive error messages are not returned to the front end.

3.2.2 Automated Testing

With limited manpower and energy, we focused on interface testing to ensure that the core functions of the system were functioning properly. The specific steps are as follows:

1. **Test tool selection**: We chose Postman as the primary tool for interface testing. Postman is powerful and easy to use, making it easy to test and debug API interfaces.

2. **Test case design**: For each API interface, we designed detailed test cases, including normal cases, exception cases, and boundary cases. Each test case contains information about the request method, request parameters, expected response, and so on.

3. **Test execution**: Use Postman to execute all designed test cases to verify that the function of the API interface meets the expectations. During testing, emphasis is placed on interface correctness, response time, and stability.

4. **Test report generation**: Through Postman's reporting function, test reports are generated to record the execution results of each test case. Test reports can help us quickly locate problems and provide reference for subsequent development and maintenance.

3.2.3 Documentation

In order to ensure the sustainability and easy maintenance of the project, we design and write detailed development documentation and user instructions. Specific contents include:

1. **Project Overview**: Describes the background, objectives, and main functions of the project.

2. **Environment configuration**: Describes the configuration steps for the development environment and production environment, including the operating system, dependency packages, and database.

3. **Code structure**: Explain the code structure of the project and the function of each module, to help developers quickly understand the project.

4. **API documentation**: Use tools like Swagger to generate API documentation detailing the functionality, parameters, and return values of each interface.

5. **Test description**: Describes how to use Postman to test the interface to ensure the quality of the code.

6. **Deployment Guide**: Provides detailed deployment steps, including Docker Compose configuration, to ensure a project can be successfully deployed to production.

3.2.4 New Functions

Building on the existing MyNewsScan platform, we have added several new features to optimise the user experience and improve the interactivity and usefulness of the platform.

3.2.4.1 Moderation System in Stages

We have improved the review system of articles and adjusted the process of one-time review to phased review. First, after the user submits the article, the system will conduct a preliminary review. Articles that pass the initial review will go to the next stage, where users can upload relevant questions for the article, which will be reviewed by the system. The phased moderation process not only reduces the mental burden of users, but also improves the accuracy and efficiency of the moderation.

3.2.4.2 User Experience Optimisation

We have also made a number of user experience optimisations, including:

1. Notification system: The new notification function, when the user's article or question passed the review, or the discussion group has a new post, will notify the user in time, enhance the user's sense of participation.

2. Email system: When an important message needs to be notified, an email is sent to the user to ensure that the user can receive the information.

3. Points and rewards system: Points and rewards are introduced to encourage users to actively participate in platform activities, such as posting articles, commenting, and participating in discussions. Points can be redeemed for rewards, increasing user engagement and loyalty.

By introducing these new features, we not only enhance the usefulness and engagement of the MyNewsScan platform, but also enhance user interactivity and a sense of community, further driving diversity and authenticity in news.

3.2.4.3 User Community Forum

We have developed a new user forum that allows users to create public and private discussion groups. In discussion groups, users can make posts and discuss articles and other topics. This feature greatly enhances the platform's interactivity, allowing users to more freely share opinions, discuss news, and participate in community events. A public discussion group allows all users to participate, while a private discussion group can set access permissions to allow only certain members to join and discuss, thus meeting the needs of different users.

3.2.5 Enhancing Security

In order to ensure the Security of MyNewsScan platform, we introduced a security mechanism that combined Spring Security and JSON Web Token (JWT) and implemented a role-based access control (RBAC) permission management model.

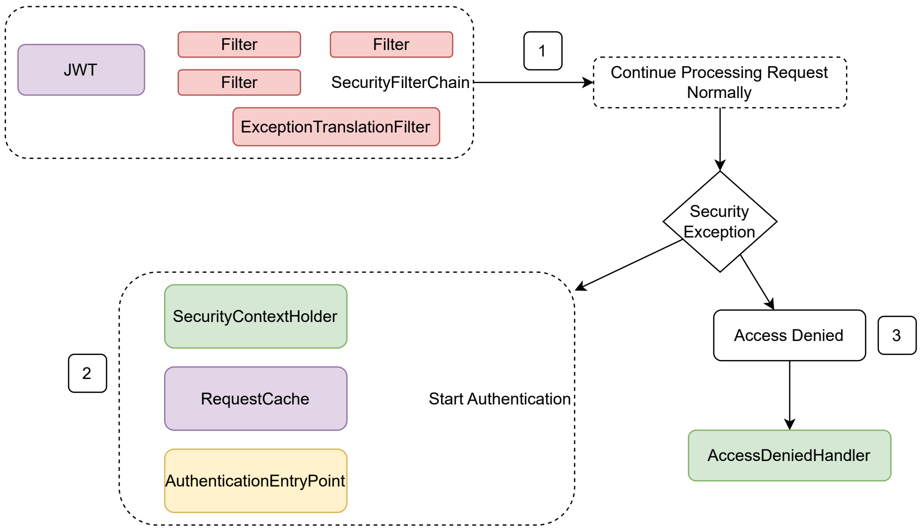


Figure 3.2: Spring Security Work Flow

3.2.5.1 Spring Security and JWT

By integrating Spring Security and JWT, we have implemented a stateless authentication system. After the user logs in successfully, the system generates a JWT and returns it to the client. The client carries the JWT in subsequent requests, and the server identifies the user by verifying the validity and integrity of the JWT, without storing session information on the server, thus improving the scalability and performance of the system.

3.2.5.2 Role-based Access Control (RBAC)

We adopted the RBAC permission management model, assigning different permissions based on the user's role (such as administrator, regular user, moderator, etc.). Each role has specific access and operation rights, ensuring that users can access and operate only the resources and functions allowed by their role. Through detailed permission control, we effectively protect user data and platform resources, prevent unauthorized access and operation, and enhance the security of the system.

Through these security measures, MyNewsScan platform not only ensures the security and privacy of user information, but also improves the reliability and anti-attack capability of the overall system, providing users with a safe and trusted news aggregation platform.

Chapter 4

Implementation

4.1 Project Refactor

First of all, the project needs to be reconstructed to better carry out subsequent development and bug repair. The content of reconstruction includes the upgrade and adjustment of dependent components, the optimization of project structure, the addition of logs, error handling and so on

4.1.1 Dependencies

Common Vulnerabilities and Exposures (CVE), is a database related to information security, collecting various information security weaknesses and vulnerabilities and giving numbers for public access [6]. The Maven repository will automatically collect CVE vulnerability information for each dependent component and report it to each project that uses the dependency, so it is best to upgrade the dependency that is reported to be vulnerable to a non-vulnerable version when the cost of upgrading the dependent component is not high.

Therefore, the updated main dependencies we used are listed below:

- Java Development Kit: 21
- SpringBoot Framework Dependencies: 3.3.0
- Lombok: 1.18.30
- MySQL: 8.0.33
- JSON Web Token: 0.9.1

- Fastjson2: 2.0.32
- MyBatis plus: 3.5.6

4.1.2 Project Structure

In the case of the previous code, all the logic was written in the controller, which works, but does not conform to MVC conventions and is difficult to maintain. Here's an example:

```java
@RequestMapping("/moderating-article")
public JSONObject moderatingArt(@RequestBody JSONObject obj) {
    int articleId = (int)obj.get("articleId");
    int approved = (int)obj.get("approved");
    int status = (int)obj.get("status");
    int moderator = (int)obj.get("moderator");
    ArrayList<Article> articles = (ArrayList<Article>) articleService.getArticleAllById(articleId);
    int moderating = articleService.moderating(articleId, approved, status, moderator);
    int uid = articles.get(0).getUid();
    if (approved == 1) {
        pointService.newPoints(articleId, uid, 5,
                "Your-article-is-approved.", new Timestamp(System.currentTimeMillis()));
        int currentPoints = userService.getUser(uid).getPoints();
        userService.updatePoints(currentPoints +5, uid);
    }
    JSONObject json = new JSONObject();
    json.put("res", moderating);
    return json;
}
```

Now let the controller handle only the data interaction with the front end, leaving the specific business logic to the service layer.

```java
@PostMapping("/moderating-article")
public JSONObject moderatingArt(@RequestBody JSONObject obj) {
    int articleId = obj.getIntValue("articleId");
    String approvedStatus = obj.getString("approved_status");
    int moderator = obj.getIntValue("moderator");
    return adminService.moderatingArt(articleId, approvedStatus, moderator);
}
```

4.1.3 Log Management

In the current project, all logging is implemented using system printing and there is no way to redirect logs to files, databases or other log processing systems. Worst of all, heavy use of *System.out.println* affects performance because every call is a blocking IO operation.

Now we use the SLF4J logging portal system that comes with the SpringBoot framework and use its default Logback logging.

