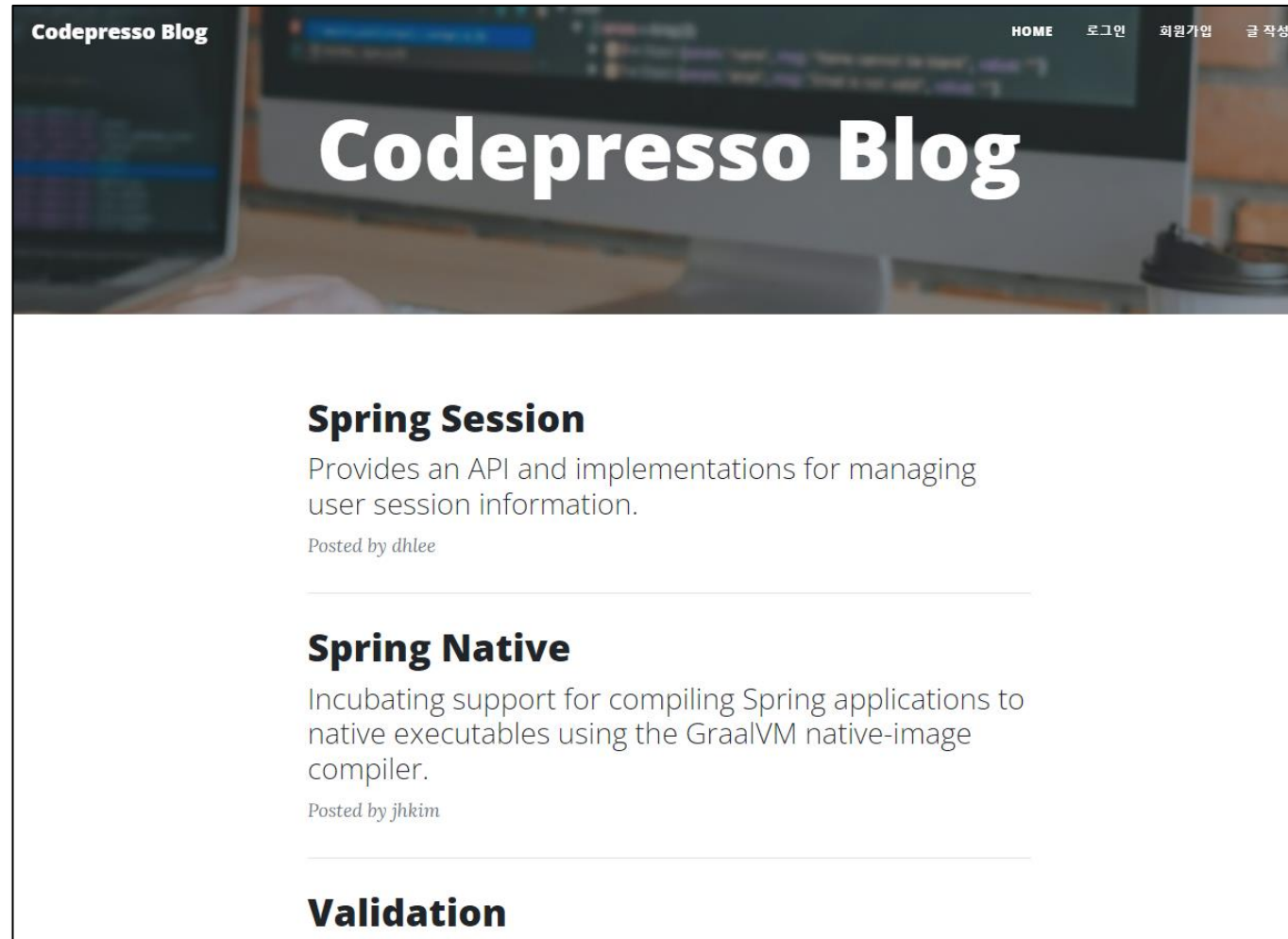


Index 화면의 Blog 글 목록 기능 개발



Index 화면의 Blog 글 목록 기능

- Index 페이지 진입 시 노출 되는 글 목록 기능



Index 화면의 Blog 글 목록 기능

- 모든 글의 목록을 조회
- Mustache를 활용
- 개발 순서
 - MyBatis Mapper 개발
 - Service 개발
 - Controller 개발
 - Mustache 파일 개발

Mapper 개발

com.codepresso.codepressoblog.mapper.
PostMapper.java

```
List<Post> fineAll();
```

resources/mybatis/mapper/
post-mapper.xml

```
<select id="fineAll" resultType="com.codepresso.codepressoblog.vo.Post">
    SELECT *
    FROM post
    ORDER BY id DESC
</select>
```

com.codepresso.codepressoblog.vo.
Post.java

```
public class Post {  
    Integer id;  
    String title;  
    String content;  
    String username;  
    Date createdAt;  
  
    public Post(Integer id, String title, String content, String username) {  
        this.id = id;  
        this.title = title;  
        this.content = content;  
        this.username = username;  
    }  
}
```

Service 개발

com.codepresso.codepressoblog.service.
PostService.java

```
@Service
public class PostService {
    Logger logger = LoggerFactory.getLogger(getClass());

    private PostMapper postMapper;

    public PostService(PostMapper postMapper) {
        this.postMapper = postMapper;
    }

    public List<Post> getAllPost() {
        return postMapper.fineAll();
    }
}
```

Controller 개발

com.codepresso.codepressoblog.controller.
IndexController.java

```
@Controller
public class IndexController {

    private PostService postService;

    public IndexController(PostService postService) {
        this.postService = postService;
    }

    @RequestMapping(value = "/")
    public String index(Model model) {
        List<Post> postList = postService.getAllPost();
        model.addAttribute("posts", postList);
        return "index";
    }
}
```

