

Profile



조성철 (Cho Seong Cheol)

1994. 8. 16

Education

2018. 12~1 2018년 NCS 기반 소프트웨어 웹 개발자 초급 과정 수료
2018. 6~8 2018년 NCS 기반 소프트웨어 웹 개발자 과정 수료
2019. 2 한남대학교 경영정보학 졸업 (학사)

Work Experiences

2018. 9~12 (주)유미테크 (인턴)
- 분산분석시스템 플랫폼 개발 프로젝트 참여
- TrustAP 프로젝트 참여
- 분산처리데이터베이스 및 시스템 구축 참여 (hadoop, munge, slurm 등 구축)

Projects

2018. 1 유지보수관리시스템 개발
2018. 3~6 (주)플랜아이 클라우드 기반 법인차량관리시스템 개발
2018. 8 SD교육그룹 포털 시스템 개발

Skills

개발언어 : Java, JSP, Oracle, Html5, XML, CSS3, JavaScript, Ajax, jQuery
프레임워크 : Spring4.0, ibatis, log4j, Bootstrap, junit3
OS : Ubuntu16.04, centos7, raspberryPi
개발 툴 : Eclipse, STS, SQL Developer, github, Postman
기타 : Azure 클라우드

Profile



조성철 (Cho Seong Cheol)

1994. 8. 16

Award

2018. 5 전국 대학생 IT 프로젝트 경연대회 최우수상

Certificates

2018. 5 정보처리기사 취득
2018. 12 TOEIC Speaking Lv.6 취득

“조성철” 은...

