게시판(질문/답변) 웹 서비스

개발자 안 승진 (2023-11-16)

목차

소스 코드 : https://github.com/sjahnsj/websvc

1. 주요 기능 및 기술 (개발 환경)

- 주요 기능
  - 회원가입
    - 비밀번호는 Bcrypt로 암호화하여 저장
  - 게시판 질문 목록 리스트 보기
    - 페이징 처리
    - 페이지 당 10개의 질문 목록 보기
  - 새 질문 등록 및 삭제
  - 상세 질문 내용 및 답변 보기
  - 답변 등록 및 삭제

- 주요 기술 및 개발 환경
  - Spring boot
    - Spring boot web framework , Tomcat, Gradle
  - Java
    - JDK17
  - View
    - Thymeleaf, HTML, CSS(Bootstrap), Javascript
  - Database
    - H2 database, JPA

2. 개발 일정

- 개발 환경 구축 및 Entity구성 : 2023-11-11
- 게시판 질문 관련 Repository, Service, Controller 구현 : 2023-11-12
- 게시판 답변 관련 Repository, Service, Controller 구현 : 2023-11-13
- 스프링 시큐리티 기반으로 회원가입, 로그인, 로그아웃 구현 : 2023-11-14
- 게시판 질문 및 답변 자료와 회원 로그인 정보 연동 구현 : 2023-11-15

3. 주요 기능 별 구현 사항

- 질문을 올리고 답변을 올리는 Q & A 성격의 게시판
- 주요한 package로는 질문을 담당하는 question, 답변을 담당하는 answer, 사용자 정보와 인증을 담당하는 user package로 나누어져 있음
- 각 package는 MVC중, 주로 Model에 해당하는 부분이 Java로 구현됨

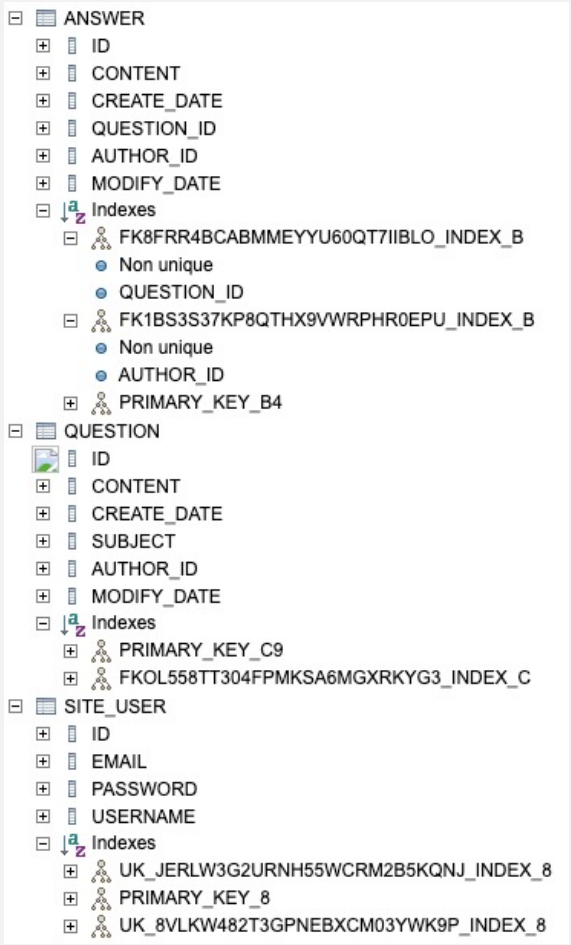
3.1 질문, 답변, 사용자 ENTITY 구성

```java
@Entity
public class Answer {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private Integer id;
  @Column(columnDefinition = "TEXT")
  private String content;
  @CreatedDate
  private LocalDateTime createDate;
  @ManyToOne
  private Question question;
  @ManyToOne
  private SiteUser author;
  private LocalDateTime modifyDate;
}
```

```java
@Entity
public class SiteUser {
  @Id
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private Long id;

  @Column(unique = true)
  private String username;

  private String password;
  @Column(unique = true)
  private String email;
}
```




```java
public class Question {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;
    @Column(length = 200)
    private String subject;
    @Column(columnDefinition = "TEXT")
    private String content;
    private LocalDateTime createDate;
    @OneToMany(mappedBy = "question", cascade = CascadeType.REMOVE)
    private List<Answer> answerList;
    @ManyToOne
    private SiteUser author;
    private LocalDateTime modifyDate;
}
```

