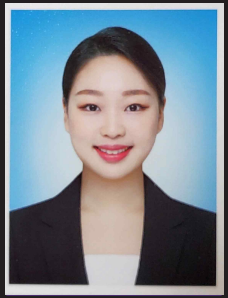
김나연

Java 개발 직무 지원자 김나연입니다



인적사항

[성명]	김나연
[생년월일]	1996년 12월 24일
[E-MAIL]	marlang0201@naver.com
[전화번호]	010-2144-6850
[Git-Hub]	https://github.com/kite1224/personal-project.git

경력사항

[기간]	2020.04 ~ 2022.04
[회사명]	한국계량측정협회
[담당업무]	kolas 시험인정위원회 지원 평가사 자격관리
[직책]	주임

학력

[기간]	2015.03 ~ 2019.08
[학교]	신한대학교
[학과(학점)]	글로벌통상경영(3.84/4.5)
[졸업구분]	졸업

교육사항

[교육기간]	2022.09~23.04
[기관명]	중앙정보처리학원
[교육시간]	09:00 ~ 18:00 (주5일)
[교육과정명]	Java 기반 풀스택개발자과정

portfolio-김나연

보유기술 소개

Java SE

-상속, 추상클래스, 인터페이스를 활용한 객체 설계
- Stream을 이용한 입출력 제어
- GUI 프로그래밍 (AWT/Swing/JavaFX,이벤트,그래픽처리)
- Collection Framework(List, Set, Map, Iterator등)를 이용한 집합 데이터 처리
- Socket, Stream, Thread를 적용한 네트워크 프로그래밍(멀티케스팅 기반 채팅 어플리케이션 구현)
- JDBC를 활용한 DBMS(Oracle, MySQL) 연동 프로그래밍
- XML/JSON파싱 및 공공데이터 포털의 Open API 연동하기
- Apache POI를 이용한 엑셀제어

Java EE (jsp/서블릿)

-서블릿 프로그래밍(생명주기의 메서드 및 XML매핑)
- Model1 패턴을 이용한 로직 객체 분리 및 재사용 (DAO, DTO 패턴 적용)
- JSP를 이용한 화면 구현(내장객체, include, beans 태그)
- RequestDispatcher를 이용한 요청 유지 및 포워딩 처리
- Tomcat 웹컨테이너 서버 환경 설정(Deployment, JNDI 설정)
- Apache Commons FileUpload , Cos.jar 를 활용한 파일업로드 처리
- Apache Commons POI를 활용한 엑셀 제어
- 공공데이터포털 Open API연동(Ajax 활용 비동기 요청 및 SAX파싱 및 데이터 처리)
- MVC프레임워 직접 제작해 보기(Command패턴, Factory 패턴 등)

SPRING framework

- web.xml 설정 및 Multiple DispatcherServlet 등록 및 매핑
- 인터페이스 및 DI를 이용한 DAO, Service 정의
- RuntimeException 재정의 및 Controller에서의 예외처리 (@ExceptionHandler)
- Mybatis Spring을 활용한 Oracle, MySQL 연동(association, collection 이용한조인)
- MultipartFile 을 이용한 파일업로드
- AOP를 이용한 공통로직의 적용(로그인, 트랜잭션, 쇼핑 카테고리 등)
- ControllerAdvice를 이용한 전역적 예외처리
- RestController를 이용한 RESTful 요청처리(ResponseEntity 기반 응답 처리)
- Java Mail API + 구글SMTP서버 활용한 메일 발송
- OAuth기반의 SNS 로그인 API (카카오, Naver, Google) 연동
- PG사 결제모듈 연동 (Toss 페이 모듈)
- BootStrap기반 쇼핑몰 프로젝트(AdminLTE 적용 관리자모드)

portfolio-김나연

보유기술 소개

ORACLE / MYSQL

- SQL(DML, DDL, DCL), Join, 서브쿼리, 집계함수
- 정규화와 Join (Inner Join, Outer Join)
- 제약조건 (primary key, not null, foreign key, default, unique) 설정
- 데이터베이스 설계 및 ERD

Mybatis / Hibernate

- Mybatis 설정 파일의 작성 – 도메인 객체 Type Alias 지정, JNDI 설정
- resultMap 이용한 조인 구현 - association(1:1), collection(1: 多)
- selectKey 를 이용한 primary key 구하기
- hashMap을 이용한 파라미터 처리
- Hibernate (ORM 프레임웍)을 이용한 게시판 CRUD 구현

Front-End

- HTML5/CSS3 기반 웹표준 퍼블리싱
- BootStrap를 이용한 반응형 웹페이지 구현
- Javascript (내장객체, ES 2015 class , DOM Api활용, Ajax비동기통신의 활용)
- Jquery를 이용한 DOM 제어
- SPA 게시판 구축(Jquery-Ajax 비동기 요청 + 스프링 REST 연동)
- Vue를 이용한 UI컴포넌트를 이용한 SPA 게시판 구현
- JS기반의 페이징 처리

리눅스 / AWS

- 우분투/Centos 설치 및 환경설정
- JDK 설치 및 PATH, CLASSPATH설정
- Tomcat 설치 및 환경설정
- 방화벽설정 (인바운드 포트 설정)
- SSH 서버 설치 및 권한
- FTP서버 설치 및 설정
- IAM 계정을 이용한 서버 인스턴스 관리