For example:

```
@Service
public class LoginService {
    private static final Logger logger = LoggerFactory.getLogger(LoginService.class);
    ...
    logger.info(”userDetails:{}”, userDetails);
    logger.info(”jwtToken:{}”, jwtToken);
    logger.error(”User-authentication-failed”, ex);
```

4.1.4 Exception Handling

Since the previous project sent error messages back to the front end, we needed to add unified error handling. In a SpringBoot project, all you need to do is configure it.

```
@RestControllerAdvice(basePackages = ”com.mns.mnsback.controller”)
@ResponseBody
@Slf4j
public class GlobalExceptionHandler {
    @ExceptionHandler(Exception.class)
    public Result exceptionHandler(Exception ex) {
        log.error(ex.getMessage());
        return Result.error(”Internal-server-error.Please-try-again-later-or-contact-the-administrator.”);
    }
}
```

4.2 Deployment

Prior to this project, the deployment approach for this project was the most primitive installation, manually setting up the server environment, installing dependencies, configuring databases and other services, and deploying program code. This approach, while intuitive, is prone to problems. For example, an inconsistent environment configuration between different servers can cause an application to not run properly on some servers. In addition, manual operation steps are numerous, increasing the possibility of error and maintenance costs.

Therefore, the deployment mode of this project is modified into Docker-Compose, that is, Docker container is used to deploy various services and programs, and Docker-

Compose is used to orchestrate services. Docker-compose enables rapid deployment and management of applications and their dependencies by defining simple configuration files.

The following diagram in Figure 4.1 shows the architecture of the project on the server.

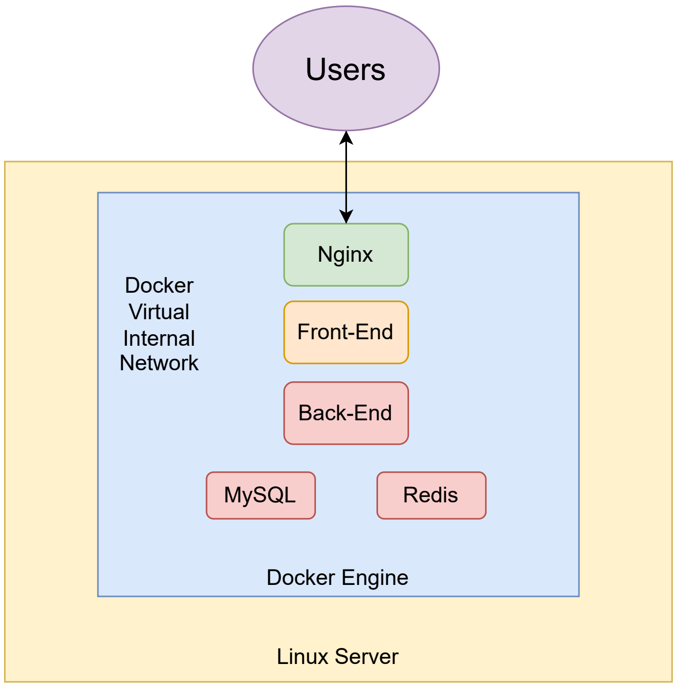


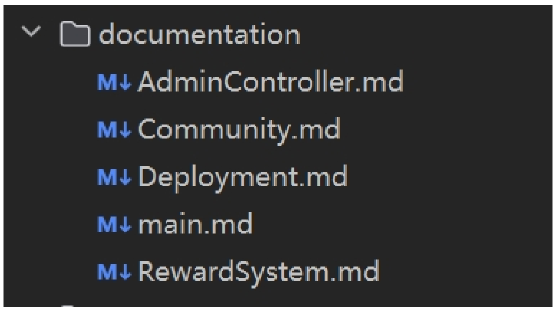
Figure 4.1: Overall Architecture of MyNewsScan Project

For specific operations, a *Dockerfile* needs to be written for both the front-end and back-end to package them into two Docker images and upload them to the Dockerhub repository. Then we only need to add the docker image coordinates to the Docker-compose.yml file in the server, which can be pulled to the server. The specific configuration code is listed in Appendix A1.

Every time this project needs to be deployed an upgrade, there are only three steps in the server: *'docker-compose down'* to close the service, *'docker-compose pull'* to pull new images automatically, and *'docker-compose up -d'* to start the new services.

4.3 Documentation

This project creates a separate folder and several markdown files for documentation. Each document clearly explains how the various parts of the system work.

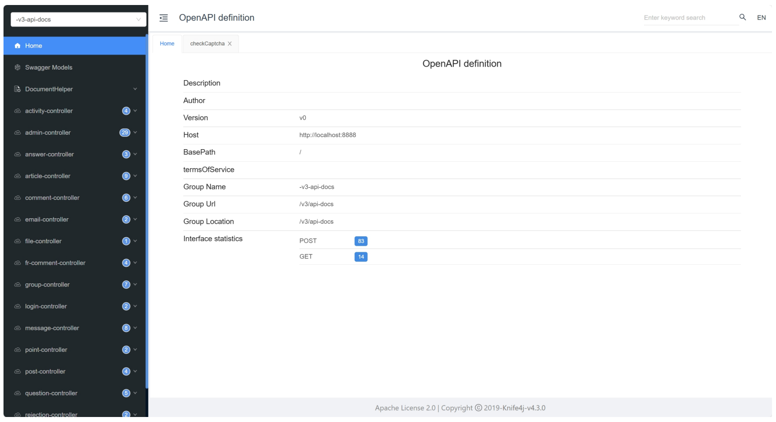


```
documentation
    AdminController.md
    Community.md
    Deployment.md
    main.md
    RewardSystem.md
```

Figure 4.2: Documentation

The *main.md* file describes the system, along with some key information and considerations. The *Deployment.md* writes every step of system deployment in detail, as well as considerations.

In addition, due to the large number of APIs on the system, the Swagger, OpenApi, and Knife4J components are used to automate the production of API documentation. When this component is introduced and configured by the SpringBoot project. The document can be accessed at *doc.html* on port 8888.



OpenAPI definition

Description	
Author	
Version	v0
Host	http://localhost:8888
BasePath	/
termsOfService	
Group Name	-v3-api-docs
Group Url	/v3/api-docs
Group Location	/v3/api-docs
Interface statistics	POST 83
	GET 14

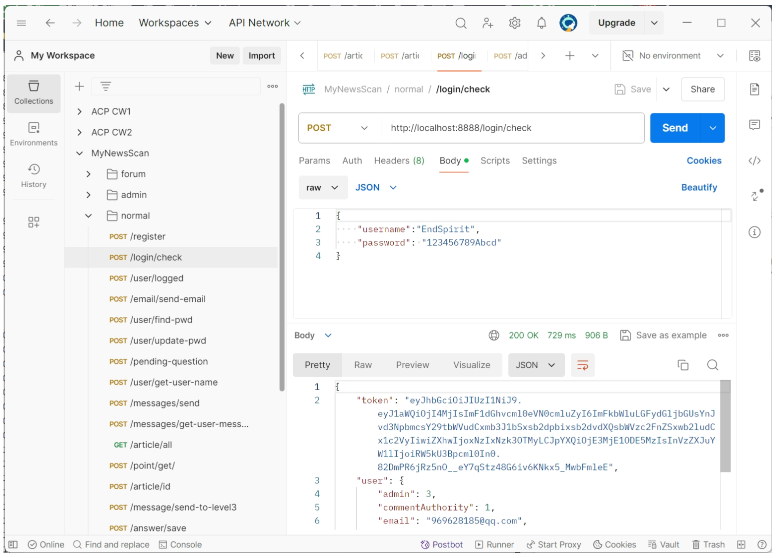
Apache License 2.0 | Copyright © 2019-Knife4j-v4.3.0

Figure 4.3: API Documentation Page

4.4 Automated Testing

Since there are not enough developers and professional test engineers, it is extremely difficult to write unit tests in the limited time available. So we only use Postman for the

API automation testing. Specifically, I created a collection in postman and sorted all apis into different folders in the collection. Best of all, subsequent programmers can test the collection directly with one single sharing operation.



POST http://localhost:8888/login/check

```
{
    "username":"EndSpirit",
    "password": "123456789Abcd"
}
```

Body 200 OK 729 ms 906 B Save as example

```
{
    "token": "eyJhbGciOiJIUzI1NiJ9.
        eyJ1aWQiOjI4MjIsImF1dGhvcml0eVN0cmluZyI6ImFkbWluLGFydGljbGUsYnJ
        vd3NpbmcsY29tbWVudCxmb3J1bSxsb2dpbixsb2dvdXQsbWVzc2FnZSxwb2ludC
        x1c2VyIiwiZXhwIjoxNzIxNzk3OTMyLCJpYXQiOjE3MjE1ODE5MzIsInVzZXJuY
        W1lIjoiRW5kU3Bpcml0In0.
        82DmPR6jRz5nO__eY7qStz48G6iv6KNkx5_MwbFmleE",
    "user": {
        "admin": 3,
        "commentAuthority": 1,
        "email": "969628185@qq.com",
```

Figure 4.4: Postman Interface

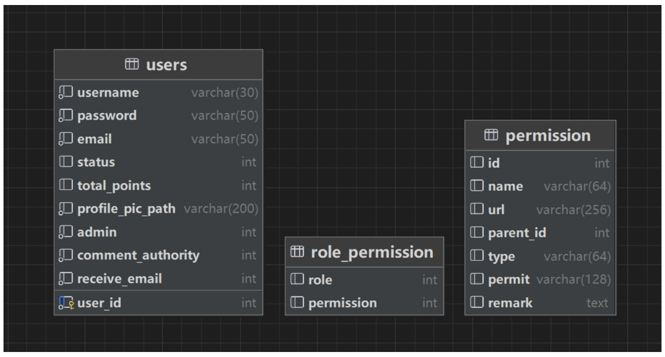
4.5 Security

As mentioned before, security is a very important part of the proper functioning of a web system. Before the project deployed security measures, only a simple JWT was used as a measure to verify the user's identity. And as mentioned earlier, the password is also incorrectly returned to the front end. Such a system is highly insecure. Also, the website is still http but not https, which is not secure for users.

