레식피

식재료 관리 서비스 제공 웹

F2

팀장 20192128 박종혁
팀원 20172048 박주형
팀원 20192051 박세영
팀원 20182262 서지우

목차

1. 주제 선정 배경

매년 약 17%의 식품 폐기물 발생의 문제점

WRAP의 '음식 폐기물 지수 보고서 2021'에 따르면, 2019년 기준으로 전 세계에서 추정 **9.3t 가량의 식품이 가정·유통·식당·외식산업 등에서 폐기**됨. 이는 소비자가 이용할 수 있는 전체 식품의 17%정도가 낭비되는 것.
문제 해결을 위해 **식자재의 유통기한을 관리하는 방법을 통해 폐기물을 줄이고자 함.**

냉장고의 식자재를 활용할 수 있도록 레시피 제공

냉장고의 **남은 식재료를 통해 요리할 수 있도록 레시피를 제공**하여 식자재 관리의 편리함을 제공함.

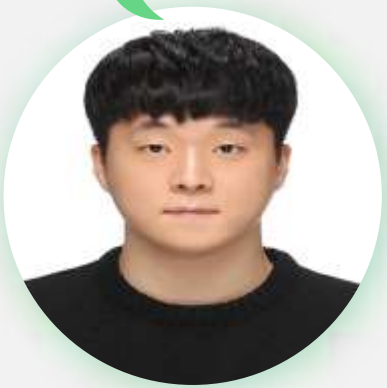
냉장고에 식자재를 등록하고, 유통기한 관리 및 레시피를 제공할 수 있는 웹 서비스 개발

2. 팀 소개 및 역할분담

팀장. 박종혁

로그인, 회원가입,
세션 구현,
파일 취합

팀원. 박주형



식자재 등록,삭제
식자재,로그인,회원가입
페이지
HTML,CSS
크롤링,데이터나열

팀원. 박세영

화면 HTML, CSS
커뮤니티 게시판, 댓글
기능 구현

팀원. 서지우

로그인,
로고 제작

2. 팀 소개 및 역할분담

<table>
<tr><th>이름
/주차</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th></tr>
<tr><td>박종혁</td><td rowspan="4">주제
선정</td><td rowspan="4">프로젝트 방향
설정</td><td rowspan="4">시스템
구조
설계</td><td rowspan="4">화면
설계,
프로토타입 작성</td><td colspan="2">고객 정보 관리
기능 설계</td><td rowspan="4">중간
점검</td><td colspan="2">회원가입 기능 구현</td><td colspan="2">로그인 기능 보완
및 세션 구현</td><td colspan="2">프로젝트 병합</td></tr>
<tr><td>박주형</td><td colspan="2">식자재 정보 관리
기능 설계</td><td colspan="2">식자재 등록,삭제</td><td colspan="3">레시피 크롤링 기능 구현</td><td rowspan="3">점검 및
오류수정</td></tr>
<tr><td>박세영</td><td colspan="2">식자재 정보 관리
기능 설계</td><td colspan="2">커뮤니티 기능 구현</td><td colspan="3">게시판 댓글 기능 구현</td></tr>
<tr><td>서지우</td><td colspan="2">고객 정보 관리
기능 설계</td><td colspan="2">로그인 기능 구현</td><td colspan="3">마이 페이지 구현</td></tr>
<tr><td>산출물</td><td></td><td>요구사
항정의
서</td><td>시스템
설계서</td><td>산출물
명세서</td><td>시스템
정의서</td><td></td><td></td><td></td><td></td><td></td><td></td><td>최종 결
과 보고
서</td></tr>
</table>

