API 활용한
서울 관광지 소개 웹 개발

서우진

목차

프로젝트 개요

주요 목표

- API를 이용한 관광지 소개를 주제로 웹 개발
- 회원가입 및 로그인 기능
- 게시판 기능

목차 페이지로 돌아가기

프로젝트 개발 일정

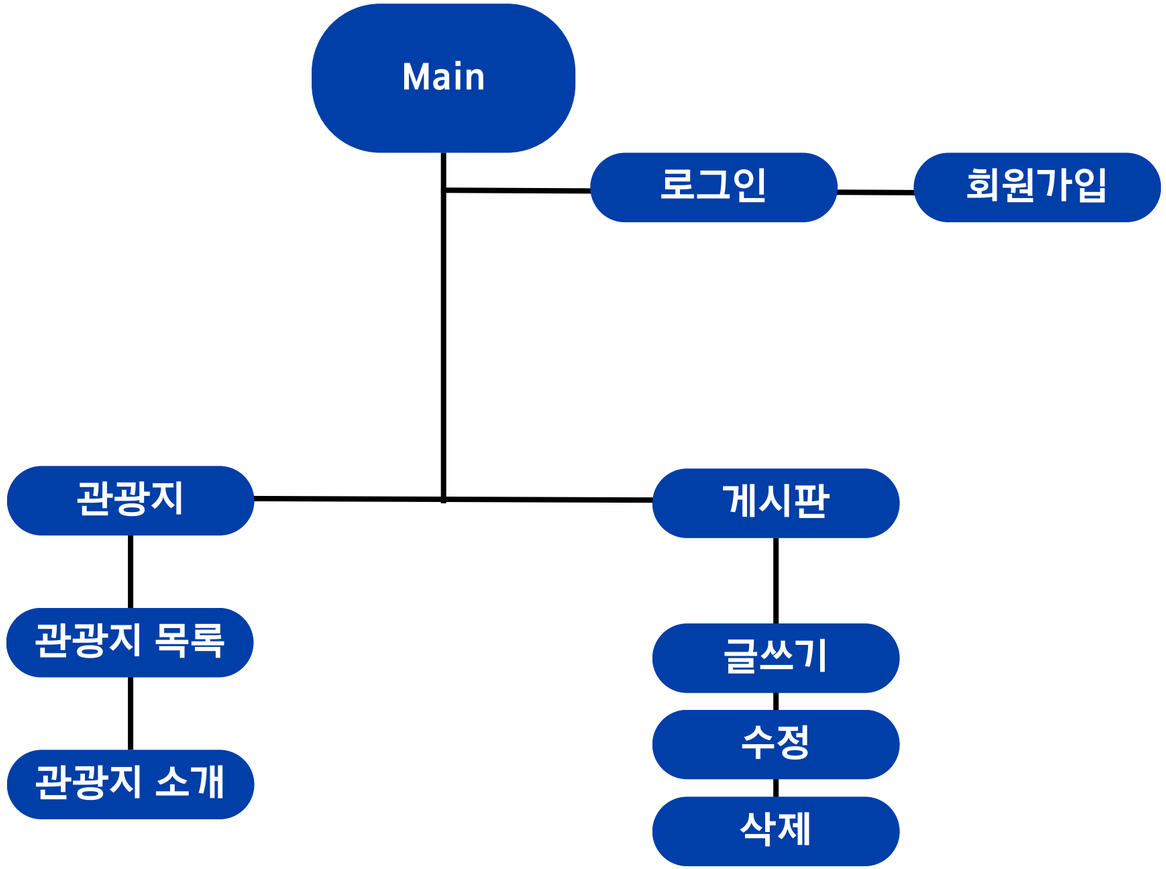
목차 페이지로 돌아가기

	11 / 20 ~ 21	11 / 21 ~ 23	11 / 24 ~ 27	11 / 28 ~ 29	11 / 30 ~ 12 / 1
주제 선정 및 기획					
DB 설계 및 연동					
웹 사이트 및 게시판 생성					
회원가입 및 로그인 기능 생성 및 적용					
수정 및 최종 테스트					