4.5.1 Back-end Security

Therefore, this project adopts a comprehensive approach of Spring Security, combined with JWT authentication, and the RBAC model to manage user permissions.

First, the permission model needed to be reformed. The project previously used the admin field in the user table to specify the levels of level 0, 1, 2, 3. Their permissions range from low to high, with level 0 being normal users who can only submit articles, questions, and comments. Level 1 is a junior administrator who can review articles and issues and manage comment rights; level 2 is a senior administrator who can increase or decrease user rights on the basis of level 1; level 3 is a super administrator who can delete user rights at all levels and change all operations. In order to avoid confusion caused by excessive reconstruction, this field was used to represent roles in the RBAC model. The final tables from the RBAC database are shown in Figure 4.6.



users	
username	varchar(30)
password	varchar(50)
email	varchar(50)
status	int
total_points	int
profile_pic_path	varchar(200)
admin	int
comment_authority	int
receive_email	int
user_id	int

role_permission	
role	int
permission	int

permission	
id	int
name	varchar(64)
url	varchar(256)
parent_id	int
type	varchar(64)
permit	varchar(128)
remark	text

Figure 4.5: Database refactor for RBAC

The next configuration is more complicated; first, we need to replace the original encoder with Spring Security's encoder, implement Spring Security's *UserDetails* class, and finally configure Spring Security's filter and various interceptors.

Here is a simplified example of the *UserDetails* implementation. All the codes below are demo; the specific codes are in Appendix A2.

```
@Service
public class UserLoginDetailsServiceImpl implements UserDetailsService {
    ...
    @Override
    public UserDetails loadUserByUsername(String username){
        User userEntity = userMapper.selectOne(new QueryWrapper<User>().eq(”username”, username));
        List<Permission> authorities = permissionMapper.selectAuthorityByUsername(username);
        StringJoiner stringJoiner = new StringJoiner(”,”, ””, ””);
        authorities.forEach(authority -> stringJoiner.add(authority.getName()));
        return new org.springframework.security.core.userdetails.User(userEntity.getUname(), userEntity.getUpwd(),
                AuthorityUtils.commaSeparatedStringToAuthorityList(stringJoiner.toString())
        );
    }
}
```

The implementation constructs Spring Security's own User type by querying the user table and the Permission table for permissions, so that the Spring Security framework can recognize the user's permissions.

After that, the original login method should be rewrite, the *UserDetails* permission is obtained and written into the *GrantedAuthority* collection, and then loaded into the security context of *SecurityContextHolder*, so that the system can normally identify and distinguish the permissions of the login user.

```
public Object doLogin(String username, String password){
    UsernamePasswordAuthenticationToken auth = new UsernamePasswordAuthenticationToken(username, password);
    Authentication authentication = authenticationManager . authenticate (auth);
    SecurityContextHolder . getContext () . setAuthentication ( authentication );
    UserDetails userDetails = (UserDetails) authentication . getPrincipal ();
    // Get user permission information
    StringJoiner authorityString = new StringJoiner(",", "", "");
    Collection<? extends GrantedAuthority> authorities = userDetails . getAuthorities ();
    for (GrantedAuthority authority : authorities ) {
        authorityString .add( authority . getAuthority ());
    }
    // User authentication successful ,
    String md5Pass = DigestUtils .md5DigestAsHex(password.getBytes());
    User user = userMapper.check(username, md5Pass);
    // JWT token generated
    String jwtToken = JWTUtils.createToken(user . getUid (), user . getUname(), authorityString . toString ());
    return ...;
```

Now that the token has been generated, Spring Security can recognise and parse the token to obtain permission information for the user when the current end requests the token. The next step is to do the overall configuration including filters, encoder, authentication provider, and other detailed configuration.

```
@Configuration
@EnableWebSecurity
public class SecurityConfiguration {
    // Injection here
     ...
    @Bean
    public PasswordEncoder passwordEncoder() {
        return new SecurityEncoder();
    }
    @Bean
    public AuthenticationManager authenticationManager( AuthenticationConfiguration configuration ){
        return configuration . getAuthenticationManager();
    }
    @Bean
    public AuthenticationProvider authenticationProvider () {
        DaoAuthenticationProvider daoAuthenticationProvider = new DaoAuthenticationProvider();
        daoAuthenticationProvider .setPasswordEncoder(passwordEncoder());
        daoAuthenticationProvider . setUserDetailsService ( userDetailsService );
        return daoAuthenticationProvider ;
    }
    @Bean
    public SecurityFilterChain defaultSecurityFilterChain (HttpSecurity httpSecurity ){
        httpSecurity . authorizeHttpRequests ( authorizeHttpRequests -> authorizeHttpRequests
        . requestMatchers ("/admin/**").hasAnyAuthority("admin")
        // set filter rules here
        );
        httpSecurity . authenticationProvider ( authenticationProvider ());
        // Set more configuration here
        httpSecurity .....

        AuthenticationManager authenticationManager = SpringContextUtils.getBean("authenticationManager");
        httpSecurity.addFilterBefore(new JwtAuthenticationFilter(authenticationManager),
            UsernamePasswordAuthenticationFilter.class);

        httpSecurity.exceptionHandling(exceptionHandling -> exceptionHandling
                .accessDeniedHandler(authAccessDeniedHandler)
                .authenticationEntryPoint(authEntryPointHandler)
        );
        return httpSecurity.build();
    }
}
```

4.5.2 HTTPS

In order to ensure the security of the MyNewsScan platform, we needed to upgrade the HTTP protocol to HTTPS protocol to encrypt the data transfer between the user and the server. HTTPS uses the SSL/TLS protocol to effectively prevent data interception and tampering, ensuring user privacy and security.

We needed to obtain the SSL certificate from the server provider and configure it in Nginx. After HTTPS is configured, an error occurred between the front and back ends. Since the back end receives HTTP requests instead of HTTPS, we needed to forward the front-end HTTPS request to the back end through Nginx as an HTTP request. The specific configuration code is shown in Appendix A3.

4.6 New Functions

4.6.1 Moderation System

The previous MyNewsScan platform required users to submit a link to an article and write relative questions all at once. Then, the moderation system will display them for the moderators. However, writing questions has a very large mental burden, in case the articles/news are not approved, the user spent mental writing questions will be wasted. Therefore, the new moderation system was restructured to separate the two stages and divide them into different states.

The detailed logic of the system is as follows and Figure 4.4:

1. To upload an article, the frontend call the */article/add-article* API only by passing the url, and save the article into the database, the article's stauts is

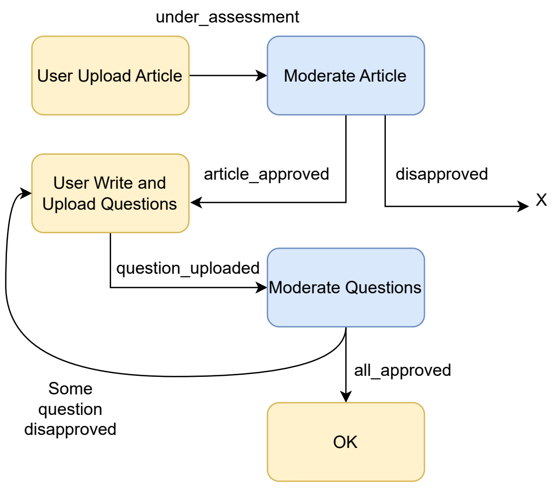


Figure 4.6: Moderation Progress

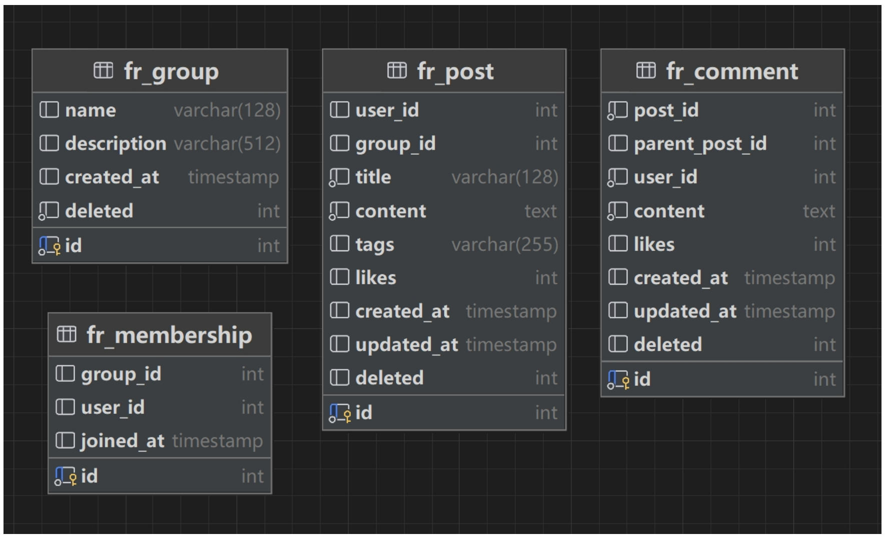
under_assessment.