3. 프로젝트 설명

식자재 관리

식자재 추가, 조회, 변경, 삭제

레시피 제공

식자재별 레시피 조회 및 링크 연결

레시피별 필요 식자재 조회

레시피 추가 및 삭제

고객 관리

로그인, 로그아웃 기능

고객 정보 관리

유통기한 관리

유통기한 입력

폐기임박 재료 수량별 조회

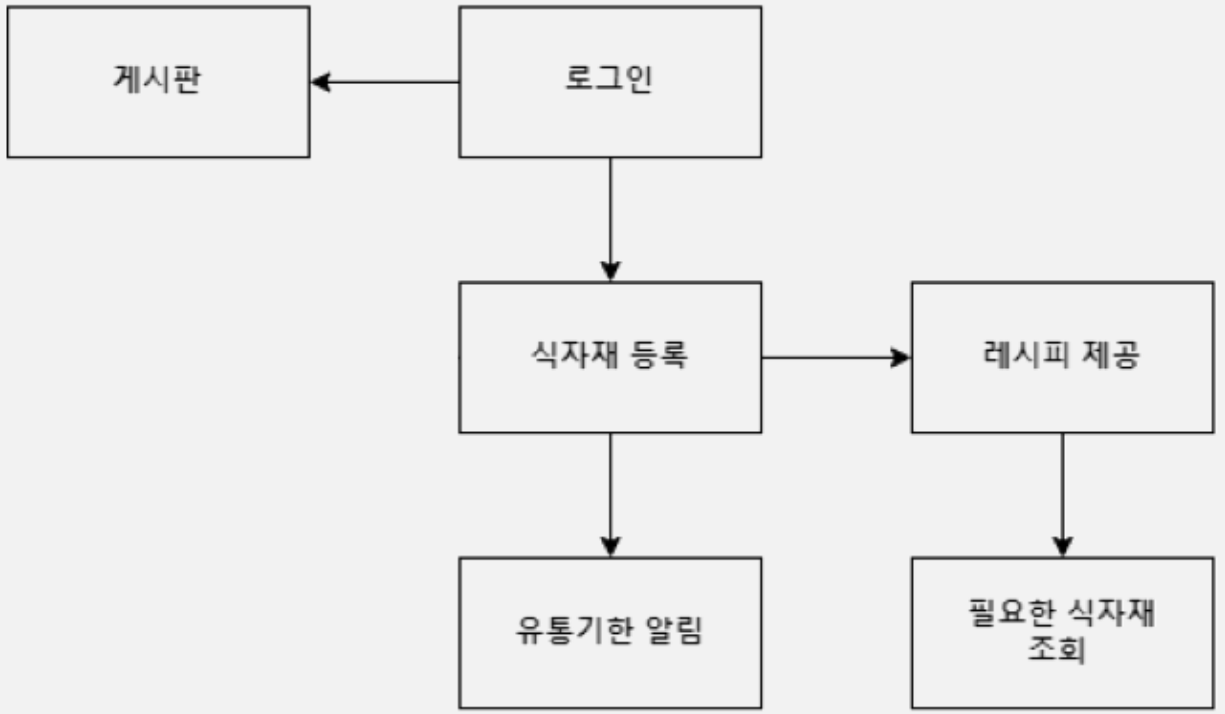
커뮤니티

게시글 조회, 추가, 수정, 삭제

댓글 조회, 추가, 수정, 삭제

카테고리) 레시피 등록,자유 글, 추천 레시피

3. 프로젝트 설명 시스템 구조



4. 활용 기술

JAVA

HTML

CSS

JS

활용 언어

화면 디자인

H2

Spring Boots Framework

Web Crawling

BootStrap

데이터 저장

자바 기반 웹 프레임 워크

레시피 추출

디자인 폼

5. 주요 화면 및 코드 설명 (1) 식자재 관리 - Entity / DTO

남은 유통기한 계산

```
    //      private int otherdate이기에 get메소드로 불러온다.
    public int getOtherdate() {

//import java.util.Date 한다
//        Date객체를 불러온다.
        Date date = new Date();
        LocalDate localDate = new java.sql.Date(date.getTime()).toLocalDate();
// date1의 날짜 형식을 "yyyy-MM-dd" 로 형태준다.
        String date1 = localDate.format(DateTimeFormatter.ofPattern("yyyy-MM-dd"));
        String date2 = getEnddate();
        LocalDate changeDate1 = LocalDate.parse(date1, DateTimeFormatter.ofPattern("yyyy-MM-dd"));
        LocalDate changeDate2 = LocalDate.parse(date2, DateTimeFormatter.ofPattern("yyyy-MM-dd"));
//return 값으로는 changeDate1,changeDate2의 일수의 차이를 출력한다.
        return Period.between(changeDate1, changeDate2).getDays();
    }
```

```
    @Column
    private String nameinfo;

    @Column
    private String image;

    @Column
    private String othermonth;

    public int getOthermonth() {

        String date1 = getStart();
        String date2 = getEnddate();

        LocalDate changeDate1 = LocalDate.parse(date1, DateTimeFormatter.ofPattern("yyyy-MM-dd"));
        LocalDate changeDate2 = LocalDate.parse(date2, DateTimeFormatter.ofPattern("yyyy-MM-dd"));
//changeDate1,changeDate2의 개월차이를 구한다.
        return Period.between(changeDate1, changeDate2).getMonths();
    }
```

Entity 등록

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

    @Column
    private String name;

    @Column
    private String start;

    @Column
    private String enddate;

    @Column
    private String kind;

    @Column
    private int otherdate;
```

DTO 등록

```
@NoArgsConstructor
@AllArgsConstructor
@ToString
@Getter
@Setter
public class DataForm {
    private Long id;
    private String name;
    private String start;
    private String enddate;
    private String kind;
    private int otherdate;
    private String nameinfo;
    private String image;
    private String othermonth;
    private String diffdays;
    private String diffdays;

    public Data toEntity(){
        return new Data(id,name,start,enddate,kind,otherdate,nameinfo,image,
    }
}
```

폐기식품 안내

```
서 남은일자를 if문에
@Column
private String diffdays;

public String getDiffdays() {
    int month = getOthermonth();
    int date = getOtherdate();
    if (month==0){
        if (date<0){
            return String.format("※폐기해야 합니다※");
        } else if (date==0) {
            return String.format("오늘까지 입니다");
        }
    }
    return String.format("%d개월 %d남음",month,date);
}
```

5. 주요 화면 및 코드 설명 (1) 식자재 관리 - Controller

식자재 등록

```
//새로운 식자재 등록
@PostMapping("/view/create")
public String homecreate(DataForm form, Model model) {
    log.info(form.toString());

    //1.dto를 entity로 변환
    Data data = form.toEntity();
    log.info(data.toString());

    //2. entity를 repository에 저장.
    Data saved = dataRepository.save(data);
    log.info(saved.toString());

    log.info(data.toString());
    model.addAttribute("data", saved);
    //repository에 저장된 정보를 배열로 불러온다.
    List<Data> Datalist = dataRepository.findAll();

    //Datalist를 오름차순 정렬
    //Datalist를 내림차순 정렬하려면 Datalist.sort(Comparator.comparing(Data::getName).reversed());
    Datalist.sort(Comparator.comparing(Data::getOtherdate));
    Datalist.sort(Comparator.comparing(Data::getOthermonth));

    log.info("Entity 변환 Data" + data.toString());

    model.addAttribute("datalist", Datalist);
//      Datalist목록 값들을 view/refrigerator 페이지에 출력한다.
    return "view/refrigerator";
```

```
//레시피 페이지에서 부류별로 정렬 클릭시 동작하는 컨트롤러
@GetMapping("/textset")
public String textset(Model model,RedirectAttributes rttr){
//      리파지터리에서 데이터를 모두 찾는다
    List<Data> textset = dataRepository.findAll();
//      모델을 사용해서 textset배열을 datalist로 지정한다.
    model.addAttribute("datalist",textset);
//      데이터 정렬 순서를 getKind 순서에 맞게 나열한다.
    textset.sort(Comparator.comparing(Data::getKind));
//      버튼 클릭시 알림메시지를 출력한다.
    rttr.addFlashAttribute("textset", "부류별로 정렬하였습니다!");
    return "view/refrigerator";
}
//레시피 페이지에서 이름별로 정렬 클릭시 동작하는 컨트롤러
@GetMapping("/nameset")
public String nameset(Model model,RedirectAttributes rttr){
//      리파지터리에 있는 데이터를 모두 찾는다.
    List<Data> nameset = dataRepository.findAll();
//      모델을 사용해서 nameset을 datalist로 지정한다.
    model.addAttribute("datalist",nameset);
//      데이터 정렬을 getName에 맞게 나열한다.
    nameset.sort(Comparator.comparing(Data::getName));
//      버튼 클릭시 알림메시지를 출력한다.
    rttr.addFlashAttribute("nameset", "이름별로 정렬하였습니다!");
//      데이터 값들을 return 값에 표시한다.
    return "view/refrigerator";
```

정렬 기능

```
public String pagemypage(Model model) {
//      리파지터리에 저장된 데이터를 모두 찾는다
    List<Data> Datalist = dataRepository.findAll();
//      Datalist를 datalist로 지정한다.
    model.addAttribute("datalist", Datalist);
//      정렬을 getOtherdate,getOthermonth순서로 나열한다.
    Datalist.sort(Comparator.comparing(Data::getOtherdate));
    Datalist.sort(Comparator.comparing(Data::getOthermonth));

    return "view/refrigerator";
}
```

새로운 식자재를 등록하고 정렬하는 기능

5. 주요 화면 및 코드 설명 (1) 식자재 관리 - Controller

식자재 삭제

```
//삭제 기능 구현
    @GetMapping("/listDelete/{id}")
    public String delete(@PathVariable Long id, Model model, RedirectAttributes rttr, DataForm form) {
//      /listDelete/{id} 지정된 id 값을 삭제했을시 발생하는 log이다. 콘솔창에 출력된다.
        log.info("삭제요청");
        //1. 대상을 가져온다
        Data target = dataRepository.findById(id).orElse( other: null);
        log.info("ddddd" + target.toString());
        //2. 대상 삭제한다.
        if (target != null) {
//          target이 null이 아니면 리파지터리에서 target을 delete한다.
            dataRepository.delete(target);
//            지정된 target 즉 id 값이 삭제되었을경우에 발생되는 알림메시지이다.
            rttr.addFlashAttribute( attributeName: "deletemsg",  attributeValue: target.getName()+"을(를) 삭제하였습니다!");
        }
```

식자재 상세 출력