끈기

도전

리더십

끈기 : 포기는 없다. 책임감으로 임무 완수

도전 : 인문에서 공학, 경영 + IT, 신기술 경험

리더십 : 실제 기업과 프로젝트에서 PM, 소통(팀원, 고객), 성공적인 결과물

Index

PART #1. 클라우드 컴퓨팅 기반 법인차량관리시스템 개발

프로젝트 개요

담당 파트

PART #2. 반응형 포털 시스템 개발

프로젝트 개요

담당 파트

PART #3. 인턴 참여 프로젝트

참여 프로젝트 및 담당 파트

01

클라우드 컴퓨팅 기반 법인차량관리시스템 개발

01

개발 개요



 **캡스톤 디자인**
한남대학교  사업단

정보시스템




plan I

01

개발 개요




개발 도구
AWS 클라우드 플랫폼
Eclipse neon2
Workbench 6.3




WAS Server
tomcat 7




운영체제
Window 10



DBMS
MySQL



개발 언어
Java8, JSP, SQL,
Javascript, html5, css3



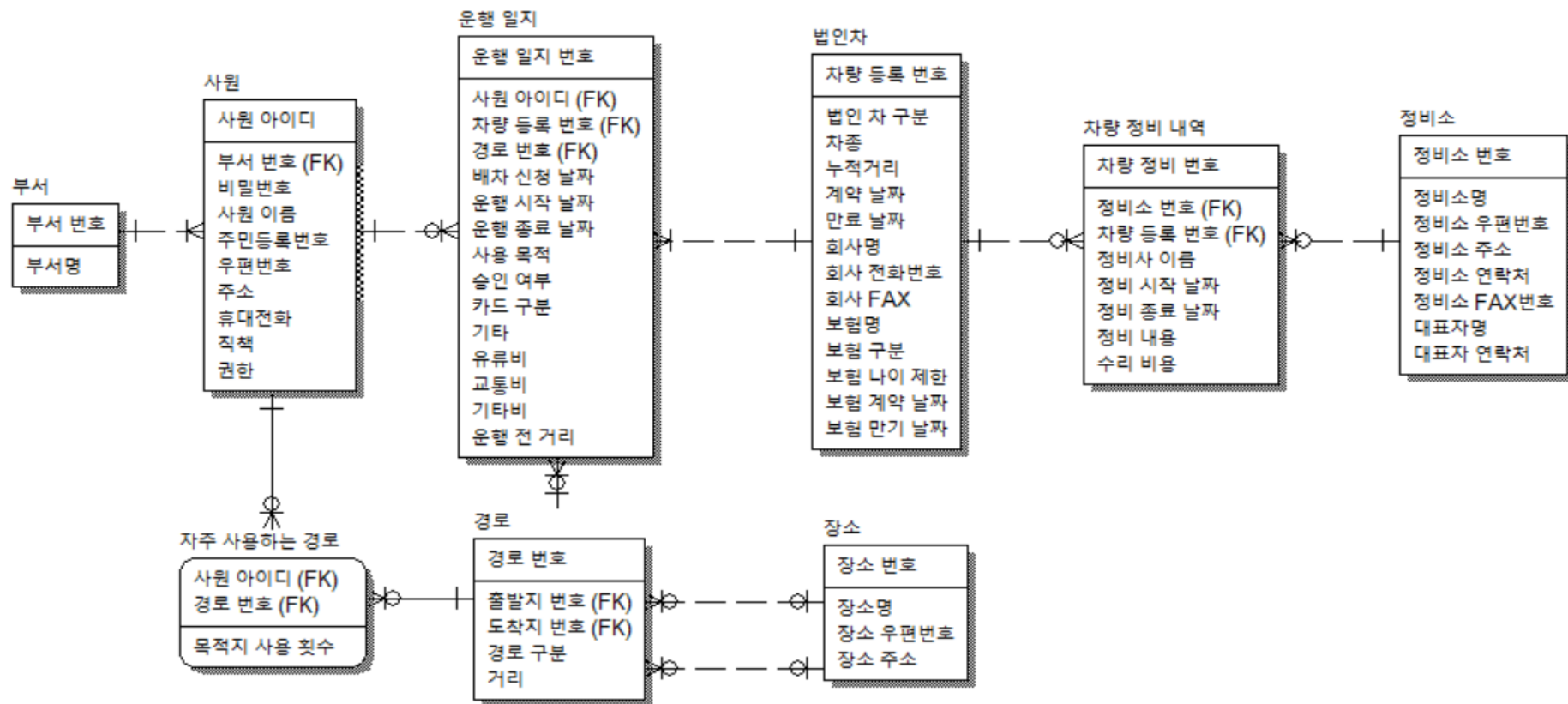
개발 패턴
MVC2
Action Factory

01

개발 개요

데이터 모델링

E-R Diagram



01

개발 개요

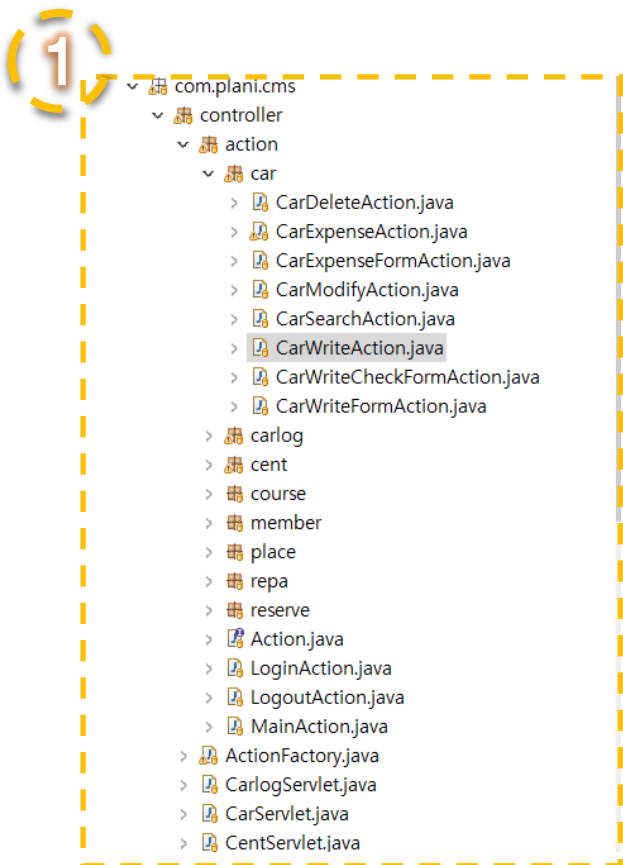
데이터 모델링

〔 CRUD Matrix 〕

Process Entity	부서 등록	사원 등록	법인 차 등록	정비소 등록	정비 내역 등록	정비 내역 조회	장소 등록	경로 등록	배차 신청	배차 승인	운행일지 작성	운행일지 조회
부서	CRUD	R										
사원		CRUD				R			R		R	R
법인 차			CRUD		R	R			R		R	R
정비소				CRUD	R							
차량 정비 내역					CUD	R						
장소							CRUD	R				
경로								CRUD	R		R	
운행일지									CUD	R	UD	R
자주 사용 하는 경로									R		CU (자동 생성)	

01

개발 개요



mvc2 action으로 구현



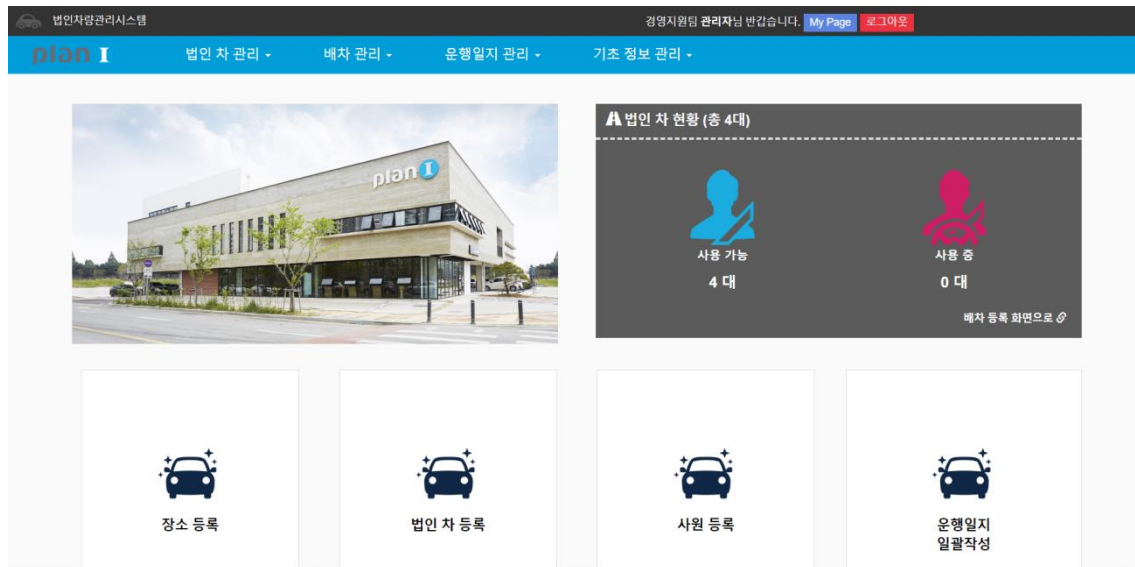
1 Action을 각 기능별로 분류하여 코딩

2 ActionFactory 객체만 사용하게 싱글톤 패턴으로 설계했고, servlet 을 통해 action command 이름을 받아서 호출

해당 디자인 패턴으로 개발을 했을 때, **유지보수**는 편리했지만, 각 객체의 value를 받을 때마다 action 클래스에 모두 적는 등 **반복 되는 코드**가 많아 개발 시간이 많이 소요됐습니다.

01

담당 파트

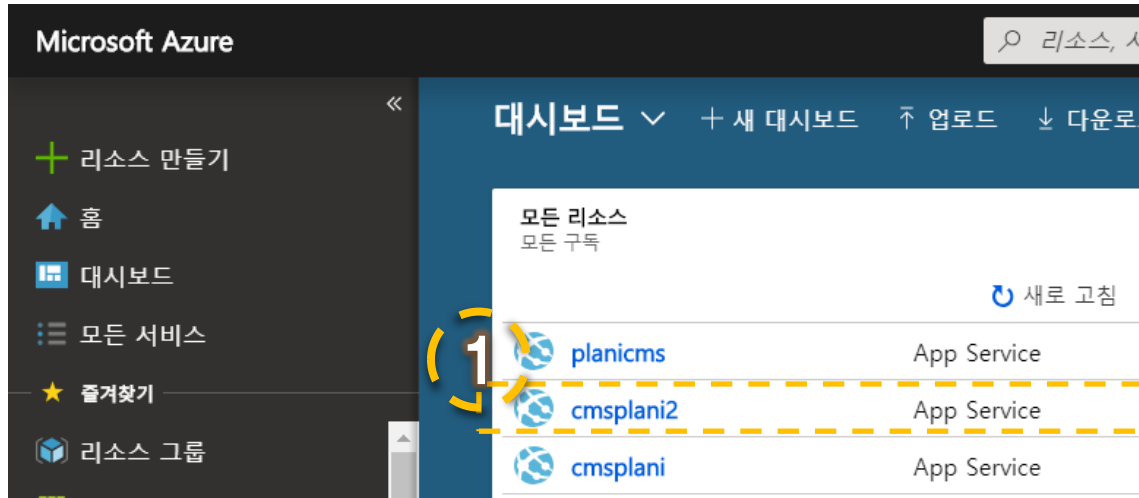


- ✓ PM 역할(고객 미팅 주도 및 의견 조율)
- ✓ Azure 클라우드 플랫폼 DB를 연동하여 개발
- ✓ 프로세스 및 데이터 모델링
- ✓ 법인차량, 정비소, 장소, 경로, 운행일지 화면 설계
- ✓ 법인차량, 정비소, 장소, 경로, 운행일지 CRUD 구현

01

담당 파트

Azure DB 연동



```
public static Connection getConnection() {  
    Connection conn = null;  
    try {  
        Class.forName("org.gjt.mm.mysql.Driver");  
        conn = DriverManager.getConnection("jdbc:mysql://127.0.0.1:56573/cms?useSSL=true&requireSSL=false&characterEncoding=UTF-8",  
            "azure", "6#vWHD_$");  
        System.out.println("DB 연결 성공 : " + conn);  
    } catch (SQLException ex) {  
        System.out.println("SQLException" + ex);  
    }  
    catch (Exception ex) {  
        System.out.println("Exception" + ex);  
    }  
    return conn;  
}
```

1

Eclipse 에서 웹 앱 deploy 한 웹 서비스가 azure 플랫폼에 정상 등록 확인

2

DB DriverManager 객체의 getConnection 매개변수에 정보를 입력하여 연동

Azure 클라우드 서비스의 한 기능인 **서버리스**를 도전해봤습니다. 인터넷 자료도 부족하여, 웹 앱을 생성하고, DB 연동까지 구현하는 데 시간이 걸리긴 했지만 그만큼 신기술을 한 번 **도전**해보고 **성공**했다는 **성취감**을 얻었습니다.