2. To review an article, call the */admin/pending-article* API to get a list of pending articles.
3. Click Approved and call the */admin/moderating-article* API, status change to *article_approved*. If the article is rejected, the user cannot change the article and the process is closed.
4. After that, the user uploads the question, calls the */question/add-question* API, and saves the question to the database. All questions' status = 0, article's status is *question_uploaded*.
5. To review the question, call the */admin/pending-article* API and */admin/pending-question* API to get a list of pending questions.
6. Click all approved button, call */admin/moderating-question* API, all questions' status = 1, call */admin/moderating-article* API, article status is *all_approved*. Now, the process is over.

During the above reconstruction process, the name of the status field in the article table of the database is reconstructed to *approve_status* and the data structure is reconstructed to *varchar* to more accurately express different moderate states.

4.6.2 User Community Forum

The User Forum is a newly designed module for users to post and discuss customised topics. The forum database is as Figure 4.7 shows.



fr_group	
name	varchar(128)
description	varchar(512)
created_at	timestamp
deleted	int
id	int

fr_post	
user_id	int
group_id	int
title	varchar(128)
content	text
tags	varchar(255)
likes	int
created_at	timestamp
updated_at	timestamp
deleted	int
id	int

fr_comment	
post_id	int
parent_post_id	int
user_id	int
content	text
likes	int
created_at	timestamp
updated_at	timestamp
deleted	int
id	int

fr_membership	
group_id	int
user_id	int
joined_at	timestamp
id	int

Figure 4.7: Forum's Database

The *fr_group* table is designed to store information about groups, and each group is a broad discussion area where users can post discussion posts on similar topics. And groups can be set to private and public to facilitate public discussion and group discussion with specific users. The *fr_membership* table is used to store the user members of a private group separately.

The *fr_post* table is used to store posts posted by users, users can fill in the title, content, tags and other content, and users can like the post. In addition, the table has a creation time and an update time for sorting. The *fr_comment* table is similar to the previous comment table, but because it is a post comment rather than an article comment, it should be distinguished separately.

After the tables of the database are set up, the API that interacts with the data in the front-end also needs to be written. The implementation of these APIs is similar to the previous work, using MyBatis to do Create, Read, Update, and Delete operations in the database, and wrap the data in the front-end.

4.6.3 User experience optimisation

4.6.3.1 Notification System

In order to improve the user experience, when the status of the user's article changes, or points change, or other situations need to be notified, there is no system to notify the previous work. Therefore, a message notification system and an email notification system were developed.

The **message notification system** depends on the newly created tables in the database. Use the *receiver_id* field in the table to send information to the user.

The **mail system** is relatively complex. Spring Boot Mail is a module provided by the Spring Framework for creating mail content, sender, receiver, and other information, and then sending the mail to the specified SMTP server. Simple Mail Transfer Protocol (SMTP) is an email transfer protocol used to transfer emails on a network [20]. The SMTP server is provided by the mail server. When MyNewsScan's server sends mail to the SMTP server, it automatically helps us find and send the mail to the recipient.

4.6.3.2 Points and Reward System

The project incentivises users to become more active in the community by creating points and rewards. The project keeps track of user points earned by maintaining a *points* table, and the *operation* field is used to record how to obtain the points. Also, there is a field *total_point* in the user table to record the total point of the user.

There are the following rules for users to obtain points:

- Open Article: point+2; operation = *open_article*
- Answer Questions: point+7 * each question; operation = *answer_question*
- Article Approved: point+25; operation = *article_approved*
- Article Disapproved: point+5; operation = *article_disapproved*
- Question Approved: point+20 per question; operation = *question_approved*

These rules are meant to encourage users to read articles and answer questions, and specifically encourage users to submit article and write good questions for them. In order not to discourage users, they can earn a small amount of points even if the uploaded articles are rejected,

4.6.3.3 Other Optimisations

In addition to the main optimizations mentioned above, there are many other optimizations carried out in this project.

- In order to help experimenters for user browsing and other data analysis needs, we provide the administrator with the function of data downloading to csv files.
- Articles are displayed in random order and refreshed at a fixed time, to ensure that users get fair information and to break the information filtering bubble. In order to assist the experimental needs of the eye tracking lab, that is, to record the user's eye focus on the news, a new database table was created to record the user name, time, and article order at each refresh of the random order, and we added the form's data to the download page.

Chapter 5

Limitations

Although the goal of the project has been achieved, there are still many limitations that can be improved.

In terms of user experience, there are still many areas that need to be improved on the website, which require front-end and back-end cooperation to improve:

- Tags for read articles are missing, it is recommended to implement a system to tag or filter read articles.
- Article sorting is inconsistent and confusing, it is recommended to maintain a consistent and logical article sorting, or provide the option to sort by date or popularity.
- The login system is not diverse enough and needs to be improved to allow users to log in with other social media accounts, such as Facebook or Google, to reduce the hassle of registering. It is recommended to introduce social sharing options for articles, provide summaries or brief descriptions of articles, add audio versions, and other accessibility features such as font sizing options.
- Lack of mobile side adaptation, currently the project only web page, and no adaptation for mobile browser web page. This may limit user acquisition and subsequent development of the project.

From a backend development perspective, the project still has some areas for improvement:

- Detailed documentation: although the project has written explanatory documentation for each system and automatically generated documentation for each API, there is still a lack of detailed explanation of each interface in the auto-generated documentation, parameter requirements, and return data structure samples.

- Lack of automated unit testing and integration testing. Since there is no professional test engineer, all the tests of this project are carried out by Postman API testing and manual web function testing. This may cause frequent bugs in the system.
- In the absence of concurrent optimization, the only way to optimize user access is to put articles into the memory-based redis database and quickly return them to the user in a cached manner. However, the disadvantage of this approach is that once the number of articles is too large, the server memory may not be enough.
- Lack of disaster recovery means that the current project only runs on a single server, once the IP address of the server is attacked or the server operator has an unexpected problem, there are no means to quickly restore the website.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This project aims to address the problem of information filter bubbles and the spread of fake news by improving and expanding the MyNewsScan news aggregation platform to create a sustainable user community. Through the efforts of the following aspects, I have achieved the main objectives of the project:

1. Fixing vulnerabilities: At the beginning of the project, I focused on fixing a large number of critical vulnerabilities that affected the normal operation of the platform, and restored the basic functions of the website, such as registration, login, article submission and browsing.

2. Improving the Moderation System: I have designed and implemented a phased article moderation system, which effectively reduces the psychological burden of users and improves the accuracy and efficiency of the moderation.

3. User Community Forums: I have developed a new user forum that allows users to create public and private discussion groups, enhancing the interactivity and user engagement of the platform.

4. Security Enhancements: By integrating Spring Security and JWT technologies and adopting a role-based access control (RBAC) model, I have significantly improved the security of our platform and the protection of user data.

5. User Experience Optimization: I have introduced a message notification system, an email system, and a points and rewards system to further enhance user engagement and loyalty.

6. Documentation and Automated Testing: In order to ensure the sustainability and easy maintenance of the project, we wrote detailed development documentation and user guides, and conducted automated interface testing through Postman.

7. Deployment and Maintenance: In addition, I assumed all deployment and maintenance operations and database management responsibilities. Ensure the normal iteration of each release and the special needs of database modifications.

Although we have made significant progress in the project and achieved most of the goals described in the Introduction, we still leave some limitations that need further improvement.

6.2 Future Work

In order to further enhance the functionality and user experience of the MyNewsScan platform, the following are the next steps that we recommend:

Improve the user experience:

1. Implement a more easy to use tagging or filtering system for read articles so that users can more easily manage reading progress.

2. Provide multiple sorting options for articles, such as sorting by date or popularity, to maintain a consistent and logical sorting of articles.

3. Improve the login system to allow users to log in with other social media accounts, such as Facebook or Google, to reduce the hassle of signing up. We can use OAuth2.0 with implemented Spring Security to implement this function [5].

4. Add social sharing options for articles, provide summary or brief descriptions, add audio versions, and other accessibility features, such as font sizing options.

5. Adapt to mobile, develop mobile browser-friendly web pages, expand user coverage. We can use Flutter to develop both IOS and Android apps at the same time [28].

Strengthen background development:

1. Supplement detailed API documentation, including detailed descriptions of each interface, parameter requirements, and examples of returned data structures, to facilitate subsequent development and maintenance.

2. Automated unit testing and integration testing are introduced to reduce frequent bugs in the system and improve system stability and reliability.
3. Optimise concurrent processing and consider caching articles in a memory-based Redis database to quickly return user requests while avoiding the problem of running out of memory.
4. Establish a disaster recovery mechanism, deploy the project on multiple servers, improve the disaster recovery capability of the system, and ensure that the website can be quickly restored when the server encounters problems.