JavaSpring 프로젝트

EduZino

프로젝트 소개

프로젝트명

EduZino

주제

온라인 강의 사이트

선정 동기

학생의 입장에서 주제를 고민하다보니,
질 좋은 강의를 시공간의 제약 없이 접근할 수 있는 온라인 강의에 대한 흥미가 생겨 선정하게 되었다

기존 사이트와의 차별점

강의 내용에 대한 문의를 게시판 형식으로만 제공하는 사이트가 많았는데, 실시간 답변을 기대하기 어렵고 답변을 놓치는 경우가 있었다
이를 보완하기 위해
강사-수강생간의 문의를 1:1 채팅으로 제공한다

개발환경

구성요소

[Data Base]
SQLPlus / Oracle Developer(Oracle XE)

[Persistence Framework] Mybatis(3.5.13)

[개발도구] Eclipse (전자정부 프레임워크 3.1)

[웹 컨테이너] Tomcat 9.0

[버전 관리] Git / GitHub

사용언어 및 스타일 적용

[사용언어] JAVA (jdk 1.8)

[스타일] HTML / CSS5 / BootStrap 4

[스크립트 언어] JavaScript

eduZino

구현 기능 소개

- SNS 로그인 기능 구현 및 aop 처리
- [관리자] 일반회원을 강사로 전환하기/ 회원 정지하기
- [관리자->유저] 공지사항 게시판
- [유저 -> 관리자] QnA 게시판
- [유저] 찜하기 기능
- [유저] 장바구니 기능
- [유저] 결제 및 결제 내역
- [강사] 강의 관리
- [관리자, 수강생(유저)] 강의시청
- [강사] 수강생 목록조회 및 검색
- [강사-수강생(유저)] Websocket을 이용한 1:1 채팅 구현

eduZino - 김나연

찜하기 기능 구현

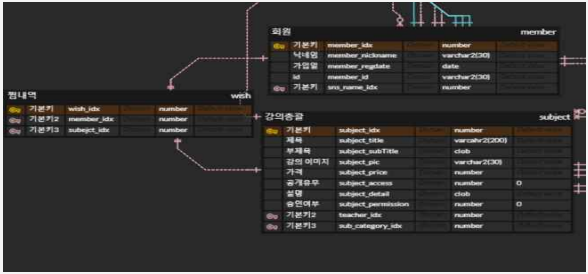
[ERD Table]

[wish]

찜한내역 내용 저장 테이블

[member]

회원 정보를 담는 테이블



[subject]

강의 전체 내용을 담은 테이블

eduZino - 김나연

찜하기 기능 구현

[WishList.jsp]

Model
데이터를 처리해 넣어주기

01 Controller 에서 전달한 데이터를 이용하여 회원이 찜한 강의를 리스트로 반환

02 찜하기 추가, 삭제 등을 수행하는 쿼리문 실행

03 DAO, MybatisDAO, Service, ServiceImpl 클래스 사용

04 Mapper에 입력된 쿼리문을 조합하여 중복체크

View
.jsp에서 구현하기

01 Vue를 사용하여 회원별 찜 목록을 비동기로 리스트 형식 반환

02 checkBox를 이용하여 원하는 항목 삭제 가능

03 장바구니로 이동 시 찜목록 비우기 기능 실행

Controller
데이터 처리하기

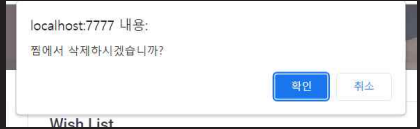
01 비동기 방식으로 DB에 저장된 목록 데이터를 추출하여 jsp로 전달

02 등록, 삭제 처리를 Service로 진행 후 ResponseEntity로 message를 반환

03 선택한 목록을 json 배열로 받아 장바구니 형식으로 변환 후 장바구니의 DB에 추가

찜하기 기능 구현

[WishList.jsp]



찜하기 기능 구현

[WishList.jsp]

```
function insertCart(){
    //체크된 개수
    let checkLng2=$("input[name='wish_idx']:checked").length;
    let arr2=[];
    let checkval; //여기에 wish_idx를 담고
    let checkval2; //여기에 subject_idx를 담을 것

    for(let i=0;i<checkLng2;i++){
        checkval = $($("input[name='hidden_wish_idx']")[i]).val(); //여는 wish_idx임
        checkval2=$($("input[name='wish_idx']")[i]).val(); //여는 subject_idx임

        let json={};
        let subject ={};

        subject["subject_idx"]=checkval2;
        json["wish_idx"]=checkval;
        json["subject"]=subject;
        arr2.push(json);
    }
    console.log("arr2 is ", JSON.stringify(arr2));
    if($("input[name=wish_idx]").is(':checked')){
        if(!confirm("장바구니에 추가하시겠습니까?")){
            return;
        }else{
            $.ajax({
                url:"/rest/cart/wishTocart",
                type:"POST",
                contentType:"application/json",
                data: JSON.stringify(arr2),
                success:function(result, status, xhr){
                    if(confirm("장바구니로 이동하시겠습니까?")){
                        location.href="/cart/list";
                    }
                    console.log(result);
                }
            });
        }
```

찜목록에서 장바구니로 이동(여러 건이라 json 배열)로 전달

```
//찜목록 장바구니에 등록 > 여러 건 등록이 가능하도록 배열로 받는다
@PostMapping("/cart/wishTocart")
@ResponseBody
public ResponseEntity<String> toCart(HttpServletRequest request, @RequestBody Wish[] wishList){
    Cart[] cartList=new Cart[wishList.length];

    //세션에서 해당 멤버의 정보를 가져온다
    HttpSession session = request.getSession();
    Member member = (Member)session.getAttribute("member");

    //여러 건이라 반복문을 돌린다
    for(int i=0; i<wishList.length; i++) {
        Wish wish = new Wish();
        wish= wishList[i];

        Subject subject = new Subject();
        subject.setSubject_idx(wish.getSubject().getSubject_idx());

        Cart cart = new Cart();
        cart.setSubject(subject);
        cart.setMember(member);
        cartList[i]=cart;
    }

    //3단계 > service에서 insert 작업 진행한다
    cartService.regist(cartList);
    //logger.info("cartList는 "+cartList);

    //4단계 > 정하는 메시지를 반환한다
    MessageUtil msg = new MessageUtil();
    msg.setMsg("장바구니 등록 성공");

    ResponseEntity entity = new ResponseEntity<MessageUtil>(msg,HttpStatus.OK);
    return entity;
}
```

