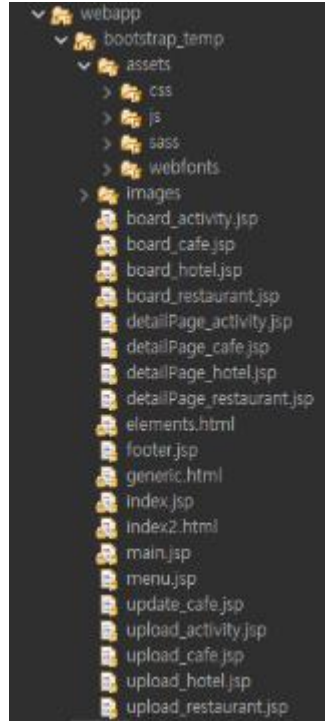
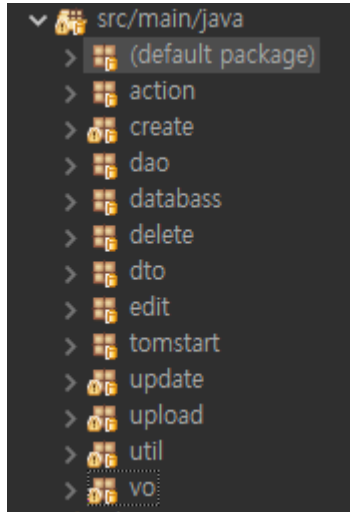


미니 프로젝트

프로젝트명 : MINI TRIP

서경원

파일 구성화 작업 및 데이터로 사용할 vo 제작



| | 카멜표기 | 스네이크표기 | | |
|-------|-----------------|-----------------|-----------|----------|
| 컬럼명 | 식당 : restaurant | 약자 : rt | | 클래스에서 타입 |
| 식당이름 | rtName | rt_name | | String |
| 지역 | rtDishloc | rt_dishloc | | String |
| 주소 | rtAddress | rt_address | | String |
| 별점 | rtRating | rt_rating | | double |
| 소개 | rtDesc | rt_desc | | String |
| 영업시간 | rtTime | rt_time | | String |
| 업종 | rtDishType | rt_dishType | | String |
| 전화번호 | rtPhoneNumber | rt_phone_number | | String |
| 태그 | rtTags | rt_tags | 업로드/수정폼에서 | String |
| 메뉴1 | rtDishItem1 | rt_dish_item1 | | String |
| 가격1 | rtDishPrice1 | rt_dish_price1 | | int |
| 메뉴2 | rtDishItem2 | rt_dish_item2 | | String |
| 가격2 | rtDishPrice2 | rt_dish_price2 | | int |
| 메뉴3 | rtDishItem3 | rt_dish_item3 | | String |
| 가격3 | rtDishPrice3 | rt_dish_price3 | | int |
| 댓글 | rtComments | rt_comments | | String |
| 이미지이름 | | 카페 : cafe | 약자:cf | 클래스에서 타입 |
| 이미지경로 | | | | |
| 카페이름 | cfName | cf_name | | String |
| 지역 | location | cf_location | | String |
| 주소 | address | cf_address | | String |
| 별점 | rating | cf_rating | | double |
| 소개 | cfDescription | cf_description | | String |
| 영업시간 | | cf_time | | String |
| 업종 | category | cf_category | | String |
| 전화번호 | phoneNumber | cf_phone_number | | String |
| 태그 | tags | cf_tags | 업로드/수정폼에서 | String |
| 메뉴1 | cfDishItem1 | cf_dish_item1 | | String |
| 가격1 | cfDishPrice1 | cf_dish_price1 | | int |
| 메뉴2 | cfDishItem2 | cf_dish_item2 | | String |
| 가격2 | cfDishPrice2 | cf_dish_price2 | | int |
| 메뉴3 | cfDishItem3 | cf_dish_item3 | | String |
| 가격3 | cfDishPrice3 | cf_dish_price3 | | int |
| 댓글 | comments | cf_comments | | String |
| 이미지이름 | | cf_image_name | | String |
| 이미지경로 | | cf_image_url | | String |

