

Actor Critic(강화학습)을 통해 증가된 Data를 이용한  
CNN(심층신경망) Model 생성 및 이를 통한 명품카드 홀더  
예측 및 중고 제품 거래 핀테크 개발

1조

정윤현, 이예진, 안수빈, 김한별, 김주현



# CONTENTS

1.	일정 .....	3
2.	역할 .....	4
3.	Tools .....	5
4.	개요 .....	7
5.	데이터 소개 .....	9
	- 데이터 선정 이유	
6.	유사 이미지 생성.....	11
7.	강화학습.....	12
	- 이미지 생성	
8.	CNN .....	13
9.	Model Test .....	15
10.	ERD .....	17
11.	Process Flow Diagram .....	18
12.	System Flow Diagram .....	19
13.	UI Flow .....	20
14.	기능 구현 .....	21
15.	향후 계획 .....	36

# 일정

		2023.03.08 ~ 2023.03.27												
		D-13	D-12	D-11	D-10	D-9	D-8	D-7	D-6	D-5	D-4	D-3	D-2	D-1
기획	주제 선정													
개발	분석													
	기능 R&D													
	프로그램 개발													
	단위 테스트													
	통합 테스트													
	디버깅													
디자인	앱 디자인													

## 역할

이름	역할	담당 업무
김주현	팀원	<ul style="list-style-type: none"><li>▶ [분석] 데이터 강화학습</li><li>▶ [개발] 로그인 및 회원가입 기능, 마이페이지(관심목록, 판매내역)</li></ul>
김한별	팀원	<ul style="list-style-type: none"><li>▶ [개발] Flask 연동 및 상품 등록, 상품 상세 페이지</li></ul>
안수빈	팀원	<ul style="list-style-type: none"><li>▶ [분석] 데이터 강화학습</li><li>▶ [개발] 사용자 정보 수정, 이미지 저장, 구매 목록</li></ul>
이예진	팀원	<ul style="list-style-type: none"><li>▶ [개발] 채팅 기능</li></ul>
정윤현	팀장	<ul style="list-style-type: none"><li>▶ [분석] 데이터 강화학습 및 CNN</li><li>▶ [개발] Flask 연동 및 메인 화면, Pay 충전 화면</li></ul>

# Tool

## Database

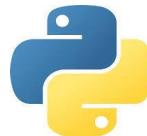


## Language



**Swift**

v5.7.2



v3.9.13

## Framework



v16.0



v14.2



v2023.1.2010391206



v1.74.2



# Tool

## 버전관리



v2.37.3



v 2.26.1



## 협업관리



## Server



# 개요



코로나19로 얹눌린 소비 심리가 표출되면서 최근 몇 년 간 해외 명품 매출이 눈에 띠는 성장세를 보이고 있다. 특히 MZ세대를 중심으로 제품을 구매하는 것이 ‘소유’ 개념이 아닌 ‘경험’에 가치를 둔 소비 행태를 보이면서 ‘리셀 시장’이 급부상하고 있다.

“MZ세대는 명품이나 한정판 신발 등을 구입하면서 얻는 차별화된 경험을 자기 표현의 수단으로 여기고 있다”

“구입에 그치는 것이 아니라 어렵게 구입한 제품에 웃돈을 얹어 되팔면서 재테크 수단으로도 활용하고 있다”

명품을 중고로 구매하거나 새 제품을 사서 일정 기간만 사용하고 되파는 문화가 만들어지면서 희소성이 높은 제품을 중심으로 ‘샤테크’, ‘스니커즈 리셀’과 같이 ‘저위험 고수익’ 재테크가 활발해지고 기존 패션 플랫폼 외에 대기업까지 시장에 뛰어드는 추세이다.

이러한 소비 경향을 파악하여 리셀할 제품을 찍어올리면 해당 제품의 품번을 판별해주고 안전한 거래까지 이루어질 수 있는 MZ세대들을 위한 쉽고 간단한 명품 리셀 플랫폼을 제공하고자 한다.



