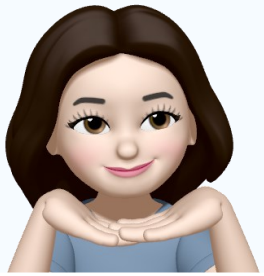
Shoe CON

About Us

함께 협동하고, 맞춰가고, 발전하는 팀원들을 소개합니다.

Introducing team members who cooperate, adapt, and develop together.

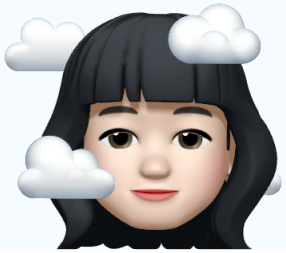


강지원

MVC/Ctrl, View 서포트, User Flow

이재환(총괄)

MVC / Model 파트(제품 부분) ,Crawling
프로그램 요구사항 분석 및 정보 취합



이민경

MVC/Ctrl, View 서포트,
Class Diagram

이재호

MVC/Model(회원),
웹크롤링 분석 및 정리, 발표

허정연

MVC/ view , Ctrl 서포트,
기능명세 제작 및 PPT 제작

Overview

전체적인 흐름을 이해하고, 기획합니다.

Understand and plan the overall flow.

기획 배경

- 팀원들의 선호하는 취미 및 특기 종합, 쇼핑을 좋아하는 공통점 발견
- MVC 패턴 및 Crawling을 이용하여 JAVA 기반 쇼핑 프로그램 기획

기획 목적

- MVC 패턴 및 Crawling의 분석 및 이해와 공부, 협업 능력 증진,
- 장바구니를 이용한 편리한 쇼핑 시스템 구축

기대 효과

- MVC 패턴 및 Crawling에 대한 이해도 상승
- 쇼핑몰 웹페이지에 대한 이해도 상승
- 이해도를 바탕으로 더욱 편리한 시스템 구축 가능

기능 요약

- 회원가입
- 로그인
- 로그아웃
- 장바구니 및 구매
- 상품 재고 관리
- 유효성 검사
- 검색(이름, 가격)

개발 환경

Java

eclipse

JSOUP

TimeTable

정해진 시간 안에 주어진 업무를 빠르게 처리할 수 있도록 정리한 타임테이블

This is a timetable organized so that given tasks can be processed quickly within a specified time.

기획 일정	1W				2W						3W	
	7	8	9	10	11	12	13	14	15	16	17	18
주제 선정 및 역할 분담 / 프로그램 설계												
View 완성, 크롤링 완성 및 데이터 무결성 검사												
Model 완성												
Ctrl 완성, 코드 종합 확인												
코드 최종 확인, PPT, 클래스 다이어그램, ERD, Userflow 최종 작성												
데드라인(마지막 날), 발표 준비 최종 확인												

Functional Specification

기능명세서

대분류	중분류	메서드명	주요 기능	코멘트
회원관리	회원가입	addMember(MemberVO mvo)	아이디 입력	중복검사
			비밀번호 / 확인	pw == pw2
			이름 입력	정규식 없음
	로그인	login(String id, String pw)	아이디 입력	로그인 성공시 관리자/ 사용자메뉴로 이동
			비밀번호 입력	
관리자	상품관리	updateCnt(ProductVO pvo)	재고 변경	setCnt(int Cnt)
	상품목록	selectAll()	상품 목록 출력	If(cnt <= 5) => 품절임박
	매출확인	salesPriceAll()	매출 출력	
사용자	상품검색	selectOne(String query)	이름으로 검색	%query%
		selectOne(int query)	가격으로 검색	round(query)
	장바구니	addCart(ProductVO pvo)	장바구니 추가	제품번호, 구매수량 전달
		buyCart(HashMap<String, Integer>)	구매하기	money - (price * cnt)

