PORTFOLIO



컴퓨터공학과 · 데이터 운영·분석 직무 지원

안녕하세요. 기술과 비즈니스를 연결하는 천세령입니다.

믿고 쓸 수 있는 데이터를 만듭니다. 정합성을 맞추고, 이력을 남기고, 결과를 검증합니다.

데이터 정합성·무결성 | 거래 데이터 이력 관리 | SQL 데이터 조회·검증 | 오류 원인 분석·대응 | 시스템 안정성·협업

E-MAIL. seryung011400@gmail.com

GITHUB. github.com/seryung0114

MAJOR. 컴퓨터공학과

KEY RESULTS.

한눈에 보는 핵심 성과

검증 · 정합성 · 이력 관리

01

검증

2건

가입 - 구매 - 리뷰 사용자 시나리오 테스트로
발견해 재현 절차와 함께 공유한 오류

관련 프로젝트 교내 중고 거래 웹사이트

02

정합성

0건

발견 오류 2건 수정, 병합 절차·상품 ID 규칙 통일 후
가입 - 구매 - 리뷰 최종 시연에서 발생한 오류

관련 프로젝트 교내 중고 거래 웹사이트

03

이력 관리

주문 당시 기준 집계

가격·고객 정보 변경 후에도
과거 매출을 당시 단가·고객 상태로 집계

관련 프로젝트 화장품 매장 판매 분석 시스템

INTRODUCTION.

기술을 활용해 데이터 문제를 직접 발견하고 해결합니다.

코드가 잘 돌아가는 것만큼 그 결과를 사람이 믿고 쓸 수 있는지가 중요하다고 생각합니다. 개발 경험을 살려 데이터를 정확하게 다루고, 그 결과가 실제 업무 판단으로 이어지도록 연결하겠습니다.



할 수 있는 것

SQL과 Java를 활용한 데이터 구조 설계 및 조회	화장품 매장 판매 분석
Python과 Firebase를 활용한 웹 기능 구현	교내 중고 거래 웹사이트
Git 협업 과정에서 발생한 문제 분석 및 해결	교내 중고 거래 웹사이트
데이터 검증과 테스트를 통한 결과 신뢰성 확보	두 프로젝트

프로필

NAME	천세령
E-MAIL	seryung011400@gmail.com
GITHUB	github.com/seryung0114

학력

2022	창덕여자고등학교 졸업
2024.03 – 재학 중	이화여자대학교 컴퓨터공학과

경력

2025.04 – 2026.07 **미래탐구 학원 · 강사 개인 조교** 1년 4개월

- 중·고등학생 학습 관리
- 시험 감독 및 채점
- 학습 자료 제작·관리
- 학생 질의응답

2024.06 – 2025.03 **블루메쯔 레스토랑 · 홀 서빙** 10개월

- 홀 서빙
- 고객 응대

어학

2026.08 – 2026.12 **ILSC Toronto English Language School · 캐나다 토론토** 5개월

- 최상위 반(Advanced 2) 도달
- AI 기반 영어 학습 앱 기획 및 영어 발표
- 다국적 학생들과 역할을 나눠 비즈니스 상황 영어 토론

2025.09.28 **TOEFL 80점**

SKILLS.

기술 및 역량

프로젝트에서 직접 사용한 기술과 강점

언어·도구

SQL	▪ MySQL 테이블 8개 팀 공동 설계 (고객 이력·주문 당시 단가 구조 포함) ▪ 주문일과 이력 유효 기간을 맞춘 주문 당시 기준 매출 집계 조회문 작성 사용한 곳 화장품 매장 판매 분석 시스템
Python	▪ Flask: 사용자 인증·로그인 상태 처리, 찜 등록·해제 기능 구현 ▪ Firebase: 사용자·상품 데이터 저장·조회 사용한 곳 교내 중고 거래 웹사이트
Java	▪ MySQL 연동 입력·조회 기능 개발 ▪ PreparedStatement 바인딩, 저장 전 누락·중복 검증, 오류 시 트랜잭션 롤백 사용한 곳 화장품 매장 판매 분석 시스템
Git	▪ 브랜치 관리, 병합, 충돌 해결 사용한 곳 교내 중고 거래 웹사이트
C	▪ 포인터·메모리 구조 등 프로그래밍 기초 학습, 수업 과제로 구현 연습 사용한 곳 컴퓨터공학 전공 수업