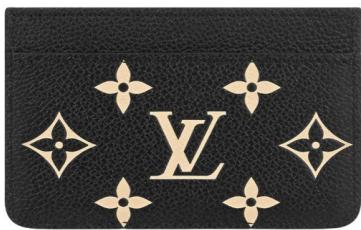
분석

# 사용할 데이터 - 600 \* 600

GUCCI 523159 96IWG 8745



LOUIS VUITTON M81022



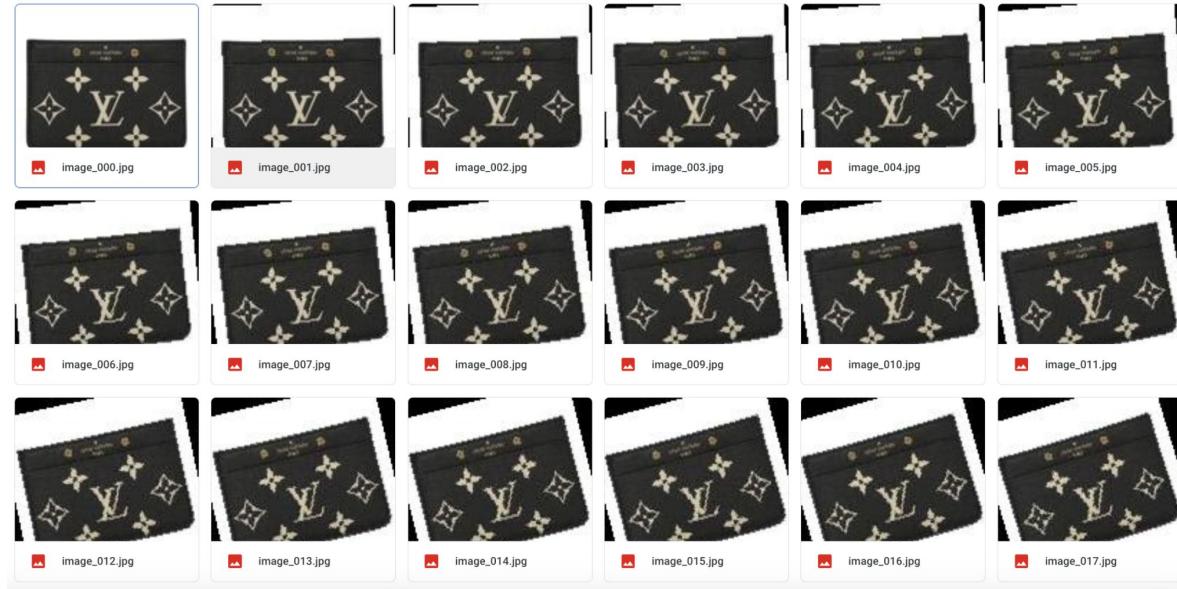
- 이미지 사이즈를 128 \* 128  
로 재설정

## ▶ 사용할 데이터 - 데이터 선정 이유



한 눈에 구별 가능한 패턴이나 LOGO가 존재하는 제품 PICK

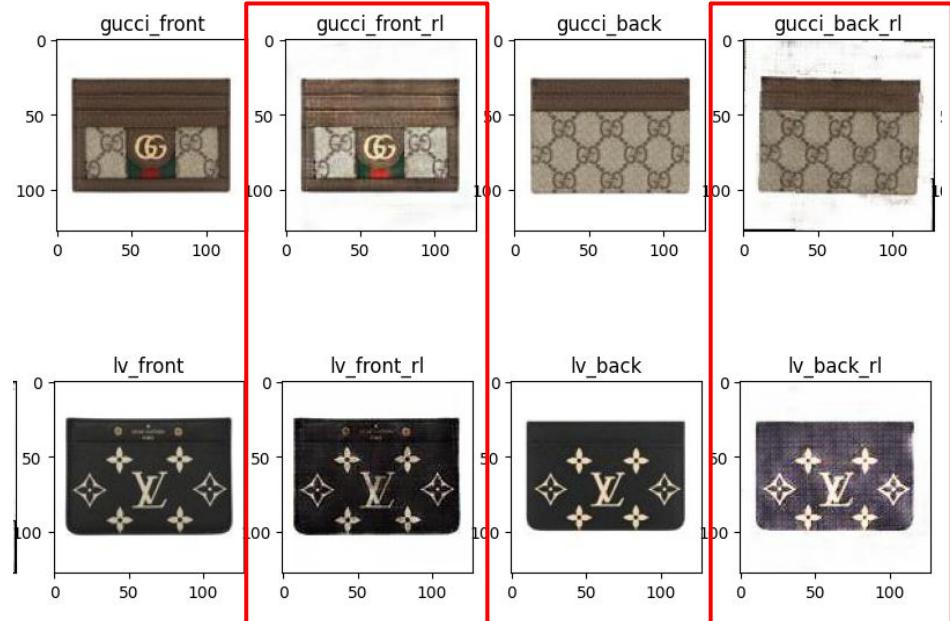
# 유사 데이터 생성



원본 이미지 데이터를 1도 간격으로 회전시킨  
회전 데이터를 생성하여 한 모델 당 360개의 유사데이터 생성

# 강화학습

Generated Images on EPOCH: 2000

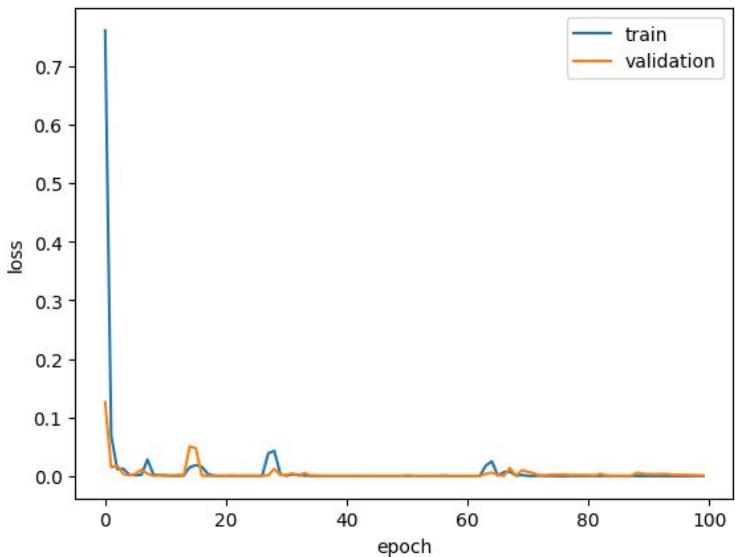


9장씩 EPOCH 2000 횟수를 주어 강화학습

# CNN

```
Model: "sequential"
-----
Layer (type)          Output Shape         Param #
=====
conv2d (Conv2D)        (None, 128, 128, 32)    896
max_pooling2d (MaxPooling2D (None, 64, 64, 32)    0
)
conv2d_1 (Conv2D)       (None, 64, 64, 64)      18496
max_pooling2d_1 (MaxPooling  (None, 32, 32, 64)    0
2D)
flatten (Flatten)       (None, 65536)           0
dense (Dense)          (None, 100)             6553700
dropout (Dropout)       (None, 100)             0
dense_1 (Dense)         (None, 4)               404
=====
Total params: 6,573,496
Trainable params: 6,573,496
Non-trainable params: 0
```