ERD

Web Crawling 을 통한 PRODUCT 테이블 생성과 MEMBER 테이블 참조

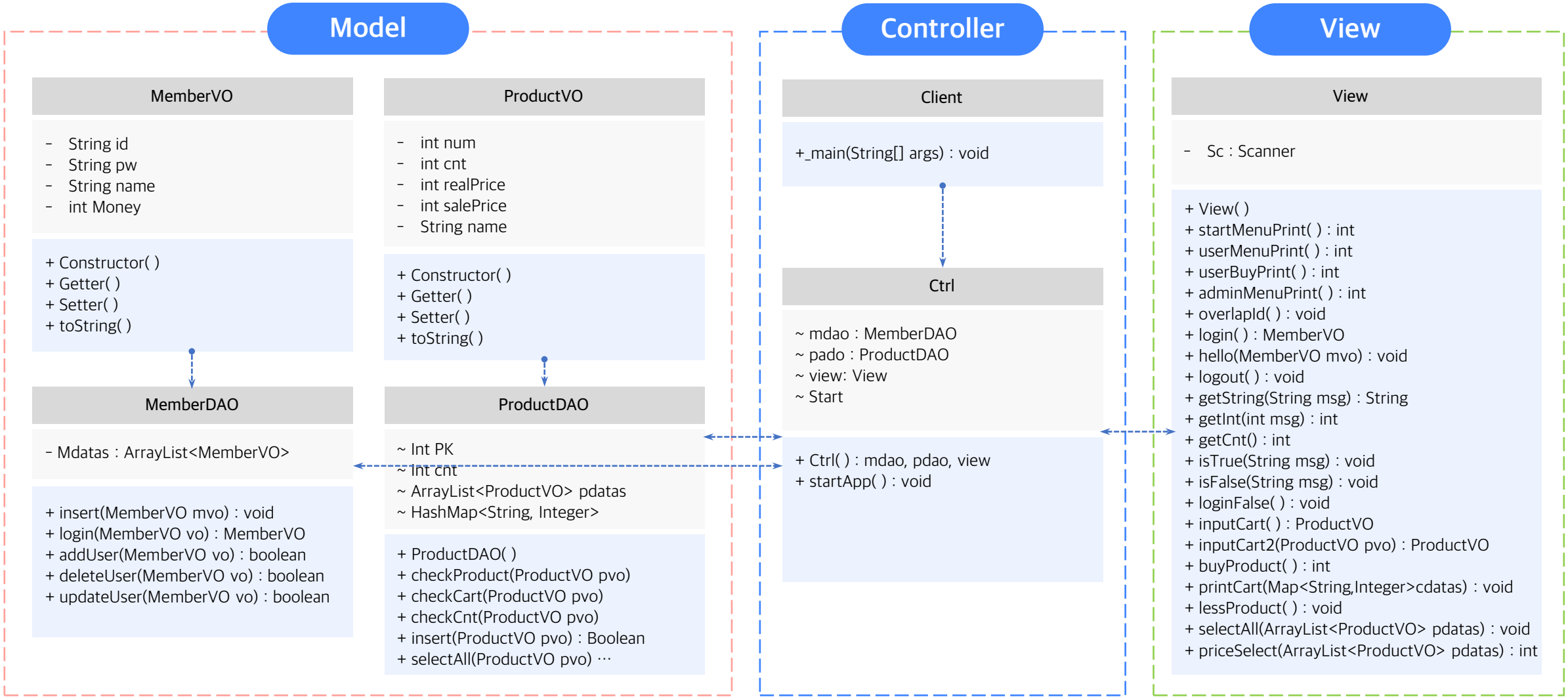
Creating a PRODUCT Table with Web Crawling and referencing the MEMBER Table

MEMBER		
id	varchar2(20)	(PK)
pw	varchar2(20)	(NN)
name	varchar2(20)	(NN)
money	varchar2(20)	
num	varchar2(20)	(FK)

PRODUCT		
num	number	(PK)
cnt	number	(NN)
realPrice	number	(NN)
salePrice	number	
name	varchar2(10)	
sum	number	(FK)

Class Diagram

Model, View, Controller 디자인 패턴에 따른 클래스 분리



User Flow


시작 화면
회원가입
로그인
프로그램 종료
ID, PW, 이름
관리자인가?
No
Yes
유효성 검사
No
회원가입 실패
관리자 메뉴
사용자 메뉴
Yes
회원가입 성공
상품관리
상품목록
매출확인
로그아웃
상품보기
로그아웃
재고변경
목록출력
매출출력
검색
장바구니
결제

Crawling Analyze

웹 크롤링을 분석하고, 파악하여 정확하게 인지합니다.