Mustache 파일 개발

resources/templates/
index.mustache

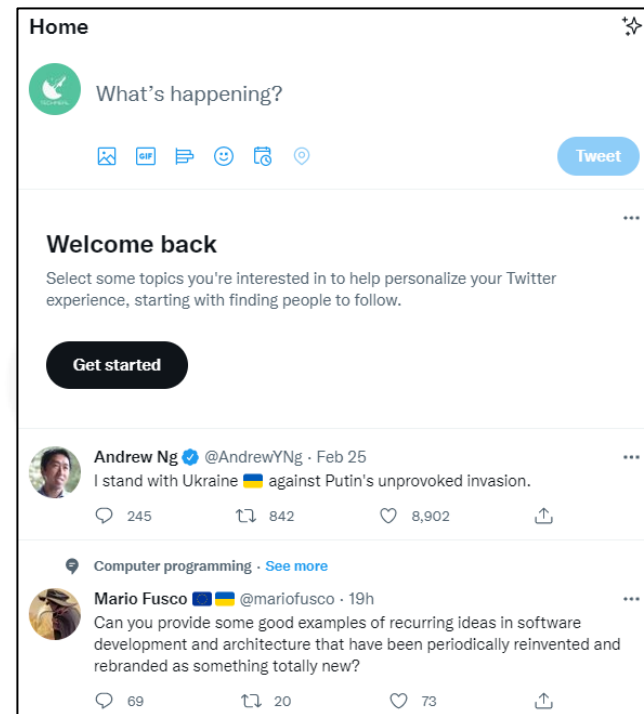
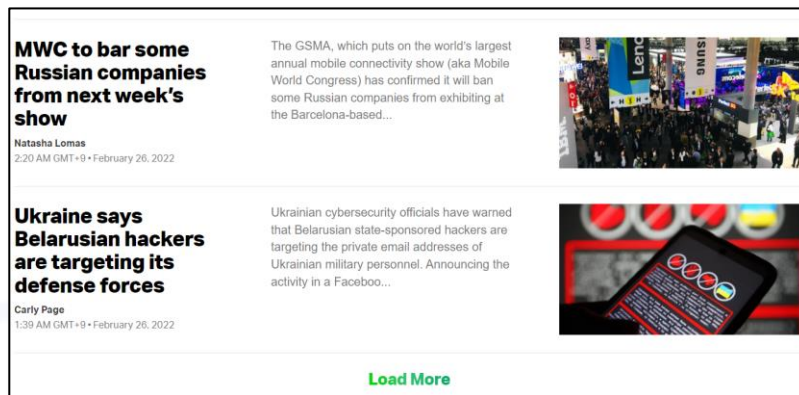
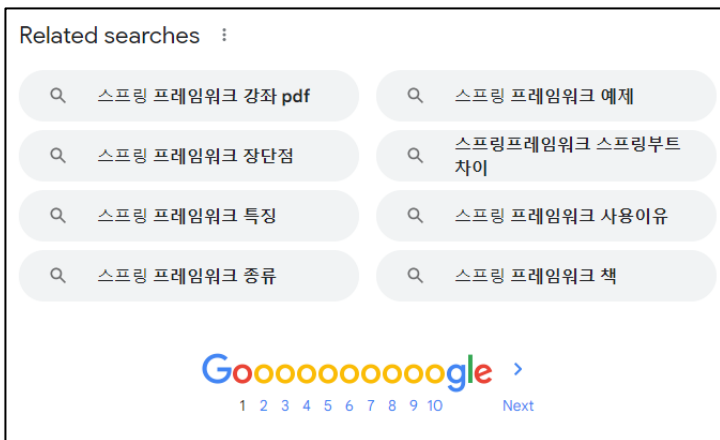
```
<!-- Main Content-->
<div class="container px-4 px-lg-5">
  <div class="row gx-4 gx-lg-5 justify-content-center">
    <div class="col-md-10 col-lg-8 col-xl-7">
      {{#posts}}
        <div class="post-preview">
          <a href="#">
            <h2 class="post-title">{{title}}</h2>
            <h3 class="post-subtitle">{{content}}</h3>
          </a>
          <p class="post-meta">
            Posted by {{username}}
          </p>
        </div>
        <hr class="my-4" />
      {{/posts}}
    </div>
  </div>
</div>
```


Index 화면의 Blog 글 목록 더 보기 기능 개발

Index 화면의 Blog 글 목록 더 보기 기능

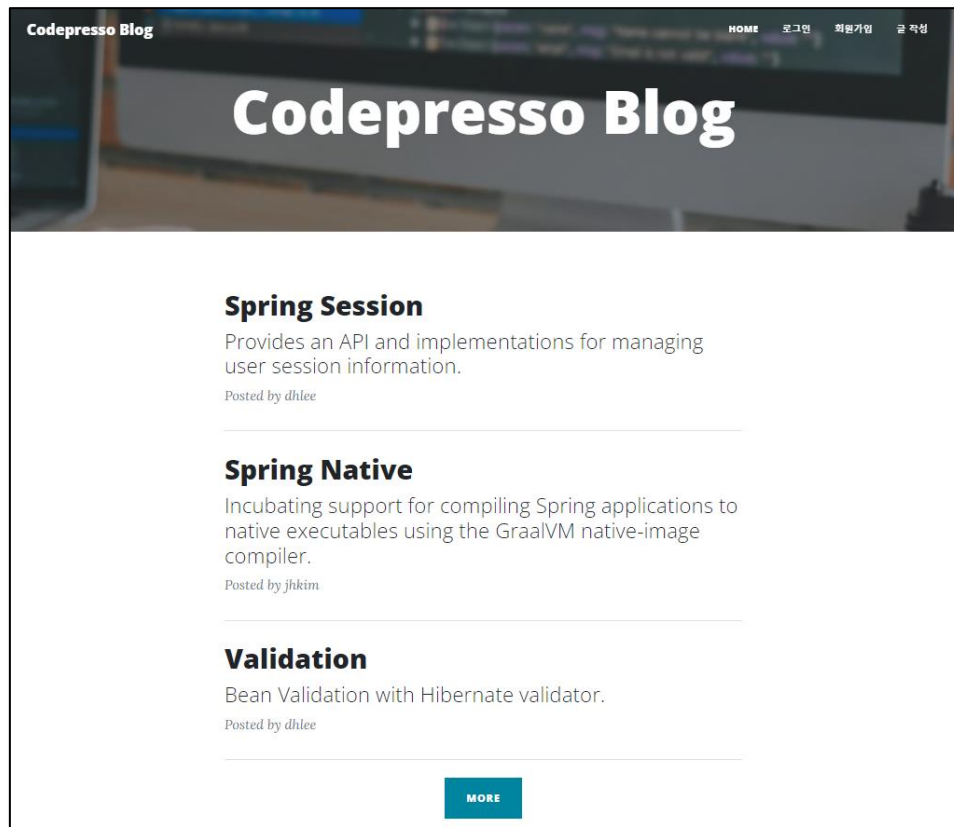
- Blog 글의 개수가 많아지면 한 번에 모든 글을 조회 불가
- Paging(더 보기) 기능으로 정해진 수 만큼의 추가 데이터를 조회
- Paging을 위해 2개의 정보를 활용
 - 조회 할 현재 Page
 - Page 당 조회 데이터의 개수

Paging 구현의 종류



Index 화면의 Blog 글 목록 기능

- Blog Index 진입 시 상위 3개의 글만 조회
- MORE 버튼 클릭 시마다 3개의 추가 데이터 조회



Mapper 개발

com.codepresso.codepressoblog.mapper.
PostMapper.java

```
List<Post> findByPage(@Param("limit") Integer limit, @Param("offset") Integer offset);
```

resources/mybatis/mapper/
post-mapper.xml

```
<select id="findByPage" resultType="com.codepresso.codepressoblog.vo.Post">
    SELECT *
    FROM post
    ORDER BY id DESC
    LIMIT #{limit} OFFSET #{offset}
</select>
```