```
//이미지 클릭시 식자재의 상세 정보 출력
    @GetMapping("imageshow/{id}")
    public String imageshow(@PathVariable Long id, Model model) {
        log.info("id="+id);
//      호출시 콘솔창에 출력된다.
        Data dataEntity = dataRepository.findById(id).orElse( other: null);
// 리파지터리에서 id값과 id값이거나 null값을 찾아서 dataEntity에 저장한다.
        model.addAttribute( attributeName: "dataid", dataEntity);
        return "view/listshow";
    }
```

5. 주요 화면 및 코드 설명 (2) 게시판 기능 - Entity / DTO

Entity 등록

```java
@Entity
@AllArgsConstructor //리팩토링
@NoArgsConstructor //디폴트 생성자 추가
@ToString //리팩토링
@Getter //id 값을 redirect
public class Article {

    @Id
    @GeneratedValue
    private Long id; //번호

    @Column
    private String division; //구분

    @Column
    private String title; //제목

    @Column
    private String content; //내용

    @Column
    private String rgiDate; //등록일

    public LocalDate getRgiDate() {
        Date date = new Date();
        LocalDate localDate = new java.sql.Date(date.getTime()).toLocalDate();
        String rgiDate = localDate.format(DateTimeFormatter.ofPattern("yyyy-MM-dd"));
        LocalDate Date = LocalDate.parse(rgiDate, DateTimeFormatter.ofPattern("yyyy-MM-dd"));

        return Date;
```

DTO 등록

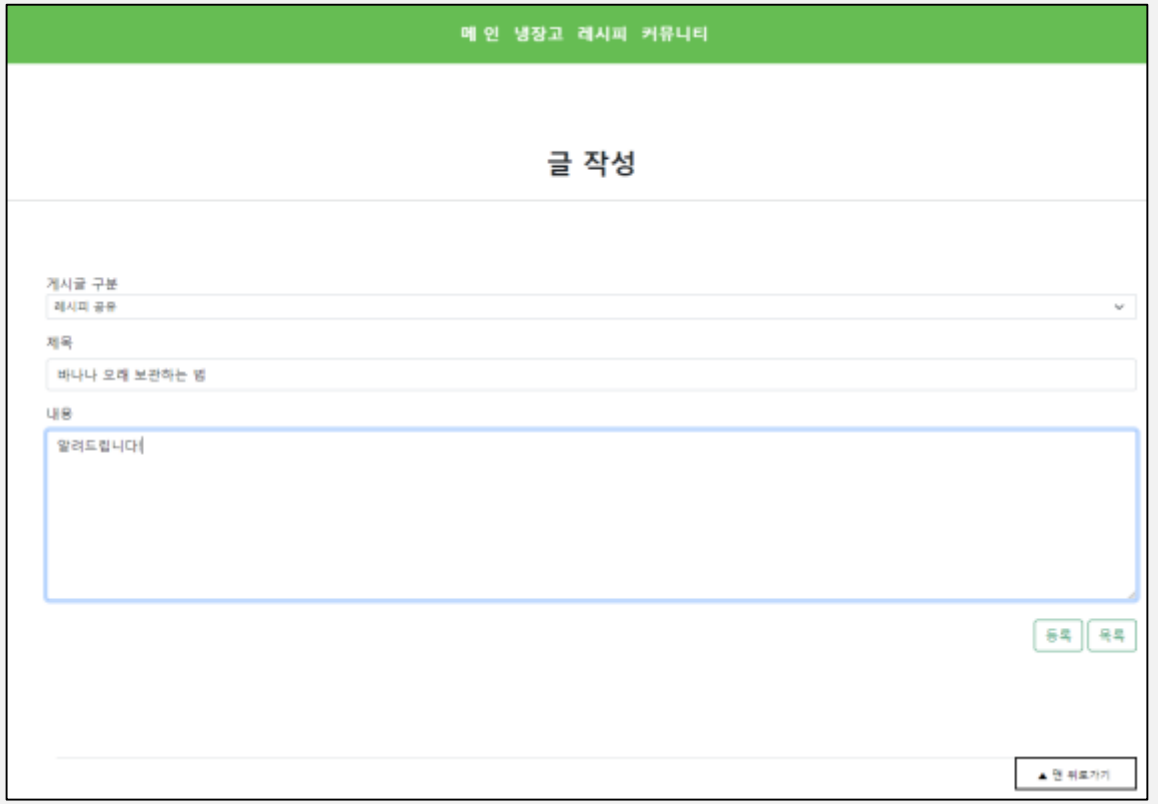
```java
@AllArgsConstructor //리팩토링
@ToString //리팩토링
public class ArticleForm {

    private Long id;
    private String division;
    private String title;
    private String content;
    private String rgiDate;


    public Article toEntity() { return new Article(id,division, title, content,rgiDate); }//entity

}
```

DTO를 DB가 이해할 수 있도록 Entity 등록을 통해 데이터를 규격화시킴

5. 주요 화면 및 코드 설명 (2) 게시판 기능 - Controller

게시글 등록

```java
@Controller
@Slf4j //로깅
public class ArticleController {

    @Autowired
    private ArticleRepository articleRepository;

    //게시판
    @GetMapping("/view/Community")
    public String newArticleForm() { return "view/Community"; }

    //등록 기능 (데이터 입력 및 저장)
    //Community.mustache
    @PostMapping("/view/new")
    public String createArticle(ArticleForm form) {
        log.info(form.toString());

        //dto->entity
        Article article = form.toEntity();
        log.info(article.toString());
        //entity->db 저장
        Article saved = articleRepository.save(article);
        log.info(saved.toString());

        return "redirect:/articles/" + saved.getId(); // redirect:글을 등록하면 방금 작성한 글을 확인
    }
```



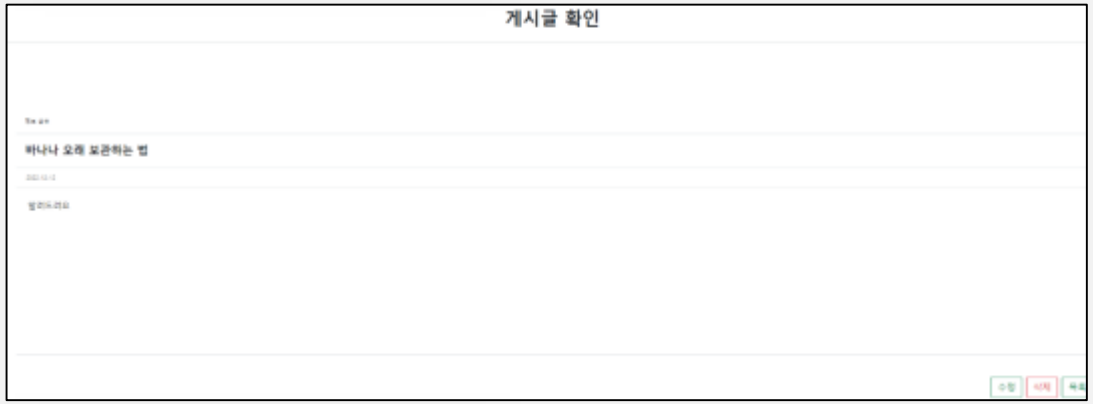
신규 게시글 등록

5. 주요 화면 및 코드 설명 (2) 게시판 기능 - Controller

게시글 조회 기능