프로젝트 개발 시스템

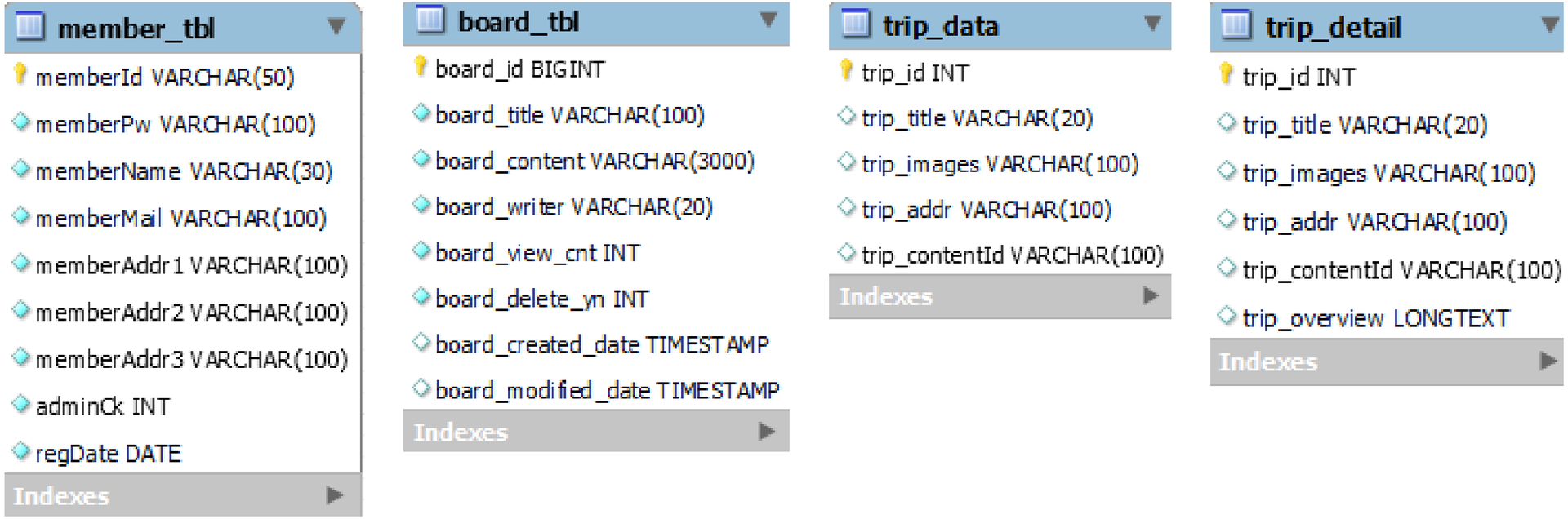
목차 페이지로 돌아가기

웹 메뉴 구조



목차 페이지로 돌아가기

DB구조



목차 페이지로 돌아가기

기능 설명 DB데이터 삽입

```
public static void main(String[] args) {
    try {
        String api_key = "CMeWOR88bn6npOdLcTx0uRLVx9I%2FFCSHaHjKDHGWW40XS4gSyH0IZltowSP%2BL6m4KfSBsE%2Fn3QcMK%2BBR7grxwA%3D%3D"; //본인의 API 키
        String dbUrl = "jdbc:mysql://localhost:3306/webproject?serverTimezone=UTC";
        String username = "root";
        String password = "1234";

        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection connDB = DriverManager.getConnection(dbUrl, username, password);

        String insertQuery = "INSERT INTO trip_data (trip_title, trip_images, trip_addr,trip_contentId) VALUES (?, ?, ?, ?)";
        PreparedStatement pstmt = connDB.prepareStatement(insertQuery);

        for (int pageNo = 1; pageNo <= 7; pageNo++) {
            String apiUrl = "http://apis.data.go.kr/B551011/KorService1/areaBasedList1?numOfRows=12&pageNo=" +
                            pageNo +
                            "&MobileOS=ETC&MobileApp=AppTest&ServiceKey=" +
                            api_key +
                            "&listYN=Y&arrange=A&contentTypeId=12&areaCode=1&sigunguCode=&cat1=A01&cat2=A0101&cat3=&_type=json";
            URL url = new URL(apiUrl);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestMethod("GET");

            BufferedReader reader = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            StringBuilder jsonBuilder = new StringBuilder();
            String line;

String jsonData = jsonBuilder.toString();
JSONObject rootNode = new JSONObject(jsonData);
JSONObject responseNode = rootNode.getJSONObject("response");
JSONObject bodyNode = responseNode.getJSONObject("body");
JSONObject itemsNode = bodyNode.getJSONObject("items");
JSONArray itemArray = itemsNode.getJSONArray("item");

for (int i = 0; i < itemArray.length(); i++) {
    JSONObject itemNode = itemArray.getJSONObject(i);
    String tripTitle = itemNode.getString("title");
    String tripImages = itemNode.optString("firstimage", "");
    String tripAddr = itemNode.optString("addr1", "");
    String tripContentId = itemNode.optString("contentid", "");

    pstmt.setString(1, tripTitle);
    pstmt.setString(2, tripImages);
    pstmt.setString(3, tripAddr);
    pstmt.setString(4, tripContentId);

    pstmt.executeUpdate();
```

각 위치에 해당하는 데이터를 설정

기능 설명 DB데이터 삽입

```
public static void main(String[] args) {
    try {
        String api_key = "CMeWOR88bn6npOdLcTx0uRLVx9I%2FFCSHaHjKDHGWW40XS4gSyH0IZltowSP%2BL6m4KfSBsE%2Fn3QcMK%2BBR7grxwA%3D%3D"; // 본인의 API 키
        String dbUrl = "jdbc:mysql://localhost:3306/webproject?serverTimezone=UTC";
        String username = "root";
        String password = "1234";

        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection connDB = DriverManager.getConnection(dbUrl, username, password);

        String selectQuery = "SELECT trip_contentId FROM trip_data";
        PreparedStatement selectPstmt = connDB.prepareStatement(selectQuery);
        ResultSet resultSet = selectPstmt.executeQuery();

        String insertQuery = "INSERT INTO trip_detail (trip_title, trip_images, trip_addr, trip_contentId, trip_overview) VALUES (?, ?, ?, ?, ?)";
        PreparedStatement pstmt = connDB.prepareStatement(insertQuery);

        while (resultSet.next()) {
            String contentId = resultSet.getString("trip_contentId");
            String apiUrl = "http://apis.data.go.kr/B551011/KorService1/detailCommon1?ServiceKey=" +
            api_key +
            "&contentTypeId=12&contentId=" +
            contentId +
            "&MobileOS=ETC&MobileApp=AppTest&defaultYN=Y&firstImageYN=Y&areacodeYN=Y&catcodeYN=Y&addrinfoYN=Y&mapinfoYN=Y&overviewYN=Y&_type=json";
            URL url = new URL(apiUrl);
            HttpURLConnection conn = (HttpURLConnection) url.openConnection();
            conn.setRequestMethod("GET");

            BufferedReader reader = new BufferedReader(new InputStreamReader(conn.getInputStream()));
            StringBuilder jsonBuilder = new StringBuilder();
            String line;

            while ((line = reader.readLine()) != null) {
                jsonBuilder.append(line);
            }
            reader.close();
            String jsonData = jsonBuilder.toString();
            JSONObject rootNode = new JSONObject(jsonData);
            JSONObject responseNode = rootNode.getJSONObject("response");
            JSONObject bodyNode = responseNode.getJSONObject("body");
            JSONObject itemsNode = bodyNode.getJSONObject("items");
            JSONArray itemArray = itemsNode.getJSONArray("item");

            // 데이터를 가져와서 DB에 저장하는 작업
            for (int i = 0; i < itemArray.length(); i++) {
                JSONObject itemNode = itemArray.getJSONObject(i);
                String tripTitle = itemNode.getString("title");
                String tripImages = itemNode.optString("firstimage", "");
                String tripAddr = itemNode.optString("addr1", "");
                String tripContentId = itemNode.optString("contentid", "");
                String tripOverview = itemNode.optString("overview", "");

                pstmt.setString(1, tripTitle);
                pstmt.setString(2, tripImages);
                pstmt.setString(3, tripAddr);
                pstmt.setString(4, tripContentId);
                pstmt.setString(5, tripOverview);

                pstmt.executeUpdate();
            }
```