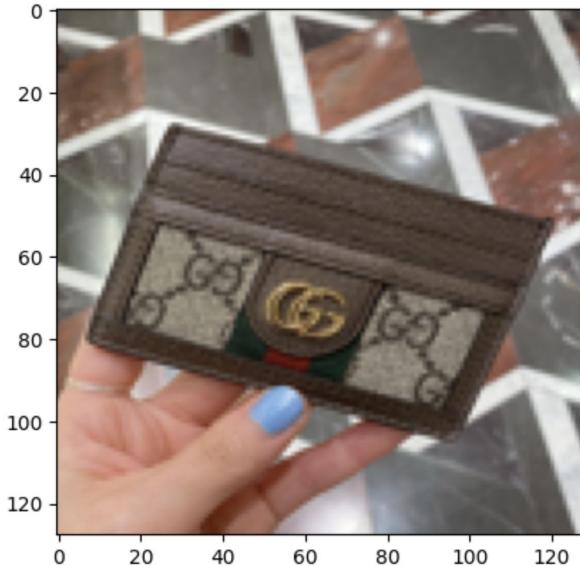
# CNN



```
Output exceeds the size limit. Open the full output data in a text editor
72/72 [=====] - 12s 163ms/step - loss: 0.7603 - accuracy: 0.7183 - val_loss: 0.1258 - val_accuracy: 0.9792
Epoch 2/100
72/72 [=====] - 11s 148ms/step - loss: 0.0686 - accuracy: 0.9848 - val_loss: 0.0149 - val_accuracy: 0.9965
Epoch 3/100
72/72 [=====] - 12s 160ms/step - loss: 0.0118 - accuracy: 0.9978 - val_loss: 0.0175 - val_accuracy: 0.9931
Epoch 4/100
72/72 [=====] - 12s 171ms/step - loss: 0.0125 - accuracy: 0.9974 - val_loss: 0.0034 - val_accuracy: 0.9983
Epoch 5/100
72/72 [=====] - 13s 177ms/step - loss: 0.0024 - accuracy: 1.0000 - val_loss: 0.0014 - val_accuracy: 1.0000
Epoch 6/100
72/72 [=====] - 12s 172ms/step - loss: 0.0015 - accuracy: 1.0000 - val_loss: 0.0036 - val_accuracy: 0.9983
Epoch 7/100
72/72 [=====] - 11s 156ms/step - loss: 0.0021 - accuracy: 0.9996 - val_loss: 0.0109 - val_accuracy: 0.9965
Epoch 8/100
72/72 [=====] - 11s 147ms/step - loss: 0.0283 - accuracy: 0.9909 - val_loss: 0.0039 - val_accuracy: 0.9983
Epoch 9/100
72/72 [=====] - 11s 156ms/step - loss: 0.0022 - accuracy: 1.0000 - val_loss: 5.1598e-04 - val_accuracy: 1.0000
Epoch 10/100
72/72 [=====] - 10s 144ms/step - loss: 0.0016 - accuracy: 0.9996 - val_loss: 0.0012 - val_accuracy: 1.0000
Epoch 11/100
72/72 [=====] - 10s 143ms/step - loss: 9.5520e-04 - accuracy: 1.0000 - val_loss: 6.2122e-04 - val_accuracy: 1.0000
Epoch 12/100
72/72 [=====] - 12s 161ms/step - loss: 2.6368e-04 - accuracy: 1.0000 - val_loss: 6.6321e-04 - val_accuracy: 1.0000
Epoch 13/100
72/72 [=====] - 11s 148ms/step - loss: 4.1546e-04 - accuracy: 1.0000 - val_loss: 0.0017 - val_accuracy: 0.9983
...
Epoch 99/100
72/72 [=====] - 11s 148ms/step - loss: 1.7360e-05 - accuracy: 1.0000 - val_loss: 0.0011 - val_accuracy: 1.0000
Epoch 100/100
72/72 [=====] - 10s 146ms/step - loss: 9.9935e-06 - accuracy: 1.0000 - val_loss: 7.6968e-04 - val_accuracy: 1.0000
```

epoch 횟수가 증가할수록 손실값이 낮아지며 예측률이 높아진다

# MODEL TEST



```
preds = model.predict(np.array(imgResize, dtype = np.int32).reshape(-1,128,128,3) / 255.0)
print(classes[np.argmax(preds)])
```