- 패키지 분류
- webapp에 사용하는 파일 분류
- vo에 사용할 데이터 네이밍

[illegible]

```
public String[] getcf_description() {
    return cf_description;
}

public String[] getcf_image_name() {
    return cf_image_name;
}

public void setcf_image_name(String[] cf_image_name) {
    this.cf_image_name = cf_image_name;
}

public String[] getcf_image_url() {
    return cf_image_url;
}

public void setcf_image_url(String[] cf_image_url) {
    this.cf_image_url = cf_image_url;
}

public void setcf_description(String[] cf_description) {
    this.cf_description = cf_description;
}

public String[] getcf_dish_name() {
    return cf_dish_name;
}

public void setcf_dish_name(String[] cf_dish_name) {
    this.cf_dish_name = cf_dish_name;
}

public int[] getcf_dish_price() {
    return cf_dish_price;
}

public void setcf_dish_price(int[] cf_dish_price) {
    this.cf_dish_price = cf_dish_price;
}

public String[] getcf_dish_image() {
    return cf_dish_image;
}

public void setcf_dish_image(String[] cf_dish_image) {
    this.cf_dish_image = cf_dish_image;
}

public int[] getcf_dish_price() {
    return cf_dish_price;
}
```

```
package databss;

import java.util.HashMap;

public class DataStore {
    private static DataStore instance = new DataStore();
    private Map<Integer, CafeVo2> map = new HashMap<>();

    private DataStore(){}

    public static DataStore getInstance() {
        return instance;
    }

    // 등록하기
    public void put(int key, CafeVo2 cafe) {
        map.put(key, cafe);
    }

    // 1개 데이터 가져오기
    public CafeVo2 get(int key) {
        return map.get(key);
    }

    // 전체 조회
    public Map<Integer, CafeVo2> getAll(){
        return map;
    }

    // 특정 key에 해당하는 데이터 삭제
    public boolean remove(int key) {
        if (map.containsKey(key)) {
            map.remove(key); // 해당 key 값에 대응되는 값을 삭제
            return true; // 삭제 성공
        } else {
            return false; // 해당 key가 없으면 삭제 실패
        }
    }
}
```

데이터로 사용할 vo 제작

```
/**
 * TODO 최초 Map에 데이터 추가
 */
public void addData(){
    for (int i = 0; i < cf_name.length; i++) {
        CafeVo2 vo2 = new CafeVo2();
        vo2.setCf_num(i);
        vo2.setCf_name(cf_name[i]);
        vo2.setCf_location(cf_location[i]);
        vo2.setCf_address(cf_address[i]);
        vo2.setCf_rating(cf_rating[i]);
        vo2.setCf_description(cf_description[i]);
        vo2.setCf_time(cf_time[i]);
        vo2.setCf_category(cf_category[i]);
        vo2.setCf_phone_number(cf_phone_number[i]);
        vo2.setCf_tags(cf_tags[i]);
        vo2.setCf_dish_item1(cf_dish_item1[i]);
        vo2.setCf_dish_price1(cf_dish_price1[i]);
        vo2.setCf_dish_item2(cf_dish_item2[i]);
        vo2.setCf_dish_price2(cf_dish_price2[i]);
        vo2.setCf_dish_item3(cf_dish_item3[i]);
        vo2.setCf_dish_price3(cf_dish_price3[i]);
        vo2.setCf_comments(cf_comments[i]);
        vo2.setCf_image_url(cf_image_url[i]);
        DataStore.getInstance().put(i, vo2);
    }
}

} // end class
```