전달받은 wish 형을 cart형으로 변경하여 insert 서비스 호출

찜하기 기능 구현

[WishList.jsp]

```
//선택 찜 비동기로 삭제
function delAsyncWish(){
let checkLng=$("input[name='wish_idx']:checked").length;
let arr=[];
let checkval;

for(let i=0;i<checkLng;i++){
    if($("input[name=wish_idx]").is(':checked')){
        checkval=$($("input[name='hidden_wish_idx']")[i]).val();
        console.log("checkval",checkval);
    }
    let json={};
    console.log("checkval",checkval);
    json["wish_idx"]=checkval;
    arr.push(json);
}

console.log("arr is ", JSON.stringify(arr));
if($("input[name=wish_idx]").is(':checked')){
    if(!confirm("찜에서 삭제하시겠습니까?")){
        return;
    }else{
        $.ajax({
            url:"/rest/cart/wish_list",
            type:"DELETE",
            contentType:"application/json",
            data: JSON.stringify(arr),
            success:function(result, status, xhr){
                console.log("삭제 완료");
                getWishList();
            }
        });
    }
}
```

찜목록 여러건 삭제(체크된 항목 여러 건 json배열로 전달)

```
@Override
@Transactional(propagation = Propagation.REQUIRED)
public void delWish(Wish[] wish)throws WishException{
    for(int i=0; i<wish.length; i++) {
        wishDAO.delWish(wish[i]);
    }
}
```

삭제 메서드들 체크된 개수만큼 반복하여 서비스에서 삭제 반복

```
//찜목록 삭제하기 > 여러건 삭제 가능하도록
@DeleteMapping("/cart/wish_list")
@ResponseBody
public ResponseEntity<MessageUtil> delWish(HttpServletRequest request, @RequestBody Wish[] wishList){

    //3단계
    wishService.delWish(wishList);

    //4단계
    MessageUtil msg = new MessageUtil();
    msg.setMsg("찜 삭제 성공");

    ResponseEntity entity = new ResponseEntity<MessageUtil>(msg,HttpStatus.OK);
    return entity;
}
```

컨트롤러에서 배열로 전달받아 서비스 호출

장바구니 기능 구현

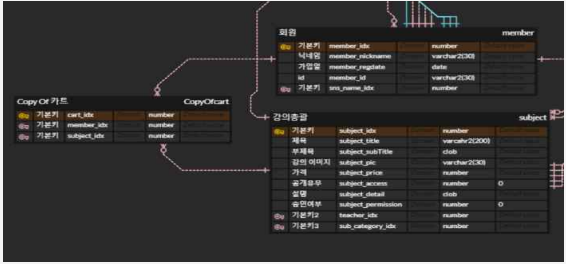
[ERD Table]

[cart]

장바구니 내용 저장 테이블

[member]

회원 정보를 담는 테이블



[subject]

강의 전체 내용을 담은 테이블

eduZino - 김나연

장바구니 기능 구현

[cartList.jsp]

Model
데이터를 처리해 넣어주기

01 Controller 에서 전달한 데이터를 이용하여 회원이 장바구니 추가한 강의를 리스트로 반환

02 장바구니 추가, 삭제 등을 수행하는 쿼리문 실행

03 DAO, MybatisDAO, Service, ServiceImpl 클래스 사용

04 Mapper에 입력된 쿼리문을 조합하여 중복체크

View
.jsp에서 구현하기

01 Vue를 사용하여 추가된 장바구니 목록을 비동기로 리스트 형식 반환

02 선택한 강의 수강료를 vue를 이용하여 비동기로 계산

03 결제 버튼 클릭 시 PG사의 결제 모듈 호출

04 결제완료 시 장바구니 비우기 기능 실행

Controller
데이터 처리하기

01 비동기 방식으로 DB에 저장된 목록 데이터를 추출하여 jsp로 전달

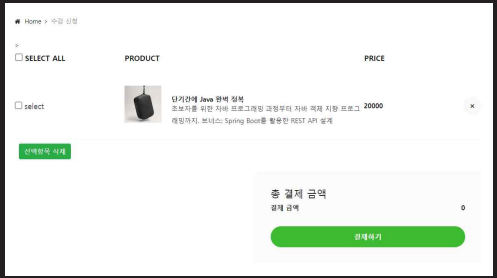
02 등록, 삭제 처리를 Service로 진행 후 ResponseEntity로 message를 반환

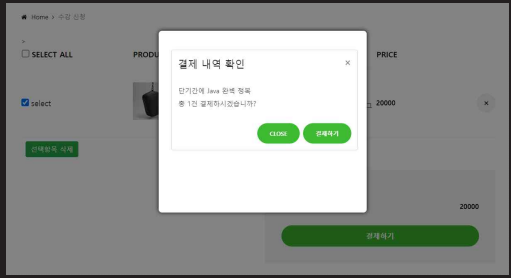
03 찜목록에서 등록 / 기본 강의 목록에서 등록을 별도로 처리

eduZino - 김나연

장바구니 기능 구현

[cartList.jsp]