강점 역량

STRENGTH 01

불일치의 원인을 찾아 기준 세우기

병합 충돌과 찜·리뷰 데이터 오매핑의 원인을 팀 작업 기준 불일치로 분석함. 공용 파일 수정 순서, 테스트 브랜치 병합 절차, 상품 고유 ID 규칙을 제안해 두 문제를 해소함.

관련 프로젝트 교내 중고 거래 웹사이트

STRENGTH 02

거래 당시 데이터 보존

주문 시점 단가를 별도 칼럼에 저장하고, 주문일과 고객 이력 유효 기간을 맞춰 조회함. 가격 변경 후에도 과거 매출을 주문 당시 기준으로 집계함.

관련 프로젝트 화장품 매장 판매 분석 시스템

STRENGTH 03

테스트를 통한 오류 발견 및 검증

가입부터 구매·리뷰까지 핵심 사용자 시나리오를 테스트해 오류 2건을 발견하고 재현 절차와 함께 공유함. 저장 전 입력값 검증과 트랜잭션 롤백으로 잘못된 데이터 저장을 차단함.

관련 프로젝트 교내 중고 거래 웹사이트 · 화장품 매장 판매 분석 시스템

PROJECT 01.

교내 중고 거래 웹사이트

5인 팀으로 Flask·Firebase 기반 교내 중고 거래 웹사이트를 개발함(약 2개월). 로그인·찜하기 기능을 구현하고, 병합 과정에서 발생한 충돌과 찜·리뷰 데이터 오매핑을 팀 작업 기준 통일로 해소함.

01

2건

가입 - 구매 - 리뷰 시나리오 테스트로 발견해 재현 절차와 함께 공유한 오류

02

0건

발견 오류 2건 수정, 병합 절차·상품 ID 규칙 통일 후 가입 - 구매 - 리뷰 최종 시연에서 발생한 오류

개요

기간	약 2개월
인원	5인 팀
담당 역할	로그인·찜하기 구현, 병합 절차 제안, 사용자 시나리오 테스트
기술 스택	Python · Flask · Firebase · Git
핵심 역량	**Python, Flask, Firebase, Git 협업 및 오류 해결**
GitHub	github.com/ChunHajin/osp-EwhaMarket-Team3