trip_data에 저장되어있는 contentId의 데이터를 불러옴

기능 설명 Member(회원가입 및 로그인)

```java
@Data
public class MemberVO {

    //회원 id
    private String memberId;

    //회원 비밀번호
    private String memberPw;

    //회원 이름
    private String memberName;

    //회원 이메일
    private String memberMail;

    //회원 우편번호
    private String memberAddr1;

    //회원 주소
    private String memberAddr2;

    //회원 상세주소
    private String memberAddr3;

    // 관리자 구분(0:일반사용자, 1:관리자)
    private int adminCk;

    //등록일자
    private int regDate;
```

```java
public interface MemberService {

    // 회원가입
    public void memberJoin(MemberVO member);

    // 아이디 중복 검사
    public int idCheck(String memberId);

    /* 로그인 */
    public MemberVO memberLogin(MemberVO member);
```

```java
@Service
public class MemberServiceImpl implements MemberService
    @Autowired
    MemberMapper membermapper;

    @Override
    public void memberJoin(MemberVO member){

        membermapper.memberJoin(member);
    }

    @Override
    public int idCheck(String memberId){

        return membermapper.idCheck(memberId);
    }

    @Override
    public MemberVO memberLogin(MemberVO member){

        return membermapper.memberLogin(member);
    }
```

```xml
<mapper namespace="com.five.mapper.MemberMapper">
<insert id="memberJoin">
  insert into member_tbl VALUES(#{memberId}, #{memberPw}, #{memberName}, #{memberMail}, #{memberAddr1}, #{memberAddr2}, #{memberAddr3}, 0, CURDATE())
</insert>

<!-- 아이디 중복검사 -->
<select id="idCheck" resultType="int">
  select count(*) from member_tbl where memberId = #{memberId}
</select>

<!-- 로그인 -->
    <select id="memberLogin" resultType="com.five.model.MemberVO">
       select memberId, memberName, adminck from member_tbl where memberId = #{memberId} and memberPw = #{memberPw}
    </select>
</mapper>
```

기능 설명 Member(회원가입 및 로그인)

```
$(document).ready(function() {
    //회원가입 버튼(회원가입 기능 작동)
    $(".join_button").click(function() {
        /* 입력값 변수 */
        var id = $('.id_input').val();                  // id 입력란
        var pw = $('.pw_input').val();                  // 비밀번호 입력란
        var pwck = $('.pwck_input').val();              // 비밀번호 확인 입력란
        var name = $('.user_input').val();              // 이름 입력란
        var mail = $('.mail_input').val();              // 이메일 입력란
        var addr = $('.address_input_3').val();         // 주소 입력란

        //$("#join_form").attr("action", "/member/join");
        //$("#join_form").submit();

        /* 아이디 유효성검사 */
        if(id == ""){
            $('.final_id_ck').css('display','block');
            idCheck = false;
        }else{
            $('.final_id_ck').css('display', 'none');
            idCheck = true;
```

```
 /* 로그인 버튼 클릭 메서드 */
 $(".login_button").click(function(){

     /* alert("로그인 버튼 작동"); */
     /* 로그인 메서드 서버 요청 */
     $("#login_form").attr("action", "/member/login");
     $("#login_form").submit();
```

```
<c:if test = "${result == 0 }">
    <div class = "login_warn">사용자 ID 또는 비밀번호를 잘못 입력하셨습니다.</div>
</c:if>
```

사용자 ID 또는 비밀번호를 잘못 입력하셨습니다.

```
/* 다음 주소 연동 */
function execution_daum_address(){

    new daum.Postcode({
        oncomplete: function(data) {
        // 팝업에서 검색결과 항목을 클릭했을때 실행할 코드를 작성하는 부분입니다.

        // 각 주소의 노출 규칙에 따라 주소를 조합한다.
        // 내려오는 변수가 값이 없는 경우엔 공백('')값을 가지므로, 이를 참고하여 분기 한다.
        var addr = ''; // 주소 변수
        var extraAddr = ''; // 참고항목 변수

        //사용자가 선택한 주소 타입에 따라 해당 주소 값을 가져온다.
        if (data.userSelectedType === 'R') { // 사용자가 도로명 주소를 선택했을 경우
            addr = data.roadAddress;
        } else { // 사용자가 지번 주소를 선택했을 경우(J)
            addr = data.jibunAddress;
        }

        // 사용자가 선택한 주소가 도로명 타입일때 참고항목을 조합한다.
        if(data.userSelectedType === 'R'){
            // 법정동명이 있을 경우 추가한다. (법정리는 제외)
            // 법정동의 경우 마지막 문자가 "동/로/가"로 끝난다.
            if(data.bname !== '' && /[동|로|가]$/g.test(data.bname)){
                extraAddr += data.bname;
            }
            // 건물명이 있고, 공동주택일 경우 추가한다.
            if(data.buildingName !== '' && data.apartment === 'Y'){
                extraAddr += (extraAddr !== '' ? ', ' + data.buildingName : data.buildingName);
            }
            // 표시할 참고항목이 있을 경우, 괄호까지 추가한 최종 문자열을 만든다.
            if(extraAddr !== ''){
                extraAddr = ' (' + extraAddr + ')';
            }
             // 주소변수 문자열과 참고항목 문자열 합치기
                addr += extraAddr;
            } else {
                addr += ' ';
            }

            // 우편번호와 주소 정보를 해당 필드에 넣는다.
            $(".address_input_1").val(data.zonecode);
            $(".address_input_2").val(addr);

            // 상세주소 입력란 disabled 속성 변경 및 커서를 상세주소 필드로 이동한다.
            $(".address_input_3").attr("readonly",false);
            $(".address_input_3").focus();


        }
```