장바구니 기능 구현

[cartList.jsp]

```
//여러건 비동기 삭제 (결제 후 장바구니 비우기)
function delAsyncCart(){
   let checkedBox=$("input[name='cart_idx']:checked");

   for(let i=0; i<checkedBox.length; i++){
      let idx = getCheckIndex($(checkedBox[i]).val());
      console.log("idx : ",idx);
      checkedList.push(cartApp.cartList[idx]);

      console.log("삭제할 때 쓸 것",checkedList);
      console.log("삭제할 때 쓸 checkedBox",checkedBox);
      }

   if(!confirm("선택하신 항목을 삭제하시겠습니까?")){
      return;
   }else{
      $.ajax({
         url:"/rest/cart/cart_list",
         type:"DELETE",
         contentType:"application/json",
         data: JSON.stringify(checkedList),
         success:function(result,status,xhr){
            console.log("장바구니 삭제 성공");
            getCartList();
         }
      });
   }
}
```

결제 완료 후 장바구니 비우기 작업

```
//카트에서 삭제하기
@Transactional(propagation = Propagation.REQUIRED)
public void delCart(Cart[] cart) throws CartException{
      for(int i=0; i<cart.length; i++) {
         cartDAO.delCart(cart[i]);
      }
   }
```

삭제 메서드들 체크된 개수만큼 반복하여 서비스에서 삭제 반복

```
//장바구니 삭제
@DeleteMapping("/cart/cart_list")
@ResponseBody
public ResponseEntity<MessageUtil> delCart(HttpServletRequest request, @RequestBody Cart[] cartList){

   //3단계
   cartService.delCart(cartList);
   //4단계
   MessageUtil msg = new MessageUtil();
   msg.setMsg("장바구니 삭제 성공");

   ResponseEntity entity = new ResponseEntity<MessageUtil>(msg,HttpStatus.OK);
   return entity;
}
```

컨트롤러에서 배열로 전달받아 서비스 호출

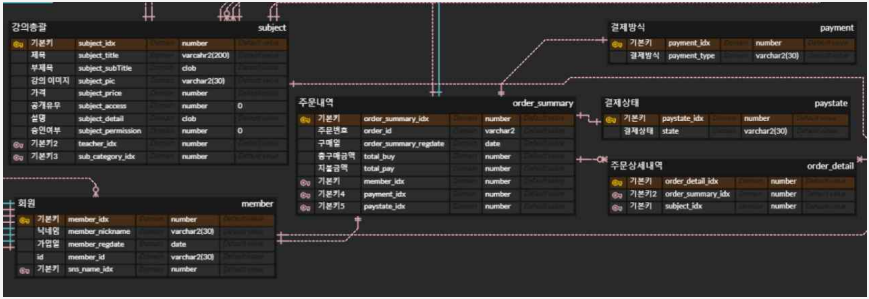
결제 및 결제내역 기능 구현

[ERD Table]

[member]
회원 정보를 담는 테이블

[order_summary]
회원별 구매한 강의 내역을 담은 테이블

[payment]
결제 상태를 담은 테이블



[subject]
강의 전체 내용을 담은 테이블

[paystate]
결제 방식을 담은 테이블

[order_detail]
결제 내용 상세를 담은 테이블

eduZino - 김나연

결제 및 결제내역 기능 구현

[cartList.jsp, paycomplete.jsp,payfail.jsp,orderList.jsp]

Model
데이터를 처리해 넣어주기

01 Controller 에서 전달한 데이터를 이용하여 결제 모듈 호출 후 결제 승인 파라미터 반환

02 결제 승인 후 DB에 결제내역, 결제 상세내역 추가

03 결제 완료 후 장바구니에서 삭제처리

04 DAO, MybatisDAO, Service, ServiceImpl 클래스 사용

View
.jsp에서 구현하기

01 회원 결제 완료 후 결제완료 창 제공 >메인 혹은 결제내역으로 이동

02 결제 실패 시 해당 에러메시지 반환

03 결제완료 시 장바구니 비움 기능 실행

04 회원별 결제내역 반환 > 상세보기 클릭 시 구매한 항목 확인 가능

Controller
데이터 처리하기

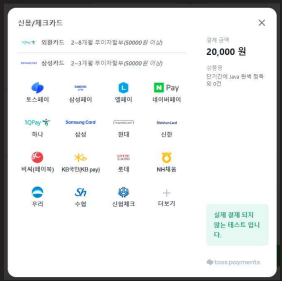
01 결제 완료 시 PG에서 제공하는 결제 파라미터를 전달받아 DB에 추가할 내용 추출

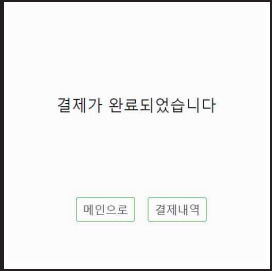
02 결제 성공, 실패 처리를 Service로 진행 후 ModelAndView로 알맞은 화면 반환

03 DB에 저장된 회원별 결제 정보를 추출하여 orderList.jsp에 반환

결제 및 결제내역 기능 구현

[cartList.jsp, paycomplete.jsp,payfail.jsp,orderList.jsp]





결제 및 결제내역 기능 구현

[cartList.jsp, paycomplete.jsp,payfail.jsp,orderList.jsp]

```
//order_id 만들어오기
function getOrderId(){
    $.ajax({
        url:"/rest/cart/orderid",
        type:"get",
        success:function(result,status,xhr){
            console.log(result);
            orderId=result;
            showPayModal(orderId);
        }

    });
}
```