– 임시 데이터를 서버 실행 시 Map 객체에 저장하기 위해서 사용한 매서드

```
package tomstart;

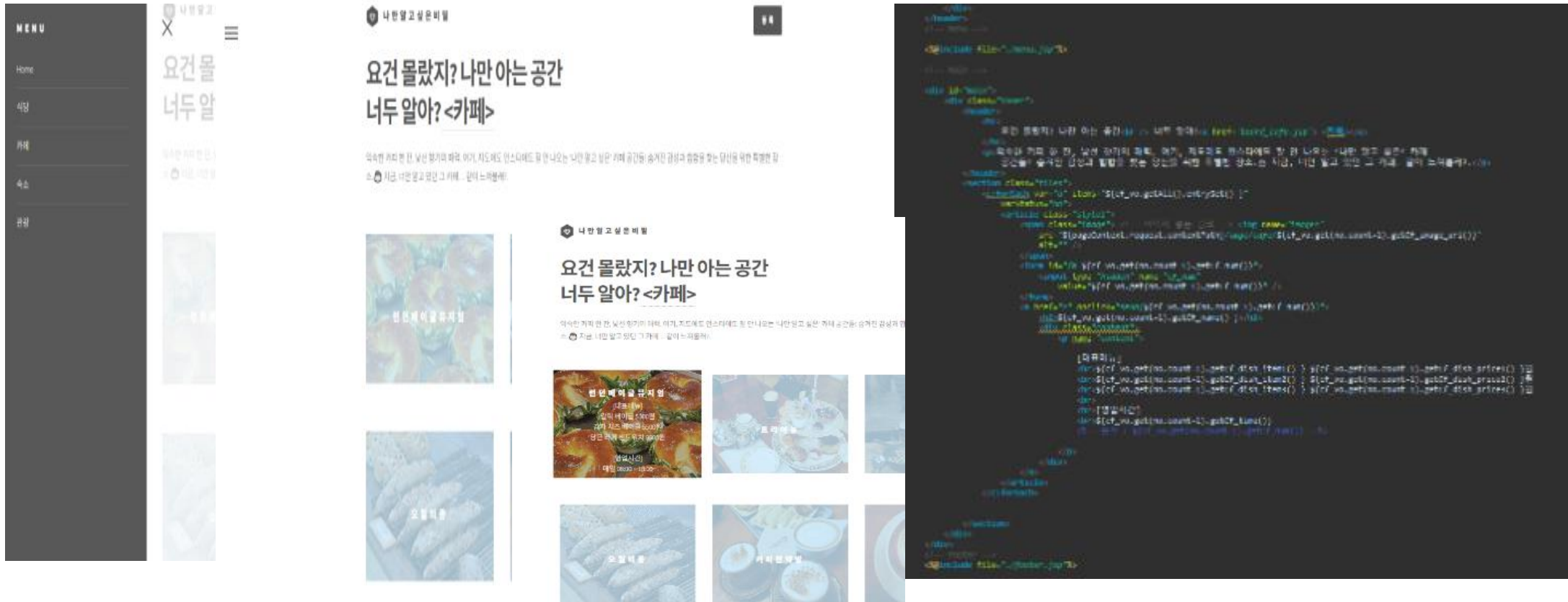
import jakarta.servlet.ServletContextEvent;

@WebListener
public class MyStartDataBass implements ServletContextListener {

    @Override
    public void contextInitialized(ServletContextEvent sce) {
        // Tomcat이 시작될 때 실행됨
        System.out.println("서버 시작됨! 여기서 초기화 작업 수행");
        CafeVo vo = new CafeVo();
        vo.addData();
    }
}
```

– 톰캣 실행시 초기화 작업할수 있는 매서드 오버라이딩해서 임시 데이터 삽입
이후에 톰캣실행시 초기화작업 할수 있는 매서드 위치를 알게됨

구현한 UI



- 전체품을 UI 구성
- jsp 파일을 이용하고 jstl을 사용해서 html 문장을 반복하고 그사이에 el을 사용해서 데이터 처리

구현한 상세페이지보기(R)



```
1 package action;
2 import jakarta.servlet.RequestDispatcher;
3
4 /**
5  * Servlet implementation class detail
6  */
7
8 @WebServlet("/detail_cafe.do")
9 public class DetailCafeAction extends HttpServlet {
10     private static final long serialVersionUID = 1L;
11
12     /**
13      * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
14      */
15     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
16         System.out.println("---상세페이지서비스체크---");
17         request.setCharacterEncoding("UTF-8");
18
19         String cf_num=request.getParameter("cf_num");
20         System.out.println(cf_num);
21
22         request.setAttribute("cf_num", cf_num);
23         String forward_page="bootstrap_tmp/detailPage_cafe.jsp";
24         RequestDispatcher disp=request.getRequestDispatcher(forward_page);
25         disp.forward(request, response);
26     }
27
28 }
```

- 게시판 이미지에서
form 태그에 데이터를 파라미터로
담아서 POST 방식으로 서블릿으로
전달하여 데이터를 가공하고 상세 페
이지에 데이터를 request에 담아서
이를 활용해서 출력

```
1 package action;
2 import jakarta.servlet.RequestDispatcher;
3
4 /**
5  * Servlet implementation class detail
6  */
7
8 @WebServlet("/detail_cafe.do")
9 public class DetailCafeAction extends HttpServlet {
10     private static final long serialVersionUID = 1L;
11
12     /**
13      * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
14      */
15     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
16         System.out.println("---상세페이지서비스체크---");
17         request.setCharacterEncoding("UTF-8");
18
19         String cf_num=request.getParameter("cf_num");
20         System.out.println(cf_num);
21
22         request.setAttribute("cf_num", cf_num);
23         String forward_page="bootstrap_tmp/detailPage_cafe.jsp";
24         RequestDispatcher disp=request.getRequestDispatcher(forward_page);
25         disp.forward(request, response);
26     }
27
28 }
```