Through the implementation of these follow-up efforts, the MyNewsScan platform will be able to provide a better user experience, further enhance user engagement and loyalty, and ultimately achieve the sustainable development goals of the project. We look forward to future work that will continue to advance the MyNewsScan platform and provide users with a diverse and trusted news aggregation platform.”

Bibliography

[1] Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. 31(2):211–236.

[2] Kevin Aslett, Zeve Sanderson, William Godel, Nathaniel Persily, Jonathan Nagler, and Joshua A. Tucker. Online searches to evaluate misinformation can increase its perceived veracity. 625(7995):548–556. Publisher: Nature Publishing Group.

[3] Eytan Bakshy, Solomon Messing, and Lada A. Adamic. Political science. exposure to ideologically diverse news and opinion on facebook. 348(6239):1130–1132.

[4] Engin Bozdag and Jeroen van den Hoven. Breaking the filter bubble: democracy and design. 17(4):249–265.

[5] Brian Campbell, C. Mortimore, and Michael B. Jones. JSON web token (JWT) bearer token profiles for OAuth 2.0.

[6] Steve Christey and Robert A Martin. Vulnerability type distributions in cve. *Mitre report, May*, 2007.

[7] Marten Deinum, Koen Serneels, Colin Yates, Seth Ladd, and Christophe Vanfleteren. Spring security. In Marten Deinum, Koen Serneels, Colin Yates, Seth Ladd, and Christophe Vanfleteren, editors, *Pro Spring MVC: With Web Flow*, pages 477–533. Apress.

[8] Nabeel Gillani, Ann Yuan, Martin Saveski, Soroush Vosoughi, and Deb Roy. Me, my echo chamber, and i: Introspection on social media polarization. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pages 823–831. International World Wide Web Conferences Steering Committee.

[9] Rajesh Gunasundaram and Randall Goya. CORS essentials : Cross origin resource sharing.

[10] Michael Jones, John Bradley, and Nat Sakimura. Json web token (jwt). Technical report, 2015.

[11] Josh Juneau. Servlets and JavaServer pages. In Josh Juneau, editor, *Jakarta EE Recipes: A Problem-Solution Approach*, pages 1–93. Apress.

[12] Shameer Kunjumohamed, Hamidreza Sattari, Alex Bretet, and Geoffroy Warin. *Spring MVC: Designing real-world web applications*. Packt Publishing Ltd, 2016.

[13] Angela M. Lee and Hsiang Iris Chyi. The rise of online news aggregators: Consumption and competition. 17(1):3–24. Publisher: Routledge _eprint: https://doi.org/10.1080/14241277.2014.997383.

[14] Qin Li, Huibiao Zhu, and Jifeng He. An inconsistency free formalization of b/s architecture. In *31st IEEE Software Engineering Workshop (SEW 2007)*, pages 75–88, 2007.

[15] Dashrath Mane, Namrata Ojha, and Ketaki Chitnis. The spring framework: An open source java platform for developing robust java applications. *International Journal of Innovative Technology and Exploring Engineering*, 3(2), 2013.

[16] Sayooran Nagulendra and Julita Vassileva. Understanding and controlling the filter bubble through interactive visualization: a user study. In *Proceedings of the 25th ACM conference on Hypertext and social media*, HT '14, pages 107–115. Association for Computing Machinery.

[17] Rolf Oppliger. *SSL and TLS: Theory and Practice*. Artech House, 2023.

[18] Eli Pariser. *The Filter Bubble: What the Internet Is Hiding from You*. The Penguin Group.

[19] Jawwad Z Raja, Mehmet Chakkol, Mark Johnson, and Ahmad Beltagui. Organizing for servitization: examining front-and back-end design configurations. *International Journal of Operations & Production Management*, 38(1):249–271, 2018.

[20] Vladimir V Riabov. Smtp (simple mail transfer protocol). *River College*, 2005.

[21] Richard Senington, Balazs Pataki, and Xi Vincent Wang. Using docker for factory system software management: Experience report. 72:659–664.

[22] Lu shao, Mahendar Goli, Abatihun Sewagegn, and Anoop Sahu. Impact of social media usage on civic engagement towards societal problems: Qualitative modelling approach. 2022:1–10.

[23] Sourabh Sharma. *Modern API Development with Spring and Spring Boot: Design highly scalable and maintainable APIs with REST, gRPC, GraphQL, and the reactive paradigm*. Packt Publishing Ltd, 2021.

[24] Laurentiu Spilca. *Spring security in action*. Simon and Schuster, 2020.

[25] Scott Trent, Michiaki Tatsubori, Toyotaro Suzumura, Akihiko Tozawa, and Tamiya Onodera. Performance comparison of PHP and JSP as server-side scripting languages. In Valérie Issarny and Richard Schantz, editors, *Middleware 2008*, pages 164–182. Springer.

[26] Craig Walls. *Spring Boot in action*. Simon and Schuster, 2015.

[27] Luke Welling and Laura Thomson. *PHP and MySQL Web development*. Sams publishing, 2003.

[28] Eric Windmill. *Flutter in action*. Simon and Schuster, 2020.

[29] Komilova Zulxumor Xokimovna. WEB 1.0, WEB 2.0, WEB 3.0 TEXNOLOGIYALARI RIVOJLANISHINING QISQACHA TARIXI. 2(16):196–200. Number: 16.

[30] Dandan Zhang, Zhiqiang Wei, and Yongquan Yang. Research on lightweight mvc framework based on spring mvc and mybatis. In *2013 sixth international symposium on computational intelligence and design*, volume 1, pages 350–353. IEEE, 2013.

Appendix A

Code Appendix

A.1 Dockerfile and Docker-compose

A.1.1 The back-end Dockerfile

```
FROM openjdk:21-jdk

WORKDIR /

ADD target/mns-back-0.0.1-SNAPSHOT.jar app.jar

EXPOSE 8888

ENTRYPOINT ["java", "-jar"]

CMD ["app.jar"]
```

A.1.2 The front-end Dockerfile

```
# Use an official Node.js runtime as a parent image
FROM node:16-alpine

# Set the working directory
WORKDIR /app

# Copy package.json and package-lock.json
COPY package*.json ./

# Install dependencies
RUN npm install

# Copy the rest of the application code
COPY . .
```

```

# Build the application
RUN npm run build

# Use an official Nginx image to serve the application
FROM nginx:stable-alpine

# Copy the built files from the previous stage
COPY --from=0 /app/dist /usr/share/nginx/html

# Expose port 80
EXPOSE 80

# No CMD is needed as nginx image has its own CMD
```

A.1.3 docker-compose.yml

```
version: '3.8'

services:
  mysql:
    image: mysql:8.0
    container_name: mysql
    environment:
      MYSQL_ROOT_PASSWORD: 2000123
      MYSQL_DATABASE: db765814530
    ports:
      - "3306:3306"
    volumes:
      - mysql-data:/var/lib/mysql
      - /home/mns-dump.sql:/docker-entrypoint-initdb.d/init.sql
    networks:
      - app-network

  backend:
    image: yuhang19/mns-backend
    container_name: backend
    ports:
      - "8888:8888"
    environment:
      SPRING_DATASOURCE_URL: jdbc:mysql://mysql:3306/db765814530?serverTimezone=UTC&useUnicode=true&
          characterEncoding=utf8
      SPRING_DATASOURCE_USERNAME: root
      SPRING_DATASOURCE_PASSWORD: 2000123
      SPRING_REDIS_HOST: redis
      SPRING_REDIS_PORT: 6379
#      SPRING_REDIS_PASSWORD: 123456
      SPRING_REDIS_DATABASE: 0
    depends_on:
      - mysql
      - redis
    networks:
      - app-network

  frontend:
    image: yuhang19/mns-frontend
    container_name: frontend
    restart: always
    expose:
      - "80"
    networks:
      - app-network

  nginx:
    image: nginx:latest
    container_name: nginx
    restart: always
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf
      - ./ssl:/etc/nginx/ssl
    depends_on:
      - backend
    networks:
      - app-network

  redis:
    image: redis:alpine
    container_name: redis
    command: ["redis-server", "--bind", "redis", "--port", "6379"]
    restart: always
    ports:
      - "6379:6379"
    networks:
      - app-network

networks:
  app-network:
    driver: bridge

volumes:
  mysql-data:
```

A.2 Spring Security related full code

```
package com.mns.mnsback.config;