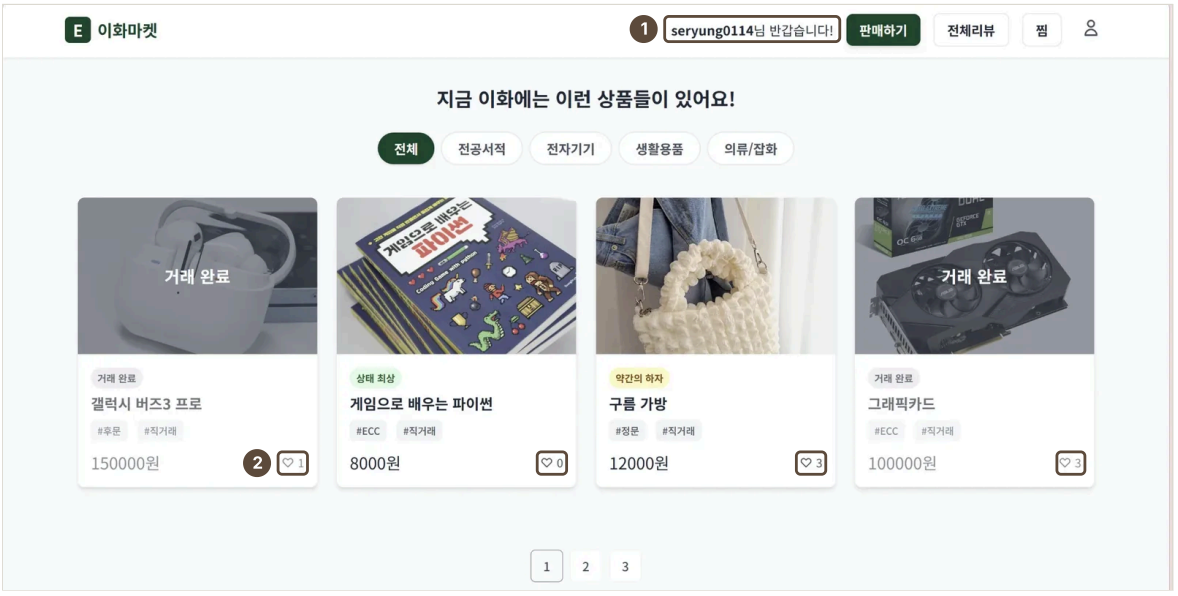
01 문제 정의

배경	기능별로는 정상 작동했으나 코드 병합 순간 문제 발생
해결하려던 문제	① 같은 파일을 동시에 수정해 병합 충돌 반복 ② 팀원마다 다른 값으로 상품을 가리켜 상품 연결 불일치
목표	병합 충돌 없는 기능 통합, 찜·리뷰 데이터의 정확한 상품 매핑

02 구현 내용

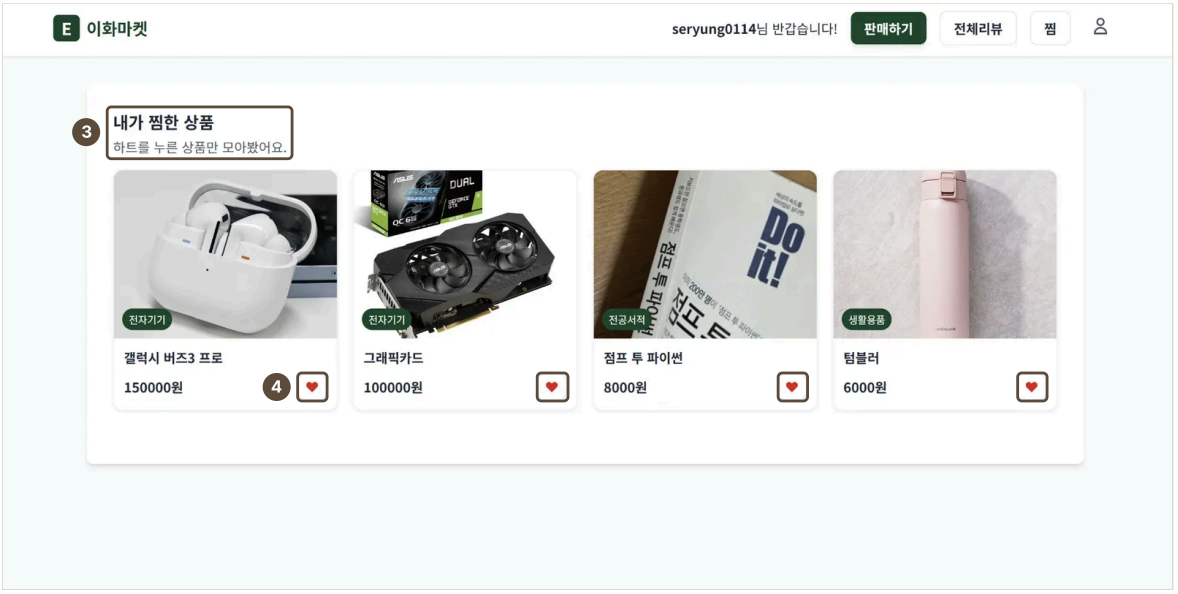
로그인	사용자 인증 및 로그인 상태 처리 (Flask)
찜하기	특정 상품의 찜 등록 및 해제
Firebase	사용자와 상품 데이터 저장 및 조회
Git 협업	테스트 브랜치에서 기능 검증 후 main 브랜치에 병합