✓ 0.0s

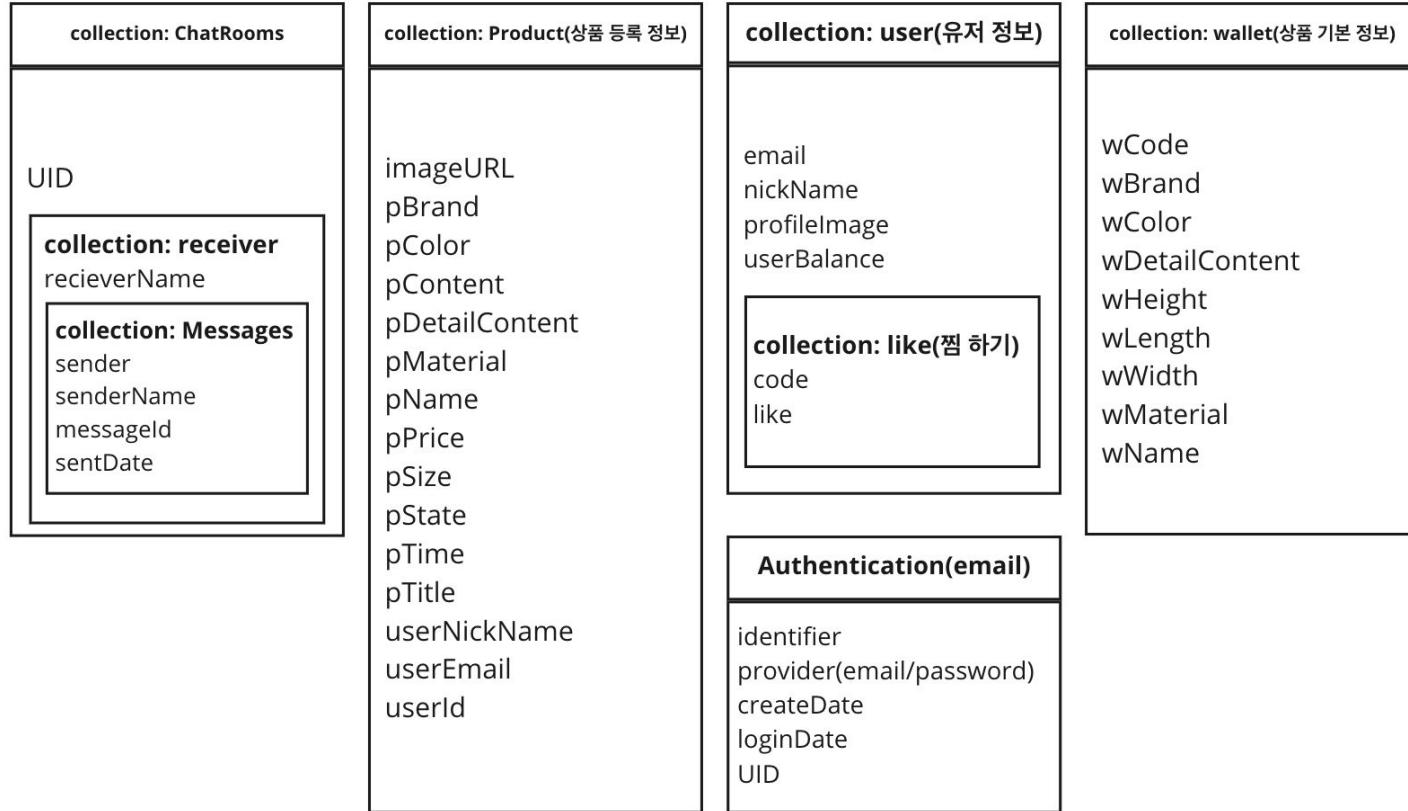
1/1 [=====] - 0s 13ms/step

구찌

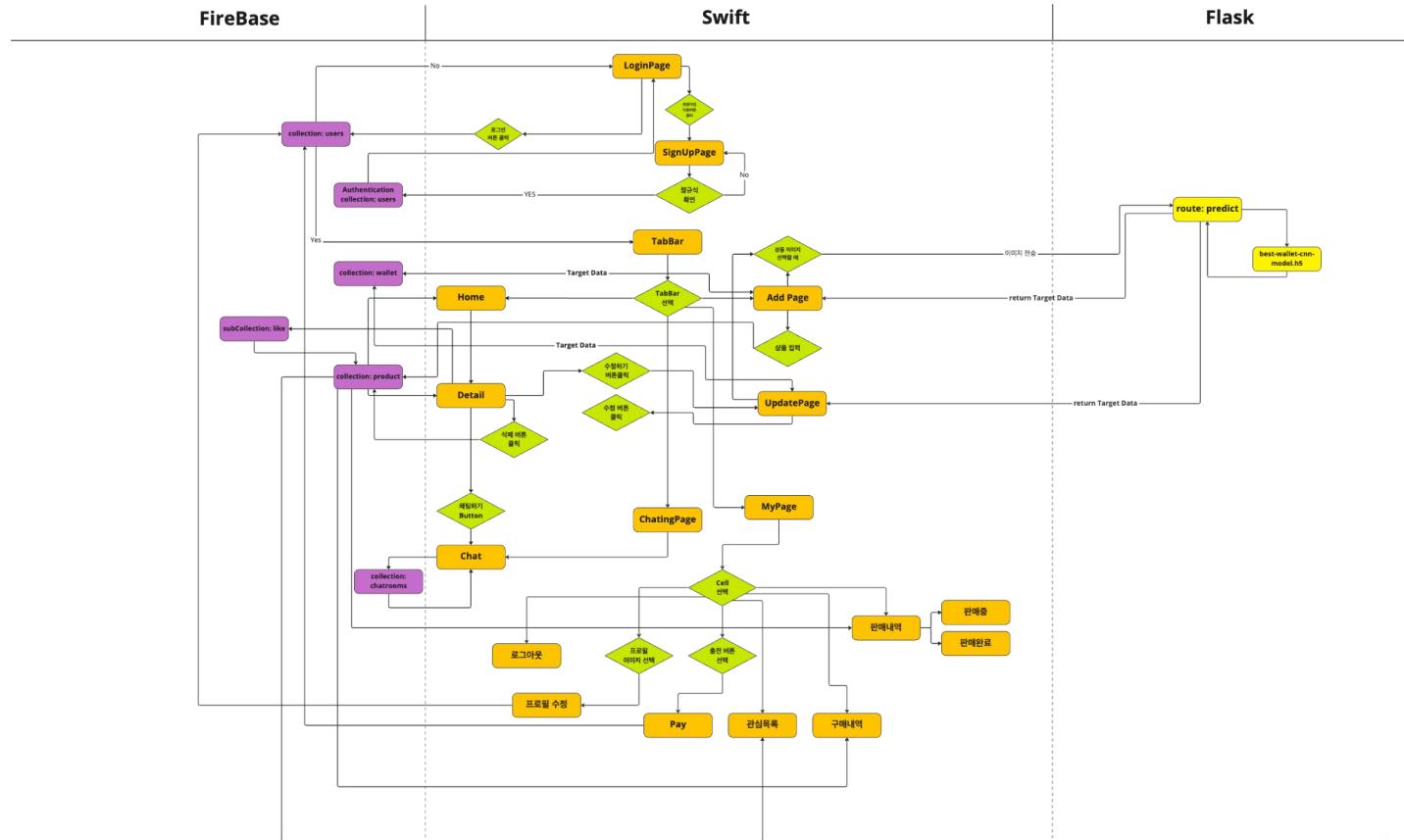


개발

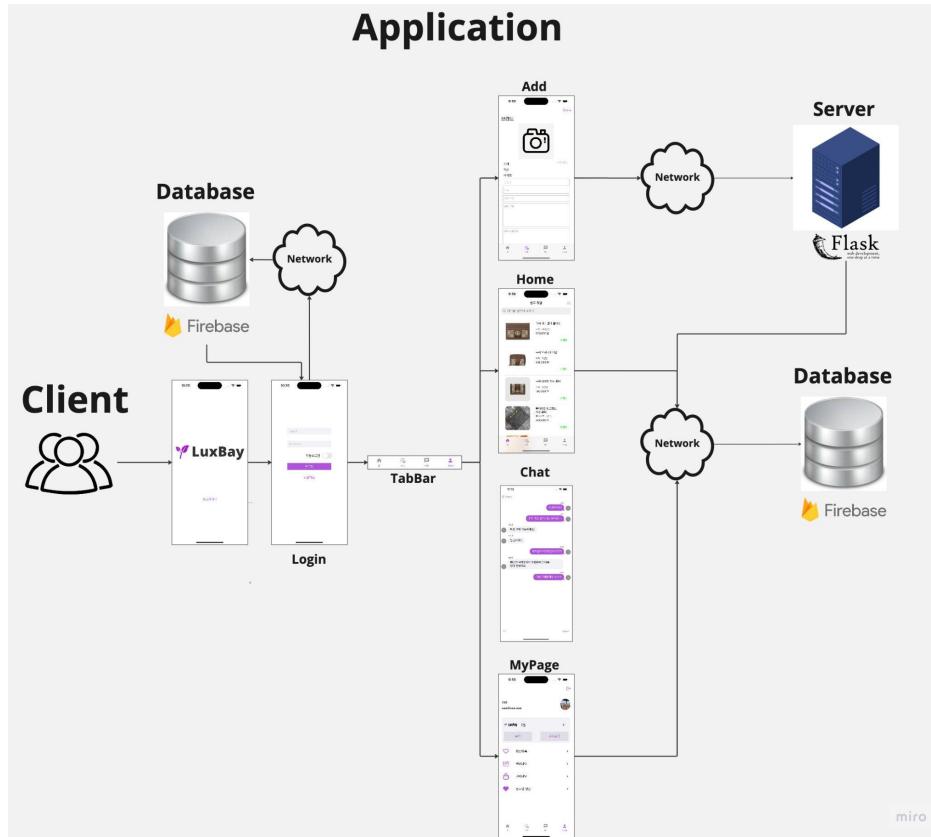
# ERD



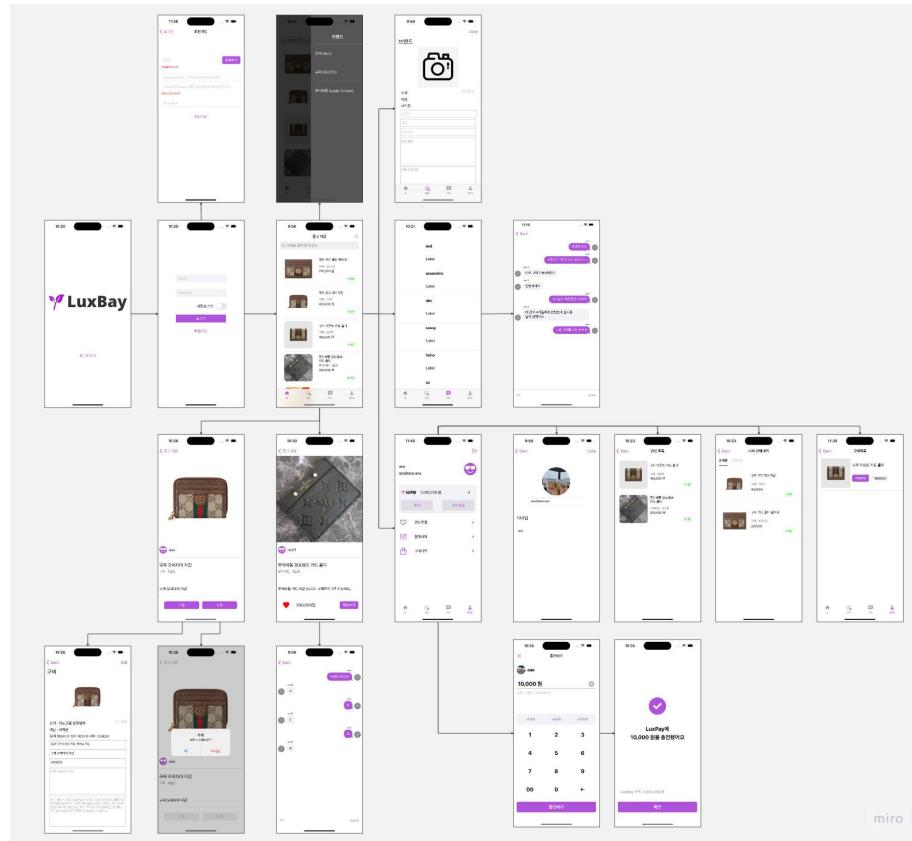