```

```java
import com.mns.mnsback.handler.AuthAccessDeniedHandler;
import com.mns.mnsback.handler.AuthEntryPointHandler;
import com.mns.mnsback.handler.JwtAuthenticationFilter;
import com.mns.mnsback.service.security.UserLoginDetailsServiceImpl;
import com.mns.mnsback.utils.SecurityEncoder;
import jakarta.annotation.Resource;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import com.mns.mnsback.utils.SpringContextUtils;
import org.springframework.http.HttpMethod;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.authentication.AuthenticationProvider;
import org.springframework.security.authentication.dao.DaoAuthenticationProvider;
import org.springframework.security.config.annotation.authentication.configuration.AuthenticationConfiguration;
import org.springframework.security.config.annotation.method.configuration.EnableMethodSecurity;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.config.annotation.web.configurers.AbstractHttpConfigurer;
import org.springframework.security.web.SecurityFilterChain;

import org.springframework.security.crypto.password.PasswordEncoder;
import org.springframework.security.web.access.AccessDeniedHandler;
import org.springframework.security.web.authentication.UsernamePasswordAuthenticationFilter;

@Configuration
@EnableWebSecurity
public class SecurityConfiguration {

    @Resource
    private UserLoginDetailsServiceImpl userDetailsService;

    @Resource
    private AuthAccessDeniedHandler authAccessDeniedHandler;

    @Resource
    private AuthEntryPointHandler authEntryPointHandler;

    @Bean
    public PasswordEncoder passwordEncoder() {
        return new SecurityEncoder();
    }

    @Bean
    public AuthenticationManager authenticationManager(AuthenticationConfiguration configuration) throws Exception
        {
        return configuration.getAuthenticationManager();
    }

    @Bean
    public AuthenticationProvider authenticationProvider() {
        DaoAuthenticationProvider daoAuthenticationProvider = new DaoAuthenticationProvider();
        daoAuthenticationProvider.setPasswordEncoder(passwordEncoder());
        daoAuthenticationProvider.setUserDetailsService(userDetailsService);
        return daoAuthenticationProvider;
    }

    @Bean
    public SecurityFilterChain defaultSecurityFilterChain(HttpSecurity httpSecurity) throws Exception {
        httpSecurity.authorizeHttpRequests(authorizeHttpRequests -> authorizeHttpRequests
                        .requestMatchers(HttpMethod.POST, "/login/**").permitAll()
                        .requestMatchers(HttpMethod.POST, "/register/**").permitAll()/
                        .requestMatchers("/admin/**").hasAnyAuthority("admin")
                        .requestMatchers("/activity/browse-history").hasAnyAuthority("user")
                        .requestMatchers("/activity/**").permitAll()
                        .requestMatchers("/article/add-article").hasAnyAuthority("article")
                        .requestMatchers("/article/change-article").hasAnyAuthority("article")
                        .requestMatchers("/article/get-article-order").hasAnyAuthority("admin")
                        .requestMatchers("/article/**").permitAll()
                        .requestMatchers(HttpMethod.GET,"/comment/**").permitAll()
                        .requestMatchers(HttpMethod.POST,"/comment/**").hasAnyAuthority("comment")
                        .requestMatchers("/email/**").permitAll()
                        .requestMatchers("/file/**").hasAnyAuthority("admin")
                        .requestMatchers("/message/**").hasAnyAuthority("message")
                        .requestMatchers("/point/**").hasAnyAuthority("point")
                        .requestMatchers("/question/**").permitAll()
                        .requestMatchers("/reject/get-by-article").permitAll()
                        .requestMatchers("/reject/add").hasAnyAuthority("admin")
                        .requestMatchers("/user/leaders").permitAll()
                        .requestMatchers("/user/**").hasAnyAuthority("user")
                        .requestMatchers("/forum/**").hasAnyAuthority("forum")
                        .anyRequest().permitAll()
//                      .anyRequest().permitAll()
        );
        httpSecurity.authenticationProvider(authenticationProvider());

        httpSecurity.formLogin(AbstractHttpConfigurer::disable);

        httpSecurity.logout(AbstractHttpConfigurer::disable);

        httpSecurity.sessionManagement(AbstractHttpConfigurer::disable);

        httpSecurity.httpBasic(AbstractHttpConfigurer::disable);

        httpSecurity.csrf(AbstractHttpConfigurer::disable);


        AuthenticationManager authenticationManager = SpringContextUtils.getBean("authenticationManager");

        httpSecurity.addFilterBefore(new JwtAuthenticationFilter(authenticationManager),
            UsernamePasswordAuthenticationFilter.class);


        httpSecurity.exceptionHandling(exceptionHandling -> exceptionHandling
                    .accessDeniedHandler(authAccessDeniedHandler)
                    .authenticationEntryPoint(authEntryPointHandler)
        );
```

```java
        return httpSecurity.build();
    }
}
```

```java
package com.mns.mnsback.handler;

import com.fasterxml.jackson.databind.ObjectMapper;
import com.mns.mnsback.domain.Result;
import com.mns.mnsback.service.normal.UserService;
import com.mns.mnsback.service.security.UserLoginDetailsServiceImpl;
import com.mns.mnsback.utils.JWTUtils;
import com.mns.mnsback.utils.SpringContextUtils;
import io.jsonwebtoken.Claims;
import jakarta.servlet.FilterChain;
import jakarta.servlet.ServletException;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.authentication.UsernamePasswordAuthenticationToken;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.web.authentication.www.BasicAuthenticationFilter;
import org.springframework.util.StringUtils;

import java.io.IOException;
import java.util.Arrays;
import java.util.Collections;
import java.util.List;
import java.util.stream.Collectors;

public class JwtAuthenticationFilter extends BasicAuthenticationFilter {

    private static final Logger logger = LoggerFactory.getLogger(JwtAuthenticationFilter.class);

    public JwtAuthenticationFilter(AuthenticationManager authenticationManager) {
        super(authenticationManager);
    }

    @Override
    protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain
            filterChain) throws IOException, ServletException {
        try {
            String jwtToken = null;
            String header = request.getHeader(”Authorization”);
            if (StringUtils.hasText(header) && header.startsWith(”Bearer␣”)) {
                jwtToken = header.substring(7);
            } else {
                jwtToken = header;
            }
            if (! StringUtils .hasLength(jwtToken)) {
                filterChain . doFilter ( request , response);
                return;
            }

            JWTUtils jwtUtils = SpringContextUtils .getBean("JWTUtils");
            if ( jwtUtils == null) {
                throw new RuntimeException();
            }


            Claims claims = jwtUtils .checkToken(jwtToken);
            //logger. info("claims: {}", claims);

            String username = (String) claims. get("username");

            String authorityString = (String) claims. get("authorityString");
            //logger. info("username: {}, authorityString : {}", username, authorityString );

            List<SimpleGrantedAuthority> authorities = Arrays.stream( authorityString . split (","))
                    .map(SimpleGrantedAuthority::new)
                    . collect ( Collectors . toList ());

            Authentication authentication = new UsernamePasswordAuthenticationToken(
                    username, null, authorities
            );

          SecurityContextHolder . getContext (). setAuthentication ( authentication );


        } catch (Exception ex) {
//              response . setCharacterEncoding("utf−8");
//              response . setContentType("application /json; charset=utf−8");
//              String value = new ObjectMapper(). writeValueAsString ( Result . error("User is not logged in! The token
    has expired! Or unknown Token resolution error!"));
//              response . getWriter (). write (value);
//              logger. error("error: {}", ex);
            logger . error ("Token■validation■error:■{}", ex .getMessage());
        }
        filterChain . doFilter ( request , response);
    }
}
```

```
package com.mns.mnsback.handler;

import com.fasterxml. jackson . databind .ObjectMapper;
import com.mns.mnsback.domain.Result;
import jakarta . servlet . http . HttpServletRequest ;
import jakarta . servlet . http .HttpServletResponse;
import org.springframework. security .core. AuthenticationException ;
import org.springframework. security .web. AuthenticationEntryPoint ;
import org.springframework.stereotype.Component;

import java.io.IOException;

@Component
public class AuthEntryPointHandler implements AuthenticationEntryPoint {

    @Override
    public void commence(HttpServletRequest request, HttpServletResponse response, AuthenticationException authException) throws IOException {
        response.setCharacterEncoding(”utf−8”);
        response.setContentType(”application/json;■charset=utf−8”);
        String value = new ObjectMapper().writeValueAsString(Result.error(”No■token!”));
        response.getWriter().write(value);
    }
}
```

```
package com.mns.mnsback.handler;

import com.fasterxml.jackson.databind.ObjectMapper;
import com.mns.mnsback.domain.Result;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;
import org.springframework.security.access.AccessDeniedException;
import org.springframework.security.web.access.AccessDeniedHandler;
import org.springframework.stereotype.Component;

import java.io.IOException;

@Component
public class AuthAccessDeniedHandler implements AccessDeniedHandler {

    @Override
    public void handle(HttpServletRequest request, HttpServletResponse response, AccessDeniedException accessDeniedException) throws IOException {
        response.setCharacterEncoding(”utf−8”);
        response.setContentType(”application/json;■charset=utf−8”);
        String value = new ObjectMapper().writeValueAsString(Result.error(”Lack■of■authority!”));
        response.getWriter().write(value);
    }
}
```

```
package com.mns.mnsback.service.security;


import com.baomidou.mybatisplus.core.conditions.query.QueryWrapper;
import com.mns.mnsback.domain.normal.User;
import com.mns.mnsback.domain.security.Permission;
import com.mns.mnsback.mapper.PermissionMapper;
import com.mns.mnsback.mapper.UserMapper;
import jakarta.annotation.Resource;
import lombok.RequiredArgsConstructor;
import org. slf4j .Logger;
import org.springframework. security .core.GrantedAuthority;
import org.springframework. security .core. authority . AuthorityUtils ;
import org.springframework. security .core. authority .SimpleGrantedAuthority;
import org.springframework. security .core. userdetails . UserDetails ;
import org.springframework. security .core. userdetails . UserDetailsService ;
import org.springframework. security .core. userdetails .UsernameNotFoundException;
import org.springframework. stereotype . Service ;

import java. util . List ;
import java. util .StringJoiner;

@Service
public class UserLoginDetailsServiceImpl implements UserDetailsService {

    private final Logger logger = org. slf4j .LoggerFactory.getLogger(UserLoginDetailsServiceImpl. class );

    @Resource
    private UserMapper userMapper;
    @Resource
    private PermissionMapper permissionMapper;

    @Override
    public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {
        User userEntity = userMapper.selectOne(new QueryWrapper<User>().eq("username", username));
        logger. info ("userEntity:■{}", userEntity );
        List<Permission> authorities = permissionMapper.selectAuthorityByUsername(username);
        StringJoiner stringJoiner = new StringJoiner(",", "", "");

         authorities .forEach( authority -> stringJoiner .add( authority .getName()));
        logger. info (" authorities :■{}", stringJoiner );

        return new org.springframework. security .core. userdetails .User( userEntity .getUname(), userEntity .getUpwd(),
                    AuthorityUtils .commaSeparatedStringToAuthorityList( stringJoiner . toString () )
        );
    }
}
```

```
package com.mns.mnsback.service. security ;

import com.alibaba. fastjson2 .JSONObject;
import com.mns.mnsback.domain.Result;
import com.mns.mnsback.domain.normal.User;
import com.mns.mnsback.mapper.PermissionMapper;
import com.mns.mnsback.mapper.UserMapper;
import com.mns.mnsback.utils.JWTUtils;
import org. slf4j .Logger;
import org. slf4j .LoggerFactory;
import org.springframework.data. redis .core.RedisTemplate;
import org.springframework. security . authentication .UsernamePasswordAuthenticationToken;
import org.springframework. stereotype . Service ;
import org.springframework.util.DigestUtils;
import org.springframework.security.authentication.AuthenticationManager;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.context.SecurityContextHolder;
import org.springframework.security.core.userdetails.UserDetails;


import jakarta.annotation.Resource;

import java.util.Collection;
import java.util.HashMap;
import java.util.Map;
import java.util.StringJoiner;

@Service
public class LoginService {

    private static final Logger logger = LoggerFactory.getLogger(LoginService.class);
    @Resource
    private UserMapper userMapper;

    @Resource
    private RedisTemplate<String,String> redisTemplate;

    @Resource
    private AuthenticationManager authenticationManager;

    public Object doLogin(String username, String password){
        try {
            UsernamePasswordAuthenticationToken auth = new UsernamePasswordAuthenticationToken(username, password
                );
            Authentication authentication = authenticationManager.authenticate(auth);
            SecurityContextHolder.getContext().setAuthentication(authentication);
            UserDetails userDetails = (UserDetails) authentication.getPrincipal();
//              logger.info("userDetails:{}", userDetails);
            // Get user permission information
            StringJoiner authorityString = new StringJoiner(",", "", "");
            Collection<? extends GrantedAuthority> authorities = userDetails.getAuthorities();
            for (GrantedAuthority authority : authorities) {
                authorityString.add(authority.getAuthority());
            }
//              logger.info("authorityString:{}", authorityString);

            // User authentication successful,
            String md5Pass = DigestUtils.md5DigestAsHex(password.getBytes());
            User user = userMapper.check(username, md5Pass);

            // jwt token generated
            String jwtToken = JWTUtils.createToken(user.getUid(), user.getUname(), authorityString.toString());
//              logger.info("jwtToken:{}", jwtToken);

            JSONObject json = new JSONObject();
```

```java
                json.put("token", jwtToken);
                json.put("user", user);
                return json;
            } catch (Exception ex) {
                logger.error("User authentication failed", ex);
                // User authentication failed, return login failure message
                return Result.error("The user name or password is incorrect!");
            }
        }
    }
```

```java
package com.mns.mnsback.domain.security;

import lombok.AllArgsConstructor;
import lombok.NoArgsConstructor;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;

import java.util.Collection;

@NoArgsConstructor
@AllArgsConstructor
public class UserDetailsEntity implements UserDetails {

    private String username;
    private String password;
    private Collection<? extends GrantedAuthority> authorities;

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        return authorities;
    }

    @Override
    public String getPassword() {
        return password;
    }

    @Override
    public String getUsername() {
        return username;
    }

    @Override
    public boolean isAccountNonExpired() {
        return true;
    }

    @Override
    public boolean isAccountNonLocked() {
        return true;
    }

    @Override
    public boolean isCredentialsNonExpired() {
        return true;
    }

    @Override
    public boolean isEnabled() {
        return true;
    }

    @Override
    public String toString() {
        return "UserDetailsEntity{" +
                "username='" + username + '\'' +
                ", password='" + password + '\'' +
                ", authorities=" + authorities +
                '}';
    }
}
```