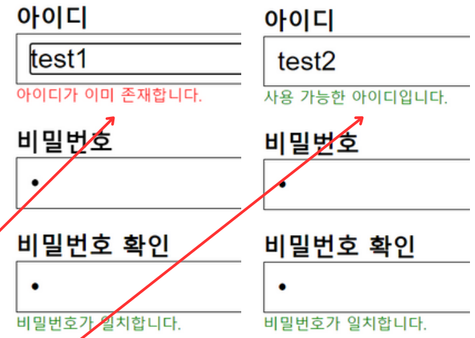
기능 설명 Member(회원가입 및 로그인)

```java
/* 로그인 */
@PostMapping("login")
public String loginPOST(HttpServletRequest request, MemberVO member, RedirectAttributes rttr){

      System.out.println("login 메서드 진입");
      System.out.println("전달된 데이터 : " + member);
    HttpSession session = request.getSession();
    MemberVO lvo = memberservice.memberLogin(member);

    if(lvo == null) {                              // 일치하지 않는 아이디, 비밀번호 입력 경우

        int result = 0;
        rttr.addFlashAttribute("result", result);
        return "redirect:/member/login";

    }

    session.setAttribute("member", lvo);           // 일치하는 아이디, 비밀번호 경우 (로그인 성공)

    return "redirect:/trip/main";

}

/* 메인페이지 로그아웃 */
@GetMapping("logout")
public String logoutMainGET(HttpServletRequest request){

    HttpSession session = request.getSession();

    session.invalidate();

    return "redirect:/trip/main";
```

```java
/* 유효성 검사 통과유무 변수 */
var idCheck = false;              // 아이디
var idckCheck = false;              // 아이디 중복 검사
var pwCheck = false;              // 비번
var pwckCheck = false;              // 비번 확인
var pwckcorCheck = false;           // 비번 확인 일치 확인
var nameCheck = false;              // 이름
var mailCheck = false;              // 이메일
var mailnumCheck = false;           // 이메일 인증번호 확인
var addressCheck = false            // 주소
```

```java
//아이디 중복검사
$('.id_input').on("propertychange change keyup paste input", function() {
        var memberId = $('.id_input').val(); // .id_input에 입력되는 값
        var data = {
            memberId: memberId
        }; // '컨트롤에 넘길 데이터 이름' : '데이터(.id_input에 입력되는 값)'

        $.ajax({
            type: "post",
            url: "/member/memberIdChk",
            data: data,
            success: function(result) {
                if (result != 'fail') {
                    $('.id_input_re_1').css('display', 'block');
                    $('.id_input_re_2').css('display', 'none');
                    idckCheck = true;
                } else {
                    $('.id_input_re_2').css('display', 'block');
                    $('.id_input_re_1').css('display', 'none');
                    idckCheck = false;
                }
```

```java
// 아이디 중복 검사
@PostMapping("memberIdChk")
@ResponseBody
public String memberIdChkPOST(String memberId) {

    int result = memberservice.idCheck(memberId);

            System.out.println("결과값 = " + result);

            if(result != 0) {

                return "fail";  // 중복 아이디가 존재

            } else {

                return "success";   // 중복 아이디 x

            }
```



기능 설명 Post(게시판)

```java
@Data
public class PostVO {
    private Long board_id;
    private String board_title;
    private String board_content;
    private String board_writer;
    private int board_view_cnt;
    private Boolean board_delete_yn;
    private Date board_created_date;
    private Date board_modified_date;
```

```java
public interface PostMapper {

    public void save(PostVO vo);

    PostVO findById(Long id);

    public int update(PostVO vo);

    public void deleteById(Long id);

    List<PostVO> findAll();

    List<PostVO> findAllPageing(Criteria cri);

    public int count();

    public int getTotal();
```

```java
public interface PostService {

    public Long savePost(PostVO vo);

    public Long updatePost(PostVO vo) ;

    public Long deletePost(Long id);

    public List<PostVO> findAllPost();

    public PostVO findPostById(Long id);

    public List<PostVO> findAllPageing(Criteria cri);

    public int getTotal();
```

```java
@Service
public class PostServiceImpl implements PostService{
    @Autowired
    PostMapper postMapper;

    @Override
    public Long savePost(PostVO vo) {
        postMapper.save(vo);
        return vo.getBoard_id();
    }



    @Override
    public Long updatePost(PostVO vo){
        postMapper.update(vo);
        return vo.getBoard_id();
    }

    @Override
    public Long deletePost(Long id) {
        postMapper.deleteById(id);
        return id;
    }

    @Override
    public List<PostVO> findAllPost(){
        return postMapper.findAll();
    }

    @Override
    public PostVO findPostById(Long id) {
        return postMapper.findById(id);
    }

    @Override
    public List<PostVO> findAllPageing(Criteria cri) {

        return postMapper.findAllPageing(cri);
    }
     @Override
        public int getTotal() {

            return postMapper.getTotal();
        }
```

```xml
<!-- 게시글 저장 -->
<insert id="save" parameterType="com.five.model.PostVO">
    INSERT INTO board_tbl (
    board_title,
    board_content,
    board_writer,
    board_view_cnt,
    board_delete_yn,
    board_created_date,
    board_modified_date
    ) VALUES (
        #{board_title},
        #{board_content},
        #{board_writer},
        0,
        0,
        now(),
        now()
    )
</insert>
<!-- 게시글 총 개수 -->
<select id="getTotal" resultType="int">
    select count(*) from board_tbl
</select>

<!-- 게시글 상세정보 조회 -->
<select id="findById" parameterType="Long" resultType="com.five.model.PostVO">
    SELECT * FROM board_tbl WHERE board_id = #{board_id}
</select>


  <!-- 게시글 수정 -->
  <update id="update" parameterType="com.five.model.PostVO">
      UPDATE board_tbl
      SET
          board_title =#{board_title},
          board_content=#{board_content},
          board_modified_date=now()
      WHERE
          board_id = #{board_id}
  </update>

  <!-- 게시글 삭제 -->
  <delete id="deleteById" parameterType="Long">
      delete from board_tbl where board_id=#{board_id}
  </delete>

  <!-- 게시글 리스트 조회 -->
  <select id="findAll" resultType="com.five.model.PostVO">
      SELECT * FROM board_tbl
  </select>

  <!-- 게시글 페이징 -->
  <select id="findAllPageing" resultType="com.five.model.PostVO">
      SELECT
          board_id, board_writer,board_title, board_content,board_created_date,board_modified_date
      FROM
          board_tbl
      ORDER BY
          board_id DESC
      LIMIT #{skip} , #{amount}
  </select>
```