01

담당 파트

사용자 UI 고려

법인차 등록

차량 등록 번호

*법인차 기본 정보

법인차 구분	<input type="text" value="구입"/>	차종	<input type="text"/>
누적 거리	<input type="text"/>	km	

*보험 정보

보험명	<input type="text"/>	보험 구분	<input type="text" value="선택"/>
보험 나이 제한	<input type="text"/>		
보험 계약 날짜	<input type="text"/>	보험 만기 날짜	<input type="text"/>

법인차 등록

차량 등록 번호

*법인차 기본 정보

법인차 구분	<input type="text" value="렌트"/>	차종	<input type="text"/>
누적 거리	<input type="text"/>	km	

*보험 정보

보험명	<input type="text"/>	보험 구분	<input type="text" value="선택"/>
보험 나이 제한	<input type="text"/>		
보험 계약 날짜	<input type="text"/>	보험 만기 날짜	<input type="text"/>

*렌탈/리스 정보

회사명	<input type="text"/>		
회사 전화번호	<input type="text"/>	회사 FAX	<input type="text"/>
계약 날짜	<input type="text"/>	계약 만료 날짜	<input type="text"/>

법인차량 상태에 따라 요구하는 속성에 다르기 때문에
사용자 UI 를 고려하여 해당 속성을 사용하여 문제를 해결

01

담당 파트

Javascript onChange 속성 사용

```
<select name="car_divi" class="form_car_select"
  onChange="change(this.options[this.selectedIndex].value)">
  <option value="선택">선택</option>
  <option value="구입">구입</option>
  <option value="렌트">렌트</option>
  <option value="리스">리스</option>
</select></td>
```

```
<script language="javascript">
  function change(style) {
    if (style == "선택") {
      paycar.style.display = "none";
      rental_lease.style.display = "none";
    }
    else if (style == "구입") {
      paycar.style.display = "inline";
      rental_lease.style.display = "none";
    }
    else if (style == "렌트") {
      paycar.style.display = "inline";
      rental_lease.style.display = "inline";
    }
    else if (style == "리스") {
      paycar.style.display = "inline";
      rental_lease.style.display = "inline";
    }
  }
  //-->
</script>
```

WebContent

- car
 - car_expense_form.jsp
 - car_pay.jsp
 - car_rental_lease.jsp
 - car_search.jsp
 - car_select.jsp
 - car_write.jsp
 - sideMenu.jsp