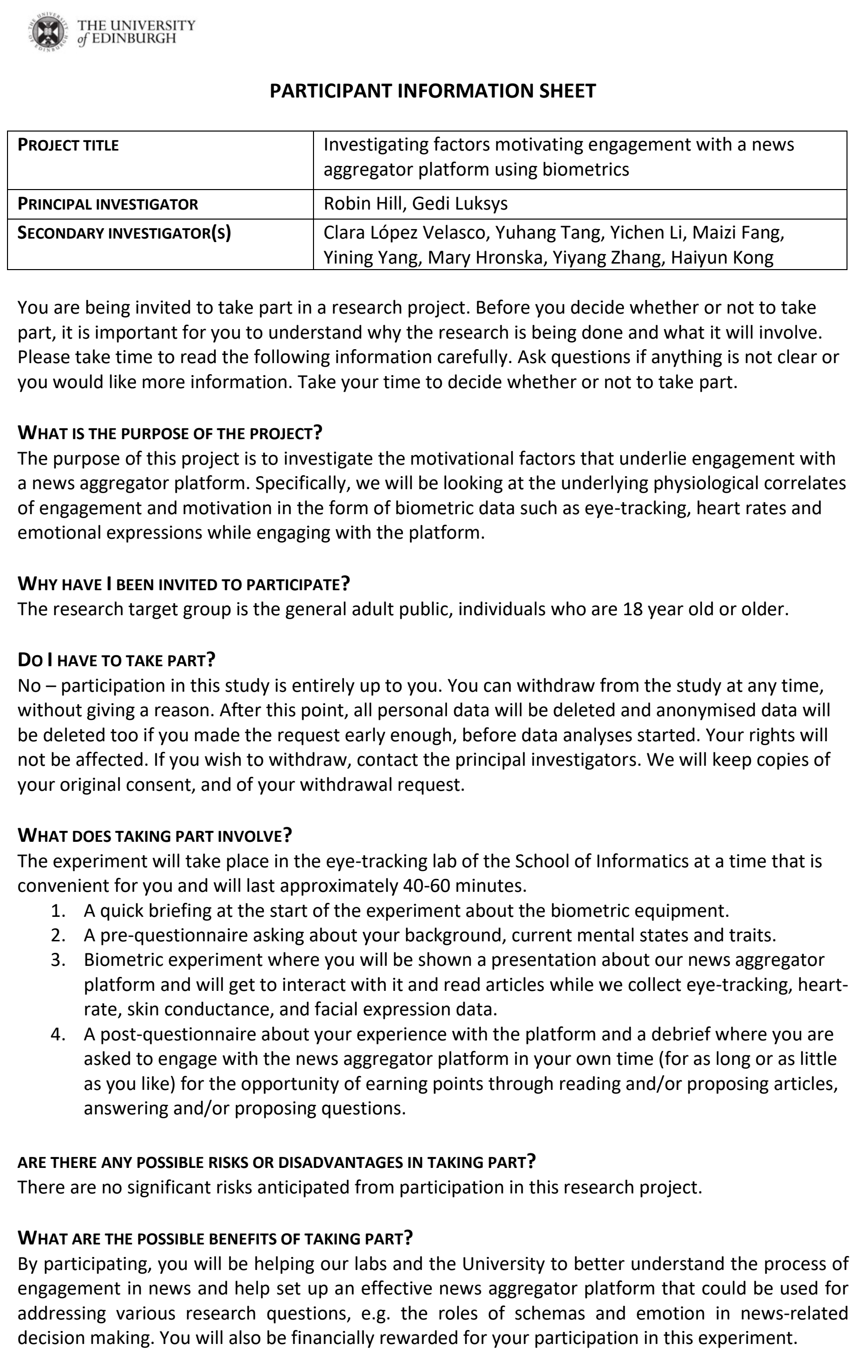
```
package com.mns.mnsback.domain.security;

import com.baomidou.mybatisplus.annotation.TableId;
import com.baomidou.mybatisplus.annotation.TableName;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import java.io.Serializable;

@Data
@AllArgsConstructor
@NoArgsConstructor
@TableName("permission")
public class Permission implements Serializable {
    @TableId
    private Integer id;
    private  String name;
    private String url;
    private String parentId;
    private String type;
    private String permit;
    private String remark;
}
```