기능 설명 Post(게시판)

```java
public class Criteria {
    /* 현재 페이지 */
    private int pageNum;

    /* 한 페이지 당 보여질 게시물 갯수 */
    private int amount;

    /* 스킵 할 게시물 수( (pageNum-1) * amount ) */
    private int skip;

    /* 기본 생성자 -> 기본 세팅 : pageNum = 1, amount = 10 */
    public Criteria() {
        this(1,10);
        this.skip = 0;
    }

    /* 생성자 => 원하는 pageNum, 원하는 amount */
    public Criteria(int pageNum, int amount) {
        this.pageNum = pageNum;
        this.amount = amount;
        this.skip = (pageNum-1)*amount;
    }

    public int getPageNum() {
        return pageNum;
    }

    public void setPageNum(int pageNum) {
        this.skip=(pageNum-1)*this.amount;
        this.pageNum = pageNum;
    }

    public int getAmount() {
        return amount;
    }

    public void setAmount(int amount) {
        this.skip=(this.pageNum-1)*amount;
        this.amount = amount;
    }

    public int getSkip() {
        return skip;
    }

    public void setSkip(int skip) {
        this.skip = skip;
    }
```

```java
@Data
public class PageMakerDTO {
    /* 시작 페이지 */
    private int startPage;

    /* 끝 페이지 */
    private int endPage;

    /* 이전 페이지, 다음 페이지 존재유무 */
    private boolean prev, next;

    /*전체 게시물 수*/
    private int total;

    /* 현재 페이지, 페이지당 게시물 표시수 정보 */
    private Criteria cri;

    /* 생성자 */
    public PageMakerDTO(Criteria cri, int total) {

        this.cri = cri;
        this.total = total;

        /* 마지막 페이지 */
        this.endPage = (int)(Math.ceil(cri.getPageNum()/10.0))*10;
        /* 시작 페이지 */
        this.startPage = this.endPage - 9;

        /* 전체 마지막 페이지 */
        int realEnd = (int)(Math.ceil(total * 1.0/cri.getAmount()));

        /* 전체 마지막 페이지(realend)가 화면에 보이는 마지막페이지(endPage)보다 작은 경우, 보이는 페이지(endPage) 값 조정 */
        if(realEnd < this.endPage) {
            this.endPage = realEnd;
        }

        /* 시작 페이지(startPage)값이 1보다 큰 경우 true */
        this.prev = this.startPage > 1;

        /* 마지막 페이지(endPage)값이 1보다 큰 경우 true */
        this.next = this.endPage < realEnd;
```

```jsp
</table>
<div class="pageInfo_wrap">
    <div class="pageInfo_area">
    <ul id="pageInfo_1" class="pageInfo_1">
        <!-- 이전페이지 버튼 -->
        <c:if test="${pageMake.prev}">
            <li class="pageInfo_btn previous"><a href="/post/list?pageNum=${pageMake.startPage-1}&amount=10">Previous</a></li>
        </c:if>

        <!-- 각 번호 페이지 버튼 -->
        <c:forEach var="num" begin="${pageMake.startPage}" end="${pageMake.endPage}">
            <li class="pageInfo_btn ${pageMake.cri.pageNum == num ? 'active':''}">
                <a href="/post/list?pageNum=${num}&amount=10">${num}</a>
            </li>
        </c:forEach>

        <!-- 다음페이지 버튼 -->
        <c:if test="${pageMake.next}">
            <li class="pageInfo_btn next"><a href="/post/list?pageNum=${pageMake.endPage+1}&amount=10">Next</a></li>
        </c:if>
    </ul>
</div>

</div>
<input type="hidden" name="pageNum" value="${pageMake.cri.pageNum }">
<input type="hidden" name="amount" value="${pageMake.cri.amount }">
```

startPage와 endPage의 값을 가지고 페이지 번호를 화면에 출력

1 2

기능 설명 Post(게시판)

게시글 목록

```
<c:forEach items="${posts}" var="post" varStatus="loop">
    <tr>
        <td><c:out value="${post.board_id}" /></td>
        <td><c:out value="${post.board_writer}" /></td>
        <td><a class="move" href='/post/view?board_id=<c:out value="${post.board_id}"/>'>
                <c:out value="${post.board_title}" />
        </a></td>
        <td><fmt:formatDate pattern="yyyy-MM-dd"
                value="${post.board_created_date}" /></td>
        <td><fmt:formatDate pattern="yyyy-MM-dd"
                value="${post.board_modified_date}" /></td>
    </tr>
</c:forEach>
```