# Process Flow Diagram



# System Flow Diagram



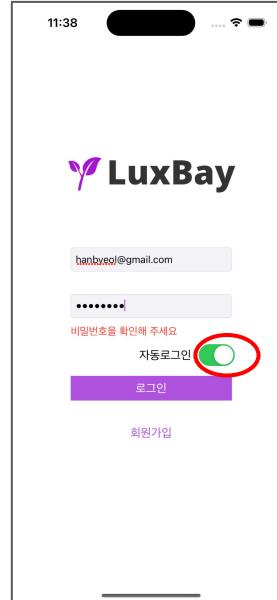
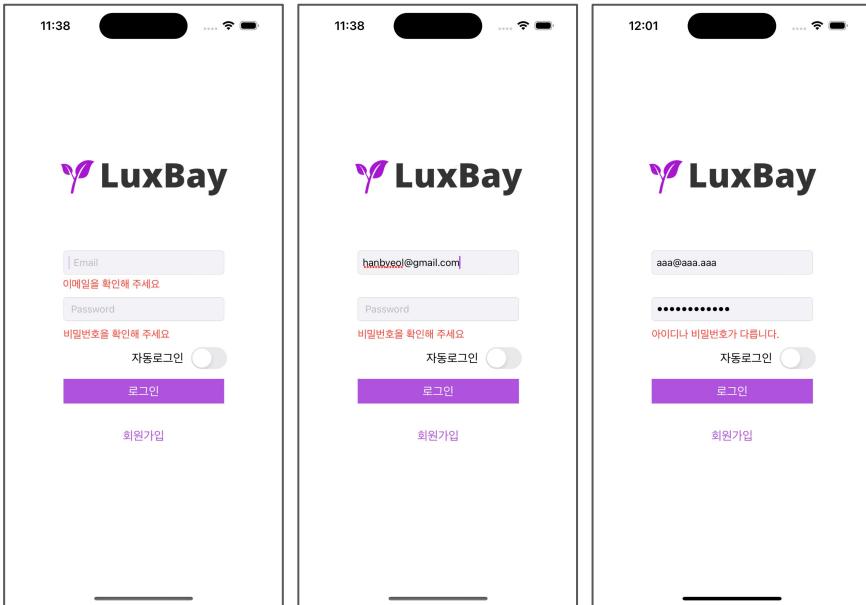
# UI Flow



# 기능 구현 (Login)

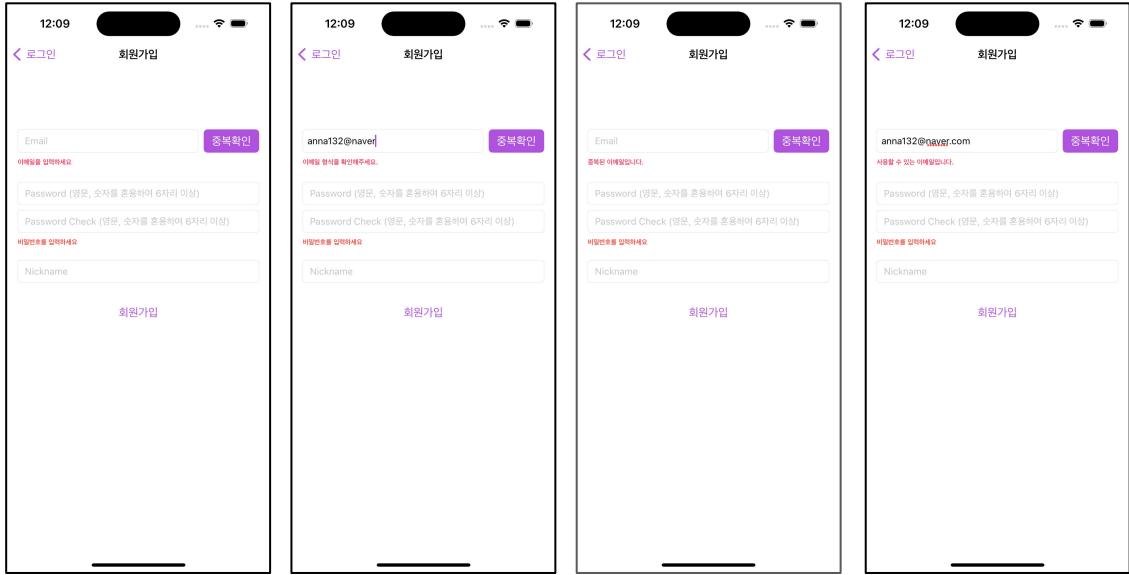
## 정규식

1. 이메일과 비밀번호를 입력하지 않음
2. 이메일 형식이 맞지 않음
3. 비밀번호 형식이 맞지 않음
4. 이메일과 해당 비밀번호가 일치하지 않음



이메일과 비밀번호 모두 입력한 후 **자동로그인** 스위치를 on 하면, 앱을 종료하고 다시 켰을 때 바로 TabBar가 보이는 Main Page로 이동