PG사(tosspayment)에서 요구하는 id 생성 메서드 호출

```
//order_id 만들기
@GetMapping("/cart/orderid")
public ResponseEntity<String> getOrderId(HttpServletRequest request){
    String orderId=fileManager.getRealTime();
    ResponseEntity entity = new ResponseEntity<String>(orderId,HttpStatus.OK);
    return entity;
}
```

orderID 생성 메서드

```
/*---------toss api 결제----------*/
function showPayModal(orderId){

var clientKey = 'test_ck_D5GePWvyJnrK0W0k6q8gLzN97Eoq'
var tossPayments = TossPayments(clientKey) // 클라이언트 키로 초기화하기

tossPayments.requestPayment('카드', { // 결제 수단 파라미터
    // 결제 정보 파라미터
    amount: cartApp.totalPay, //order_summary.totalbuy
    orderId: ${cartList[0].member.member_idx}+orderId,  //이건 멤버idx+시간 조합으로 만들까봐
    orderName: checkedList[0].subject.subject_title+"외 "+(checkedList.length-1)+"건",  //여러 건 결제시
    //subject_title  >
    customerName: {member.member_nickname}, //member.member_nickname
    successUrl: 'http://localhost:7777/pay/payment', //결제가 완료되었습니다.> 완료 메서드 호출
    failUrl: 'http://localhost:7777/pay/payfail', //결제가 실패했습니다.> 실패 메서드 호출
  })
  .catch(function (error) {
    if (error.code === 'USER_CANCEL') {
      // 결제 고객이 결제창을 닫았을 때 에러 처리
      alert("결제창을 닫았습니다.");
      location.href='http://localhost:7777/cart/list';
    } else if (error.code === 'INVALID_CARD_COMPANY') {
      // 유효하지 않은 카드 코드에 대한 에러 처리
      alert("유효하지 않은 카드입니다.");
      location.href='http://localhost:7777/cart/list';
    }
  })

}
```

jsp에서 PG사가 요구하는 양식에 해당하는 파라미터값 입력

결제 및 결제내역 기능 구현

[cartList.jsp, paycomplete.jsp,payfail.jsp,orderList.jsp]

```
//결제 완료 (콜백주소)
@GetMapping("/pay/payment")
public ModelAndView cartTopay(HttpServletRequest request, OrderSummary order){

   String orderId=request.getParameter("orderId");
   int amount=Integer.parseInt(request.getParameter("amount"));
   String paymentKey=request.getParameter("paymentKey");

   logger.info("orderId는 "+orderId);
   logger.info("amount는 "+amount);
   logger.info("paymentKey는 "+paymentKey);

   //결제 완료 시 넘어오는 파라미터(json)를 String 으로 받음
   String payResult=orderService.getPay(orderId, amount, paymentKey);
   logger.info("payresult는 "+payResult);

   JSONObject obj = new JSONObject(payResult);
   String method = obj.getString("method"); //결제방법
   String state = obj.getString("status"); //결제상태
   int balanceAmount =  obj.getInt("balanceAmount");

   logger.info("method "+method);
   logger.info("status "+state);

   //order_summary에 등록

   HttpSession session = request.getSession();
   Member member = (Member)session.getAttribute("member");
```

```
   Payment payment = new Payment();
   if(method.equals("간편결제")) {
      payment.setPayment_idx(2); //간편결제는 2
   }else {
      payment.setPayment_idx(1);
   } //이전 payResult의 method 키값으로 대체
   payment.setPayment_type(method);

   Paystate paystate = new Paystate(); //payResult의 status 키값으로 대체
   if(state.equals("DONE")) {
      paystate.setPaystate_idx(2); //결제완료
      paystate.setState("결제완료");
   }else {
      paystate.setPaystate_idx(1); //결제실패
      paystate.setState("결제실패");
   }

   //orderSummary에 들어갈 내용들을 채워준다
   order.setMember(member); //회원번호
   order.setOrder_id(orderId); //주문번호
   order.setPayment(payment);//결제방법
   order.setPaystate(paystate);//결제상태
   order.setTotal_buy(amount); //구매금액
   order.setTotal_pay(balanceAmount); //실제 결제금액(포인트사용 후)

   //3단계> 여기서 orderSummary, orderDetail -insert / cart-del 완료
   orderService.regist(order);

   //4단계
   ModelAndView mav = new ModelAndView("/user/order/paycomplete");
   return mav;
}
```

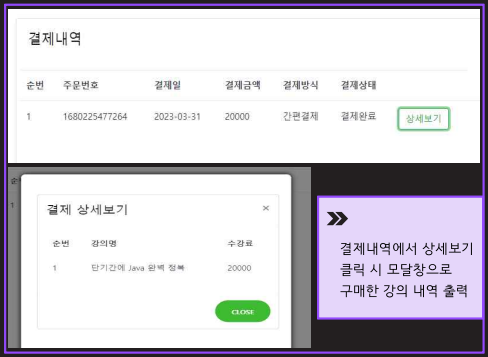
» controller에서 결제 파라미터를 전달받아 추가할 DB값을 추출 (주문내역, 주문 상세내역 등) 후 값을 set하고 service에서 메서드를 호출한다

결제 및 결제내역 기능 구현

[cartList.jsp, paycomplete.jsp,payfail.jsp,orderList.jsp]

```
//주문메서드
@Transactional(propagation = Propagation.REQUIRED)
public void regist(OrderSummary orderSummary) {


   //1) 주문요약 넣기> pk 반환
   orderSummaryDAO.insert(orderSummary);

   //2) 디테일 넣기
   List<Cart> cartList=cartDAO.selectAll(orderSummary.getMember());
   for(int i=0; i<cartList.size(); i++) {
      OrderDetail orderDetail = new OrderDetail();
      orderDetail.setOrderSummary(orderSummary);
      orderDetail.setSubject(cartList.get(i).getSubject());

      logger.info("getSubject는 "+orderDetail.getSubject());
      orderDetailDAO.insert(orderDetail);

      //카트에서 삭제
      cartDAO.delCart(cartList.get(i));
   }
}
```