```java
//조회 기능 (=> http://localhost:8080/articles/1)
@GetMapping("/articles/{id}")
public String show(@PathVariable Long id, Model model){
    log.info("id="+id);

    //id를 데이터로 가져오기
    Article articleEntity = articleRepository.findById(id).orElse( other: null);
    //가져온 데이터 모델에 등록
    model.addAttribute( attributeName: "article", articleEntity);
    //페이지 설정
    return "view/show";
}

//목록 조회 (=> http://localhost:8080/articles)
@GetMapping("/articles")
public String index(Model model) {
    //모든 article 가져오기
    List<Article> articleEntityList = articleRepository.findAll();
    //뷰로 전달
    model.addAttribute( attributeName: "articleList",articleEntityList);
    //페이지 설정
    return "view/index";
}
```

게시글 조회



게시글 목록 조회

등록한 게시글 조회 및 목록조회

5. 주요 화면 및 코드 설명 (2) 게시판 기능 - Controller

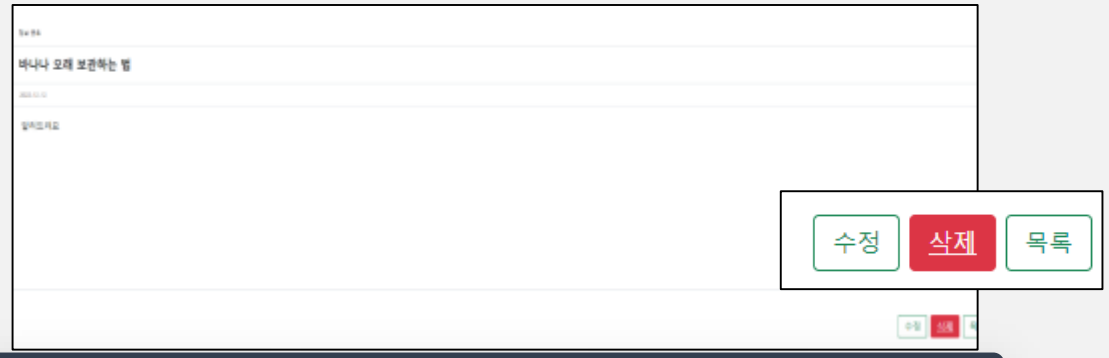
게시글 수정

```
@GetMapping("/articles/{id}/articleEdit")
public String articleEdit(@PathVariable Long id, Model model) {
    //수정할 데이터 가져오기
    Article articleEntity = articleRepository.findById(id).orElse( other: null);
    //모델에 데이터 등록
    model.addAttribute( attributeName: "article",articleEntity);
    //페이지 설정
    return "view/articleEdit";
}

@PostMapping("/articles/update")
public String update(ArticleForm form) {
    log.info(form.toString());

    //dto-> entity
    Article articleEntity = form.toEntity();
    log.info(articleEntity.toString());
    //entity->db
    Article target = articleRepository.findById(articleEntity.getId()).orElse( other: null);
    if(target != null){
        articleRepository.save(articleEntity);
    }
    //리다이렉트
    return "redirect:/articles/" + articleEntity.getId();
}
```

게시판 상세 페이지에서 수정을 누르면 기존 내용을 출력

게시글 삭제

```
@GetMapping("/articles/{id}/delete")
public String delete(@PathVariable Long id, RedirectAttributes rttr) {
    log.info("삭제 요청");

    //삭제 대상 가져오기
    Article target = articleRepository.findById(id).orElse( other: null);
    log.info(target.toString());
    //대상 삭제
    if(target != null) {
        articleRepository.delete(target);
        rttr.addFlashAttribute( attributeName: "msg", attributeValue: "삭제가 완료되었습니다.");
    }
    //리다이렉트
    return "redirect:/articles";
}
```



게시판 상세 페이지에서 삭제 버튼 클릭

5. 주요 화면 및 코드 설명 (3) 댓글 기능 -추가

새 댓글 추가 폼

```
<div class="card m-3" id="comments-new">
    <div class="card-body" >
        <!--댓글 작성 폼-->
        <form>
            <!--닉네임 입력-->
            <div class="mb-3">
                <label class="form-label">닉네임</label>
                <!-- <input class="form-control form-control-sm" id="new-comment-nickname"> -->
            </div>
            <!--댓글 본문 입력 입력-->
            <div class="mb-3">
                <label class="form-label">댓글 내용</label>
                <textarea class="form-control form-control-sm" rows="4" id="new-comment-body"></textarea>
            </div>

            <!--히든 필드-->
            {{#article}}
                <input type="hidden" id="new-comment-article-id" value="{{id}}">
            {{/article}}

            <!--전송 버튼-->
            <button type="button" class="btn btn-outline-primary btn-sm" id="comment-create-btn">댓글 작성</button>
        </form>
    </div>

</div>
```

새 댓글 추가 JS 코드

```
{
    //댓글 생성 버튼 변수화(id가 comment-create-btn인 대상)
    const commentCreateBtn = document.querySelector("#comment-create-btn");

    //버튼 클릭 이벤트를 감지
    commentCreateBtn.addEventListener("click", function() {
        //새 댓글 객체 생성
        const comment = {
            nickname: document.querySelector("#new-comment-nickname").value,
            body: document.querySelector("#new-comment-body").value,
            article_id: document.querySelector("#new-comment-article-id").value
        };

        //댓글 객체 출력
        console.log(comment);

        // fetch()
        const url = "/api/articles/" + comment.article_id + "/comments";
        fetch(url, {
            method: "post",
            body: JSON.stringify(comment),
            headers: {
                "Content-Type": "application/json"
            }
        }).then(response => {
            //http 응답 코드에 따른 메세지 출력
            const msg = (response.ok) ? "댓글이 등록되었습니다." : "댓글 등록 실패";
            alert(msg);
            //현재 페이지 새로고침
            window.location.reload();
        })
    });
}
```

댓글 생성 버튼 변수화 후 클릭 이벤트를 감지하여, 추가 메시지를 출력하도록 하고 페이지 새로고침

5. 주요 화면 및 코드 설명 (3) 댓글 기능 -컨트롤러 작성

댓글 목록 조회 및 생성

```java
@RestController
public class CommentApiController {
    @Autowired
    private CommentService commentService;

    //댓글 목록 조회
    @GetMapping("/api/articles/{articleId}/comments")
    public ResponseEntity<List<CommentDTO>> comments(@PathVariable Long articleId) {
        //서비스에게 위임
        List<CommentDTO> dtos = commentService.comments(articleId);
        //결과 응답
        return ResponseEntity.status(HttpStatus.OK).body(dtos); //성공한다는 가정하에
    }