구현한 등록페이지 기능추가(C)

카페 게시판 게시물 작성

카페 이름을 입력하세요

명칭

★★★★★

| | | |
|-------------------|-------------|------------------------|
| 지역 입력(주소, 경기, 부산) | 상세주소를 입력하세요 | |
| 원화번호를 입력하세요 | 영업시간을 입력하세요 | 업종을 입력하세요(예: 편의점, 음식점) |
| 메뉴명을 입력하세요 | 가격을 입력하세요 | |
| 메뉴명을 입력하세요 | 가격을 입력하세요 | |
| 메뉴명을 입력하세요 | 가격을 입력하세요 | |

소개를 내용

그림 첨부 (카페 소개 설명 가능)

☐ 파일 선택 ☐ 이미지 파일 업로드

☐ 이미지 선택

```

// Cafe 등록 페이지에 입력된 데이터 처리
@RequestMapping("/cafe/new")
public void newCafe(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    // Cafe 등록 폼에서 입력된 데이터 처리
    String name = request.getParameter("name");
    String location = request.getParameter("location");
    String address = request.getParameter("address");
    String rating = request.getParameter("rating");
    String description = request.getParameter("description");
    String time = request.getParameter("time");
    String category = request.getParameter("category");
    String phone_number = request.getParameter("phone_number");
    String dish_item = request.getParameter("dish_item");
    String dish_price = request.getParameter("dish_price");
    String dish_item2 = request.getParameter("dish_item2");
    String dish_price2 = request.getParameter("dish_price2");
    String dish_item3 = request.getParameter("dish_item3");
    String dish_price3 = request.getParameter("dish_price3");
    String image_url = request.getParameter("image_url");


    // Cafe 등록 데이터베이스에 저장
    Cafe cafe = new Cafe();
    cafe.setName(name);
    cafe.setLocation(location);
    cafe.setAddress(address);
    cafe.setRating(Integer.parseInt(rating));
    cafe.setDescription(description);
    cafe.setTime(time);
    cafe.setCategory(category);
    cafe.setPhone_number(phone_number);
    cafe.setDish_item(dish_item);
    cafe.setDish_price(Integer.parseInt(dish_price));
    cafe.setDish_item2(dish_item2);
    cafe.setDish_price2(Integer.parseInt(dish_price2));
    cafe.setDish_item3(dish_item3);
    cafe.setDish_price3(Integer.parseInt(dish_price3));
    cafe.setImage_url(image_url);

    // Cafe 등록 데이터베이스에 저장
    CafeDAO dao = new CafeDAO();
    dao.insertCafe(cafe);

    // Cafe 등록 성공 메시지 출력
    response.sendRedirect(request.getContextPath() + "/cafe/list");
}

```

카페



카페 정보

상호명:

지역: 서울

주소: 서울

연락처: 010-123-1234

영업시간: 9시

평점: 5.0

레스토랑 소개

맛있는음식점

-등록 페이지에 데이터를 입력하고 jsp에서 form 태그에 데이터를 담아 등록서블릿으로 보내서 서블릿에서 데이터를 받아서 등록하는 작업

-서블릿을 통해 가공된 데이터를 업로드.jsp에 출력하면 위와같이 새로운 데이터로 페이지가 나온다

구현한 수정페이지보기(U)

