개인정보 보호기능을 강화한 SNS 웹사이트 개발

이수정*, 류경록*, 이승예O
*인하공업전문대학 컴퓨터정보과
O인하공업전문대학 컴퓨터정보과
e-mail: sjlee@inhatc.ac.kr*, rkr555@naver.com*, seungye1209@naver.comO

Development of an SNS Website with Enhanced Privacy Features

Soojung Lee*, GyeoungRok Ryu*, SeungYe LeeO
*Dept. of Computer Science, Inha Technical Collage,
ODept. of Computer Science, Inha Technical Collage

요 약

본 논문에서는 보안성과 사용자 경험을 최적화하여 환경 보호를 위한 SNS 웹사이트를 개발하였다. 이를 위해 Spring Security와 JWT를 통해 보안성을 강화하고, React를 사용하여 빠르고 반응성이 향상된 인터페이스를 구현하였다. 또한, SNS 로그인과 JPA를 도입하여 사용자 편의성을 높였으며, 독립된 두 개의 서버 아키텍처를 통해 개발 효율성과 보안성을 극대화하였다. 이를 통해 환경 보호 활동을 촉진할 수 있는 효과적인 플랫폼을 제공하였다.

▸ Keyword : 보안 강화(Security Enhancements), 환경 보호(Environmental Protection) ,Spring boot, React
SNS(Social Networking Service)

I. Introduction

현대 사회에서는 환경 보호와 지속 가능한 발전이 중요한 과제로 떠오르고 있다. 제안하는 SNS 플랫폼은 사용자들이 환경 보호 활동을 사진이나 글로 기록하고 공유할 기회를 제공하여, 친환경 활동과 봉사활동을 장려하고 사회적 가치를 실현한다. 이를 통해 사용자 간 상호작용과 지식 공유가 촉진되어 더 많은 사람들이 환경 보호에 동참하고, 지속 가능한 발전에 기여할 것으로 기대된다.

또한 SNS 특성상 개인정보 보호가 중요한 문제로 떠오르고 있으므로, 이를 강화한 SNS 웹사이트 개발의 필요성을 인지하였다. 이에 사용자들이 안심하고 플랫폼을 이용할 수 있도록 본 연구를 진행하였다.

II. Preliminaries

1. Develop Environment

1.1 BCryptPasswordEncoder

암호화된 비밀번호의 저장은 사용자 정보 보호의 핵심적인 부분이다. BCryptPasswordEncoder는 해시 함수를 사용하여 원본 비밀번호를 암호화하며, 이 과정에서 생성된 해시는 원본 데이터를 복원할 수 없다.

사용자의 비밀번호는 BCryptPasswordEncoder를 통해 암호화하여 데이터베이스에 저장하였다. 각 비밀번호에는 고유의 Salt가 추가되어 동일한 비밀번호라도 다른 해시 값을 생성함으로써 보안을 더욱 강화하였다.

1.2 두 개의 서버 아키텍처

프론트엔드(React)와 백엔드(SpringBoot)를 독립된 두 개의 서버로 구성하였다. 프론트엔드 서버는 사용자 인터페이스의 반응성을 향상시켜 사용자 요청을 처리하고, 백엔드 서버는 비즈니스 로직과 데이터베이스 연동을 담당하여 시스템의 효율성을 극대화한다. 이러한 분리된 아키텍처는 개발 작업의 효율성을 높이며, 유지보수와 확장성을 강화한다.

배포 시에는 이러한 두 서버를 하나로 빌드하여 통합 배포함으로써, 운영 환경에서의 일관성과 안정성을 유지하였다.

1.3 SNS 로그인과 JPA 도입

사용자의 편의성을 높이기 위해 SNS 로그인 기능을 도입하였으며, 이를 통해 사용자는 보다 편리하게 접근할 수 있다.

또한 JPA(Java Persistence API)를 사용하여 데이터베이스 작업을 객체 지향적으로 관리할 수 있게 되었고, 이는 데이터 접근과 조작의 복잡성을 줄이며 애플리케이션의 유지보수성과 확장성을 향상시켰다.