    //댓글 생성
    @PostMapping("/api/articles/{articleId}/comments")
    public ResponseEntity<CommentDTO> create(@PathVariable Long articleId, @RequestBody CommentDTO dto) {
        //서비스에게 위임
        CommentDTO createdDTO = commentService.create(articleId, dto);

        //결과 응답
        return ResponseEntity.status(HttpStatus.OK).body(createdDTO);
    }
```

댓글 수정 및 삭제

```java
//댓글 수정
@PatchMapping("/api/comments/{id}")
public ResponseEntity<CommentDTO> update(@PathVariable Long id, @RequestBody CommentDTO dto) {
    //서비스에게 위임
    CommentDTO updatedDTO = commentService.update(id, dto);

    //결과 응답
    return ResponseEntity.status(HttpStatus.OK).body(updatedDTO);
}

//댓글 삭제
@DeleteMapping("/api/comments/{id}")
public ResponseEntity<CommentDTO> delete(@PathVariable Long id) {
    //서비스에게 위임
    CommentDTO updatedDTO = commentService.delete(id);

    //결과 응답
    return ResponseEntity.status(HttpStatus.OK).body(updatedDTO);
}
```

Controller 파일에서 댓글 기능 수행 코드를 작성한 후 View로 넘겨줌

5. 주요 화면 및 코드 설명 (3) 댓글 기능 -수정

댓글 수정 폼

```
{{#commentDtos}}
    <div class="card m-2" id="comments-{{id}}"> <!--m : 마진-->
        <div class="card-header">
            {{nickname}}
            <!-- 모달 트리거 버튼-->
            <!--     수정 버튼-->
            <button type="button"
                    class="btn btn-sm btn-primary"
                    data-bs-toggle="modal"
                    data-bs-target="#comment-edit-modal"
                    style="..."
                    data-bs-id="{{id}}"
                    data-bs-nickname="{{
                    data-bs-body="{{body
                    data-bs-article-id="
```


```
<div class="modal fade" id="comment-edit-modal" tabindex="-1">
    <div class="modal-dialog">
        <div class="modal-content">
            <div class="modal-header">
                <h1 class="modal-title fs-5" id="exampleModalLabel">댓글 수정
                <button type="button" class="btn-close" data-bs-dismiss="mo
            </div>
            <div class="modal-body">
                <form>
                    <!--댓글 수정 폼-->
                    <form>
                        <!--댓글 본문 입력 필드-->
                        <div class="mb-3">
                            <label class="form-label">댓글 내용</label>
                            <textarea class="form-control form-control-sm" rows="4" id="edit-comment-body"></textarea>
                        </div>

                        <!--히든 인풋-->
                        <input type="hidden" id="edit-comment-id">
                        <input type="hidden" id="edit-comment-article-id">

                        <!--완료 버튼-->
                        <button type="button" class="btn btn-outline-primary btn-sm" id="comment-update-btn">수정 완료</button>
```

댓글 수정 JS 코드

```
{
    //모달 요소 선택
    const commentEditModal = document.querySelector("#comment-edit-modal"); //id값 #으로 가져오기

    //모달 이벤트 감지
    commentEditModal.addEventListener("show.bs.modal",function(event) {
        //트리거 버튼 선택
        const triggerBtn = event.relatedTarget;

        //데이터 가져오기
        const id = triggerBtn.getAttribute("data-bs-id");
        const nickname = triggerBtn.getAttribute("data-bs-nickname");
        const body = triggerBtn.getAttribute("data-bs-body");
        const articleId = triggerBtn.getAttribute("data-bs-article-id");

        //데이터 반영 => 히든 인풋이 변경됨
        document.querySelector("#edit-comment-nickname").value = nickname;
        document.querySelector("#edit-comment-body").value = body;
        document.querySelector("#edit-comment-id").value = id;
        document.querySelector("#edit-comment-articleId").value = articleId;
    });
}

{
    //수정 완료 버튼
    const commentUpdateBtn = document.querySelector("#comment-update-btn");

    //클릭 이벤트 감지 및 처리
    commentUpdateBtn.addEventListener("click", function() {
        //수정 댓글 객체 생성
        const comment = {
            id: document.querySelector("#edit-comment-id").value,
            nickname: document.querySelector("#edit-comment-nickname").value,
            body: document.querySelector("#edit-comment-body").value,
            article_id: document.querySelector("#edit-comment-article-id").value
        };

        console.log(comment);

        //수정 REST API 호출 -fetch()
        const url = "/api/comments/" + comment.id;
        fetch(url, {
            method: "PATCH",                    //PATCH 요청
            body: JSON.stringify(comment),      //수정된 댓글 객체를 JSON으로 전달
            headers : {
                "Content-Type": "application/json"
            }
        }).then(response => {
            //http 응답 코드에 따른 메시지 출력
            const msg = (response.ok) ? "댓글이 수정 되었습니다." : "댓글 수정 실패";
```

댓글 수정 모달 이벤트를 통한 댓글 수정 후 안내창 출력 및 페이지 새로고침

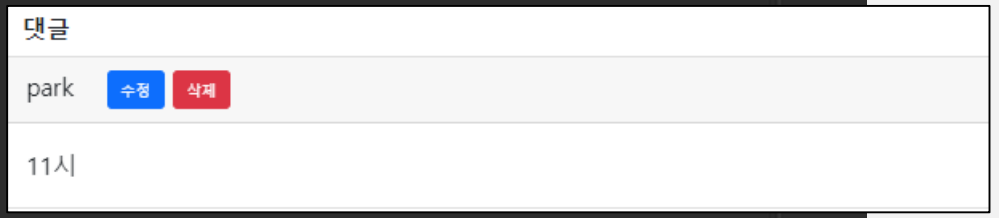
5. 주요 화면 및 코드 설명 (3) 댓글 기능 -삭제

댓글 삭제 폼

```
<div id="comments-list">
    {{#commentDTOS}}
        <div class="card m-2" id="comments-{{id}}"> <!--m : 마진-->
            <div class="card-header">
                {{nickname}}
                <!-- 모달 트리거 버튼-->
                <!--    수정 버튼-->
                <button type="button"
                        class="btn btn-sm btn-primary"
                        data-bs-toggle="modal"
                        data-bs-target="#comment-edit-modal"
                        style="..."
                        data-bs-id="{{id}}"
                        data-bs-nickname="{{nickname}}"
                        data-bs-body="{{body}}"
                        data-bs-article-id="{{articleId}}">수정</button>
                <!--    삭제버튼-->
                <button type="button"
                        class="btn btn-sm btn-danger comment-delete-btn"
                        style="..."
                        data-comment-id="{{id}}">삭제</button>
            </div>
            <div class="card-body">
                {{body}}
            </div>
        </div>
    {{/commentDTOS}}
```

댓글 삭제 JS 코드

```
//삭제 버튼 선택
const commentDeleteBtns = document.querySelectorAll(".comment-delete-btn"); //class값 가지고 버튼 가져오기

//삭제 버튼 이벤트를 처리
commentDeleteBtns.forEach(btn => {
    //각 버튼의 이벤트 처리 등록
    btn.addEventListener("click", (event) => {
        //1. 이벤트 발생 요소 선택 (클릭 이벤트가 발생된 버튼을 가져옴)
        const commentDeleteBtn = event.srcElement;

        //2. 삭제 댓글 id 가져오기
        const commentId = commentDeleteBtn.getAttribute("data-comment-id"); //id 가져오기
        console.log(`삭제 버튼 클릭: ${commentId}번 댓글`);       // 백틱 문자열) "삭제 버튼 클릭: " + commentId + "번 댓글"; 와 동일!