3.2 질문데이터 관련 CRUD기능

```java
public Page<Question> getList(int page) {
    List<Sort.Order> sorts = new ArrayList<>();
    sorts.add(Sort.Order.desc(property:"createDate"));
    Pageable pageable = PageRequest.of(page, pageSize:10, Sort.by(sorts));
    return this.questionRepository.findAll(pageable);
}
public Question getQuestion(Integer id) {
    Optional<Question> question = this.questionRepository.findById(id);
    if (question.isPresent()) {
        return question.get();
    } else {
        throw new DataNotFoundException(message:"question not found");
    }
}
public void create(String subject, String content, SiteUser user) {
    Question q = new Question();
    q.setSubject(subject);
    q.setContent(content);
    q.setCreateDate(LocalDateTime.now());
    q.setAuthor(user);
    this.questionRepository.save(q);
}
public void modify(Question question, String subject, String content) {
    question.setSubject(subject);
    question.setContent(content);
    question.setModifyDate(LocalDateTime.now());
    this.questionRepository.save(question);
}
public void delete(Question question) {
    this.questionRepository.delete(question);
}
```

```java
public interface QuestionRepository extends JpaRepository<Question, Integer> {
    Question findBySubject(String subject);

    Question findBySubjectAndContent(String subject, String content);

    List<Question> findBySubjectLike(String subject);
    Page<Question> findAll(Pageable pageable);
}
```

- Question data 접근을 위한 JPA repository 및 service의 CRUD구현
- JPA의 페이지 지원 패키지를 이용하여 페이지 단위로 질문 리스트를 findall 함수로 가져와서 View로 전달

3.3 질문 리스트의 페이지 단위 표시

```
<div th:if="${!paging.isEmpty}">
  <ul class="pagination justify-content-center">
    <li
      class="page-item"
      th:classappend="${!paging.hasPrevious} ? 'disabled'"
    >
      <a class="page-link" th:href="@{|?page=${paging.number-1}|}">
        <span>이전</span>
      </a>
    </li>
    <li
      th:each="page: ${#numbers.sequence(0, paging.totalPages-1)}"
      th:if="${page >= (paging.number / 10)*10 and page <= (paging.number / 10)*10 + 9}"
      th:classappend="${page == paging.number} ? 'active'"
      class="page-item"
    >
      <a
        th:text="${page}"
        class="page-link"
        th:href="@{|/question/list?page=${page}|}"
      ></a>
    </li>
    <li class="page-item" th:classappend="${!paging.hasNext} ? 'disabled'">
      <a class="page-link" th:href="@{|?page=${paging.number+1}|}">
        <span>다음</span>
      </a>
    </li>
  </ul>
</div>
```

- 질문 리스트를 페이지 단위로 Controller로 부터 Model을 이용하여 “paging”속성으로 가져옴
- 질문 리스트 하단에는 현재 보여지는 페이지 번호를 기준으로 10개의 페이지 번호를 선택 할 수 있도록 페이지 번호 링크를 표시

3.4 답변 데이터 관련 CRUD기능

```java
public void create(Question question, String content, SiteUser auther) {
    Answer answer = new Answer();
    answer.setContent(content);
    answer.setCreateDate(LocalDateTime.now());
    answer.setQuestion(question);
    answer.setAuthor(auther);
    this.answerRepository.save(answer);
}

public Answer getAnswer(Integer id) {
    Optional<Answer> answer = this.answerRepository.findById(id);
    if (answer.isPresent()) {
        return answer.get();
    } else {
        throw new DataNotFoundException(message:"answer not found");
    }
}

public void modify(Answer answer, String content) {
    answer.setContent(content);
    answer.setModifyDate(LocalDateTime.now());
    this.answerRepository.save(answer);
}

public void delete(Answer answer) {
    this.answerRepository.delete(answer);
}
```

```java
public interface AnswerRepository extends JpaRepository<Answer, Integer> {
}
```