» 주문내역, 상세내역 입력 후 장바구니 비우기 실행



» 결제내역에서 상세보기 클릭 시 모달창으로 구매한 강의 내역 출력

JavaSwing 프로젝트

영화 예매 프로그램

영화 예매 프로그램 프로젝트 소개

프로젝트명

영화 예매 프로그램

주제

시간, 좌석 선택까지 가능한 영화 예매 프로그램

선정 동기

회원가입, 로그인, 예매 기능등을 종합하여 사용할 수 있는 주제를 생각하다, 좋아하는 영화를 예매하는 프로그램을 선정하게 됨

JavaSwing 프로젝트 개발환경

구성요소

[Data Base]
SQLPlus / Oracle Developer(Oracle XE)

[개발도구] Eclipse

[버전 관리] Git / GitHub

사용언어 및 스타일 적용

[사용언어] JAVA (jdk 1.8)

[스타일] HTML5 / CSS3

[스크립트 언어] JavaScript

메인 기능

로그인/회원가입 시 DB에 데이터 저장

현재 상영작 클릭 시 상세보기

영화 예매

회원별 예매내역 확인

Movie Reservation - 김나연

주요 기능 구현

[ERD Table]

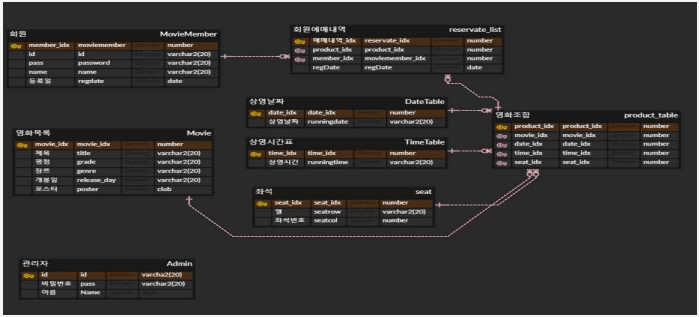
[Movie Member]
회원 정보를 담는 테이블

[Movie]
영화의 정보를 담는 테이블

[Admin]
관리자 정보를 담는 테이블

[Reservate_List]
회원별 영화 예매 내역 정보를 담는 테이블

[Product_Table]
영화별 날짜/시간/좌석의 정보를 담는 테이블



[DateTable]
영화의 상영날짜 정보를 담는 테이블

[TimeTable]
영화의 상영 시간 정보를 담는 테이블

[Seat]
영화의 날짜, 시간별 좌석 정보를 담는 테이블

로그인,회원가입 기능 구현

```
//회원가입 메서드
public void regist() {
    MovieMember movieMember = new MovieMember(); // 새 dto
    String str1 = t_id.getText();
    String str2 = new String(t_pass.getPassword());
    String str3 = t_name.getText();

    if (str1.isEmpty() != true && str2.isEmpty() != true && str3.isEmpty() != true) {
        JOptionPane.showMessageDialog(this, "사용가능한 아이디입니다.");
        movieMember.setId(str1);
        movieMember.setPass(str2);
        movieMember.setName(str3);

        int result = movieMemberDAO.insert(movieMember);
        if (result > 0) {
            JOptionPane.showMessageDialog(this, "Join Complete");
            t_id.setText("");
            t_pass.setText("");
            t_name.setText("");
            reservationMain.showHide(reservationMain.LOGINPAGE);
        }
    } else {
        JOptionPane.showMessageDialog(this, "모든 사항을 입력해주세요");
```

» 아이디, 이름, 비밀번호가 비어있지 않은 경우 DB에 추가

```
//아이디 중복검사
public void idTest() {
    String id = t_id.getText();

    boolean result = movieMemberDAO.idCheck(id);
    if (result) {
        JOptionPane.showMessageDialog(this, "중복된 아이디입니다.");
    } else {
        regist();
        idFlag = true;
    }
```

» DB에 있는 아이디의 경우 가입이 불가능

```
public void loginCheck() {
    MovieMember movieMember_login = new MovieMember(); //빈 dto 선언
    movieMember_login.setId(t_id.getText());
    movieMember_login.setPass(new String(t_pass.getPassword()));
    movieMember = movieMemberDAO.select(movieMember_login);
    movieMember.getMoviemember_idx();

    String strId = t_id.getText();
    String strPass = new String(t_pass.getPassword());


      if(strId.isEmpty()==true && strPass.isEmpty()==true) {
          JOptionPane.showMessageDialog(this, "Login Fail");
      }else {
          JOptionPane.showMessageDialog(this, "Login Complete");
          reservationMain.showHide(reservationMain.LOGINCOMPLETE);
          t_id.setText("");
          t_pass.setText("");

          LoginComplete l_com = (LoginComplete)reservationMain.page[ReservationMain.LOGINCOMPLETE];
          l_com.getHistory();

      }
```

» 아이디와 비밀번호 입력시 DB에 존재하면 로그인 완료창으로 이동

예매 기능 구현


```
combo_date.addItemListener(new ItemListener() {
    public void itemStateChanged(ItemEvent e) {
        int index=combo_date.getSelectedIndex();
        selectedDate = dateList.get(index-1);//선택한 날짜정보

    }
});

combo_time.addItemListener(new ItemListener() {
    public void itemStateChanged(ItemEvent e) {
        int index=combo_time.getSelectedIndex();
        selectedTime = timeList.get(index-1);//선택한 시간정보

    }
});
```