SD교육그룹 포털 시스템 개발

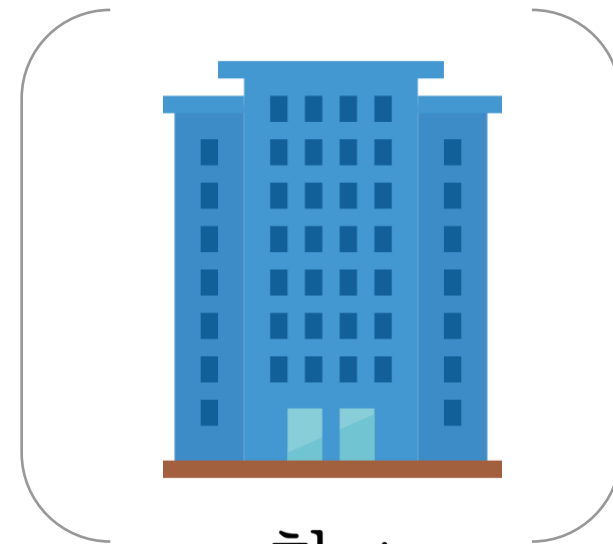
02

개발 개요

정보시스템 프로젝트



청년취업아카데미
INNOBIZ
기술혁신형중소기업




수학의
Z선
Mathematics
Academy

02


개발 개요




개발 도구
STS
sqldeveloper



WAS Server
tomcat 8.5



운영체제
Window 10



DBMS
Oracle 11g



개발 언어
Java8, JSP, SQL
ajax, Javascript
html5, css3



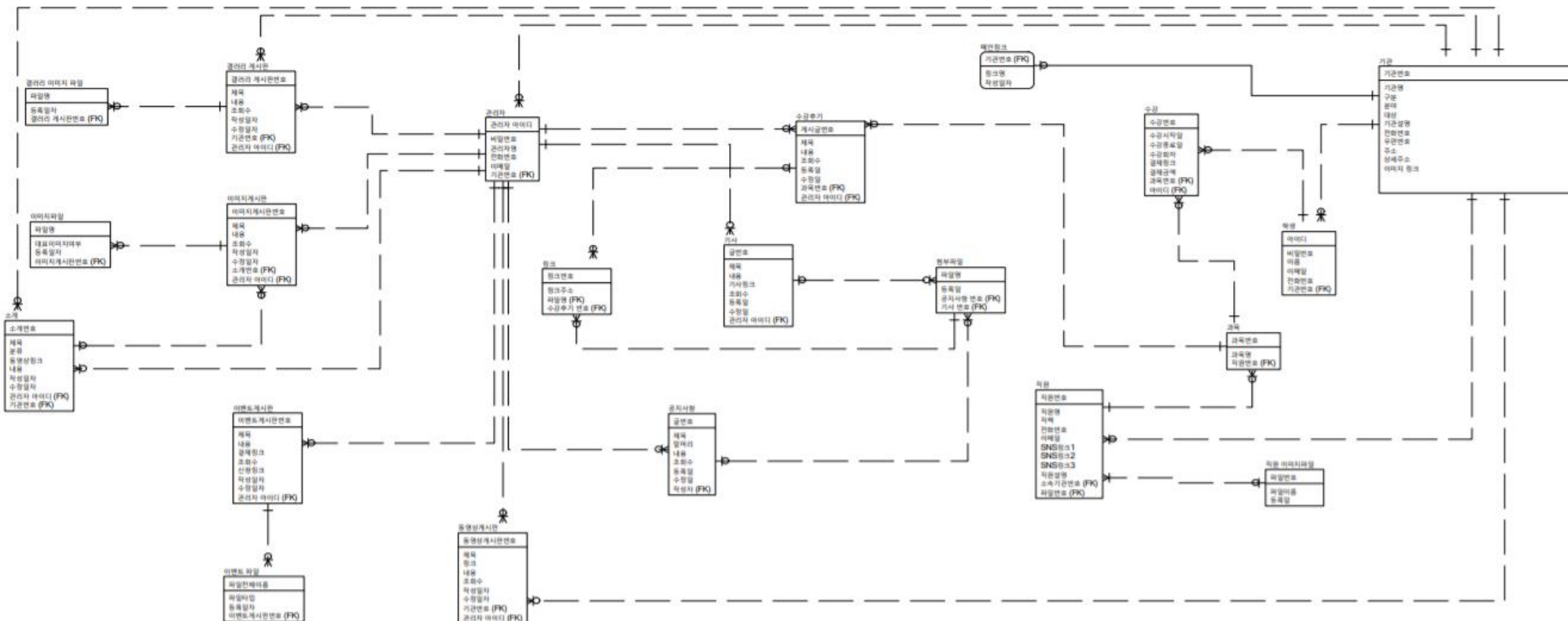
개발 패턴
Spring 4.0
ibatis
Bootstrap3.3

02

개발 개요

데이터 모델링

E-R Diagram



02

개발 개요

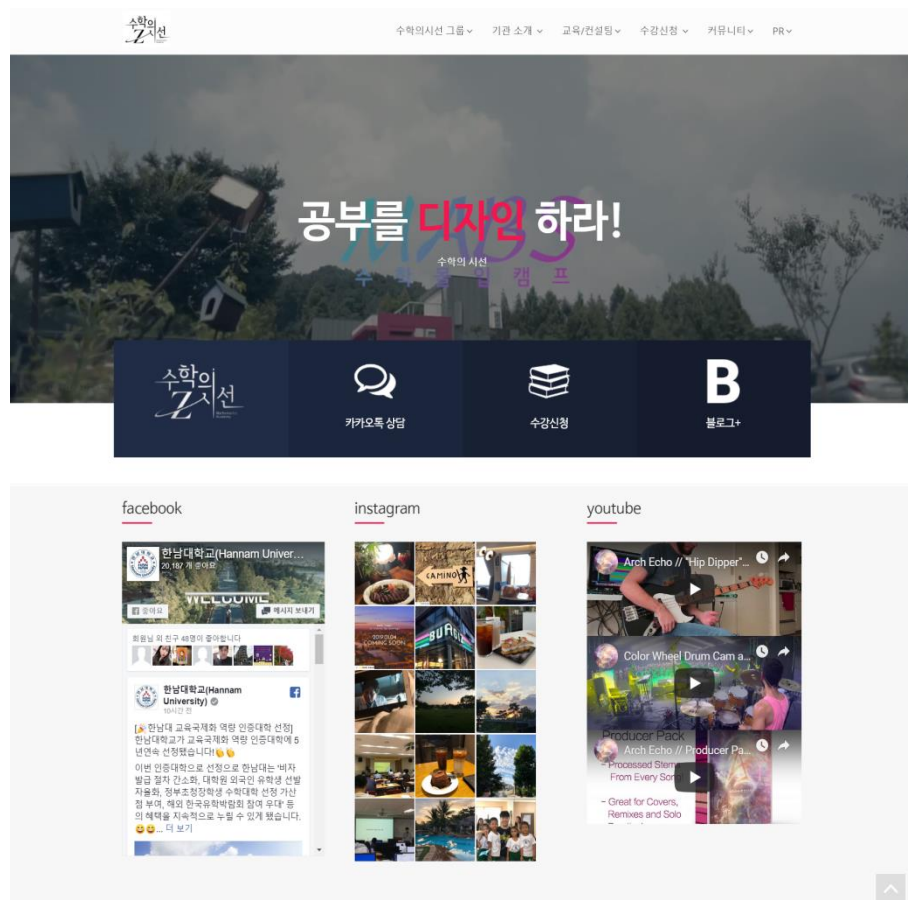
데이터 모델링

[CRUD Matrix]

Function \ Entity		기관	직원	직원 이미지파일	과목	수강	메인링크	수강후기	첨부파일	기사	공지사항	링크	관리자	프로그램 이미지	프로그램 이미지파일	소개(프로그램)	동영상게시판	이벤트게시판	이벤트파일	학생	사진게시판	사진파일
커뮤니티 관리	공지사항 등록								CRUD		CRUD	CRUD	R									
	이벤트게시판 등록												R					CRUD	CRUD			
	사진게시판 등록	R											R								CRUD	CRUD
	동영상게시판 등록	R											R				CRUD					
PR 관리	보도자료 등록								CRUD	CRUD			R									
	수강후기 등록	R			R			CRUD				CRUD	R									
그룹 관리	기관 등록	CRUD					CRUD						R									
수강 관리	과목 등록	R	R		CRUD								R									
	수강 등록	R	R		R	CRUD							R							R		
기초정보 관리	관리자 등록	R											CRUD									
	학생 등록	R											R							CRUD		
	직원 등록	R	CRUD	CRUD									R									
프로그램 관리	프로그램 등록	R											R			CRUD						
	프로그램 이미지 등록	R											R	CRUD	CRUD	R						