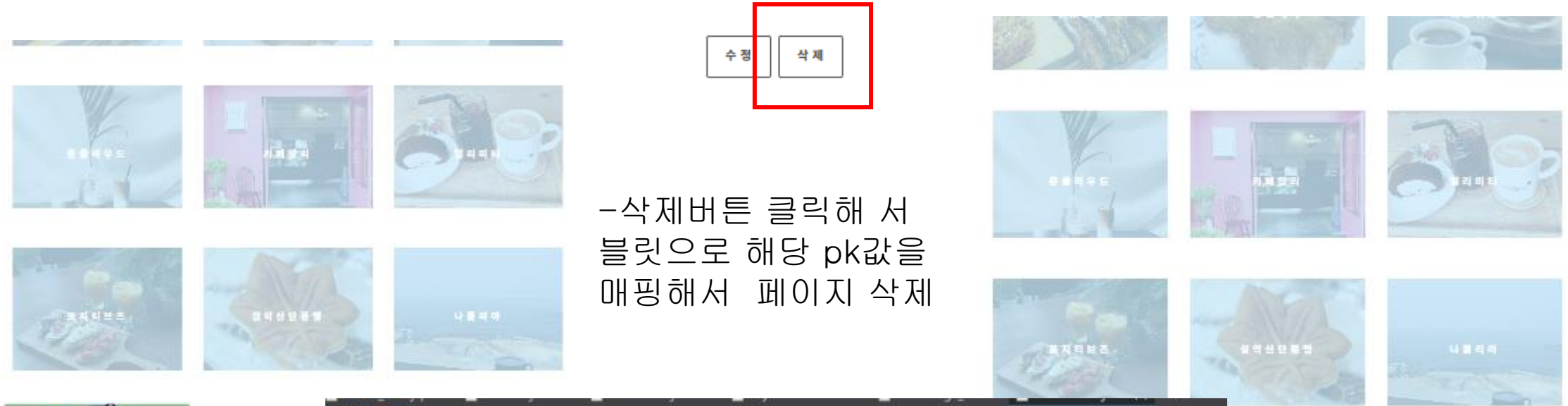
[illegible]

- 상세 페이지에서 수정 버튼을 누르고 받아온 데이터를 input 에 넣고 수정할값을 변경하고 수정버튼을 클릭하여 데이터들을 form에 담아서 서블릿으로 전송하여 최초 생성한 Map객체에서 데이터 수정

[illegible]

- 서블릿을 통해 가공한 데이터를 다시 request에 담아서 상세페이지 jsp에서 el을 사용해서 출력

구현한 페이지 삭제(D)



-삭제버튼 클릭해 서
블릿으로 해당 pk값을
매핑해서 페이지 삭제



```
1 package delete;
2
3 import jakarta.servlet.RequestDispatcher;
4
5 /**
6  * Servlet implementation class DeleteCafe
7  */
8 @WebServlet("/delete_cafe.do")
9 public class DeleteCafe extends HttpServlet {
10     private static final long serialVersionUID = 1L;
11
12     /**
13      * @see HttpServlet#service(HttpServletRequest request, HttpServletResponse response)
14      */
15     protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
16         // TODO Auto-generated method stub
17         System.out.println("-----가짜삭제 서비스 서비스-----");
18         int cf_num=integer.parseInt(request.getParameter("cf_num"));
19         DataStore.getInstance().remove(cf_num);
20
21         DataStore cf_vo = DataStore.getInstance();
22         request.setAttribute("cf_vo", cf_vo);
23         String forward_page="bootstrap_temp/board_cafe.jsp";
24         RequestDispatcher disp=request.getRequestDispatcher(forward_page);
25         disp.forward(request, response);
26         System.out.println(DataStore.getInstance().getAll());
27     }
28 }
29 }
```