2. Requirements Analysis and Design

2.1 작업 분할 구조도

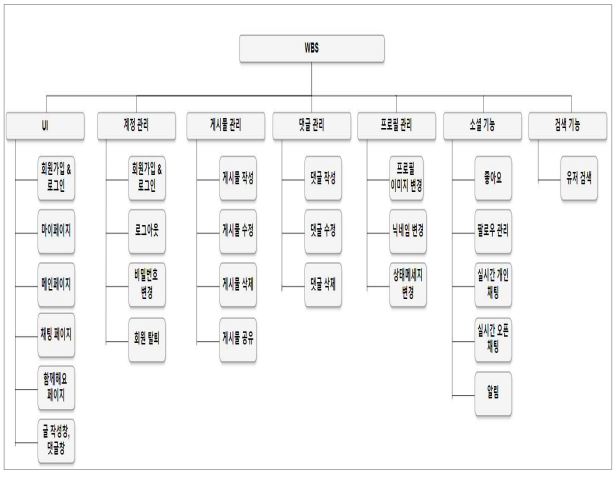

Fig. 1. WBS

SNS 웹사이트 개발을 위한 요구사항 분석 및 정의를 수행하였고, 이를 기반으로 작업 분할 구조도(WBS: Work Breakdown Structure)를 설계하였다.

Fig.1은 UI, 계정 관리, 게시물 관리, 댓글 관리, 프로필 관리 등 다양한 기능을 포함한 상위 카테고리들이 주요 항목으로 구성된 WBS이다. 각 항목은 사용자의 활동에 따라 구체적인 작업 패키지로 세분화되어, 시스템 개발의 체계적인 계획을 가능하게 하였다.

2.2 모델링

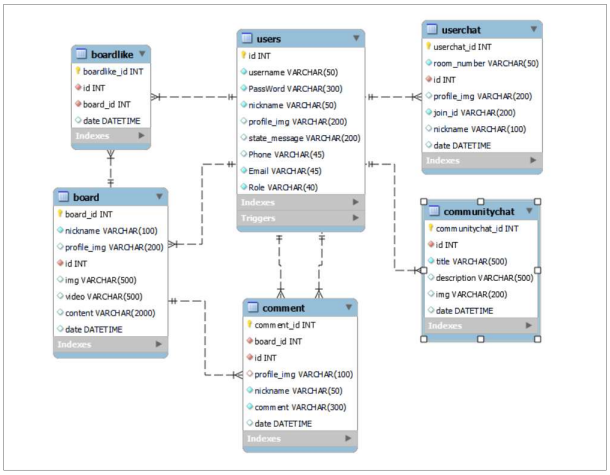

Fig. 2. ERD using MYSQL

Fig. 2는 MySQL 데이터베이스의 엔티티-관계 다이어그램(ERD)을 통해 사용자 데이터의 처리 및 저장 구조를 보여준다. 주요 정보를 저장하는 'users' 테이블은 사용자 ID, 사용자명, 비밀번호 등을 포함하며, 다른 테이블과는 외래키를 통해 연결되어 참조 무결성을 보장한다.

예를 들어, 'boardlike' 테이블은 사용자와 게시물 간의 '좋아요' 상호작용을 관리하며, 데이터의 일관성과 정확성을 유지하는 구조적 설계를 특징으로 한다.

III. The Proposed Scheme

1. SNS 로그인

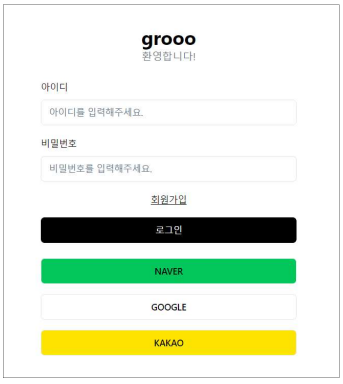

Fig. 3. Login Page

본 시스템은 사용자 편의성을 위해 OAuth 2.0 프로토콜을 활용한 SNS 로그인 기능을 구현하였다. 이를 통해 사용자는 네이버, 구글, 카카오와 같은 주요 소셜 네트워크 서비스 계정을 이용하여 간편하게 로그인할 수 있고, 기존 방식인 아이디와 비밀번호를 이용한 로그인 또한 가능하다.

소셜 인증에 성공할 시 해당 유저의 데이터를 DB에 저장하여 관리한다. 이때 패스워드는 BCryptPasswordEncoder를 적용하여 안전하게 관리한다.