Service 개발

com.codepresso.codepressoblog.service.
PostService.java

```
public List<Post> getPostByPage(Integer page, Integer size) {  
    return postMapper.findByPage(size, (page-1) * size);  
}
```

limit

offset

page	size	offset	limit
1	3	0	3
2	3	3	3
3	3	6	3
4	3	9	3

Controller 개발

com.codepresso.codepressoblog.controller.
PostController.java

```
@RestController
public class PostController {
    Logger logger = LoggerFactory.getLogger(getClass());

    private PostService postService;

    public PostController(PostService postService) {
        this.postService = postService;
    }
}
```

Controller 개발 – RequestDto, ResponseDto

■ DTO

- Data Transfer Object
- 데이터를 저장하여 다른 곳으로 전송하기 위한 목적의 객체

■ 계층형 아키텍처에서 서로 다른 계층으로 데이터를 전송

■ REST API에서 요청 데이터나 응답 데이터의 전송

Lombok

- 반복적으로 작성해야 하는 java 코드를 자동 생성해주는 라이브러리
 - getter
 - setter
 - constructor
 - toString, equals, ...
 - logging
- Annotation을 사용하여 코드를 생성
 - @Getter, @Setter, @AllArgsConstructor

Lombok 주요 Annotation

- **@NoArgsConstructor**

- 파라미터가 없는 생성자를 생성

- **@AllArgsConstructor**

- 모든 멤버변수를 초기화하는 생성자를 생성

- **@Getter**

- 모든 멤버변수에 대한 Getter 메소드를 생성

- **@Setter**

- 모든 멤버변수에 대한 Setter 메소드를 생성

Lombok 주요 Annotation

■ @ToString

- 모든 멤버변수의 데이터가 출력 될 수 있도록 toString 메소드 자동 생성

■ @NonNull

- 메소드 파라미터의 Null 체크 후 Null 일 경우 NullPointerException 발생

■ @Slf4j

- 로깅을 위한 Logger 객체 자동 생성

Lombok 사용 시 주의할 점

■ @Setter의 @Getter의 남용

- 객체에 대한 캡슐화가 깨질 가능성 존재
- Setter의 경우 객체의 데이터 변경 가능
- Getter의 경우 외부에 공개되면 안되는 데이터가 공개될 수 있음

Controller 개발 – RequestDto

com.codepresso.codepressoblog.controller.dto
PostRequestDto.java

```
@Setter
public class PostRequestDto {
    Integer id;
    String title;
    String content;
    String username;

    public Post getPost() {
        return new Post(this.id, this.title, this.content, this.username);
    }
}
```

Controller 개발 – ResponseDto

com.codepresso.codepressoblog.controller.dto
PostResponseDto.java

```
@Getter
public class PostResponseDto {
    Integer id;
    String title;
    String content;
    String username;

    public PostResponseDto(Post post) {
        this.id = post.getId();
        this.title = post.getTitle();
        this.content = post.getContent();
        this.username = post.getUsername();
    }
}
```

Controller 개발

com.codepresso.codepressoblog.controller
PostController.java

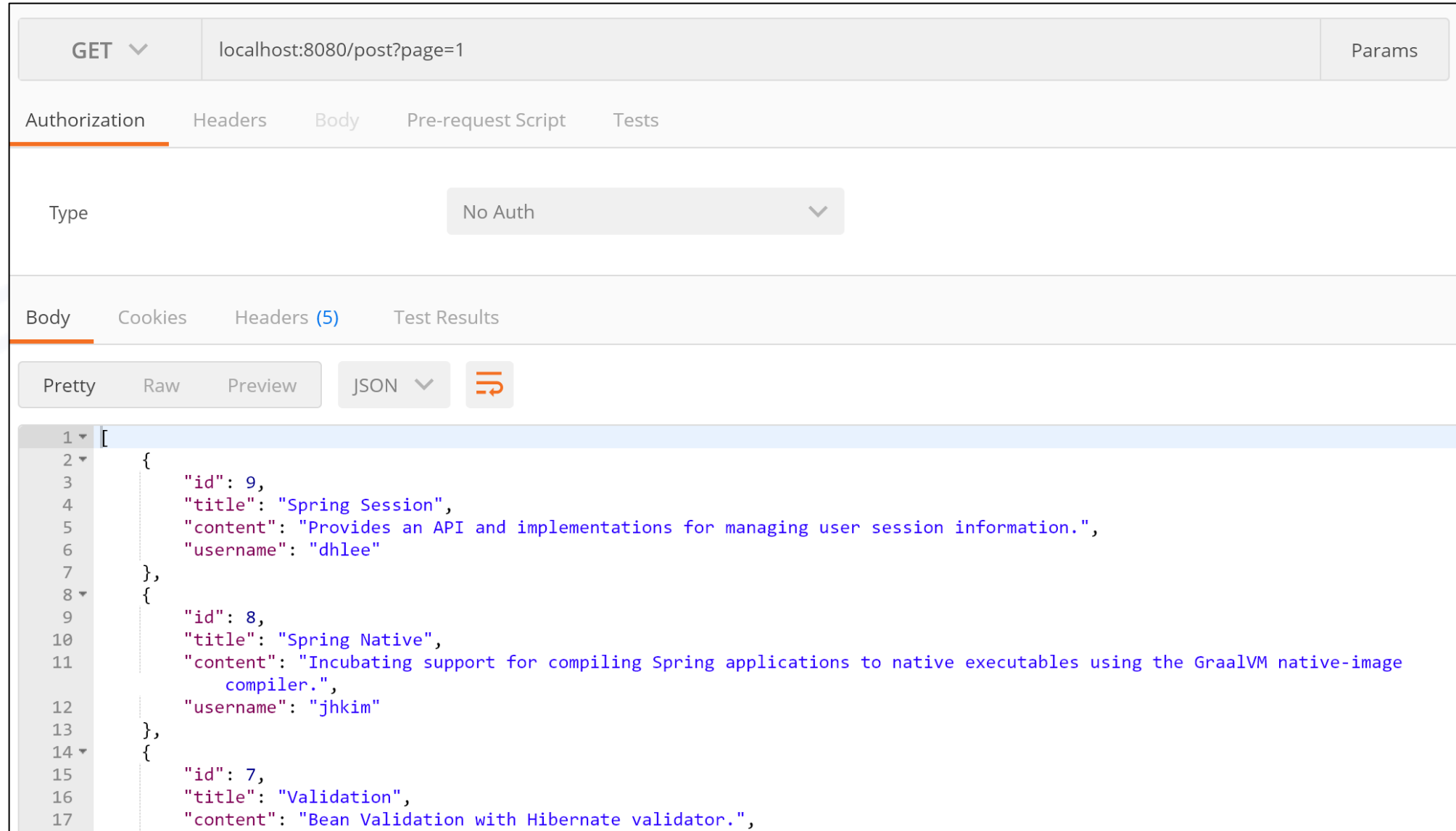
```
@GetMapping("/post")
public List<PostResponseDto> getPostList(@RequestParam Integer page) {
    List<Post> postList = postService.getPostByPage(page, 3);

    List<PostResponseDto> postResponseDtoList = new ArrayList<>();
    for(Post post : postList) {
        postResponseDtoList.add(new PostResponseDto(post));
    }

    return postResponseDtoList;
}
```

```
public PostResponseDto(Post post) {
    this.id = post.getId();
    this.title = post.getTitle();
    this.content = post.getContent();
    this.username = post.getUsername();
}
```