구현한 페이지 연결작업

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>Insert title here</title>
8 </head>
9 <body>
10 <nav id="menu">
11 <h2>Menu</h2>
12 <ul>
13 <li><a href="${pageContext.request.contextPath}/bootstrap_temp/main.jsp">Home</a></li>
14 <li><a href="${pageContext.request.contextPath}/bootstrap_temp/board_restaurant.jsp">식당</a></li>
15 <li><a href="${pageContext.request.contextPath}/bootstrap_temp/board_cafe.jsp">카페</a></li>
16 <li><a href="${pageContext.request.contextPath}/bootstrap_temp/board_hotel.jsp">호텔</a></li>
17 <li><a href="${pageContext.request.contextPath}/bootstrap_temp/board_activity.jsp">관광</a></li>
18 <li><a href="${pageContext.request.contextPath}/bootstrap_temp/elements.html">Elements</a></li>
19 </ul>
20 </nav>
21 </body>
22 </html>
```

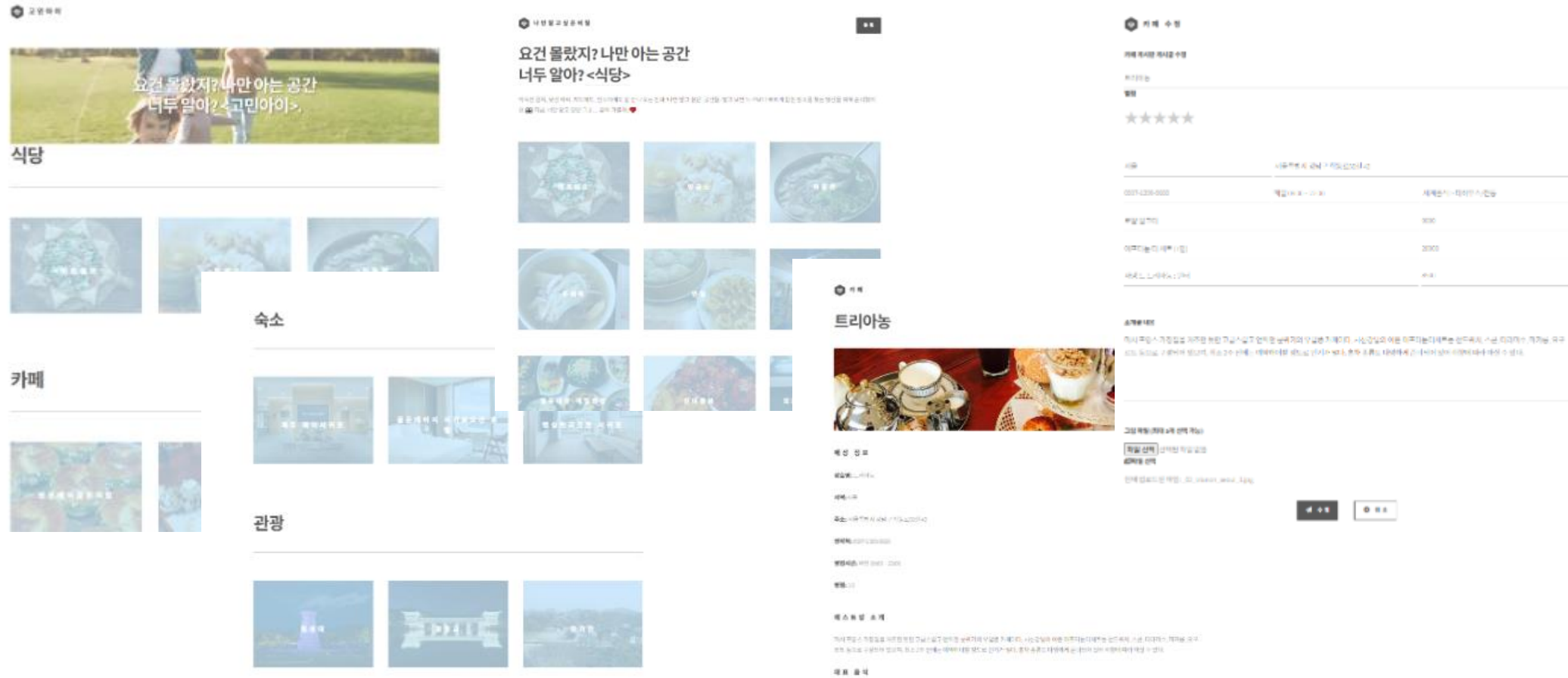
```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>Insert title here</title>
8 </head>
9 <body>
10 <div id="footer">
11 <div class="inner">
12 <section>
13 <h2>내 단골집 알려줘!</h2>
14 </section>
15 <ul class="copyright">
16 <li>&copy; (주) 예민아이</li>
17 <li>Contact us: test@sensitivekids.com</li>
18 <li>Phone: +82 4534 9985</li>
19 <li><a href="#top" class="back-to-top">Back to Top</a></li>
20 </ul>
21 </div>
22 </div>
23 </body>
24 </html>
```

```
78 <!-- Menu -->
79
80 <%@include file="./menu.jsp"%>
81
82 <!-- Main -->
83
```

```
<!-- Footer -->
<%@include file="./footer.jsp"%>
```

-공통으로 사용되는 menu 와 footer는 따로 .jsp 파일로 만들어서 include로 관리함
이후 각페이지로 이동하는 경로 설정

구현한 페이지들 및 소감



- 소감 : 페이지에 데이터를 가져오고 데이터를 서블릿으로 보내는 작업들에 오류 발생도 많고 데이터를 가공해서 출력하는 작업들이 쉽지않았다. 하지만 이번 프로젝트를 통해서 데이터의 이동 및 흐름을 파악 할수 있었고 데이터 가공에 도움이 많이되는 경험이었다.