실행 화면



로그인 후 메인 화면

① 로그인 상태(사용자 이름) 표시 ② 상품별 찜 수



내가 찜한 상품 화면

③ 사용자가 찜한 상품만 모아 표시 ④ 찜 등록·해제 버튼

03 핵심 문제 해결

사례 1 **병합 충돌 반복** Git 협업

문제	원인	해결	결과
동일 파일 동시 수정으로 병합 시 충돌 반복 발생	공용 파일 수정 순서 및 브랜치 병합 규칙 부재	공용 파일 수정 순서 지정, main 반영 전 테스트 브랜치에서 통합 결과를 먼저 확인하는 병합 절차 도입	**이후 병합 시 충돌 미발생**

내 역할 원인을 팀 작업 기준 불일치로 판단, 팀원 의견을 모아 각자 작업 방식은 유지하고 공용 파일 수정 순서와 테스트 브랜치 병합 절차만 정하는 방안 제안·적용

사례 2 **찜·리뷰 데이터 오매핑** 데이터 정합성

문제	원인	해결	결과
찜·리뷰 데이터가 대상이 아닌 다른 상품에 연결	팀원별 상품 참조 기준(식별값) 불일치로 데이터 매핑 혼선 발생	상품 고유 ID로 식별 기준 통일 및 팀 내 규칙 공유	**찜·리뷰 데이터의 대상 상품 정상 매핑**

내 역할 담당한 찜 기능과 리뷰의 오연결 원인 분석, 상품 식별자를 고유 ID로 통일하는 규칙 제안 및 팀 공지

04 결과·검증

지표	판단 기준	전	후
병합 충돌	병합 시 충돌 미발생	충돌 반복 발생	**충돌 미발생**
찜·리뷰 매핑	대상 상품과 1:1 일치	타 상품으로 오연결	**대상 상품 정상 연결**
최종 시연	주요 시나리오 오류 없이 통과	테스트 중 오류 2건 확인	**오류 0건으로 시연 통과**

검증 방법	회원가입부터 구매·리뷰 작성까지 핵심 사용자 시나리오(E2E)를 직접 수행하며 검증함. 발견한 오류 2건은 재현 절차와 함께 공유 후 수정 완료함.

05 AI 도구 활용

활용 도구	ChatGPT
활용 목적	Flask 로그인 처리 및 Firebase 연동 방식 참고, Git 병합 충돌 해결 방법 조사
검증 과정	생성된 코드의 구조와 동작을 확인하고 실제 프로젝트 환경에 맞게 수정한 뒤 테스트 브랜치에서 기능 검증
직접 수행	상품 식별자 규칙 통일, 병합 절차 제안, 사용자 흐름 테스트 및 오류 수정

06 배운 점

배운 점	시스템 안정성은 개별 기능의 완성도뿐 아니라 구성 요소 간 일관성에서 비롯된다는 점
다시 한다면	개발 시작 전 공용 파일 수정 규칙·브랜치 전략·상품 ID 기준 우선 합의, 병합 시마다 사용자 시나리오 테스트 반복

PROJECT 02.

화장품 매장 판매 분석 시스템

6인 팀으로 Java·MySQL 기반 화장품 매장 판매 분석 시스템을 개발함(약 3주). 제품별 매출, 고객층(연령대·피부 타입)별 구매 패턴, 가격 변동이 판매량과 매출에 미치는 영향 추적을 목표로 함. 팀 공동 DB 설계(테이블 8개)에 참여하고, 입력·조회 기능 구현을 담당함.