It analyzes, understands, and recognizes web crawling accurately.

MVC / Model _ Cart

데이터 수집 및 가공
가공한 정보를 토대로 핵심 로직(비즈니스 메서드) 작성

```
public int buyCart(HashMap<String, Integer> cdatas) {
// 구매 로직
ProductVO pvo=new ProductVO();
// ProductVO 객체 생성
int nowSalesPrice=0; // 총 매출액
int oneCartPrice=0; // 장바구니 구매 금액
if(cdatas.isEmpty()) { // 카트가 비어있다면
return nowSalesPrice; } boolean isExists;
// 제품이 있는지 확인하는 flag
String productName;
// 판매 제품 이름 (장바구니x, 판매하고 있는 제품)
int buyCnt; // 장바구니에 담은 제품 수량
int shortCnt=0; // 모자란 재고 (담은 수량 - 재고)
while(true) { for(int i=0; i<pdatas.size(); i++) {
productName=pdatas.get(i).getName(); // 비교 제품 이름
isExists = cdatas.containsKey(productName);
// 제품이 있다면 true, 없다면 false 저장
if(isExists) { // 제품이 있다면
```

```
buyCnt=cdatas.get(productName); // 장바구니에 담은 제품의 숫자
if(pdatas.get(i).getCnt()<buyCnt){ // 재고 확인
shortCnt=buyCnt-pdatas.get(i).getCnt();          // 모자란 재고
buyCnt=pdatas.get(i).getCnt();          // 재고만큼 구매 수량 변경
}
salesNameDatas.add(pdatas.get(i).getName());          // 구매한 제품 이름 저장
oneCartPrice+=(pdatas.get(i).getSalePrice()*buyCnt); // 장바구니 총 구매 금액 저장
nowSalesPrice=(pdatas.get(i).getSalePrice()*buyCnt); // 총 매출 저장
salesPriceDatas.add(nowSalesPrice);     // 구매한 제품 가격 * 수량 저장
pdatas.get(i).setCnt(pdatas.get(i).getCnt()-buyCnt); // 구매한만큼 재고 소진
pvo.setName(productName); // 구매하려한 제품 이름 저장
pvo.setCnt(shortCnt); // 구매하지 못한 수량 저장
cantBuyDatas.add(pvo);          // 구매하지 못한 제품의 이름과 수량
cdatas.remove(productName);          // 구매한 물품 장바구니에서 비우기
}
if(cdatas.size()<=0) {          // 장바구니가 비워지면
break;
}
return oneCartPrice;
```

MVC / View

화면구성 : UI(User Interface) / UX(User Experience) 디자인
경계값 및 유효성 검사(입력값 확인 등)

```
public class View {
  private Scanner sc; // 스캐너
  private int mAction; // 메인 메뉴 개수
  private int uAction; // 구매자 메뉴 개수
  private int aAction; // 관리자 메뉴 개수
  private int bAction; // 구매 메뉴 개수
  private int sAction; // 검색 메뉴 개수

  public View() { // 멤버변수 초기화 ( 생성자 )
    sc = new Scanner(System.in);
                    // View를 처음 실행될때 스캐너 생성되도록
    mAction = 3;
    uAction = 1;
    aAction = 3;
    bAction = 3;
    sAction = 2;
  }
```

```
 public boolean inputPw(MemberVO mvo) {
// 비밀 번호 더블체크
    System.out.print("PW CHECK : ");
    String pw2 = sc.next();
    System.out.println();
    if (mvo.getPw().equals(pw2)) {
      return true;
    }
    return false;
  }
```