2. Spring Security & JWT

Spring Security와 JWT를 사용하여 Access Token과 Refresh Token을 분리하여 사용한다. Access Token은 짧은 유효 기간을 가지며, 사용자 요청시마다 인증을 위해 사용된다. 반면, Refresh Token은 긴 유효 기간을 가지며, Access Token이 만료되었을 때 새로운 Access Token을 발급받는 데 사용된다.

이러한 구조는 보안성을 강화하고, 사용자의 지속적인 인증 경험을 개선한다. 또한, 토큰의 유효성 검증은 Spring Security 설정에서 JWTFilter를 통해 자동으로 수행되며, 클라이언트에서 오는 모든 요청에 대해 유효성 검증이 이루어진다.

3. 팔로우 및 언팔로우 기능

Fig. 4. Before Follow

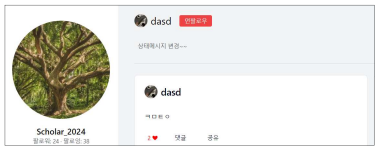

Fig. 5. After Follow

본 시스템에서는 팔로우 및 언팔로우 기능을 제공한다. 사용자가 다른 사용자의 페이지에 접근하면, 상단에 팔로우 버튼이 나타난다. 만약 사용자가 이미 해당 사용자를 팔로우한 상태라면, 이 버튼은 언팔로우 버튼으로 변경된다.

이러한 기능은 React의 useEffect Hook을 사용하여 구현되었으며, 조건문을 통해 사용자 상태에 따라 동적으로 처리된다. 이로써 사용자 인터페이스는 실시간으로 업데이트된다.

4. 채팅 바로가기

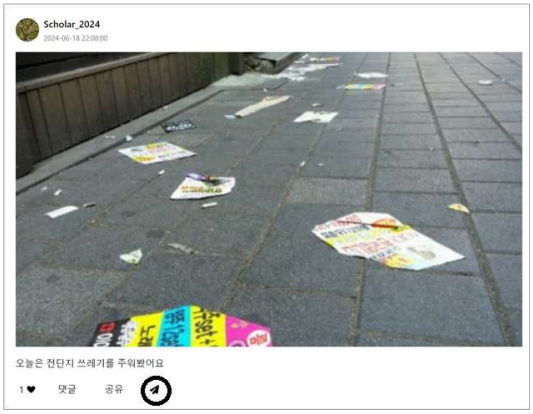

Fig. 6. navigate to the chat room

Fig. 6 하단의 공유 버튼 오른쪽에 위치한 비행기 모양 버튼을 클릭하면 해당 글을 작성한 사용자와의 채팅방이 즉시 표시된다. 이때 해당 사용자와의 채팅 기록이 없는 경우, 새로운 채팅방이 자동으로 생성되고 기존에 채팅 내역이 존재할 경우, 해당 채팅방으로 바로 연결된다. 이 기능은 사용자가 다른 사용자와의 소통을 보다 원활하게 할 수 있도록 설계되었다.

5. 일반 채팅과 오픈 채팅

채팅은 STOMP를 통한 실시간 채팅으로 구현된다. 사용자가 채팅방 목록에서 특정 채팅방을 선택하면, 선택한 채팅방의 ID로 구독(subscribe)한다. 이렇게 구독된 채팅방에서는 해당 방의 모든 메시지를 실시간으로 수신할 수 있다.

Fig. 7. User Chat

일반 채팅은 기존 사용자들 간의 1:1 또는 그룹 채팅을 의미한다. 이 채팅 방식에서는 사용자가 직접 채팅 상대를 초대해야 하며, 초대받지 않은 타인은 채팅방에 접근할 수 없다. 따라서, 일반 채팅은 폐쇄적이고 사적인 대화를 위한 환경을 제공한다.