# 기능 구현 (SignUp)



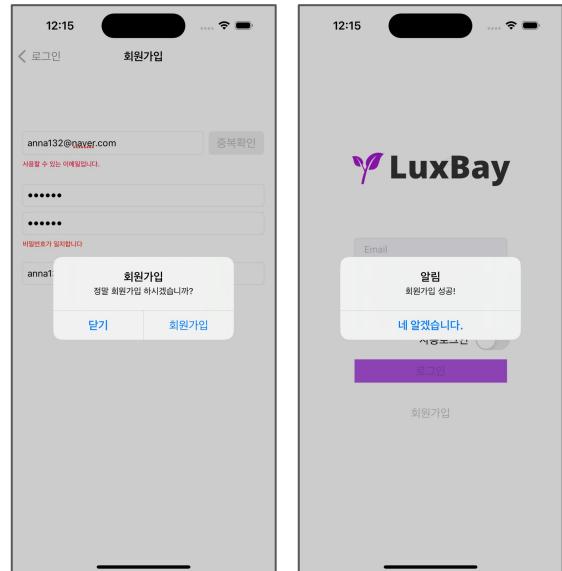
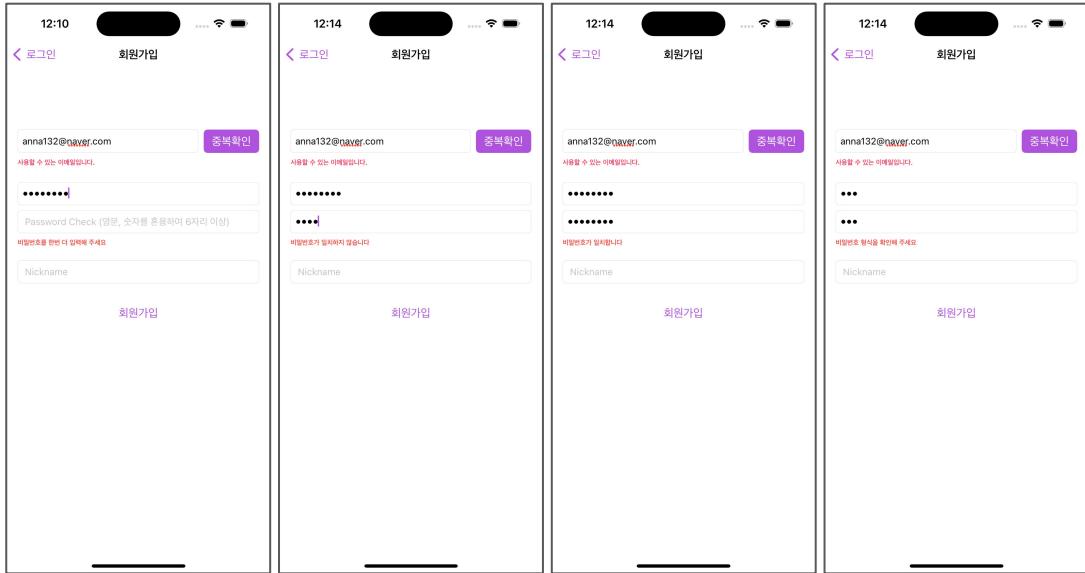
## 정규식

-이메일과 형식 및 이메일 중복 확인

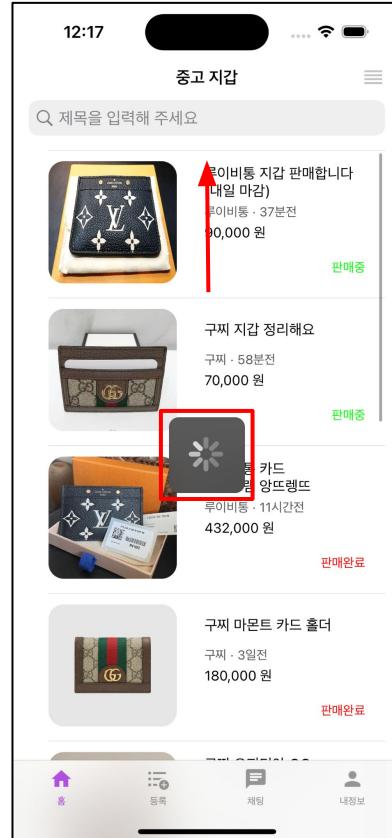
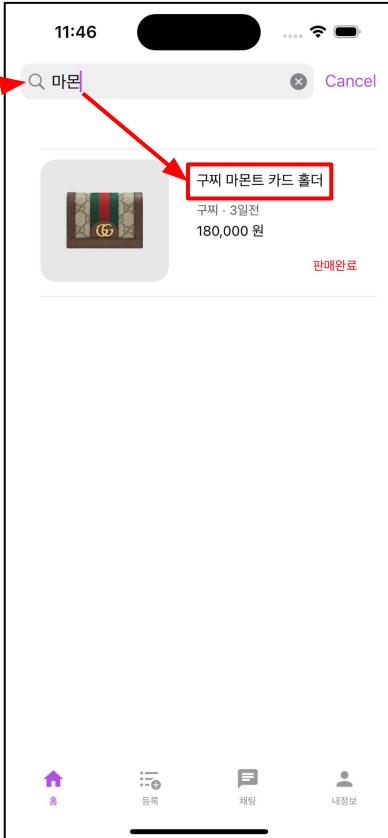
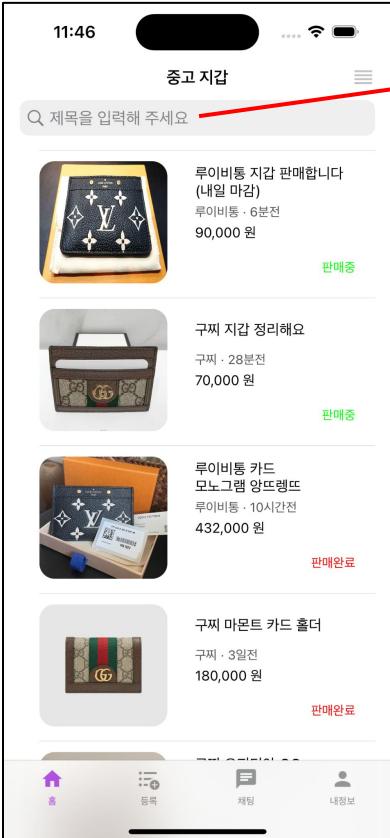
# 기능 구현 (SignUp)