REST API 테스트




GET ▼ localhost:8080/post?page=1 Params

Authorization Headers Body Pre-request Script Tests

Type No Auth ▼

Body Cookies Headers (5) Test Results

Pretty Raw Preview JSON ▼ 

```
1 [
2   {
3     "id": 9,
4     "title": "Spring Session",
5     "content": "Provides an API and implementations for managing user session information.",
6     "username": "dhlee"
7   },
8   {
9     "id": 8,
10    "title": "Spring Native",
11    "content": "Incubating support for compiling Spring applications to native executables using the GraalVM native-image compiler.",
12    "username": "jhkim"
13  },
14  {
15    "id": 7,
16    "title": "Validation",
17    "content": "Bean Validation with Hibernate validator.",
18  }
19 ]
```


Index 화면의 Blog 글 목록 더 보기 기능 개발

Index 화면의 Blog 글 목록 더 보기 기능

- **MORE 버튼 클릭 시 다음 page 데이터 조회**
 - page 별 블로그 글 데이터 조회 API 호출
 - jQuery Ajax 사용
- **Response 데이터를 활용하여 HTML 태그를 append**

IndexController 수정

com.codepresso.codepressoblog.controller.
IndexController.java

```
@RequestMapping(value = "/")  
public String index(Model model) {  
    List<Post> postList = postService.getPostByPage(1, 3);  
    model.addAttribute("posts", postList);  
    return "index";  
}
```

Mustache 화면 개발

resources/templates/
index.mustache

```
<div class="container px-4 px-lg-5">
  <div class="row gx-4 gx-lg-5 justify-content-center">
    <div class="col-md-10 col-lg-8 col-xl-7">
      {{#posts}}
      <div class="post-preview">
        <a href="#">
          <h2 class="post-title">{{title}}</h2>
          <h3 class="post-subtitle">{{content}}</h3>
        </a>
        <p class="post-meta">
          Posted by {{username}}
        </p>
      </div>
      <hr class="my-4" />
      {{/posts}}
      <div id="more-posts"></div>
      <div class="d-flex justify-content-center mb-4">
        <a class="btn btn-primary text-uppercase" id="more" current-page="1">More</a>
      </div>
    </div>
  </div>
</div>
```

추가 데이터를
append하기 위한 tag

클릭 이벤트 처리 위한 id와
현재 page 정보 저장

jQuery Ajax 개발

resources/static/js/
ajax.js

```
$("#more").click(function(){  
    var next_page = parseInt($("#this").attr(name: "current-page")) + 1;  
  
    $.ajax( url: {  
        method: "GET",  
        url: "/post",  
        data: {"page": next_page}  
    })  
})
```

current-page 정보를 가져 와서
문자열을 숫자로 형 변환 후
1을 더함

GET /post?page=1 을 호출

jQuery Ajax 개발

resources/static/js/
ajax.js

```
.done(function(response) {  
    for(var post of response) {  
        $("#more-posts").append("<div class=\"post-preview\">" +  
            "<a href=\"#\">" +  
            "<h2 class=\"post-title\">" +  
            post.title +  
            "</h2>\n" +  
            "<h3 class=\"post-subtitle\">" +  
            post.content +  
            "</h3></a><p class=\"post-meta\">Posted by " +  
            post.username +  
            "</p></div><hr class=\"my-4\" />");  
    }  
});  
$(this).attr( name: "current-page", next_page);  
});
```

응답 받은 글 리스트 요소를
0번째 부터 순회

응답 받은 글 데이터를
사용하여 생성한 HTML을
#more-posts 영역에 append

Spring Session

Provides an API and implementations for managing
user session information.

Posted by dhlee

current-page 값을
next_page 데이터로 변경

Blog 글 상세 보기 기능 개발

Blog 글 상세 보기 기능

- Index의 글 목록에서 특정 글 클릭 시 해당 글의 상세 페이지로 진입
- Mustache Template Engine을 활용
- Blog 글 id로 단일 글 데이터 조회

Mapper 개발

com.codepresso.codepressoblog.mapper.
PostMapper.java

```
Post findOne(@Param("id") Integer id);
```

resources/mybatis/mapper/
post-mapper.xml

```
<select id="findOne" resultType="com.codepresso.codepressoblog.vo.Post">
    SELECT *
    FROM post
    WHERE id=#{id};
</select>
```

Service 개발

com.codepresso.codepressoblog.service.
PostService.java

```
public Post getPostById(Integer id) {  
    return postMapper.findOne(id);  
}
```

Controller 개발

com.codepresso.codepressoblog.controller.
PostPageController.java

```
@Controller
public class PostPageController {

    private PostService postService;

    public PostPageController(PostService postService) {
        this.postService = postService;
    }

    @RequestMapping("/post/{id}")
    public String getPostDetailPage(Model model, @PathVariable Integer id) {
        Post post = postService.getPostById(id);
        model.addAttribute("post", post);
        return "post_datail";
    }
}
```