영화별로 날짜, 시간을 콤보박스에서 선택

```
//예매 전롱받기
public void showConfirm() {
    String value = (String) table.getValueAt(table.getSelectedRow(), 0);
    Movie movie = reservationMain.movieDAO.select(Integer.parseInt(value));
    String movieTitle=movie.getTitle();

    System.out.println(movieTitle);

    String date = (String)combo_date.getSelectedItem();
    System.out.println(date);

    String time = (String)combo_time.getSelectedItem();
    System.out.println(time);

    if( JOptionPane.showConfirmDialog(this, "선택 영화 : "+ movieTitle+"\n"+"선택 날짜 : "+date+"\n"+"선택 시간 : "
    +time + " 에 예매하시겠습니까?", "영화선택",JOptionPane.YES_NO_OPTION) == JOptionPane.OK_OPTION) {
        reservationMain.showHide(reservationMain.SEATSELECTEPAGE);
    }

    //System.out.println(deliverMovie());
}
```

좌석 선택 전 영화명, 날짜, 시간을 컨펌 후 좌석 예매칸으로 이동

예매 기능 구현

```
protected void paintComponent(Graphics g) {
    Graphics2D g2 = (Graphics2D)g;
    g2.clearRect(0, 0, 130, 120);

    g2.setColor(color);
    g2.fillRect(0, 0, 120, 100);

    g2.setFont(new Font("Times New Roman",Font.BOLD,fontSize));
    g2.setColor(Color.DARK_GRAY);


    g2.drawString(seatTable.getSeatRow()+seatTable.getSeatCol(), 35, 25);
}
```

Graphics를 사용하여 객체로 좌석 디자인

```
//좌석 생성
public void createSeat() {
    for (int i = 0; i < seatList.size(); i++) {
        rowList = seatList.get(i);
        for (int a = 0; a < rowList.size(); a++) {
            SeatTable seatTable = rowList.get(a); // 좌석의 dto
            SeatDesign seatDesign = new SeatDesign(seatTable, 20, this);
            p_seat.add(seatDesign);
        }
    }

}

// 선택한 좌석 번호를 받아와보자
public void getSeat() {
    seatList = (ArrayList) seatTableDAO.selectAll();
}
```

좌석 생성 및 선택한 좌석의 idx 받아오기

```
//예매내역 테이블에 한 건 추가하자(reservate_list)
public void insertReservate() {
    history = new History();
    //1) moviemember_idx받아오기
    LoginPage loginPage = (LoginPage)reservationMain.page[ReservationMain.LOGINPAGE];
    history.setMovieMember(loginPage.movieMember);
    System.out.println(loginPage.movieMember.getMoviemember_idx()+" member_idx");

    //2. product_Idx 가져오기
    Product getProduct = getProduct();
    history.setProduct(getProduct);

    int result = historyDAO.insert(history);
    if(result>0) {
        System.out.println("history에 1건 추가 완료");

    }
}
```

선택한 영화, 날짜, 시간, 좌석에 해당하는 product_idx를 회원별 예매내역에 추가

회원별 예매내역 기능 구현

```
// 멤버별로 예약내역을 가져올 메서드
public void getHistory() {
    history = new History();
    loginPage = (LoginPage) reservationMain.page[ReservationMain.LOGINPAGE];
    history.setMovieMember(loginPage.movieMember);
    int h_m_idx = history.getMovieMember().getMoviemember_idx();

    la_complete.setText(history.getMovieMember().getName()+"'s Reservate List");

    model.historyList = historyDAO.selectAll(h_m_idx);
    table.updateUI();
}
```

- 회원별로 저장된 product_idx를 사용하여 예매내역 호출

```
//예매내역 취소하기
public void delete() {
    if (JOptionPane.showConfirmDialog(this, "예매를 취소하시겠습니까?", "예매 취소",
            JOptionPane.YES_NO_OPTION) == JOptionPane.OK_OPTION) {
        int result = historyDAO.delete(detail.getHistory_idx());
        if (result > 0) {
            JOptionPane.showMessageDialog(this, "예매가 취소되었습니다");
            getHistory();
        }
    }
}
```

- 예매 취소 시 회원별 예매 DB에서 삭제

```
// 예매내역 상세보기
public void getDetail() {
    detail = model.historyList.get(table.getSelectedRow());
    t_title.setText(detail.getProduct().getMovie().getTitle());
    t_date.setText(detail.getProduct().getDate().getRunningDate());
    t_time.setText(detail.getProduct().getTime().getRunningTime());
    t_seat.setText(detail.getProduct().getSeat().getSeatRow() + detail.getProduct().getSeat().getSeatCol());

}
```

- 회원별 DB에 저장된 예매내역의 상세내역을 출력

JavaScript 프로젝트

1. 미로게임
2. 캐릭터 슬롯머신

미로게임 프로젝트 소개

프로젝트명

미로찾기 게임

주제

2차원 배열을 이용한 미로찾기게임

선정 동기

이차원 배열이라는 개념이 흥미롭게 느껴져 알아보던 중, 해당 개념을 접목하여 미로 맵을 구현하면 어떨까 하는 생각을 하게 되면서 탈출 게임을 제작하게 됨

개발환경 소개

JavaScript 프로젝트 개발환경

구성요소

[개발도구] Visual Studio

[OS] Window 10

사용언어 및 스타일 적용

[사용언어] JAVA (jdk 1.8)