01

당시 기준 집계

가격·고객 정보 변경 후에도 과거 매출을 주문 당시 기준으로 집계

02

검증 + 롤백

잘못된 데이터와 일부만 저장되는 데이터 차단

개요

기간	약 3주
인원	6인 팀
담당 역할	DB 공동 설계(테이블 8개), 입력·조회 기능 구현
기술 스택	Java · MySQL · SQL
핵심 역량	**Java, MySQL, SQL 데이터 모델링 및 데이터 정합성**
GitHub	비공개 저장소 (팀원 계정) · 테이블 구조·SQL 발췌 보기

01 문제 정의

분석 목표	제품별 매출, 고객층(연령대·피부 타입)별 구매 패턴, 가격 변동이 판매량과 매출에 미치는 영향 추적
해결하려던 문제	가격·고객 정보(연령대·피부 타입) 변경 시 지난 주문까지 새 값으로 집계되어 변화 전후 비교 불가
설계 목표	주문 당시 가격·고객 상태 보존, 잘못되거나 일부만 저장된 데이터 차단

02 구현 내용

고객 이력 테이블	고객 정보(연령대·피부 타입 등) 변경 시 유효 기간과 함께 이력으로 저장 (MySQL)
주문 단가	주문 시점의 단가를 주문 데이터의 별도 칼럼에 저장
당시 기준 집계	주문일과 고객 이력의 유효 기간을 맞춰 당시 상태 기준으로 매출 집계 (SQL)
입력·저장	PreparedStatement 바인딩, 저장 전 필수값·고객/상품 존재·중복 검증, 오류 시 트랜잭션 롤백 (Java)

데이터베이스 구조

Customer	고객	
PK	customer_id	INT
	first_name	VARCHAR(50)
	last_name	VARCHAR(50)
UQ	phone	VARCHAR(20)
UQ	email	VARCHAR(100)
	birth_date	DATE

Customer_History	고객 프로필 이력	
PK	history_id	INT
FK	customer_id → Customer	INT
	skin_type	VARCHAR(30)
	age_group	VARCHAR(20)
	grade	VARCHAR(30)
	start_date	DATE
	end_date	DATE

Orders	주문	
PK	order_id	INT
FK	customer_id → Customer	INT
	order_date	DATE

Order_Items	주문 상세	
PK	order_items_id	INT
FK	order_id → Orders	INT
FK	product_id → Product	INT
	quantity	INT
	ordered_price	INT

Product	상품	
PK	product_id	INT
FK	brand_id → Brand	INT
FK	category_id → Cosmetic_Category	INT
IDX	product_name	VARCHAR(100)
	current_price	INT

Price_History	가격 변동 이력	
PK	price_history_id	INT
FK	product_id → Product	INT
	old_price	INT
	new_price	INT
	start_date	DATE
	end_date	DATE

Brand	브랜드	
PK	brand_id	INT
	brand_name	VARCHAR(50)
	company_name	VARCHAR(50)
	brand_tier	VARCHAR(20)
	contact_number	VARCHAR(20)

Cosmetic_Category	카테고리	
PK	category_id	INT
	category_name	VARCHAR(50)

MySQL · 테이블 8개 · 뷰 2개 · 인덱스 1개 (PK 기본키, FK 외래키, UQ 중복 불가, IDX 조회용 인덱스)

강조한 칼럼: 주문 당시 단가·고객 프로필을 찾는 데 쓰는 칼럼

SQL 쿼리 예시

```
CREATE VIEW Customer_Profile_Order_View AS
SELECT
    ci.customer_id,
    ch.skin_type, ch.age_group, ch.grade,
    o.order_date,
    oi.quantity, oi.ordered_price,
    pr.product_name, cat.category_name
FROM Customer ci
JOIN Customer_History ch
    ON ci.customer_id = ch.customer_id
LEFT JOIN Orders o
    ON ci.customer_id = o.customer_id
   AND o.order_date BETWEEN ch.start_date
                        AND IFNULL(ch.end_date, NOW())
LEFT JOIN Order_Items oi
    ON o.order_id = oi.order_id
LEFT JOIN Product pr
    ON oi.product_id = pr.product_id
LEFT JOIN Cosmetic_Category cat
    ON pr.category_id = cat.category_id;
```

createSchema.sql 발췌 (일부 칼럼 생략, 줄바꿈만 정리)