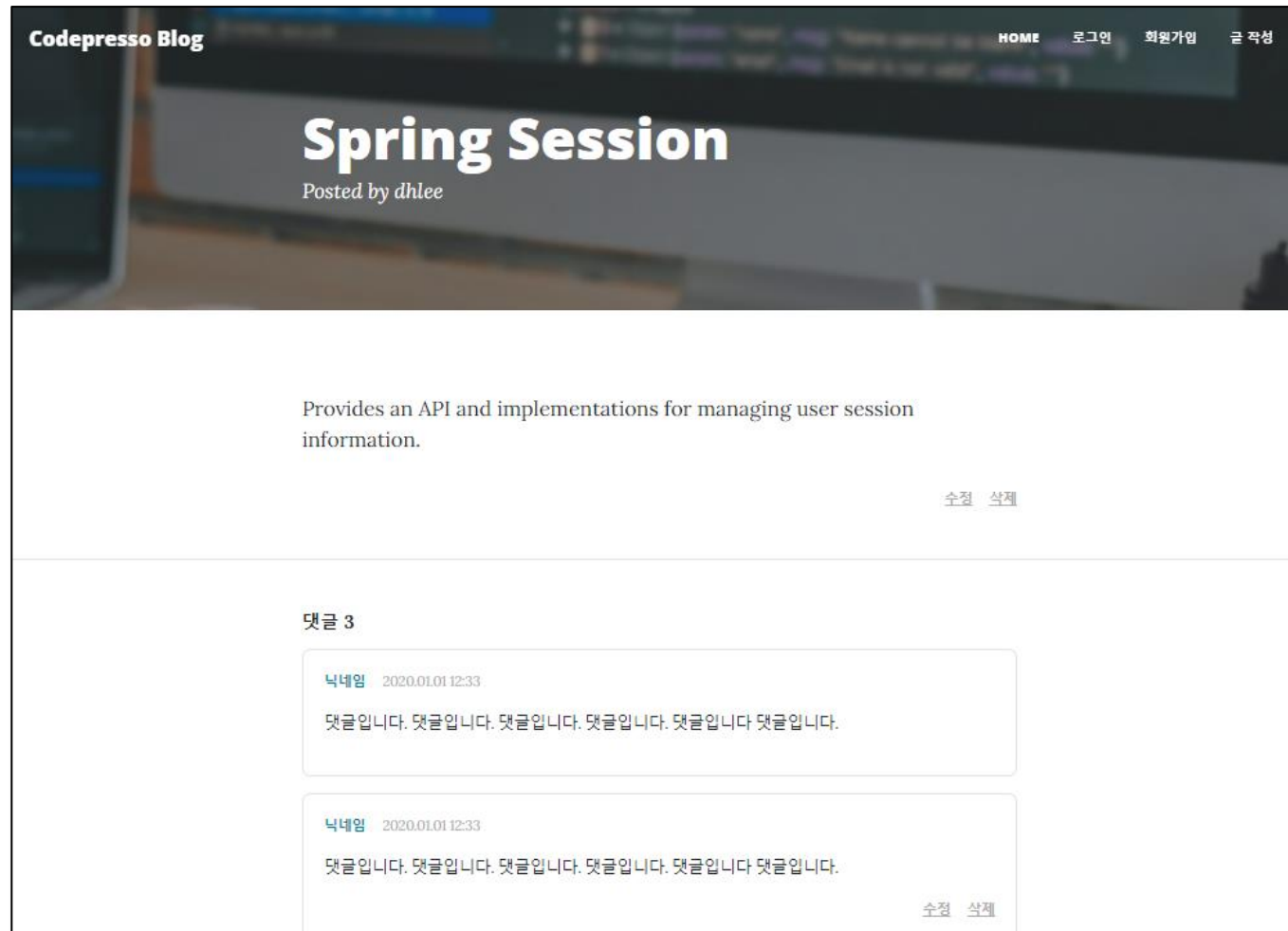
Mustache 화면 개발

resources/templates/
post_detail.mustache

```
<!-- Page Header-->
<header class="masthead" style="background-image: url('https://codepresso-online-
  <div class="container position-relative px-4 px-lg-5">
    <div class="row gx-4 gx-lg-5 justify-content-center">
      <div class="col-md-10 col-lg-8 col-xl-7">
        <div class="post-heading">
          <h1>{{post.title}}</h1>
          <span class="meta"> Posted by {{post.username}} </span>
        </div>
      </div>
    </div>
  </div>
</header>
<!-- Post Content-->
<article class="mb-4" style="...">
  <div class="container px-4 px-lg-5">
    <div class="row gx-4 gx-lg-5 justify-content-center border-bottom pb-5">
      <div class="col-md-10 col-lg-8 col-xl-7">
        <p>{{post.content}}</p>
        <div class="edit_btns">
          <button>수정</button>
          <button>삭제</button>
        </div>
      </div>
    </div>
  </div>
</article>
```

Mustache 화면 개발 테스트

■ localhost:8080/post/9 접속



Mustache 화면 개발

- index 화면 글 목록에서 글 상세 페이지 링크 추가

resources/templates/
index.mustache

```
<!-- Main Content-->
<div class="container px-4 px-lg-5">
  <div class="row gx-4 gx-lg-5 justify-content-center">
    <div class="col-md-10 col-lg-8 col-xl-7">
      {{#posts}}
      <div class="post-preview">
        <a href="/post/{{id}}">
          <h2 class="post-title">{{title}}</h2>
          <h3 class="post-subtitle">{{content}}</h3>
        </a>
        <p class="post-meta">
          Posted by {{username}}
        </p>
      </div>
    </div>
  </div>
</div>
```

Mustache 화면 개발

- index 화면 글 목록에서 글 상세 페이지 링크 추가

resources/static/js/
ajax.js

```
.done(function(response) {  
  for(var post of response) {  
    $("#more-posts").append("<div class=\"post-preview\">" +  
      "<a href=\"/post/\" + post.id + \"\">" +  
      "<h2 class=\"post-title\">" +  
      post.title +  
      "</h2>\n" +  
      "<h3 class=\"post-subtitle\">" +  
      post.content +  
      "</h3></a><p class=\"post-meta\">Posted by " +  
      post.username +  
      "</p></div><hr class=\"my-4\" />");  
  }  
  $(this).attr( name: "current-page", next_page);  
});
```

Blog 글 저장 기능 개발 1

Blog 글 저장 기능 개발

- “글 작성” 메뉴 클릭 시 글 작성 화면으로 진입
- Input Form에 정보를 입력한 후 “등록” 버튼 클릭
 - 버튼에 대한 클릭 이벤트 처리
 - ajax로 'POST /post' REST API 호출
 - 작성 된 글 정보는 Request Body 데이터로 전송
- 글 작성이 완료되면 Index 화면으로 자동으로 이동

Blog 글 저장 기능 개발 순서

■ Backend

- Mapper 개발
- Service 개발
- Controller(REST API) 개발
- REST API 테스트

■ Frontend

- '등록' 버튼 클릭 이벤트 처리
- 버튼 클릭 시 form에 작성 된 데이터를 가져옴
- ajax로 REST API 호출
- API 정상 응답 시 index 화면으로 이동(redirect)

Mapper 개발

com.codepresso.codepressoblog.mapper.
PostMapper.java

```
Integer save(@Param("post") Post post);
```

resources/mybatis/mapper/
post-mapper.xml

```
<insert id="save">
    INSERT INTO post(title, content, username)
    VALUES (#{post.title}, #{post.content}, #{post.username});
</insert>
```