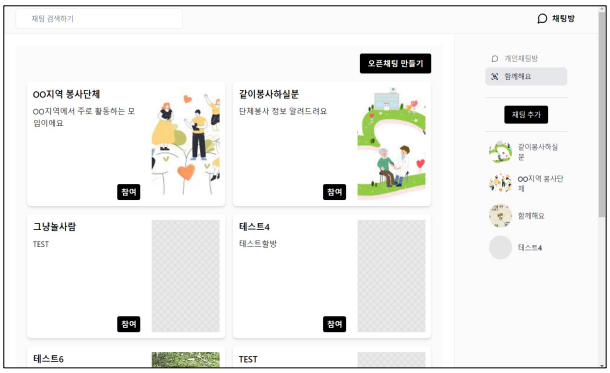

Fig. 8. Community Chat

오픈 채팅은 본 서비스에서 '함께해요'라는 이름으로 제공된다. 사용자가 '함께해요' 버튼을 클릭하면 현재 활성화된 오픈 채팅방의 전체 목록이 표시되며, 사용자는 자신의 의지에 따라 원하는 방에 자유롭게 참여할 수 있다. 오픈 채팅은 다양한 주제에 대해 자유롭게 의견을 교환할 수 있는 개방적이고 참여적인 대화 환경을 제공한다.

6. 알림 기능

Fig. 9

Fig. 10

알림 기능은 SideBar를 렌더링 할 때 useEffect Hook을 이용하여 데이터를 가져오도록 구현되었다. 현재 데이터베이스에서 사용자가 읽지 않은 알림이 존재하는 경우, Fig. 10과 같이 시각적으로 표시된다.

알림은 사용자가 다른 사용자에게서 팔로우를 받았을 때, 사용자의 게시물에 댓글이 달리거나 '좋아요'를 받았을 때 생성된다.

알림 버튼을 클릭하면 Axios를 통한 서버 요청으로 알림 상태를 업데이트한다.

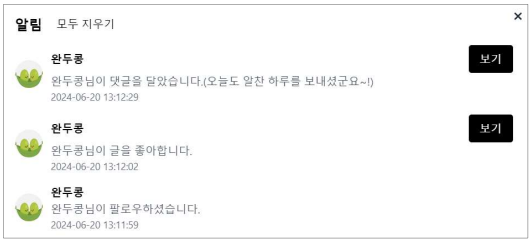

Fig. 11. Detailed notification

Fig. 11의 알림 화면에서 댓글과 좋아요 알림에 해당 게시물을 확인할 수 있는 "보기" 버튼을 제공한다. 이를 클릭하면 알림이 발생한 게시물이 표시되어 내용을 확인할 수 있다. 또한, 알림 화면에서는 모든 알림 내역을 삭제하는 "모두 지우기" 버튼이 제공된다.

7. 이미지업로드

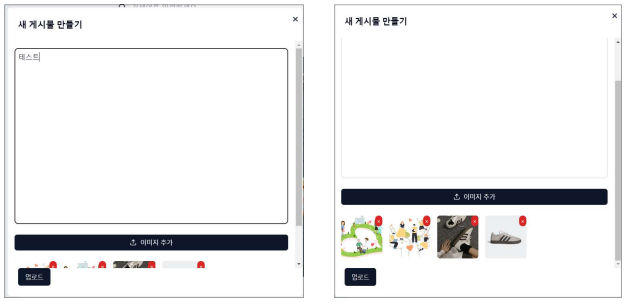

Fig. 12. Writing Page

SideBar(Fig.9)에서 글쓰기 버튼을 클릭하면 Fig.12와 같은 Writing Page가 나타난다. 사용자가 이미지 업로드를 할 경우, 업로드된 이미지는 하단에 미리보기 형태로 표시되며, 드래그 앤 드롭(Drag and Drop) 방식으로 순서를 변경할 수 있다.

이미지 파일의 이름 중복을 방지하기 위해 UUID를 사용하며, 업로드된 이미지는 AWS S3에 저장된다.

Ⅳ. Conclusions

본 연구에서는 SNS를 기반으로 환경 보호와 지속 가능한 발전을 촉진하는 플랫폼을 개발하였으며, 이와 동일한 기능을 제공하는 앱(Application)으로도 개발하여 사용자가 필요에 따라 손쉽게 접근할 수 있도록 하였다.

데이터의 안전성을 위해 저장 과정에서 암호화 기술을 적용해 외부 공격으로부터 사용자 정보를 보호하였다. 이러한 조치는 사용자들이 서비스를 안심하고 이용하게 하며, 환경 문제에 대한 공공의 인식을 높이고 지속 가능한 미래를 위한 사회적 행동을 촉진할 것으로 기대된다.

References

[1] http://en.wikipedia.org/wiki/OAuth

[2] Stomp Protocol Specification, Version 1.2. (2012). Retrieved from Stomp GitHub

[3] Jones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT). RFC 7519. Retrieved from IETF Tools