MVC / Contorlloer

Model과 View를 조합하여 설계를 바탕으로
사용자가 실제로 사용할 수 있도록 서비스 구축 (Model <-> View 연동)

```
public Ctrl() { // 멤버변수 초기화 (생성자)
    mdao = new MemberDAO();
    pdao = new ProductDAO();
    view = new View();
  }
  public void startApp() { // 프로그램 시작
    WebCrawling.sample(pdao); // 웹 크롤링 샘플 데이터 가져오기
    while(true) {
      int action = view.startMenuPrint(); // 시작 메뉴
      if (action == 1){ // 로그인
        MemberVO mvo = view.login(); // 아이디, 패스워드 입력
        mvo = mdao.login(mvo);
                    // 로그인 시도 ( null 또는 로그인 정보 담긴 객체 반환 )
        if(mvo==null) { // mvo가 null이라면
          view.loginFalse(); // 로그인 실패 출력
          continue;
        }
        view.hello(mvo); // 로그인 성공 출력
```

```
} else if (action ==2) {// 장바구니 담기
                    ProductVO pvo = new ProductVO(); // ProductVO 객체 생성
                    pvo.setNum(view.buyProduct());
                    // 어떤 상품 담을지 받아와서 pvo객체에 저장
                    if(!pdao.checkProduct(pvo)) { // 존재하지 않는 상품이라면
                      view.nullProduct(); // 존재하지 않는 상품 출력
                      continue;
                    }
                    pvo.setCnt(view.getCnt());
                    // 몇개를 담을 것인지 받아와서 pvo객체에 저장
                    pvo=pdao.checkCart(pvo);
                    // 이미 장바구니에 있는 상품인지 확인하고 구매수량 저장
                    if(!pdao.checkCnt(pvo)) { // 재고가 없을때
                      view.zeroProduct(); // 재고 없음 출력
                      continue;
                    }
                    // 재고가 있다면
                    cdatas=pdao.addCart(pvo); // 장바구니에 담기
                    view.printCart(cdatas); // 장바구니 목록 출력
                    continue;
```

MVC / Crawling

화면구성 : UI(User Interface) / UX(User Experience) 디자인
경계값 및 유효성 검사(입력값 확인 등)

```
/*
 * removeNotNumeric(String str)
 * : replaceAll 메서드를 사용했고 정규식 \W로
 * 알파벳 + 숫자 + _ 가 아닌 문자를 "" 로 바꾸
어줌 (==해당 문자를 없앰)
 */
private static String removeNotNumeric(String str) {
                return str.replaceAll("\\W", "");
                }
private static int toInt(String str) {
                return Integer.parseInt(str);
                }
```

```
/*
* 크롤링해서 가져온 가격 데이터는 dao에서 연산 시에 필요하기 때문에
* String 타입이 아닌 int 타입으로 변환이 필요함
* 그러나 가격 정보를 해당 페이지에서 크롤링해서 가져오면
* 원화(\) 표시와 반점(,)이 함께 text로 가져오게 됨
* 이를 int타입으로 형변환 시켜주기 위해 두 가지 메서드를 생성해줌
*/
String realPrice = itrRP.next().text(); // 할인 전 가격(원래 가격)
String salePrice = itrSP.next().text(); // 판매 가격
int realIntPrice = toInt(removeNotNumeric(realPrice));
int saleIntPrice = toInt(removeNotNumeric(salePrice));
// 숫자만 남기고 모두 제거 후 Int로 형변환
pvo.setSalePrice(realIntPrice);
pvo.setRealPrice(saleIntPrice);

pdatas.add(pvo); // 중복 검사를 위해 객체 저장
pdao.insert(pvo);
                }
```

CODE Demonstrate

드디어 ShoeCON의 코드를 발표합니다!

We finally release the code for ShoeCON!

Error

에러사항을 확인하고 해결방안을 파악하였습니다.

I checked the error and figured out the solution.

크롤링을 하던 와중 NoSuchElementException 예외가 발생
확인해보니 존재하지 않는 것을 가져오려고 할 때 해당 예외가 발생

1. 정보를 가져오려는 위치가 잘못됨.
2. 위치는 정상적이지만 정보를 가져오려고 시도할 때 문제가 있음.

CODE

```
Exception in thread "main" java.util.NoSuchElementException
        at java.base/java.util.ArrayList$Itr.next(ArrayList.java:1000)
        at model.WepCrawlig.main(WepCrawlig.java:36)
```

해결방안

타겟사이트의 html 태그가 잘못 되었을 수도 있겠다고 생각함.