Service 개발

com.codepresso.codepressoblog.service.
PostService.java

```
public boolean savePost(Post post) {  
    Integer result = postMapper.save(post);  
    return result == 1;  
}
```

Controller 개발

com.codepresso.codepressoblog.controller.
PostController.java

```
@PostMapping("/post")
public String createPost(@RequestBody PostRequestDto postDto) {
    Post post = postDto.getPost();
    postService.savePost(post);

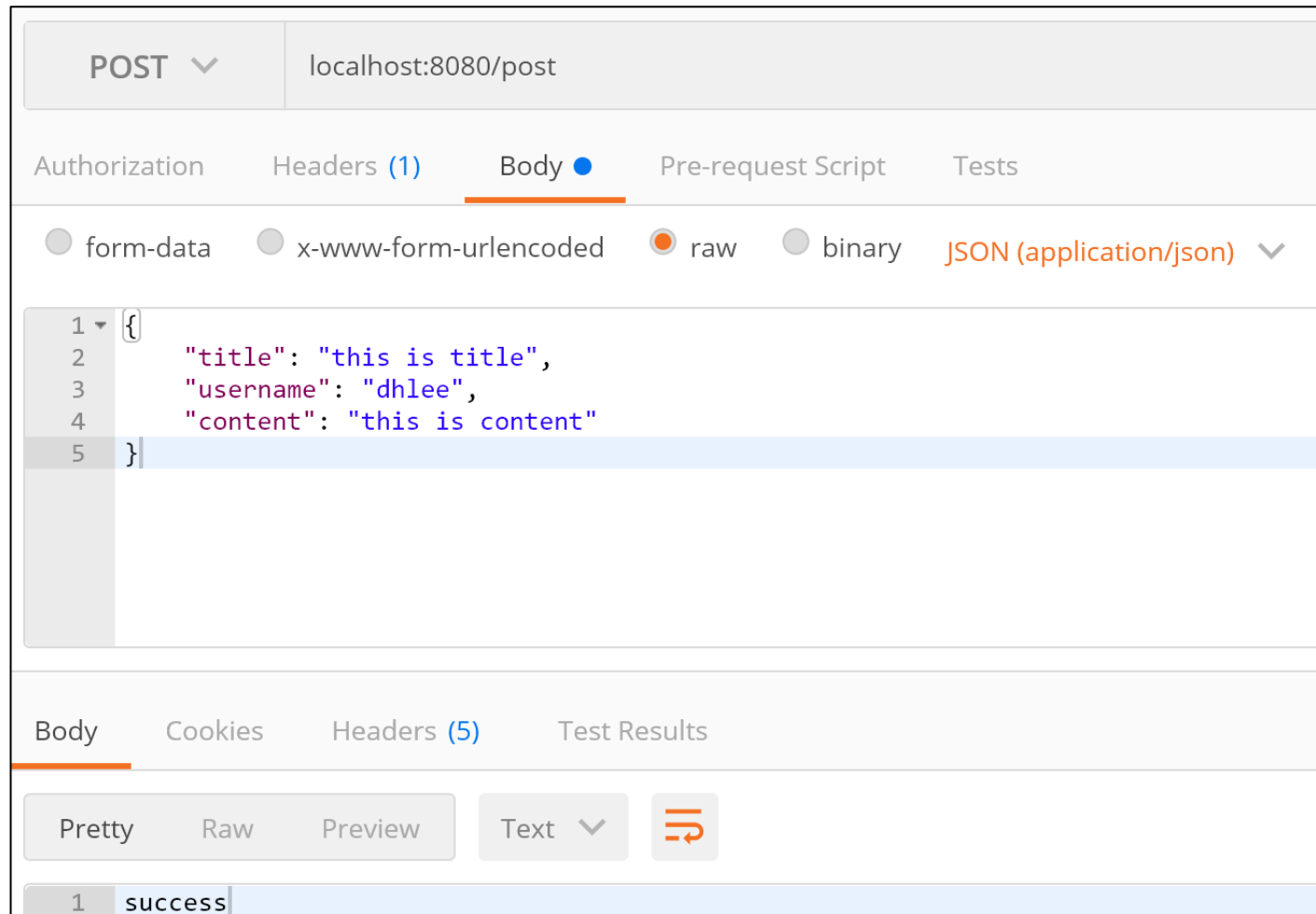
    return "success";
}
```

com.codepresso.codepressoblog.controller.dto.
PostRequestDto.java

```
public Post getPost() {
    return new Post(this.id, this.title, this.content, this.username);
}
```

REST API 테스트

■ POST /post 호출



Blog 글 저장 기능 개발 2

Blog 글 저장 기능 개발 순서

■ Backend

- Mapper 개발
- Service 개발
- Controller(REST API) 개발
- REST API 테스트

■ Frontend

- '등록' 버튼 클릭 이벤트 처리
- 버튼 클릭 시 form에 작성 된 데이터를 가져옴
- ajax로 REST API 호출
- API 정상 응답 시 index 화면으로 이동(redirect)

글 작성 화면 Controller 개발

com.codepresso.codepressoblog.controller.
PostPageController

```
@RequestMapping("/post/create")  
public String getPostCreatePage() {  
    return "post_write";  
}
```

Mustache 화면 개발

resources/templates/
post_write.mustache

```
<!-- Post Content-->
<article class="mb-4" style="...">
  <div class="container px-4 px-lg-5">
    <div class="row gx-4 gx-lg-5 justify-content-center border-bottom pb-5">
      <div class="col-md-10 col-lg-8 col-xl-7">
        <input type="text" id="post-title" class="edit mb-2" style="..." placeholder="글 제목을 입력하세요"/>
        <input type="text" id="post-username" class="edit mb-2" style="..." placeholder="사용자 이름을 입력하세요"/>
        <textarea class="edit" name="" id="post-content" cols="30" rows="10" placeholder="글 본문을 입력하세요"></textarea>
        <div class="comment_btn">
          <button id="create_button">등록</button>
        </div>
      </div>
    </div>
  </div>
</article>
```

jQuery Ajax 개발

resources/static/js/
ajax.js

```
$("#create_button").click(function(){  
    var title = $("#post-title").val();  
    var username = $("#post-username").val();  
    var content = $("#post-content").val();  
  
    $.ajax({ url: {  
        method: "POST",  
        url: "/post",  
        data: JSON.stringify({ value: {  
            "title": title,  
            "username": username,  
            "content": content  
        }},  
        contentType: "application/json"  
    })  
})
```

val()을 호출하여
각 Form에 작성 된 데이터를 반환 함

JSON.stringify() 없으면
query string 형태로 데이터 전송 됨

전송하는 데이터의 형식을
Server에게 알려줌(HTTP Header)