게시글 정보

```
<div class="btn_wrap">
    <a class="btn" id="list_btn">목록 페이지</a>
    <a class="btn" id="update_btn">수정 완료</a>
    <a class="btn" id="delete_btn">삭제</a>
    <a class="btn" id="cancel_btn">수정 취소</a>
</div>
<form id="infoForm" action="/post/update" method="get">
    <input type="hidden" id="board_id" name="board_id" value='<c:out value="${pageInfo.board_id}"/>'>
</form>
<script>
    let form = $("#infoForm"); // 페이지 이동 form(리스트 페이지 이동, 조회 페이지 이동)
    let mForm = $("#updateForm"); // 페이지 데이터 수정 from

    /* 목록 페이지 이동 버튼 */
    $("#list_btn").on("click", function(e) {
        if (confirm("목록으로 이동하겠습니다")) {
            form.attr("action", "/post/list");
            form.submit();
        }
    });

    $("#update_btn").on("click", function(e) {
        if (confirm("수정되었습니다")) {
            mForm.submit();
        }
    });

    /* 취소 버튼 */
    $("#cancel_btn").on("click", function(e) {
        if (confirm("수정을 취소하겠습니다")) {
            form.attr("action", "/post/view");
            form.submit();
        }
    });
    /* 삭제 버튼 */
    $("#delete_btn").on("click", function(e) {
        if (confirm("삭제되었습니다")) {
            form.attr("action", "/post/delete");
            form.attr("method", "post");
            form.submit();
        }
    });
</script>
```

게시글 수정

```
<table class="tb tb_row">
    <div class="input_wrap">
        <label>등록일</label> <input name="board_created_date"
            readonly="readonly" value="${pageInfo.board_created_date}">
    </div>
    <div class="input_wrap">
        <label>수정일</label> <input name="board_modified_date"
            readonly="readonly" value="${pageInfo.board_modified_date}">
    </div>
    <div class="input_wrap">
        <label>제목</label> <input name="board_title" readonly="readonly"
            value="${pageInfo.board_title}">
    </div>
    <div class="input_wrap">
        <label for="board_writer">작성자</label> <input id="board_writer"
            name="board_writer" readonly="readonly"
            value="${pageInfo.board_writer}">
    </div>
    <div class="input_wrap">
        <label>내용</label>
        <textarea rows="3" name="board_content" readonly="readonly">${pageInfo.board_content}</textarea>
    </div>
</table>
```

게시글 작성

```
<tbody>
    <tr>
        <th scope="row">등록일</th>
        <td colspan="3"><input type="text" id="createdDate"
            name="board_created_date" readonly
            value="<%=new java.text.SimpleDateFormat("yyyy-MM-dd").format(new java.util.Date())%>" /></td>
    </tr>
    <tr>
        <th>제목 <span class="es">필수 입력</span></th>
        <td colspan="3"><input type="text" id="title" name="board_title" maxlength="50" placeholder="제목을 입력해 주세요." /></td>
    </tr>
    <tr>
        <th>이름 <span class="es">필수 입력</span></th>
        <td colspan="3"><input type="text" id="writer" name="board_writer" maxlength="10"
            placeholder="이름을 입력해 주세요." /></td>
    </tr>
    <tr>
        <th>내용 <span class="es">필수 입력</span></th>
        <td colspan="3">
            <textarea id="content" name="board_content" cols="150" rows="10" placeholder="내용을 입력해 주세요.">
            </textarea>
        </td>
    </tr>
</tbody>
```

기능 설명 Trip(관광지)

```java
@Data
public class TripVO {
    private String trip_title;
    private String trip_images;
    private String trip_addr;
    private String trip_contentId;
    private String trip_overview;
```

```java
public interface TripMapper {
    List<TripVO> getAllTrips();

    List<TripVO> getFindTrips();

    TripVO Findbydetail(String contentId);
    }
```

```java
public interface TripService {

    public List<TripVO> getAllTrips();

    public List<TripVO> getFindTrips();

    public TripVO Findbydetail(String contentId);
```

```java
@Service
public class TripServiceImpl implements TripService{
    @Autowired
    TripMapper tripMapper;

    @Override
    public List<TripVO> getAllTrips() {

        return tripMapper.getAllTrips();
    }

    @Override
    public List<TripVO> getFindTrips() {

        return tripMapper.getFindTrips();
    }

    @Override
    public TripVO Findbydetail(String contentId) {

        return tripMapper.Findbydetail(contentId);
    }
```

```xml
  <select id="getAllTrips" resultType="com.five.model.TripVO">
      SELECT trip_title, trip_images, trip_addr, trip_contentId FROM trip_data
   </select>

   <select id="getFindTrips" resultType="com.five.model.TripVO">
      SELECT trip_title, trip_images, trip_addr, trip_contentId, trip_overview FROM trip_detail
   </select>

   <select id="Findbydetail" parameterType="String" resultType="com.five.model.TripVO">
      SELECT * FROM trip_detail WHERE trip_contentId = #{trip_contentId}
   </select>
/mapper>
```

기능 설명 Trip(관광지)