A.3 HTTPS

```
events {
    worker_connections 1024;
}

http {
    server {
        listen 80;
        server_name mynewsscan.eu;

        # Redirect HTTP to HTTPS
        location / {
            return 301 https://$host$request_uri;
        }
    }

    server {
        listen 443 ssl;
        server_name mynewsscan.eu;

        ssl_certificate /etc/nginx/ssl/mynewsscan.eu_ssl_certificate.cer;
        ssl_certificate_key /etc/nginx/ssl/_.mynewsscan.eu_private_key.key;
        ssl_trusted_certificate /etc/nginx/ssl/_.mynewsscan.eu_ssl_certificate_INTERMEDIATE.cer;

        ssl_protocols TLSv1.2 TLSv1.3;
        ssl_ciphers HIGH:!aNULL:!MD5;

        location /api/ {
            # Forward all requests to backend
            proxy_pass http://backend:8888/;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
        }

        location / {
            # If you have static files served by frontend, you can set it up here
            proxy_pass http://frontend:80;
            proxy_set_header Host $host;
            proxy_set_header X-Real-IP $remote_addr;
            proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
            proxy_set_header X-Forwarded-Proto $scheme;
            add_header Cache-Control "no-cache, no-store, must-revalidate"; add_header Pragma "no-cache";
                add_header Expires 0;
        }
    }
}
```

Appendix B

Participants’ information sheet



THE UNIVERSITY *of* EDINBURGH

PARTICIPANT INFORMATION SHEET

PROJECT TITLE	Investigating factors motivating engagement with a news aggregator platform using biometrics
PRINCIPAL INVESTIGATOR	Robin Hill, Gedi Luksys
SECONDARY INVESTIGATOR(S)	Clara López Velasco, Yuhang Tang, Yichen Li, Maizi Fang, Yining Yang, Mary Hronska, Yiyang Zhang, Haiyun Kong

You are being invited to take part in a research project. Before you decide whether or not to take part, it is important for you to understand why the research is being done and what it will involve. Please take time to read the following information carefully. Ask questions if anything is not clear or you would like more information. Take your time to decide whether or not to take part.

WHAT IS THE PURPOSE OF THE PROJECT?
The purpose of this project is to investigate the motivational factors that underlie engagement with a news aggregator platform. Specifically, we will be looking at the underlying physiological correlates of engagement and motivation in the form of biometric data such as eye-tracking, heart rates and emotional expressions while engaging with the platform.

WHY HAVE I BEEN INVITED TO PARTICIPATE?
The research target group is the general adult public, individuals who are 18 year old or older.

DO I HAVE TO TAKE PART?
No – participation in this study is entirely up to you. You can withdraw from the study at any time, without giving a reason. After this point, all personal data will be deleted and anonymised data will be deleted too if you made the request early enough, before data analyses started. Your rights will not be affected. If you wish to withdraw, contact the principal investigators. We will keep copies of your original consent, and of your withdrawal request.

WHAT DOES TAKING PART INVOLVE?
The experiment will take place in the eye-tracking lab of the School of Informatics at a time that is convenient for you and will last approximately 40-60 minutes.

1. A quick briefing at the start of the experiment about the biometric equipment.
2. A pre-questionnaire asking about your background, current mental states and traits.
3. Biometric experiment where you will be shown a presentation about our news aggregator platform and will get to interact with it and read articles while we collect eye-tracking, heart-rate, skin conductance, and facial expression data.
4. A post-questionnaire about your experience with the platform and a debrief where you are asked to engage with the news aggregator platform in your own time (for as long or as little as you like) for the opportunity of earning points through reading and/or proposing articles, answering and/or proposing questions.

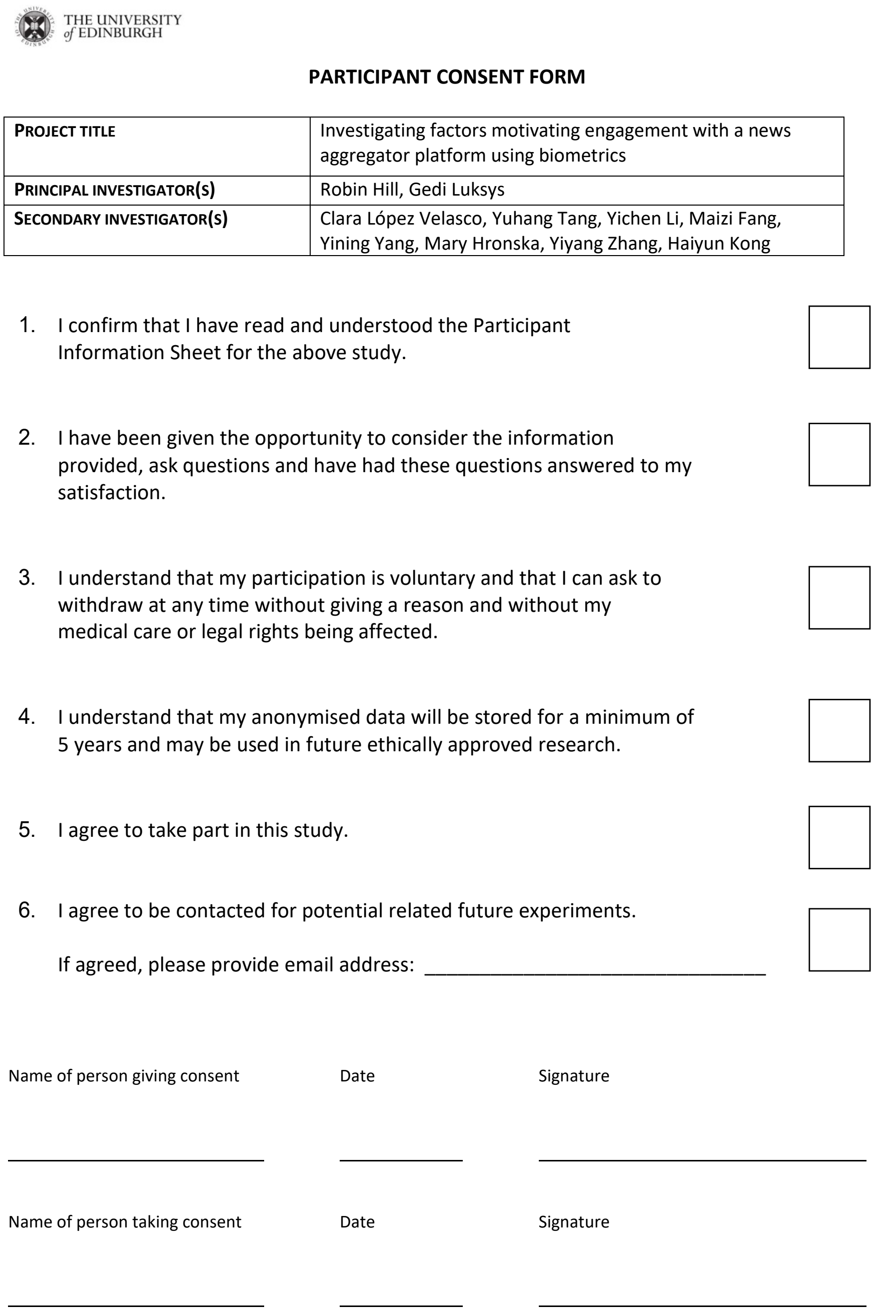
ARE THERE ANY POSSIBLE RISKS OR DISADVANTAGES IN TAKING PART?
There are no significant risks anticipated from participation in this research project.

WHAT ARE THE POSSIBLE BENEFITS OF TAKING PART?
By participating, you will be helping our labs and the University to better understand the process of engagement in news and help set up an effective news aggregator platform that could be used for addressing various research questions, e.g. the roles of schemas and emotion in news-related decision making. You will also be financially rewarded for your participation in this experiment.

Figure B.1: MyNewsScan Participants' information sheet 2024

Appendix C

Participants’ consent form



THE UNIVERSITY *of* EDINBURGH

PARTICIPANT CONSENT FORM

PROJECT TITLE	Investigating factors motivating engagement with a news aggregator platform using biometrics
PRINCIPAL INVESTIGATOR(S)	Robin Hill, Gedi Luksys
SECONDARY INVESTIGATOR(S)	Clara López Velasco, Yuhang Tang, Yichen Li, Maizi Fang, Yining Yang, Mary Hronska, Yiyang Zhang, Haiyun Kong

1. I confirm that I have read and understood the Participant Information Sheet for the above study. ☐
2. I have been given the opportunity to consider the information provided, ask questions and have had these questions answered to my satisfaction. ☐
3. I understand that my participation is voluntary and that I can ask to withdraw at any time without giving a reason and without my medical care or legal rights being affected. ☐
4. I understand that my anonymised data will be stored for a minimum of 5 years and may be used in future ethically approved research. ☐
5. I agree to take part in this study. ☐
6. I agree to be contacted for potential related future experiments. ☐

   If agreed, please provide email address: ________________________________

Name of person giving consent	Date	Signature
________________________	___________	______________________________
Name of person taking consent	Date	Signature
________________________	___________	______________________________

Figure C.1: MyNewsScan Participants' consent form 2024