jQuery Ajax 개발

resources/static/js/
ajax.js

```
.done(function(response) {  
    console.log("Post creation success!");  
    window.location.href = "/";  
});  
});
```

window.location은 url과 관련 된 정보를
저장하고 있음

window.location.href는 현재 페이지의 URL
정보를 저장하며, 값을 변경하면 해당 URL로
이동

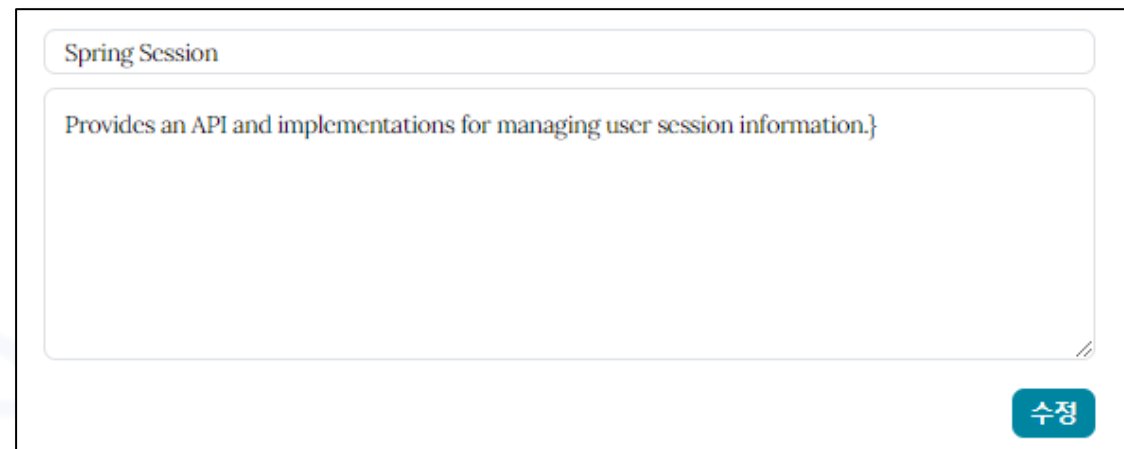
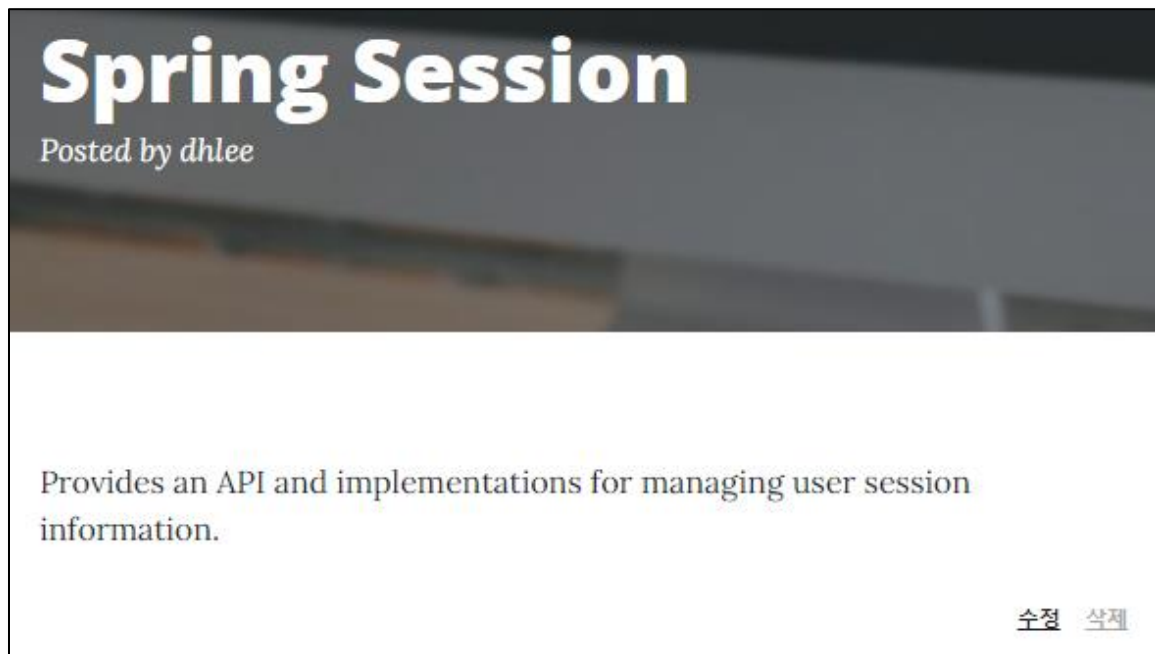
웹 서비스의 Root 경로(index)로
페이지를 이동(Redirect) 함

Blog 글 수정 기능 개발 1

Blog 글 수정 기능 개발

- 글 상세 페이지에서 “수정” 버튼을 클릭 시 수정 페이지 진입
- 글 입력 Form에 기존에 작성 된 데이터 노출(제목, 본문)
- 데이터 수정 후 “수정” 버튼 클릭 시 DB 저장 후 글 상세 페이지로 Redirect

Blog 글 수정 기능 개발



Blog 글 저장 기능 개발 순서

■ Backend

- 글 수정 REST API를 위한 Mapper 개발
- 글 수정 REST API를 위한 Service 개발
- 글 수정 REST API를 위한 Controller(REST API) 개발
- 글 수정 REST API 테스트
- 글 수정 화면을 위한 PostPageController 개발

■ Frontend

- '수정' 버튼 클릭 이벤트 처리
- 버튼 클릭 시 form에 작성 된 데이터를 가져옴
- ajax로 REST API 호출
- API 정상 응답 시 글 상세 화면으로 이동(redirect)

Mapper 개발

com.codepresso.codepressoblog.mapper.
PostMapper.java

```
Integer update(@Param("post") Post post);
```

resources/mybatis/mapper/
post-mapper.xml

```
<update id="update">  
    UPDATE post  
    SET title=#{post.title}, content=#{post.content}  
    WHERE id=#{post.id};  
</update>
```

Service 개발

com.codepresso.codepressoblog.service.
PostService.java

```
public boolean updatePost(Post post) {  
    Integer result = postMapper.update(post);  
    return result == 1;  
}
```