02

담당 파트

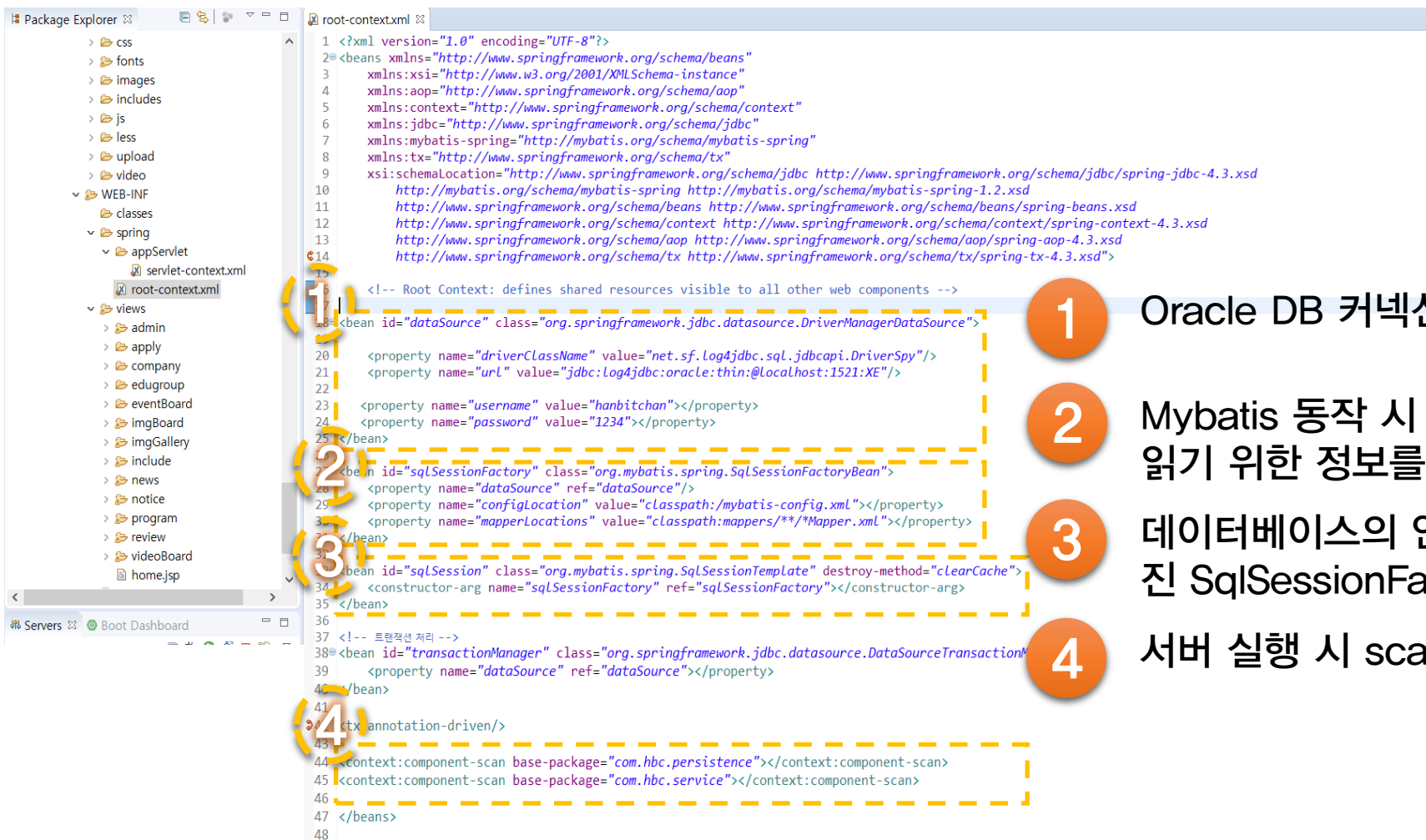


- ✓ PM 역할(고객 미팅 주도 및 의견 조율)
- ✓ 개발 환경 설정(DB 설계 종합, 템플릿 세팅, 스프링 환경 설정 등)
- ✓ 학생 수강신청 로그인 기능 구현
- ✓ 과목, 수강 관리자, 사용자 CRUD 구현
- ✓ ajax 를 이용한 비동기식 DB 처리
- ✓ SNS(페이스북, 인스타그램, 유튜브) 플러그인 연동 구현

02

담당 파트

root-context.xml 환경설정



The screenshot shows the Package Explorer on the left with the project structure. The main editor displays the root-context.xml file. Annotations 1 through 4 are placed over specific parts of the XML code:

- 1: Points to the `<bean id="dataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">` block, which defines the database connection properties.
- 2: Points to the `<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">` block, which defines the MyBatis configuration.
- 3: Points to the `<bean id="sqlSession" class="org.mybatis.spring.SqlSessionTemplate" destroy-method="clearCache">` block, which defines the MyBatis session.
- 4: Points to the `<context:component-scan base-package="com.hbc.persistence">` and `<context:component-scan base-package="com.hbc.service">` blocks, which define the packages to be scanned for components.

1

Oracle DB 커넥션 정보를 작성

2

Mybatis 동작 시 config 파일 적용과 *Mapper.xml를 읽기 위한 정보를 작성

3

데이터베이스의 연결과 sql 실행에 대한 모든 것을 가진 SqlSessionFactory 객체 등록

4

서버 실행 시 scan 할 패키지 설정(DAO와 Service)

02

담당 파트

로그인 기능

[수학의시선 그룹 ▾](#)[기관 소개 ▾](#)[교육/컨설팅 ▾](#)[수강신청 ▾](#)[커뮤니티 ▾](#)[PR ▾](#)

수강신청

[수강신청](#)[교육연구소](#)

학원생 로그인

아이디

비밀번호

학원생 계정으로 로그인

🏠 홈으로 돌아가기

1

수강신청 메뉴에서 학원생 로그인 요구

2

학원생 아이디, 비밀번호 입력

3

로그인 버튼을 이용해 계정 로그인

로그인 기능

02

담당 파트

```
34 @RequestMapping(value = {"login", "/"}, method = RequestMethod.GET)
35 public void login(Model model) throws Exception {
36
37     model.addAttribute("appCompList", service.appCompList());
38
39 }
40
41 @RequestMapping(value = "/loginPost", method = RequestMethod.POST)
42 public void loginPOST(StudentLoginDTO dto, HttpSession session, Model model) throws Exception {
43
44     StudentVO vo = stuService.login(dto);
45
46     if (vo == null) {
47         return;
48     }
49
50     model.addAttribute("studentVO", vo);
51
52 }
53
54 @RequestMapping(value = "/logout", method = RequestMethod.GET)
55 public String logout(HttpSession session) {
56     session.invalidate();
57
58     return "redirect:/apply/login";
59 }
```

```
9 </head>
10 <body>
11 <script type="text/javascript">
12     self.location = "/apply/list";
13 </script>
14 </body>
15 </html>
```

```
61 /**
62  * 전체 리스트 페이지에 보여줄 수강 정보를 조회하는 메소드
63  *
64  * @param cri
65  * @param model
66  * @throws Exception
67  */
68 @RequestMapping(value = "/list", method = RequestMethod.GET)
69 public void listApply(@ModelAttribute("cri") AppSearchCriteria cri, Model model, HttpServletRequest request) throws Exception {
70
71     Logger.info(cri.toString());
72
73     HttpSession session = request.getSession();
74     String stuid = ((StudentVO) session.getAttribute("stuLogin")).getStuid();
75     Logger.info("학생 아이디" + stuid);
76
77     cri.setStuid(stuid);
78
79
80     model.addAttribute("list", service.userListSearchCriteria(cri));
81     model.addAttribute("count", service.userListSearchCount(cri));
82     model.addAttribute("appCompList", service.appCompList());
83
84     PageMaker pageMaker = new PageMaker();
85     pageMaker.setCri(cri);
86
87     pageMaker.setTotalCount(service.listSearchCount(cri));
88
89     Logger.info(pageMaker.getTotalCount() + "페이지개수");
90
91     model.addAttribute("pageMaker", pageMaker);
92
93 }
```

특정 학생의 수강과목
리스트 출력

- 1 login 화면/사이드메뉴에 기관명 노출
- 2 login 성공 시 list 호출
- 3 logout 기능

[로그인 기능]

02

담당 파트

1

1 servlet-context.xml에 인터셉터 등록

1

```
29 <beans:bean id="StudentLoginInterceptor" class="com.hbc.interceptor.StudentLoginInterceptor"></beans:bean>
30 <beans:bean id="StudentAuthInterceptor" class="com.hbc.interceptor.StudentAuthInterceptor"></beans:bean>
```