        //3. 삭제 API 호출 및 처리
        const url = `/api/comments/${commentId}`;   //백틱 문자열에 변수 삽입 가능
        fetch(url, {
            method: "DELETE"
        }).then(response => {
            //댓글 삭제 실패 처리
            if (!response.ok) {
                alert("댓글 삭제 실패");
                return;
            }
            //댓글 삭제 성공 시, 댓글을 화면에서 지움
            const target = document.querySelector(`#comments-${commentId}`);
            target.remove();
```



수정과 마찬가지로 모달 이벤트를 감지하여 삭제 버튼의 이벤트 수행

5. 주요 화면 및 코드 설명 (4) 레시피 조회(Web Crawling)

웹 크롤링 태그 추출


```
style="padding: 20px 0 0 0;">…</div>
<div class="blank_bottom"></div>
<!-- Swiper -->
▶ <div class="view2_box" style="margin-top: 10px; marg
in-bottom: 10px;">…</div>
<!-- Initialize Swiper -->
▶ <script>…</script>
<div class="blank_bottom"></div>
... ▼ <div class="view_step"> == $0
  ▼ <div class="best_tit">
      <b>조리순서</b>
      <span>Steps</span>
    ▶ <div class="best_tit_rmn">…</div>
    </div>
  ▶ <div id="stepDiv1" class="view_step_cont media ste
    p1">…</div>
  ▶ <div id="stepDiv2" class="view_step_cont media ste
    p2">…</div>
  ▶ <div id="stepDiv3" class="view_step_cont media ste
    p3">…</div>
```

웹 크롤링 대상 페이지에 접속하여 크롤링할 부분 HTML 태그를 찾아옴

5. 주요 화면 및 코드 설명 (4) 레시피 조회(Web Crawling)

웹 크롤링 코드 작성

```
@GetMapping("/recipelist2")
public String goRegister1(Model model) throws IOException{
    Document doc = Jsoup.connect( url: "https://www.10000recipe.com/recipe/6992986").get();
    //크롤링할 사이트를 불러온다
//      크롤링한 페이지에서 추출할 html 태그 코드를 찾아서 입력한다.
    Elements ingredient = doc.select( cssQuery: "div[class=ready_ingre3]");
    Elements foodname = doc.select( cssQuery: "div[class=view2_summary st3]").select( query: "h3");
    Elements infoList1 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr1]");
    Elements infoList2 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr2]");
    Elements infoList3 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr3]");
    Elements infoList4 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr4]");
    Elements infoList5 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr5]");
    Elements infoList6 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr6]");
    Elements infoList7 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr7]");
    Elements infoList8 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr8]");
    Elements infoList9 = doc.select( cssQuery: "div[class=view_step]").select( query: "[id=stepdescr9]");
```

```
//recipelist2 접속시 콘솔창에 데이터 출력된다.
System.out.println("foodname"+foodname);
System.out.println(infoList1.text());
System.out.println(infoList2.text());
System.out.println(infoList3.text());
System.out.println(infoList4.text());
System.out.println(infoList5.text());
System.out.println(infoList6.text());
System.out.println(infoList7.text());
System.out.println(infoList8.text());
System.out.println(infoList9.text());


//태그에서 불러온 데이터를 텍스트만 추출한다.
String Ingredient = ingredient.text();
String Foodname = foodname.text();
String infolist1 = infoList1.text();
String infolist2 = infoList2.text();
String infolist3 = infoList3.text();
String infolist4 = infoList4.text();
String infolist5 = infoList5.text();
String infolist6 = infoList6.text();
String infolist7 = infoList7.text();
String infolist8 = infoList8.text();
String infolist9 = infoList9.text();
```

크롤링 태그 찾아온 후 사이트를 불러와 추출

5. 주요 화면 및 코드 설명 (4) 레시피 조회(Web Crawling)

웹 크롤링 코드 작성

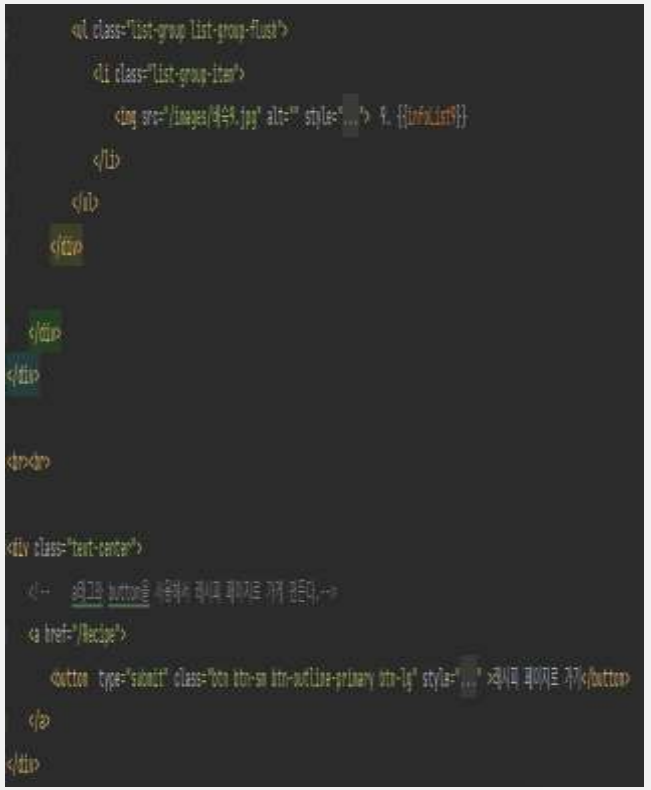
```
model.addAttribute( attributeName: "foodname",Foodname);
model.addAttribute( attributeName: "infoList1",infolist1);
model.addAttribute( attributeName: "infoList2",infolist2);
model.addAttribute( attributeName: "infoList3",infolist3);
model.addAttribute( attributeName: "infoList4",infolist4);
model.addAttribute( attributeName: "infoList5",infolist5);
model.addAttribute( attributeName: "infoList6",infolist6);
model.addAttribute( attributeName: "infoList7",infolist7);
model.addAttribute( attributeName: "infoList8",infolist8);
model.addAttribute( attributeName: "infoList9",infolist9);
model.addAttribute( attributeName: "Ingredient",Ingredient);