- Answer data 접근을 위한 JPA repository 및 service의 CRUD구현

3.5 사용자 데이터 관련 CRUD기능

```java
public SiteUser create(String username, String email, String password) {
  SiteUser user = new SiteUser();
  user.setUsername(username);
  user.setEmail(email);
  user.setPassword(passwordEncoder.encode(password));
  this.userRepository.save(user);
  return user;

}
public SiteUser getUser(String username) {
      Optional<SiteUser> siteUser = this.userRepository.findByusername(username);
      if (siteUser.isPresent()) {
          return siteUser.get();
      } else {
          throw new DataNotFoundException(message:"siteuser not found");
      }
  }
```

```java
public interface UserRepository extends JpaRepository<SiteUser, Long> {
  Optional<SiteUser> findByusername(String username);
}
```

- User data 접근을 위한 JPA repository 및 service의 CRUD구현

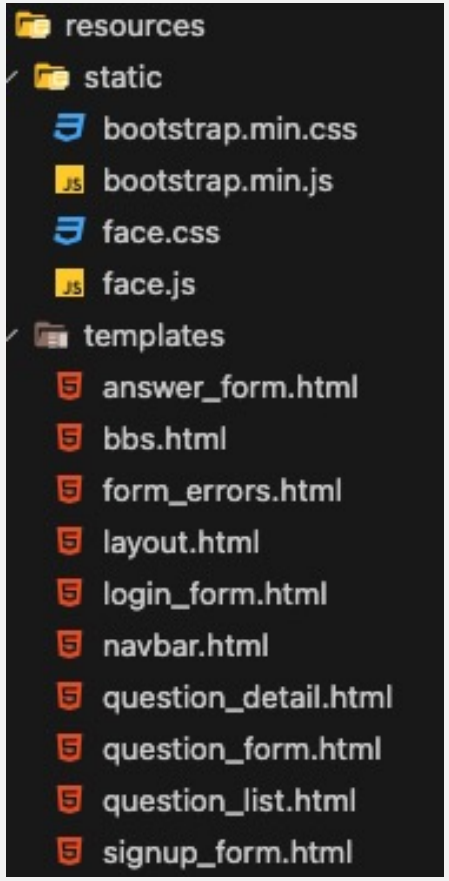
3.6 로그인/로그아웃 및 암호화 설정

```
SecurityFilterChain filterChain(HttpSecurity http) throws Exception {
  http.authorizeHttpRequests((authorizeHttpRequests) -> authorizeHttpRequests
      .requestMatchers(new AntPathRequestMatcher(pattern:"/bbs"),
      new AntPathRequestMatcher(pattern:"/user/login"),
      new AntPathRequestMatcher(pattern:"/user/signup"),
      new AntPathRequestMatcher(pattern:"/*.css"),
      new AntPathRequestMatcher(pattern:"/*.js")).permitAll()
      .requestMatchers(new AntPathRequestMatcher(pattern:"/question/**")).hasRole(UserRole.USER.name()).anyRequest().authenticated())
      .csrf((csrf) -> csrf
          .ignoringRequestMatchers(new AntPathRequestMatcher(pattern:"/h2-console/**")))
      .headers((headers) -> headers
          .addHeaderWriter(new XFrameOptionsHeaderWriter(XFrameOptionsHeaderWriter.XFrameOptionsMode.SAMEORIGIN)))
      .formLogin((formLogin) -> formLogin.loginPage(loginPage:"/user/login").defaultSuccessUrl(defaultSuccessUrl:"/question/list"))
      .logout((logout) -> logout
          .logoutRequestMatcher(new AntPathRequestMatcher(pattern:"/user/logout"))
          .logoutSuccessUrl(logoutSuccessUrl:"/bbs")
          .invalidateHttpSession(invalidateHttpSession:true));
  return http.build();
}
```

```
@Bean
PasswordEncoder passwordEncoder() {
  return new BCryptPasswordEncoder();
}
```

- Security filter 설정 : 각 URL 별로 인증 및 허가 설정
- 비밀번호 암호화를 위해 사용되는 Bcrypt를 Bean으로 설정

4. 주요 소스 트리



- /static : 게시판 HTML style을 설정하는 bootstrap css 화일 및 게시판 홈 화면의 시계 style을 설정하는 css화일
- /template : question, answer, user 관련 사용자 요청 정보를 표시한 HTML

- question, answer, user 데이터 접근 및 사용자 요청을 처리하는 controller와 JPA관련 부분 구현 화일