2

```
public class StudentLoginInterceptor extends HandlerInterceptorAdapter{

    private static final String LOGIN = "stuLogin";
    private static final Logger LOGGER = LoggerFactory.getLogger(StudentLoginInterceptor.class);

    @Override
    public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler,
        ModelAndView modelAndView) throws Exception {

        HttpSession session = request.getSession();

        ModelMap modelMap = modelAndView.getModelMap();
        Object studentVO = modelMap.get("studentVO");

        if (studentVO != null) {
            LOGGER.info("new login success - " + studentVO);

            session.setAttribute(LOGIN, studentVO);

            response.sendRedirect("/apply/list");
        }
    }

    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler)
        throws Exception {

        HttpSession session = request.getSession();

        if (session.getAttribute(LOGIN) != null) {

            LOGGER.info("clear login data before");
            session.removeAttribute(LOGIN);
        }

        return true;
    }
}
```

3

```
public class StudentAuthInterceptor extends HandlerInterceptorAdapter {

    private static final Logger LOGGER = LoggerFactory.getLogger(StudentAuthInterceptor.class);

    private void saveDest(HttpServletRequest req) {
        String uri = req.getRequestURI();
        String query = req.getQueryString();

        if (query == null || query.equals("null")) {
            query = "";
        } else {
            query = "?" + query;
        }
    }

    if (req.getMethod().equals("GET")) {
        LOGGER.info("dest : " + (uri + query));
        req.getSession().setAttribute("dest", uri + query);
    }
}

@Override
public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler)
    throws Exception {
    HttpSession session = request.getSession();

    if (session.getAttribute("stuLogin") == null) {
        LOGGER.info("current user is not logged");

        saveDest(request);
        response.sendRedirect("/apply/login");

        return false;
    }

    return true;
}
}
```


2,3

LoginInterceptor와 AuthInterceptor로
분리하여 interceptor를 작성

02 담당 파트

[CRUD_등록 기능]

SD 포털 관리자모드



관리자

로그아웃

관리자 메뉴

- 프로그램 관리
- 커뮤니티 관리
- PR관리
- 수강 관리
- 기초정보관리

관리자 화면

수강 등록

학원생*

수강 과목*

교육연구소

수강 기간*

03/18/2019 12:00 AM

 ~

03/30/2019 12:01 AM

수강횟수*

매주 화, 목

수강료*

500,000

결제 링크*

결제 링크를 입력하세요.

수강 설명*

심화 수학반

취소

등록

학생 조회 - Chrome

localhost/admin/apply/search

학생 목록

이름

키워드를 입력하세요

Q 검색

소속 기관	학생 아이디	이름	전화번호
교육연구소	s001	홍길동	010-5848-4848
교육연구소	s002	허균	010-8848-4848

1

- 1 등록된 학원생 조회 후 선택
- 2 등록된 기관 선택
- 3 2번에서 선택된 기관의 수강 과목 조회 후 선택

02

담당 파트

CRUD 기능

```
11 public class ApplyVO extends CourseVO{
12
13     // 1. 변수 선언
14     private int appnum;
15     private Date sdate;
16     private Date edate;
17     private String dayinfo;
18     private String price;
19     private String link;
20     private String spec;
21     private int cournum;
22     private String stuid;
23     private String stuname;
24
25     // 2. setter, getter 선언
26     public int getAppnum() {
27         return appnum;
28     }
29
30     public void setAppnum(int appnum) {
31         this.appnum = appnum;
32     }
33
34     public Date getSdate() {
35         return sdate;
36     }
37
38     public void setSdate(Date sdate) {
39         this.sdate = sdate;
40     }
41
42     public Date getEdate() {
43         return edate;
44     }
45 }
```

```
1 public interface ApplyDAO {
2
3     /**
4      * 수강 테이블에 데이터를 등록하는 메소드
5      *
6      * @param vo : 등록 할 ApplyVO 객체
7      * @throws Exception
8      */
9     public void create(ApplyVO vo) throws Exception;
10
11     /**
12      * 수강 테이블의 전체 조회하는 메소드
13      *
14      * @return : ApplyVO 리스트
15      * @throws Exception
16      */
17     public List<ApplyVO> listAll() throws Exception;
18
19     /**
20      * 수강 테이블에 저장 된 데이터를 수정하는 메소드
21      *
22      * @param vo : 수정할 ApplyVO 객체
23      * @throws Exception
24      */
25     public void update(ApplyVO vo) throws Exception;
26
27     /**
28      * 수강 테이블의 저장 된 데이터를 삭제하는 메소드
29      *
30      * @param appnum : 삭제 할 수강번호
31      * @throws Exception
32      */
33 }
```

```
1 Repository
2 public class ApplyDAOImpl implements ApplyDAO {
3
4     @Inject
5     SqlSession sqlSession;
6
7     private static final String namespace = "com.hbc.mappers.ApplyMapper";
8
9     @Override
10    public void create(ApplyVO vo) throws Exception {
11        sqlSession.insert(namespace + ".create", vo);
12    }
13
14    @Override
15    public List<ApplyVO> listAll() throws Exception {
16        return sqlSession.selectList(namespace + ".listAll");
17    }
18
19    @Override
20    public void update(ApplyVO vo) throws Exception {
21        sqlSession.update(namespace + ".update", vo);
22    }
23
24    @Override
25    public void delete(int appnum) throws Exception {
26        sqlSession.delete(namespace + ".delete", appnum);
27    }
28 }
```

1

VO 클래스 설계

2,3

CRUD 인터페이스 DAO 설계 구현

02

담당 파트

CRUD 기능

```
12 public interface ApplyService {
13
14     public void register(ApplyVO vo) throws Exception;
15
16     public ApplyVO read(Integer appnum) throws Exception;
17
18     public void modify(ApplyVO vo) throws Exception;
19
20     public void remove(Integer cournum) throws Exception;
21
22     public List<ApplyVO> listAll() throws Exception;
23
24     public List<CompanyVO> listComp() throws Exception;
25
26
27 @Service
28 public class ApplyServiceImpl implements ApplyService {
29
30     @Inject
31     private ApplyDAO dao;
32
33     @Override
34     public void register(ApplyVO vo) throws Exception {
35         dao.create(vo);
36     }
37
38     @Override
39     public ApplyVO read(Integer appnum) throws Exception {
40         return dao.read(appnum);
41     }
42
43     @Override
44     public void modify(ApplyVO vo) throws Exception {
45         dao.update(vo);
46     }
47
48     @Override
49     public void remove(Integer appnum) throws Exception {
50         dao.delete(appnum);
51     }
52 }
```

```
90 <!-- 수강을 등록해 주는 쿼리 -->
91 <insert id="create">
92     INSERT INTO TBL_APPLY ( appnum
93     , sdate
94     , edate
95     , dayinfo
96     , spec
97     , price
98     , link
99     , cournum
100     , stuid )
101
102     VALUES ( seq_apply.nextval
103     , #{sdate}
104     , #{edate}
105     , #{dayinfo}
106     , #{spec}
107     , #{price}
108     , #{link}
109     , #{cournum}
110     , #{stuid} )
111
112 </insert>
```

```
100 <!-- 기관의 이름을 조회하는 쿼리 -->
101 <select id="compList" resultType="com.hbc.domain.CompanyVO">
102     SELECT c.compname, c.compnum
103     FROM TBL_COMPANY c
104 </select>
105
106 <!-- 특정 기관의 해당 과목을 조회하는 쿼리 -->
107 <select id="courList" resultType="com.hbc.domain.CourseVO">
108     SELECT c.COURNUM
109     , c.COURNAME
110     FROM TBL_COURSE c, TBL_COMPANY co, TBL_EMPLOYEE e
111     WHERE e.empnum = c.empnum
112     AND co.compnum = e.compnum
113     AND co.compname = #{compname}
114 </select>
```