//infolist 들을 모델로 묶어서 "view/register"에 출력한다.
return "view/register2";
```

```
foodname<h3>우리집 배숙</h3>
재료는 배, 시중에서 판매하는 맛밤, 생강청, 꿀스틱입니다.
배의 속을 수저로 정리하여 재료를 담을 수 있도록 정리합니다.
맛밤을 넣습니다.
생강청을 넣습니다
꿀스틱을 넣습니다.
냄비에 물을 붓고 중탕하여 끓입니다.
배뚜껑을 살포시 닫아서 끓이는 것, 잊지 말아 주세요^^
중탕으로 끓이니 배의 색이 갈색으로 바뀌었습니다.
우리집만의 배숙이 완성되었습니다.
```

모델로 묶어서 출력하는 작업

5. 주요 화면 및 코드 설명 (4) 레시피 조회(Web Crawling)

웹 크롤링 화면 구성

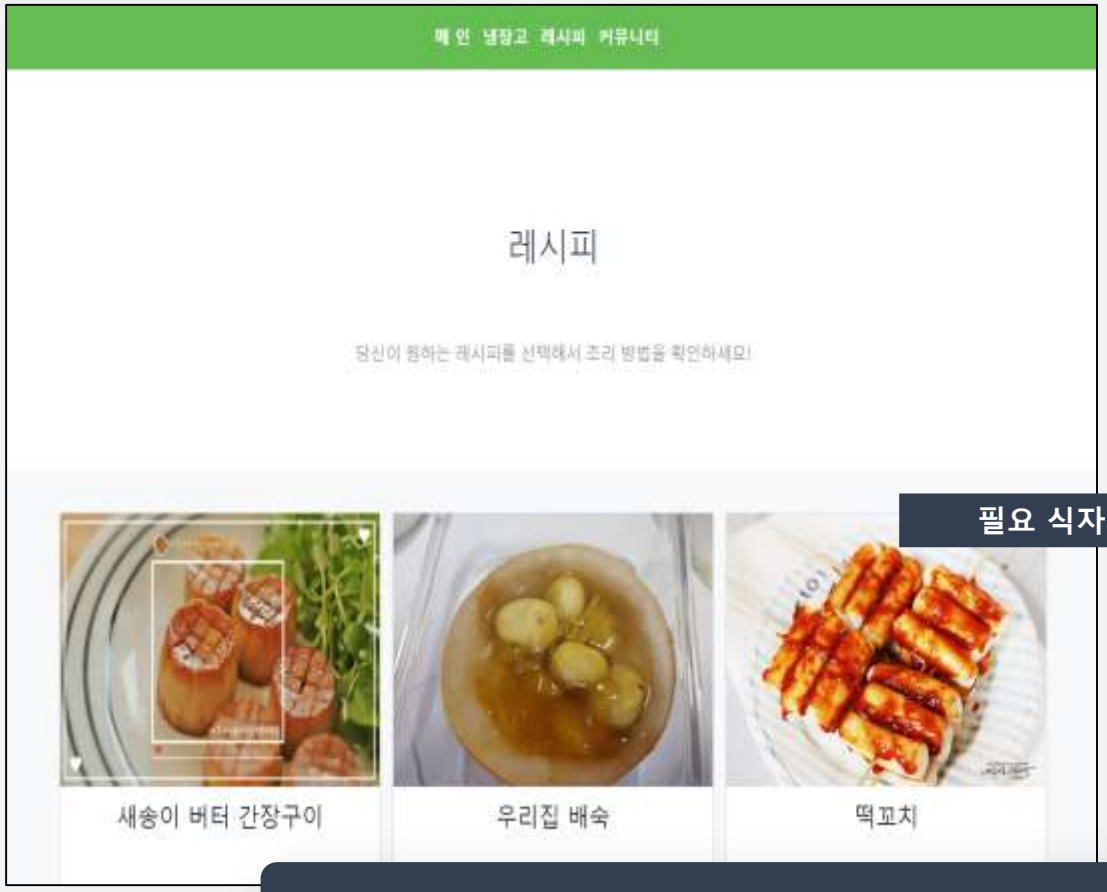


출력시킬 페이지의 HTML 구조와 CSS 작업을 해놓은 상태에서
Model로 치환한 값들을 HTML코드에 입력

5. 주요 화면 및 코드 설명 (4) 레시피 조회(Web Crawling)

레시피 목록 조회

레시피 상세 화면



필요 식자재

레시피

레시피 화면 접속 시 원하는 레시피를 클릭 시, 상세 화면에 재료와 설명을 출력

5. 주요 화면 및 코드 설명 (5) 로그인 구현 – 아이디 중복 확인

아이디 중복 확인

```java
@Controller
@Slf4j //로깅
public class UserController {

    @Autowired
    private UserService userService;


    //등록안내페이지
    @GetMapping("/view/join/info")
    public String info() { return "view/join/info"; }

    //등록페이지
    @GetMapping("/view/join/reg")
    public String reg() { return "view/join/reg"; }

    // 중복체크
    @ResponseBody
    @GetMapping("/view/join/dupCheck")
    public Map<String, String> dupCheck(@RequestParam Map<String, String> param) {
        Map<String, String> map = new HashMap<~>();
        String userId = param.get("userId");
        int dupCnt = userService.dupCheck(userId);
        map.put("dupCheckCnt", String.valueOf(dupCnt));
        return map;
    }
```

```javascript
function dupCheck() {
    //alert("중복체크");
    var userId = $("#userId").val();
    //alert("id = " + userId);
    if ( userId == "" ) {
        alert("아이디를 입력하세요.");
        return;
    }
    var data = { "userId" : userId }
    $.ajax ({
        url: "/view/join/dupCheck",
        data: data,
        type:"get",
        success: function ( result ) {
            //alert(result);
            //alert(result.dupCheckCnt);
            if ( result.dupCheckCnt == "0" ) {
                alert(userId + "는 사용가능한 아이디입니다.");
            } else {
                alert(userId + "는 이미 사용중인 아이디입니다.");
            }
        }
    });
}
```

아이디 중복이 있는지 확인하는 작업

5. 주요 화면 및 코드 설명 (5) 로그인 구현 - 회원인증

회원 휴대폰 인증

```java
// 인증번호 전송
@ResponseBody
@GetMapping("/view/join/authSend")
public Map<String, String> authSend(@RequestParam Map<String, String> param) {
    Map<String, String> map = new HashMap<~>();
    String phone = param.get("phone");
    String authNo = numberGen( len: 4,  dupCd: 1);
    System.out.println("인증번호 :::::: " + authNo);
    userService.authSend(phone, authNo);
    map.put("result", "1");
    return map;
}

// 인증번호 체크
@ResponseBody
@GetMapping("/view/join/checkAuthNo")
public Map<String, String> checkAuthNo(@RequestParam Map<String, String> param) {
    Map<String, String> map = new HashMap<~>();
    String phone = param.get("phone");
    String authNo = param.get("authNo");
    int checkCnt = userService.checkAuthNo(phone,authNo);
    map.put("result", String.valueOf(checkCnt));
    return map;
}
```