getElementsByClass 메소드를 사용하여 찾아줌
정상적으로 크롤링이 가동했고 문제 해결

정규표현식 패턴과 관련된 오류로
split() 메소드나 replaceAll() 메소드 등을 사용할 때 나타날 수 있는 예외

Crawling에서 특수문자를 제거해주기 위해
replaceAll()을 사용한 메소드를 만들어 사용했는데 문제발생

CODE

```
Exception in thread "main" java.util.regex.PatternSyntaxException
: Unexpected internal error
        at java.base/java.util.regex.Pattern.error(Pattern.java:2028)
        at java.base/java.util.regex.Pattern.compile(Pattern.java:1789)
        at java.base/java.util.regex.Pattern.<init>(Pattern.java:1430)
        at java.base/java.util.regex.Pattern.compile(Pattern.java:1069)
        at java.base/java.util.regex.Pattern.replaceAll(String.java:2942)
        at model.WepCwraling.removeNotNumeric(WepCrawling.java:57)
        at model.WepCwraling.main(WepCrawling.java:42)
```

해결방안

특수문자를 사용할 때 역슬래시(\ )를 붙여줌으로써 해결

Improvements

코드를 이해하고 문제인식과 해결방안을 학습하였습니다.

I learned how to use code and learned how to recognize problems and solve them.

개요

- 코드 최신화
- MVC 패턴 미숙

문제 인식

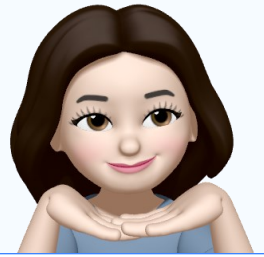
- 코드가 최신화가 되지 않아 서로의 코드가 달라서 코드 디버깅에 문제가 있었다.
- MVC 패턴을 분리하여 협업하는 게 처음이라 설계 부분에서 고생했다.

해결 방안

- git을 사용하여 code를 최신화하고 협업 효율을 늘리는 방법
- 프로젝트를 진행하며 이해도가 높아졌기 때문에 다음 프로젝트에서는 더욱 발전한 설계를 할 수 있을 것으로 보임

review

강지원

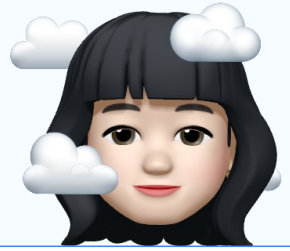


처음 mvc 패턴으로 프로젝트를 진행했는데
생각보다 어려운 부분도 많고 재밌는 부분도 많았습니다.
확실히 프로젝트를 통해 mvc패턴에대한
이해도가 높아진 것 같습니다.
다음 프로젝트에선 경험해보지못한 모델 파트도
진행해보고싶습니다

이재환

Crawling을 사용하며 많은 공부가 되었고
MVC 패턴으로 각 파트별로 분담해서 프로젝트를 진행하니
협업이 잘되어서 좋았습니다. 분명 어려운 부분도 있었지만
열정적인 팀원분들과 소통하여 잘 헤쳐나가
완성하게되어 기쁩니다.

이민경

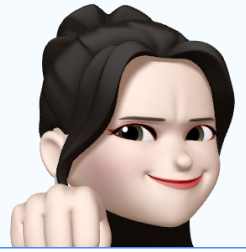


공부할 때 컨트롤 부분을 제일 어려워했었던 기억이 있어서
도전해 볼까 싶은 마음으로 뛰어들게 되었습니다.
처음에는 어떻게 짜야 할지 막막했으나 프로젝트를 진행하면서
전체적인 코드 흐름을 읽는 능력이 많이 향상된 것 같습니다.
또한, 뷰도 같이 작성해 보면서 사용자 편의성에 대해서도
다시 한번 생각해 볼 수 있어 좋았습니다.

이재호

MVC모델을 활용한 프로젝트를 진행하면서
실제 프로젝트 진행과정을 간접적으로 체험할 수 있었습니다.
특히 역할분담을 통하여 각자 작업을 따로 진행하는 부분과,
이를 합치는 부분에서 여러 문제가 발생하기도 했으나,
해결하면서 많은 공부가 되었던 것 같습니다.

허정연



팀원분들과 함께하니 어려웠던
MVC패턴의 이해도가 정말 많이 높아짐을 느꼈습니다.
모두의 열정과 노력덕분에 협업이 즐겁고
더욱 열심히 하게 된 것 같습니다.
View파트를 진행하면서 편한 UI를 더욱 고려하게 되어
많은 공부가 되었던 프로젝트였습니다

QnA Time

Thank you!