1

Service 인터페이스 설계 및 구현

2

수강 등록 쿼리 작성

3

기관의 이름 조회하는 쿼리 작성

4

특정 기관의 해당 과목을 조회하는 쿼리 작성

02

담당 파트

〔 CRUD_등록 기능 〕

```
33 @Controller
34 @RequestMapping("/admin/apply/*")
35 public class AdminApplyController {
36
37     private static final Logger logger = LoggerFactory.getLogger(AdminCourseController.class);
38
39     @Inject
40     private ApplyService service;
41     @Inject
42     private StudentService stuService;
43
44     /**
45      * 수강을 등록하는 페이지로 이동하는 메소드
46      *
47      * @param apply
48      * @param model
49      * @throws Exception
50      */
51     @RequestMapping(value = "/register", method = RequestMethod.GET)
52     public void registerGET(@ModelAttribute("cri") AppSearchCriteria cri, ApplyVO apply, Model model) throws Exception {
53         model.addAttribute("list", service.listComp());
54         logger.info("regist get.....");
55     }
56
57     /**
58      * 수강 등록 페이지에서 수강을 등록해주는 페이지
59      *
60      * @param apply : 등록할 ApplyVO 객체
61      * @param rttr
62      * @return
63      * @throws Exception
64      */
65
66     @RequestMapping(value = "/register", method = RequestMethod.POST)
67     public String registerPOST(ApplyVO apply, RedirectAttributes rttr) throws Exception {
68
69         logger.info("regist post.....");
70         logger.info("등록 : " + apply.toString());
71
72         service.register(apply);
73         rttr.addFlashAttribute("msg", "SUCCESS");
74
75         return "redirect:/admin/apply/list";
76     }
77 }
```

1

등록 API 개발(GET, POST)

02

담당 파트

script로 ajax api 호출

```
61<script type="text/javascript">
62  //AJAX select box
63  function compSelect(compname) {
64    $.ajax({
65      type : "POST",
66      url : "selectAjax",
67      dataType : "json",
68      contentType: "application/x-www-form-urlencoded; charset=UTF-8",
69      data : {
70        param : compname
71      },
72      success : function(result) {
73
74        //SELECT BOX 초기화
75        $("#cournum").find("option").remove().end().append(
76          "<option value=''>선택</option>");
77
78        //배열 개수 만큼 option 추가
79        $.each(result, function(i) {
80          $("#cournum").append(
81            "<option value='"+result[i].cournum+"'>"+ result[i].courname
82            + "</option>");
83        });
84      },
85      error : function(jqXHR, textStatus, errorThrown) {
86        alert("오류가 발생하였습니다.");
87      }
88    });
89  }
90</script>
```

ajax 통신

(2)

```
87 public void selectAjax(HttpServletRequest req, HttpServletResponse res, String param) {
88   try {
89     // ajax에서 받은 기관의 이름을 compname 변수에 담음
90     String compname = param;
91
92     System.out.println(compname);
93
94     // 기관 번호와 명만 조회하여 리스트에 담은 객체 생성
95     List<CompanyVO> complist = service.listComp();
96
97     // 리스트 확인
98     for (CompanyVO vo : complist) {
99       System.out.println(vo);
100     }
101
102     // 특정 기관명을 조회하여 해당하는 과목번호와 이름을 조회하여 리스트 객체 생성
103     List<CourseVO> courlist = service.listCour(compname);
104
105     // 리스트 확인
106     for (CourseVO vo : courlist) {
107       System.out.println(vo);
108     }
109
110     List<CourseVO> courJsonList = new ArrayList<CourseVO>();
111
112     for (int idx = 0; idx < complist.size(); idx++) {
113       if (compname.equals(complist.get(idx).getCompname())) {
114         for (int i = 0; i < courlist.size(); i++) {
115           courJsonList.add(courlist.get(i));
116         }
117       }
118     }
119
120     // jsonArray에 추가
121     JSONArray jsonArray = new JSONArray();
122     for (int i = 0; i < courJsonList.size(); i++) {
123       jsonArray.add(courJsonList.get(i));
124     }
125
126     // jsonArray 넘김
127     PrintWriter pw = res.getWriter();
128     pw.print(jsonArray.toString());
129     pw.flush();
130     pw.close();
131
132   } catch (Exception e) {
133     System.out.println("Controller error");
134   }
135 }
136 }
```

List 컬렉션을 이용해 데이터를 불러오고, json으로 데이터 포맷 (JSONArray 객체)

처음 **ajax**를 이용해 데이터를 **비동기식**으로 처리했는데, **사용자 입장**에서 고려했을 때 반드시 필요한 기술이라고 생각

02

담당 파트

〔 CRUD_조회 기능 〕

1

SD 포털 관리자모드

관리자

관리자 메뉴

- 프로그램 관리
- 커뮤니티 관리
- PR관리
- 수강 관리
- 기초정보관리

관리자 화면

수강 목록

과목명

키워드를 입력하세요

Q 검색

수강 번호	수강 과목	학생명	수강 기관	수강 기간	수강 횟수	수강료
3	기초 수학반	허균	교육연구소	2019년 3월 1일 ~ 2019년 3월 31일	매주 월, 수	400,000원
2	심화 수학반	홍길동	교육연구소	2019년 3월 18일 ~ 2019년 4월 6일	매주 화, 목	500,000원

1

신규 등록

1 수강 전체 목록 조회

2

SD 포털 관리자모드

관리자

관리자 메뉴

- 프로그램 관리
- 커뮤니티 관리
- PR관리
- 수강 관리
- 기초정보관리

관리자 화면

수강 목록

과목명

기초

Q 검색

수강 번호	수강 과목	학생명	수강 기관	수강 기간	수강 횟수	수강료
3	기초 수학반	허균	교육연구소	2019년 3월 1일 ~ 2019년 3월 31일	매주 월, 수	400,000원