```javascript
function authSend() {
    //alert("인증번호전송");
    var phone = $("#phone").val();
    if ( phone == "" ) {
        alert("휴대폰 번호를 입력하고 인증번호 전송을 클릭하세요.");
        return;
    }
    var data = { "phone" : phone }
    $.ajax ({
        url: "/view/join/authSend",
        data: data,
        type:"get",
        success: function ( result ) {
            if ( result.result == "1" ) {
                alert("인증번호가 전송되었습니다.");
            } else {
                alert("오류입니다. 다시 시도하세요.");
            }
        }
    });
}

function authCheck() {
    //alert("인증번호체크");
    var phone = $("#phone").val();
    var authNo = $("#certifi").val();
    if ( phone == "" || authNo == "" ) {
        alert("인증번호를 입력하세요.");
        return;
    }
    var data = { "phone" : phone, "authNo" : authNo }
    $.ajax ({
        url: "/view/join/checkAuthNo",
        data: data,
        type:"get",
        success: function ( result ) {
            if ( result.result == "1" ) {
                alert("인증이 완료되었습니다.");
            } else {
                alert("인증번호가 틀리었지 않습니다.");
            }
        }
```

회원의 휴대폰으로 인증번호를 보내고 인증번호가 맞는지 확인하는 과정

5. 주요 화면 및 코드 설명 (5) 로그인 구현 – 회원가입 및 DB저장

회원가입 처리

```java
// 가입처리
@ResponseBody
@PostMapping("/view/join/joinOk")
public Map<String, String> joinOk(@RequestParam Map<String, String> param) {
    Map<String, String> map = new HashMap<>();
    String userId = param.get("userId");
    String userNm = param.get("userNm");
    String userPw = param.get("userPw");
    String phone = param.get("phone");
    String email = param.get("email");
    userService.insertUser(userId, userNm, userPw, phone, email);
    map.put("result", "1");
    return map;
}

private String numberGen(int len, int dupCd ) {

    Random rand = new Random();
    String numStr = ""; //난수가 저장될 변수

    for(int i=0;i<len;i++) {

        //0~9 까지 난수 생성
        String ran = Integer.toString(rand.nextInt(bound: 10));

        if(dupCd==1) {
            //중복 허용시 numStr에 append
            numStr += ran;
        }else if(dupCd==2) {
            //중복을 허용하지 않을시 중복된 값이 있는지 검사한다
            if(!numStr.contains(ran)) {
                //중복된 값이 없으면 numStr에 append
                numStr += ran;
            }else {
                //생성된 난수가 중복되면 루틴을 다시 실행한다
                i--1;
            }
        }
    }
    return numStr;
}
```

```javascript
function regOk() {
    con = confirm("가입하시겠습니까?");
    if ( con ) {
        var userId = $("#userId").val();
        var userNm = $("#userNm").val();
        var userPw = $("#userPw").val();
        var phone = $("#phone").val();
        var email = $("#email").val();

        var data = { "phone" : phone, "email" : email, "userId" : userId, "userNm" : userNm, "userPw" : userPw}
        $.ajax ({
            url: "/view/join/joinOk",
            data: data,
            type:"post",
            success: function ( result ) {
                if ( result.result == "1" ) {
                    alert("가입이 완료되었습니다.");
                    document.location.href = "/home";
                } else {
                    alert("가입에 실패하였습니다. 다시시도하세요.");
                }
            }
        });
    }
}
```

회원가입을 완료하고 회원의 정보를 DB로 보내는 과정

5. 주요 화면 및 코드 설명 (5) 로그인 구현 - 세션

로그인 및 세션

```java
@Controller
@RequiredArgsConstructor //롬복
public class LoginController {

    @Autowired
    private UserService userService;


    @GetMapping("/view/login/login")
    public String login() { return "view/login/login"; }

    @ResponseBody
    @PostMapping("/view/join/loginProc")
    public Map<String, String> loginProc(@RequestParam Map<String, String> param, HttpServletRequest request) {
        Map<String, String> map = new HashMap<>();
        String userId = param.get("userId");
        String userPw = param.get("userPw");
        User user = userService.getLoginUser(userId, userPw);
        if ( user != null && user.getUserId() != null ) {
            map.put("result", "1");
            HttpSession session = request.getSession();  //세션가져
            session.setAttribute( name: "userId", user.getUserId());
            session.setAttribute( name: "userNm", user.getUserNm());
            session.setAttribute( name: "phone", user.getPhone());
            session.setAttribute( name: "email", user.getEmail());
        } else {
            map.put("result", "0");
        }

        return map;
    }
}
```

```javascript
function loginOk() {
    var userId = $("#userId").val();
    var userPw = $("#userPw").val();

    if ( userId == "" || userPw == "" ) {
        alert("아이디/비밀번호를 입력하세요.");
        return;
    }

    var data = { "userId" : userId, "userPw" : userPw}
    $.ajax ({
        url: "/view/join/loginProc",
        data: data,
        type:"post",
        success: function ( result ) {
            if ( result.result == "1" ) {
                document.location.href = "/home";
            } else {
                alert("로그인에 실패하였습니다. 아이디/비밀번호를 확인하세요.");
            }
        }
    });
}
```

DB에 있는 데이터를 기반으로 로그인 및 세션 활성화

5. 주요 화면 및 코드 설명 (5) 로그인 기능 구현 및 세션

회원약관

회원가입

회원약관과 사용자의 정보를 입력하여 회원가입

5. 주요 화면 및 코드 설명 (5) 로그인 기능 구현 및 세션

로그인

세션

로그인 및 세션기능과 로그아웃 활성화

6. 기대효과

식자재 등록을 통한 냉장고의 식품 유통기한 관리의 편리성

남은 식자재를 활용한 레시피 정보 제공

커뮤니티를 활용한 회원간 정보 공유

출처

http://pptbizcam.co.kr/
https://www.10000recipe.com/
https://www.flaticon.com
https://www.flaticon.com/free-icons/boyicons created by Freepik
https://www.flaticon.com/free-icons/womanicons created by dDara
https://www.flaticon.com/free-icons/cook created by Eucalyp
https://www.flaticon.com/free-icons/shelf-life icons created by Freepik
https://www.flaticon.com/free-icons/project icons created by Uniconlabs
https://www.flaticon.com/free-icons/login icons created by Freepik
https://www.flaticon.com/free-icons/community icons created by Freepik
https://www.flaticon.com/free-icons/java icons created by juicy_fish
https://www.flaticon.com/kr/free-icons/css-3 아이콘 제작자: Freepik
https://www.flaticon.com/kr/free-icons/js 아이콘 제작자: Freepik
https://www.flaticon.com/free-icons/database icons created by Freepik
https://www.flaticon.com/free-icons/crawl icons created by mynamepong
https://www.flaticon.com/free-icons/bootstrap icons created by Vitaly Gorbachev
http://www.newsearth.kr/news/articleView.html?idxno=984