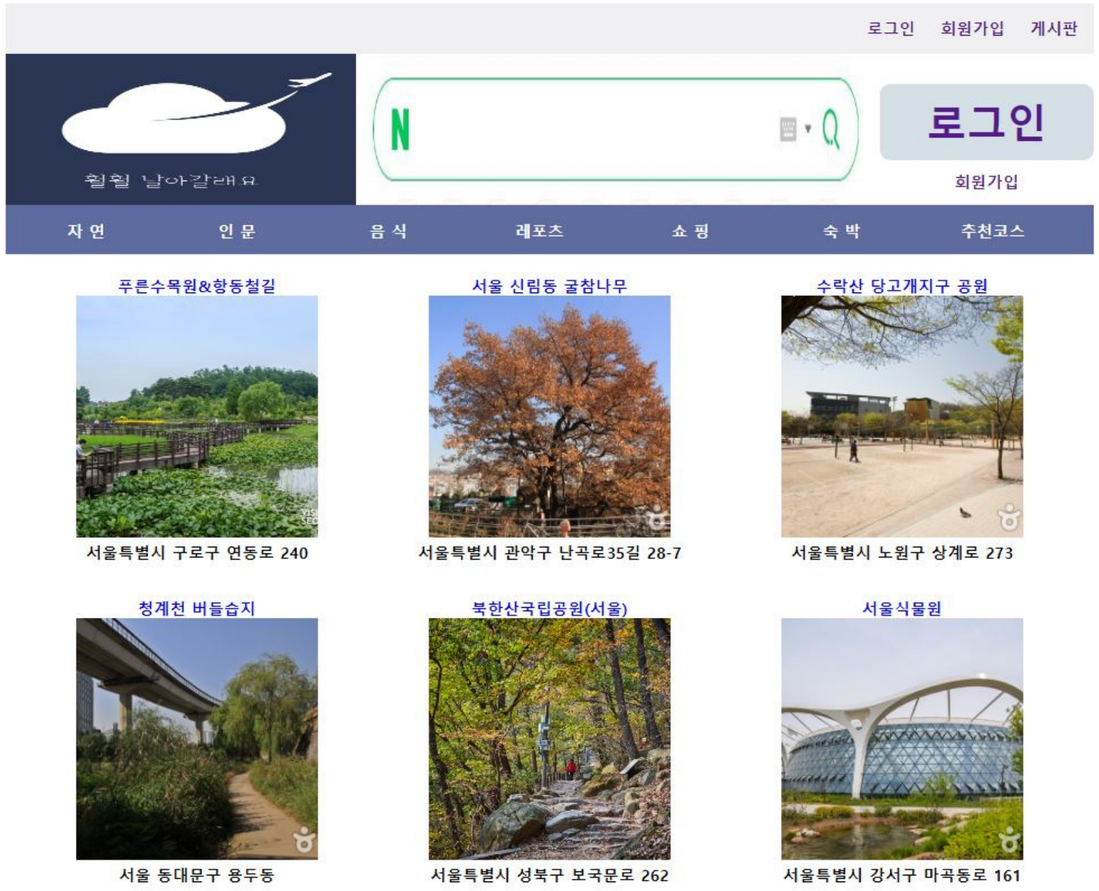
```java
@GetMapping("/main")
public String mainPage(Model model) {
    List<TripVO> allTrips = tripService.getAllTrips();

    // 데이터가 6개 미만인 경우, 전체 데이터를 전송
    if (allTrips.size() <= 6) {
        model.addAttribute("trips", allTrips);
    } else {
        // 데이터가 6개 이상인 경우, 랜덤으로 6개만 선택하여 전송
        List<TripVO> randomTrips = new ArrayList<>();
        Random random = new Random();
        Set<Integer> selectedIndexes = new HashSet<>();

        while (selectedIndexes.size() < 6) {
            int randomIndex = random.nextInt(allTrips.size());
            if (!selectedIndexes.contains(randomIndex)) {
                selectedIndexes.add(randomIndex);
                System.out.println(selectedIndexes);
                randomTrips.add(allTrips.get(randomIndex));
            }
        }

        model.addAttribute("trips", randomTrips);
    }
```

```java
@GetMapping("trips")
public String getAllTrips(Model model) {
    List<TripVO> trips = tripService.getAllTrips();
    model.addAttribute("trips",trips);
    System.out.println(trips);
    return "/trip/displayTrips";
}
@GetMapping("detail")
public String getFindTrips(String trip_contentId,Model model) {
    TripVO detail = tripService.Findbydetail(trip_contentId);
    model.addAttribute("details",detail);
    System.out.println(detail);
    return "/trip/displaydetail";
}
```

```jsp
<c:forEach var="trip" items="${trips}">
    <div class="trip-info">
        <p>
            <a href="/trip/detail?trip_contentId=${trip.trip_contentId}">${trip.trip_title}</a>
        </p>
        <c:choose>
            <c:when test="${empty trip.trip_images}">
                <img class="trip-images"
                    src="https://lh3.googleusercontent.com/oucDcDAOVG1fLnm_XdisT0Sp3c72sI8JuR_PWKWmeXl
                    alt="${trip.trip_title}" />
            </c:when>
            <c:otherwise>
                <img class="trip-images" src="${trip.trip_images}"
                    alt="${trip.trip_title}" />
            </c:otherwise>
        </c:choose>
        <p>${trip.trip_addr}</p>
    </div>
</c:forEach>
```

```javascript
var tripAddr = "${details.trip_addr}";
var tripTitle = "${details.trip_title}";
var mapContainer = document.getElementById('map'), // 지도를 표시할 div
mapOption = {
    center : new kakao.maps.LatLng(33.450701, 126.570667), // 지도의 중심좌표
    level : 3
// 지도의 확대 레벨
};

// 지도를 생성합니다
var map = new kakao.maps.Map(mapContainer, mapOption);

// 주소-좌표 변환 객체를 생성합니다
var geocoder = new kakao.maps.services.Geocoder();

// 주소로 좌표를 검색합니다
geocoder
        .addressSearch(
                tripAddr,
                function(result, status) {

                    // 정상적으로 검색이 완료됐으면
                    if (status === kakao.maps.services.Status.OK) {

                        var coords = new kakao.maps.LatLng(result[0].y,
                                result[0].x);

                        // 결과값으로 받은 위치를 마커로 표시합니다
                        var marker = new kakao.maps.Marker({
                            map : map,
                            position : coords
                        });

                        // 인포윈도우로 장소에 대한 설명을 표시합니다
                        var infowindow = new kakao.maps.InfoWindow(
                                {
                                    content : '<div style="width:150px;text-align:center;padding:6px 0;">'
                                            + tripTitle + '</div>'
                                });
                        infowindow.open(map, marker);

                        // 지도의 중심을 결과값으로 받은 위치로 이동시킵니다
                        map.setCenter(coords);
                    }
                });
```