1

신규 등록

2 과목명으로 “기초” 를 검색하여 조회

02

담당 파트

CRUD_조회 기능

1

수강 목록 조회하는
쿼리(전체/검색)

```
<select id="listSearch" resultType="com.hbc.domain.ApplyVO">
<![CDATA[
SELECT *
FROM ( SELECT rownum rnum
      , a.appnum
      , a.link
      , a.spec
      , a.sdate
      , a.edate
      , a.dayinfo
      , a.price
      , co.compname
      , c.courname
      , s.stuid
      , s.stuname
FROM TBL_COURSE c, TBL_COMPANY co, TBL_EMPLOYEE e, TBL_APPLY a, TBL_STUDENT s
WHERE e.empnum = c.empnum
      AND co.compnum = e.compnum
      AND e.empnum = c.empnum
      AND a.cournum = c.cournum
      AND s.stuid = a.stuid
      AND a.appnum > 0
)]>
<include refid="search"></include>
<![CDATA[
ORDER BY a.appnum DESC
)
WHERE ( rnum >= #{pageStart} AND rnum <= #{pageEnd} )
ORDER BY appnum DESC
]]>
</select>
```

```
<sql id="search">
<if test="searchType != null">
<if test="searchType == 'c'.toString()">
AND c.courname LIKE '%' || #{keyword} || '%'
</if>
<if test="searchType == 't'.toString()">
AND co.compname LIKE '%' || #{keyword} || '%'
</if>
<if test="searchType == 'ct'.toString()">
AND ( c.courname LIKE '%' || #{keyword} || '%'
      OR co.compname LIKE '%' || #{keyword} || '%' )
</if>
</if>
</sql>
```

```
<select id="listSearchCount" resultType="int">
<![CDATA[
SELECT COUNT(a.appnum)
FROM TBL_COURSE c, TBL_COMPANY co, TBL_EMPLOYEE e, TBL_APPLY a, TBL_STUDENT s
WHERE e.empnum = c.empnum
      AND co.compnum = e.compnum
      AND e.empnum = c.empnum
      AND a.cournum = c.cournum
      AND s.stuid = a.stuid
      AND a.appnum > 0
]]>
<include refid="search"></include>
</select>
```

2

검색 결과 count 조회(페이징)

02

담당 파트

[CRUD_조회 기능]

1

```
160 @RequestMapping(value = "/list", method = RequestMethod.GET)
161 public void listPage(@ModelAttribute("cri") AppSearchCriteria cri, Model model) throws Exception {
162
163     logger.info(cri.toString());
164
165     model.addAttribute("list", service.listSearchCriteria(cri));
166
167     System.out.println("조회" + service.listSearchCriteria(cri));
168
169     PageMaker pageMaker = new PageMaker();
170     pageMaker.setCri(cri);
171
172     pageMaker.setTotalCount(service.listSearchCount(cri));
173
174     System.out.println(pageMaker.getTotalCount());
175     logger.info(pageMaker.getTotalCount() + "페이지개수");
176
177     model.addAttribute("pageMaker", pageMaker);
178
179 }
```

1

조회 API 개발(GET)

02 담당 파트

SNS 플러그인 오픈 소스



02

담당 파트

SNS 플러그인 오픈 소스



```
<script>
(function(d, s, id) {
  var js, fjs = d.getElementsByTagName(s)[0];
  if (d.getElementById(id))
    return;
  js = d.createElement(s);
  js.id = id;
  js.src = 'https://connect.facebook.net/ko_KR/sdk.js#xfbml=1&version=v3.1';
  fjs.parentNode.insertBefore(js, fjs);
})(document, 'script', 'facebook-jssdk');
</script>
<h3 class="mt-0 line-bottom line-height-1 theme-colored">facebook</h3>
<div class="fb-page"
  data-href="https://www.facebook.com/hannam.university"
  data-tabs="timeline" data-small-header="false"
  data-adapt-container-width="true" data-hide-cover="false"
  data-show-facepile="true">
  <blockquote cite="https://www.facebook.com/facebook"
    class="fb-xfbml-parse-ignore">
    <a href="https://www.facebook.com/facebook">Facebook</a>
  </blockquote>
</div>
```



```
<script>
$('.instagram-feed').each(function() {
  var current_div = $(this);
  var instagramFeed = new Instafeed({
    target : current_div.attr('id'),
    get : 'user',
    userId : current_div.data('userid'),
    accessToken : current_div.data('access_token'),
    resolution : 'low_resolution',
    limit : 9,
    template : '',
    after : function() {
    }
  });
  instagramFeed.run();
});
</script>
<h3 class="mt-0 line-bottom line-height-1 theme-colored">instagram</h3>
<div id="instafeed1" class="instagram-feed" data-userid="1212394882"
  data-access_token="1212394882.2aabac8.41ebc9f020b14e69aa38848469fef29e"
  data-limit="15" data-resolution="low_resolution"></div>
```



```
<script>
var url = 'https://www.googleapis.com/youtube/v3/search?key=AIzaSyB5mV2xVDs0ePgYBbYYe4QTbMYc9-kYmIA&channel'
+ 'Id=UCiXDVCqPmUHMcl0R1rFERpw&part=snippet,id&order=date&maxResults=3';
var videos = null;

$(document).ready(
  function() {
    $.ajax({ type : 'GET',
      url : url,
      data : 'JSON',
      success : function(data) {
        // alert(data.items[0].id.videoId);
        videos = data;
        for (var i = 0; i < videos.items.length; i++) {
          var ifrm = document.createElement("iframe");
          ifrm.setAttribute("src", ("https://www.youtube.com/embed/" + videos.items[i].id.videoId));
          ifrm.style.width = "100%";
          ifrm.style.height = "100%";
          document.getElementById("ytyt").appendChild(ifrm);
        }
      }
    });
  });
</script>
```

03

인턴 참여 프로젝트

03

참여 프로젝트



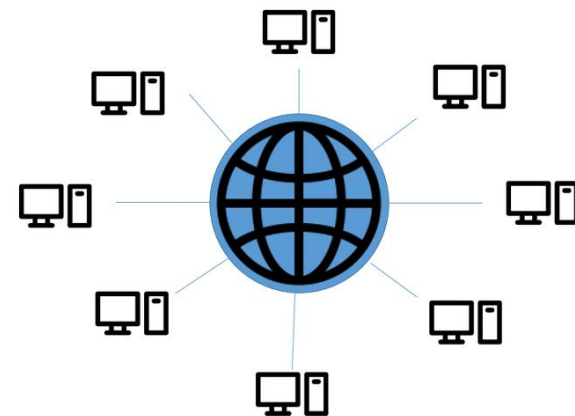
분산분석시스템 플랫폼 개발

- ∴ Angular JS, Java, Spring(boot), JPA 기반
- ∴ 기본 CRUD 개발
- ∴ 관리자 로그인 기능 구현 (Spring security)
- ∴ 프로세스 종료 및 시스템 종료 기능 구현



블록체인 기반 TrustAP 개발 프로젝트

- ∴ Ubuntu16.04 기반
- ∴ Atom보드에 리눅스 셋업, AP 구축 및 테스트
- ∴ Legacy AP 사이에 연결한 커넥터 AP 구축 및 테스트
- ∴ TrustAP 테스트 및 유지보수



분산처리데이터베이스 및 시스템 구축

- ∴ Centos7 기반
- ∴ hadoop 시스템 구축 및 테스트
- ∴ munge, slurm 구축 및 테스트
- ∴ Postgresql 구축 및 성능 테스트

감사합니다.