Controller 개발

com.codepresso.codepressoblog.controller.
PostController.java

```
@PutMapping("/post")
public String updatePost(@RequestBody PostRequestDto postDto) {
    Post post = postDto.getPost();
    postService.updatePost(post);

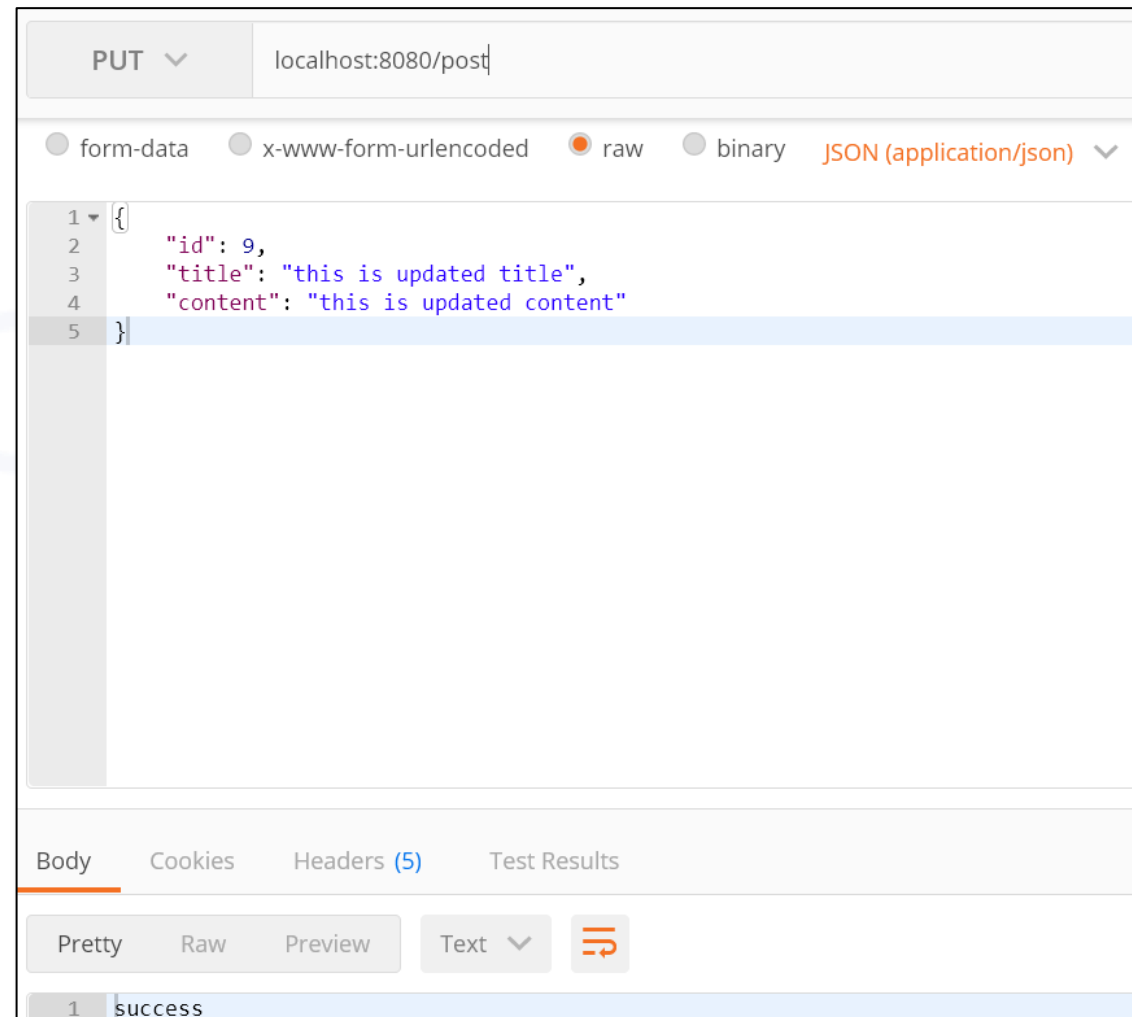
    return "success";
}
```

com.codepresso.codepressoblog.controller.dto.
PostRequestDto.java

```
public Post getPost() {
    return new Post(this.id, this.title, this.content, this.username);
}
```

REST API 테스트

■ PUT /post 호출



Blog 글 수정 기능 개발 2

Blog 글 저장 기능 개발 순서

■ Backend

- 글 수정 REST API를 위한 Mapper 개발
- 글 수정 REST API를 위한 Service 개발
- 글 수정 REST API를 위한 Controller(REST API) 개발
- 글 수정 REST API 테스트
- 글 수정 화면을 위한 PostPageController 개발

■ Frontend

- '수정' 버튼 클릭 이벤트 처리
- 버튼 클릭 시 form에 작성 된 데이터를 가져옴
- ajax로 REST API 호출
- API 정상 응답 시 글 상세 화면으로 이동(redirect)

Controller 개발

com.codepresso.codepressoblog.controller.
PostPageController.java

```
@RequestMapping("/post/edit/{id}")  
public String getPostCreatePage(Model model, @PathVariable Integer id) {  
    Post post = postService.getPostById(id);  
    model.addAttribute("post", post);  
    return "post_edit";  
}
```

수정 Form에 기존에 작성 된
제목, 본문 데이터를 노출하기 위해
post 객체 조회 후 model에 세팅

Mustache 화면 개발

resources/templates/
post_edit.mustache

```
<!-- Post Content-->
<article class="mb-4" style="...">
  <div class="container px-4 px-lg-5">
    <div class="row gx-4 gx-lg-5 justify-content-center border-bottom pb-5">
      <div class="col-md-10 col-lg-8 col-xl-7">
        <input type="text" id="edit-post-title" class="edit mb-2" style="height: 30px;" value="{{post.title}}"/>
        <textarea class="edit" name="" id="edit-post-content" cols="30" rows="10">{{post.content}}</textarea>
        <input type="hidden" id="edit-post-id" value="{{post.id}}" />
        <div class="comment_btn">
          <button id="edit_button">수정</button>
        </div>
      </div>
    </div>
  </div>
</article>
```


jQuery Ajax 개발

resources/static/js/
ajax.js

```
$("#edit_button").click(function(){  
    var id = $("#edit-post-id").val();  
    var title = $("#edit-post-title").val();  
    var content = $("#edit-post-content").val();  
  
    $.ajax({url: {  
        method: "PUT",  
        url: "/post",  
        data: JSON.stringify(value: {  
            "id": id,  
            "title": title,  
            "content": content  
        })),  
        contentType: "application/json"  
    })  
})
```

jQuery Ajax 개발

resources/static/js/
ajax.js

```
.done(function(response) {  
    console.log("Post creation success!");  
    window.location.href = "/post/" + id;  
});  
});
```

id 정보를 사용하여
수정 한 글의
상세 조회 화면으로 이동(Redirect)

Mustache 화면 개발

resources/templates/
post_detail.mustache

```
<div class="col-md-10 col-lg-8 col-xl-7">
  <p>{{post.content}}</p>
  <div class="edit_btns">
    <button><a href="/post/edit/{{id}}">수정</a></button>
    <button>삭제</button>
  </div>
</div>
```