구현 페이지 메인 및 카테고리 페이지



구현 페이지 상세 페이지

장소명

강서습지생태공원

주소

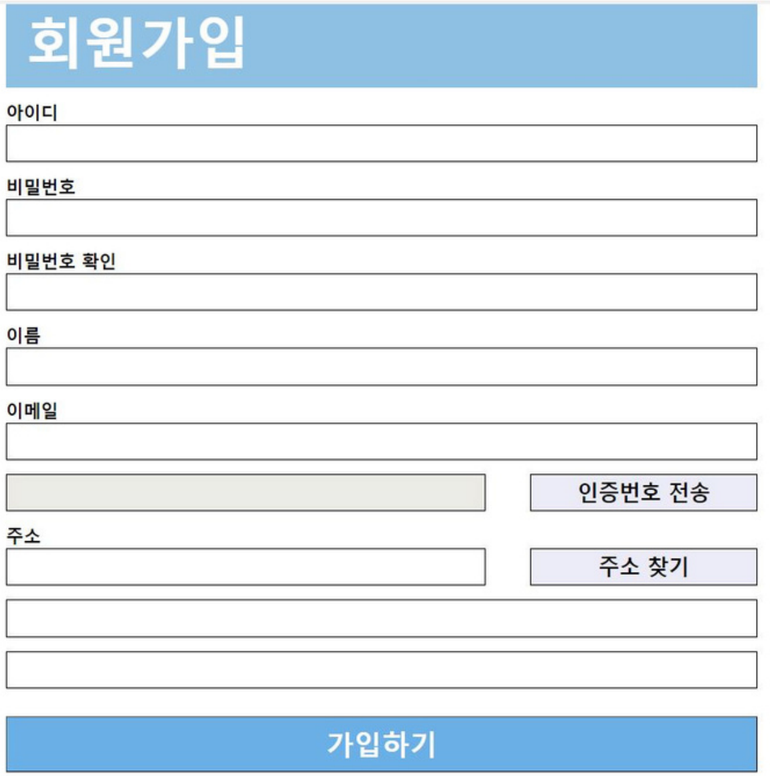
서울특별시 강서구 양천로27길 279-23

장소 소개

방화대교 남쪽 끝에서 행주대교 남쪽 끝 사이 한강 둔치에 있는 생태공원이다. 서울시 생태관광명소 30개소 중 하나로 선정된 강서습지생태공원은 담수지·저습지 등을 조성하고 습생·수생식물을 심은 습지공원으로 2002년 7월에 개원되었다. 하중도, 자연관찰로, 습초지 등 습지 생태계를 복원하여 하천의 자연정화 기능을 제공한다. 갈대밭·버드나무숲이 어우러진 습지를 가로질러 두 곳의 탐방로와 철새 조망대 등이 있다. 여름과 겨울에는 철새들이 찾아들며, 다양한 생태학습 프로그램을 운영하고 있다. 또한 방화대교 하부, 가족 피크닉장 앞, 행주대교 하부에는 체력단련기구 등을 갖춰 시민의 여가활동과 스포츠 공간으로서의 기능을 하고있다. * 면적 : 370,000㎡

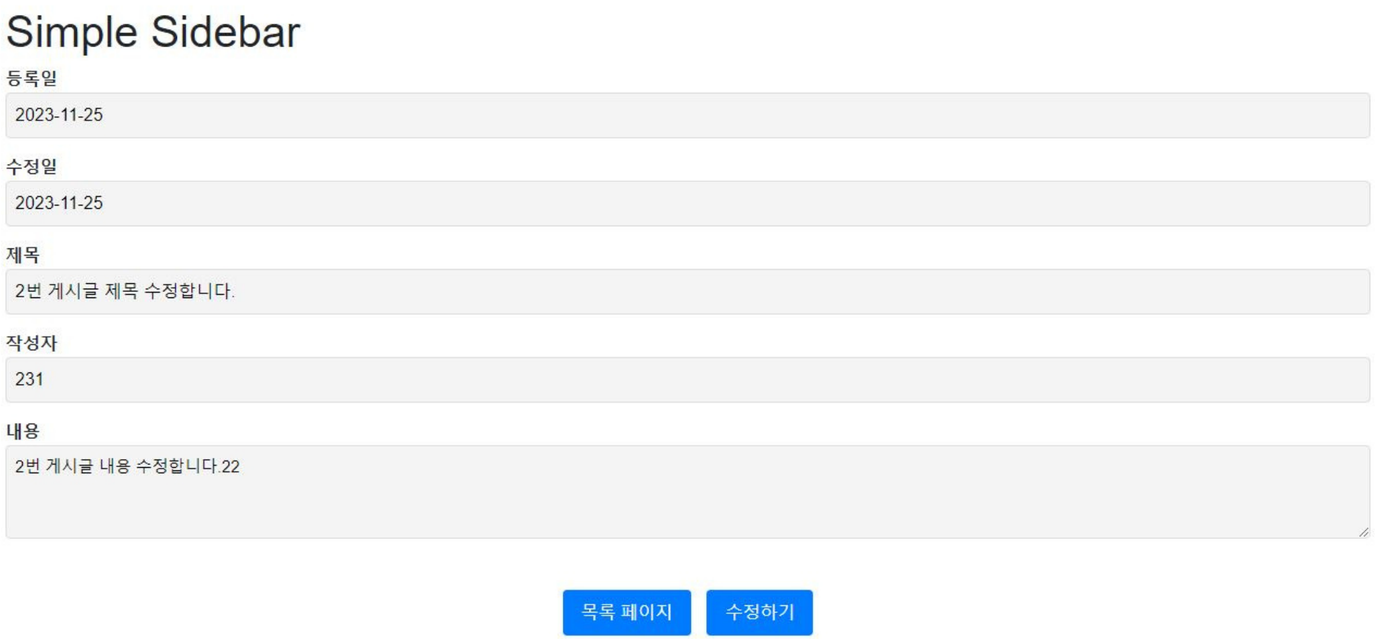
찾아가는 길

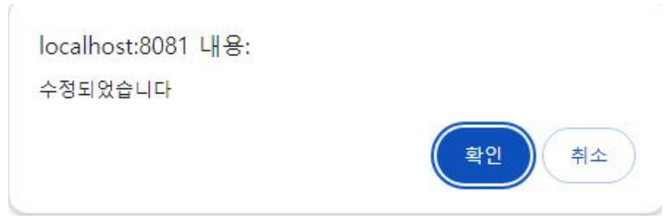
구현 페이지 로그인 및 회원가입 페이지



구현 페이지 게시판 페이지

구현 페이지 게시판 상세보기 페이지





감사합니다

Thank you