핵심 조건	주문일이 프로필 적용 기간(start_date ~ end_date) 안에 있는 이력만 연결 → 주문 당시 피부 타입·연령대·등급 조회 (피부 타입·연령대별 분석 메뉴에서 사용)
end_date NULL	현재 적용 중인 이력. IFNULL로 현재 시점(NOW())까지를 적용 기간으로 처리. DATE인 order_date는 DATETIME인 NOW()와 비교할 때 그날 0시로 맞춰 비교되므로 오늘 주문도 포함
LEFT JOIN	주문이 없는 프로필 이력도 결과에 유지
작성 범위	테이블·뷰 구조는 팀 공동 설계, 이 구조를 활용한 입력·조회 기능(Java)은 직접 구현

03 핵심 문제 해결

사례 1 **가격·고객 정보 변경 시 과거 매출 왜곡** 거래 데이터 이력 관리

문제	원인	해결	결과
가격·고객 정보 변경 시 지난 매출까지 새 값으로 재계산되어 가격 변동·고객층별 분석 왜곡 우려	현재 값만 참조하는 구조로 주문 당시 가격·고객 상태 미보존	주문 시점 단가 별도 칼럼 저장, 고객 이력에 유효 기간을 두어 주문일 기준 당시 상태 조회	**가격 변경 후에도 과거 매출의 주문 당시 기준 집계 확인**

내 역할 이력·주문 단가 구조 팀 공동 설계, 주문일 기준 당시 상태 조회 기능 구현

사례 2 **잘못된 데이터와 일부만 저장된 데이터** 데이터 무결성

문제	원인	해결	결과
필수값 누락·없는 고객/상품·중복 상품 저장, 저장 중 오류 시 일부 데이터만 남을 위험	저장 전 검증 부재 시 잘못된 값 유입, 여러 단계로 나뉜 저장 중 오류 시 일부만 반영	저장 전 입력값 검증, PreparedStatement 값 바인딩, 저장 중 오류 시 트랜잭션 롤백	**잘못된 데이터와 일부만 저장된 데이터 차단**

내 역할 저장 전 입력값 검증 로직 작성, 저장 로직 트랜잭션 처리(오류 시 롤백) 구현

04 결과·검증

지표	판단 기준	결과
과거 매출 집계	가격 변경 후에도 주문 당시 단가·고객 상태로 집계	**주문 당시 기준 집계 확인**
잘못된 입력	누락·중복 값 입력 시 저장 차단	**저장 차단 확인**

검증 방법 가격 변경 후 과거 매출을 다시 조회해 주문 당시 기준 집계 여부를 확인함. 누락·중복 값을 직접 입력해 저장 차단 여부를 확인함.

05 AI 도구 활용

활용 도구	ChatGPT
활용 목적	고객 이력 테이블(유효 기간) 설계 방식 조사, PreparedStatement·트랜잭션 예시 코드 참고
검증 과정	가격 변경 후 과거 매출 재조회로 당시 기준 집계 확인, 누락·중복 값 입력으로 저장 차단 여부 확인
직접 수행	고객 이력·주문 단가 구조 공동 설계, 입력값 검증 로직 작성, 저장 로직 트랜잭션 처리(오류 시 롤백)

06 배운 점

배운 점	은행 거래 기록처럼 거래 당시의 가격과 고객 상태가 그대로 남아야 분석 결과를 신뢰할 수 있다는 점
다시 한다면	가격·고객 정보 변경 상황의 테스트 케이스화 및 반복 검증, 이력 유효 기간 규칙의 팀 공통 문서화 선행