[스타일] HTML5 / CSS3

[스크립트 언어] JavaScript

메인 기능

2차원 배열을 통한 미로 맵 구현

특정 위치 도착 시 생명력 추가

시간 제한 기능

성공 or 실패 시 이미지 추가

주요 기능 구현

[code]

```
let mapArray=[
[0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1],
[1,0,1,1,1,1,1,1,0,0,0,0,0,0,0,1,1,1,1],
[1,0,0,0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,2,0,1],
[1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1],
[2,0,0,0,0,0,0,0,1,1,1,0,0,0,0,0,1,1,1,1,1],
[1,1,1,1,1,1,1,0,0,0,0,1,1,1,1,1,1,1,1,1,1],
[1,0,0,0,1,1,1,1,1,1,1,1,1,1,0,0,0,0,1,1,1],
[1,1,1,0,1,0,0,0,0,1,1,1,1,1,0,1,1,0,1,1,1],
[1,0,0,0,0,1,0,1,1,0,0,0,0,0,0,1,1,0,0,0,1],
[1,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,1,1,0,1],
[1,0,0,0,0,0,0,1,1,2,1,1,0,0,0,0,0,0,0,0,1],
[1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,0,1],
[1,2,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,1,1],
[1,1,1,0,1,1,0,1,0,1,1,1,1,1,1,1,1,1,1,1,1],
[1,1,1,0,1,1,1,1,0,1,1,0,0,0,0,1,1,0,1,1,1],
[0,0,0,0,1,1,1,1,1,0,1,0,0,1,1,0,0,0,0,1,1],
[3,1,1,0,1,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1,1],
[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]
];
```

2차원 배열로 미로 구성
0: 길, 1: 벽 2: 아이템 3: 도착 지점

```
switch(this.type){
    case 0: this.img.src ="./images/floor2.png"; break;
    case 1:this.img.src="./images/wall.png"; break;
    case 2:this.img.src="./images/item.png";
            itemArray.push(this);break;
    case 3: this.img.src="./images/Home.png";break;
}
```

switch 사용하여 각 숫자에 이미지 추가

```
function moveX(n){
    let result = mapArray[stepY][stepX+n];

    if(result !=1){
        stepX+=n;//좌표계산
        meBox.stepX=stepX;
    }
}

function moveY(n){
    let result = mapArray[stepY+n][stepX];
    if(result !=1){
        stepY+=n;//좌표계산
        meBox.stepY=stepY;
    }

    check();
}

function keyEvent(e){
    switch(e.keyCode){
        case 37: moveX(-1);break; //좌
        case 38: moveY(-1);break; //상
        case 39: moveX(1);break; //우
        case 40: moveY(1);break; //하
    }

    meBox.tick();
```

키 이벤트로 움직임 제어, 벽(1)을 만나는 경우 반대 방향 조건을 주어 지나갈 수 없도록 지정

주요 기능 구현

[code]

```
function chance(){
    ///(container,src, x,y,width,height,velX,velY)
    for(let i=0; i<3; i++){
        let chance=new Chance(detailArea,"./images/chance.png",detailArea,600+(80*i),80,80,0,0);
        chanceArray.push(chance);
        chanceCnt++;
    }
}
```

› 생명은 3개로 설정, 클래스로 만들어 화면에 부착

› 아이템 추가 지점을 지나가면 생명력 배열에 1 추가

```
function createItem(){
    //꽃 이미지를 밟으면 목숨배열에 1이 추가되도록 하고싶음
    let result;
    for(let i=0; i<itemArray.length; i++){
        result = (meBox,itemArray[i]);
        if(itemArray[i].x == meBox.x && itemArray[i].y == meBox.y){
            //alert("닿았다");
            let chance=new Chance(detailArea,"./images/chance.png",detailArea,600+(chanceCnt*80),80,80,0,0);
            chanceArray.push(chance);
            chanceCnt++;
            itemArray[i].img.src="./images/floor2.png";
            itemArray[i].type=0;
            itemArray.splice(i,1);
        }
    }
}
```

캐릭터 슬롯머신 프로젝트 소개

프로젝트명

캐릭터 슬롯머신

주제

자바스크립트로 구현한 옷입히기 게임

선정 동기

어린 시절 즐겨하던 옷입히기 게임을 javaScript로 구현해보고 싶은 생각이 들어 구현하게됨

구현 기능 소개

메인 기능

- 각 배열에 추가된 이미지를 따라 일정 속도로 움직이기
- 선택된 이미지 캐릭터에 추가

주요 기능 구현

[code]

```
let hairNameArray=["hair1.png","hair2.png","hair3.png","hair4.png","hair5.png","hair6.png","ha
let hairArray=[];
let faceNameArray=["face1.png","face2.png","face3.png","face4.png","face5.png","face6.png","
let faceArray=[];
let shirtNameArray=["shirt1.png","shirt2.png","shirt3.png","shirt4.png","shirt5.png","shirt6.png
let shirtArray=[];
```

← 1. 각 배열에 이미지 추가
✓ 2. 클래스로 작성한 객체들을 호출
↓ 3. 각 테마 마다 별도로 flag를 주어 각각 멈출 수 있도록 작성

```
function createHair(){
    //1.헤어 칸에 헤어 이미지를 컨베이어로 붙이자.
    //멈춤 버튼 추가하자
    //container,src,x,y,width,height,velX,velY
    for(let i=0; i<hairNameArray.length;i++){
        let hair=new Hair(hairArea,"./images/"+hairNameArray[i],i*250,0,250,250,-22,0);
        hairArray.push(hair);
    }
```

```
function move(){
    if(flag1){
        for(let i=0; i<hairArray.length;i++){
            hairArray[i].tick();
            hairArray[i].render();
        }
```

감사합니다.

김나연