## 정규식

- 비밀번호 형식 및 일치 확인 (비밀번호 형식이 안맞으면 일치해도 회원가입 불가)
- 이메일, 비밀번호, 닉네임 입력 여부 확인

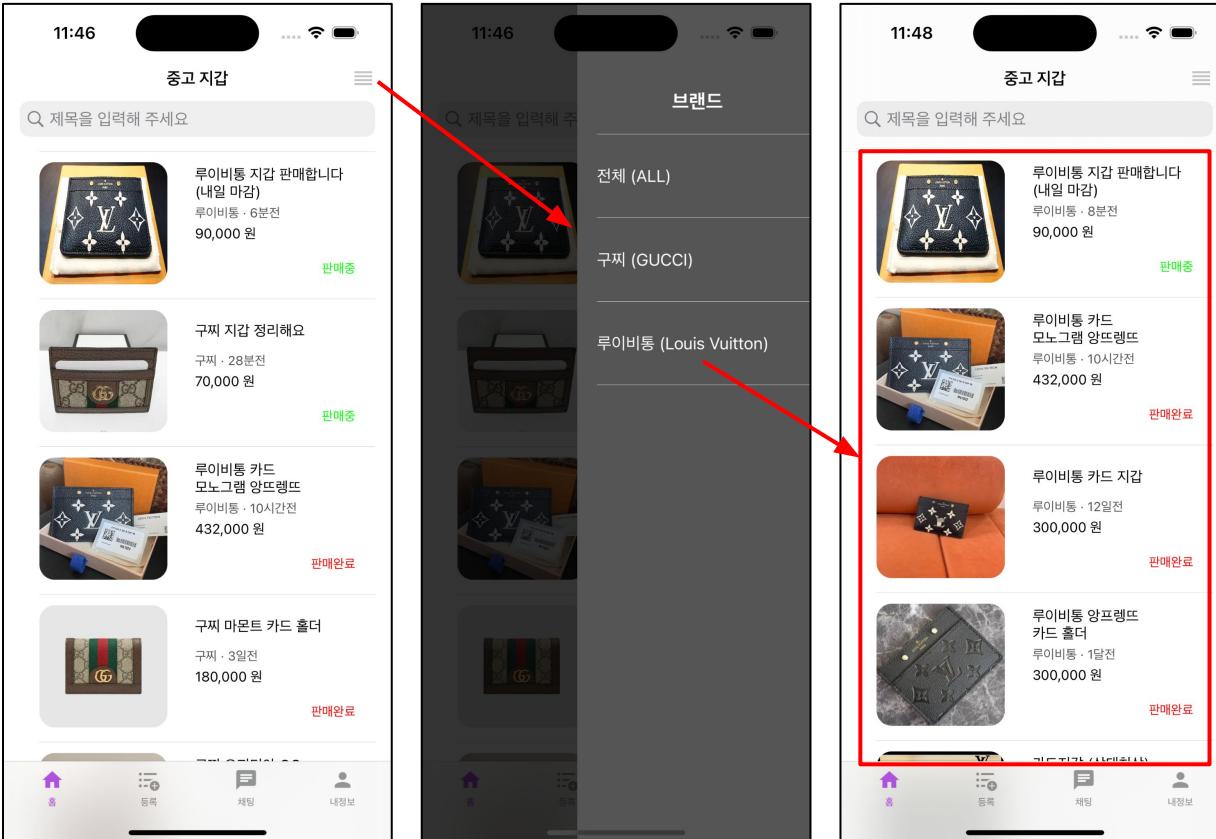


# 기능 구현 (Main)



- TableView 형식으로 사용자들이 입력한 상품목록 표시
- 제목, 카테고리, 시간, 가격, 판매상태를 표시
- 제목을 상단의 SearchBar에서 검색 가능
- 최상단으로 스크롤 시 새로고침

# 기능 구현 (Main)



- NavigationBar의 좌측 버튼 클릭 시 브랜드별 검색 가능
- 전체, 구찌, 루이비통 카테고리로 분류
- 클릭한 카테고리별로 검색 목록 표시(상단 SearchBar 사용시 카테고리 검색 유지)

# 기능 구현 (Detail)

판매 완료된 상품의 디테일 페이지

상품 상세설명

찜하기



내가 올린  
상품 디테일  
페이지  
(  
수정, 삭제 )

# 기능 구현 (Update & Delete)



## 수정하기

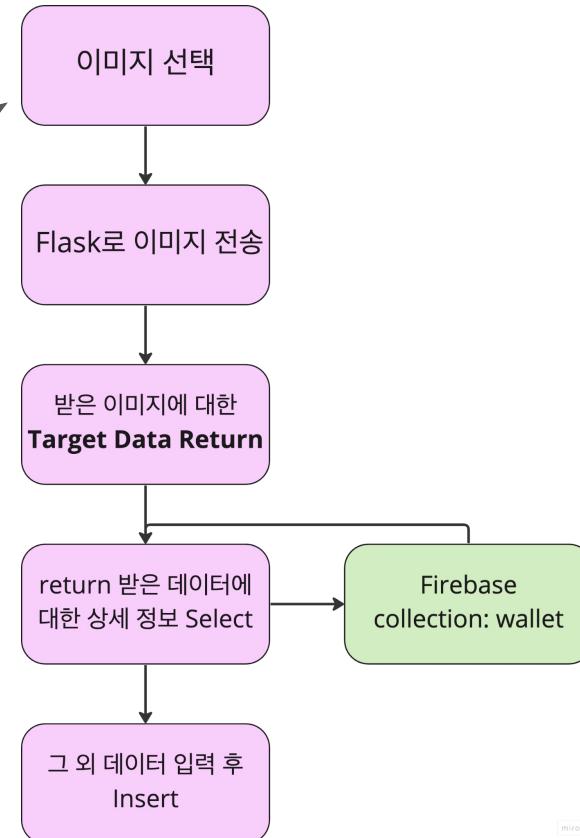
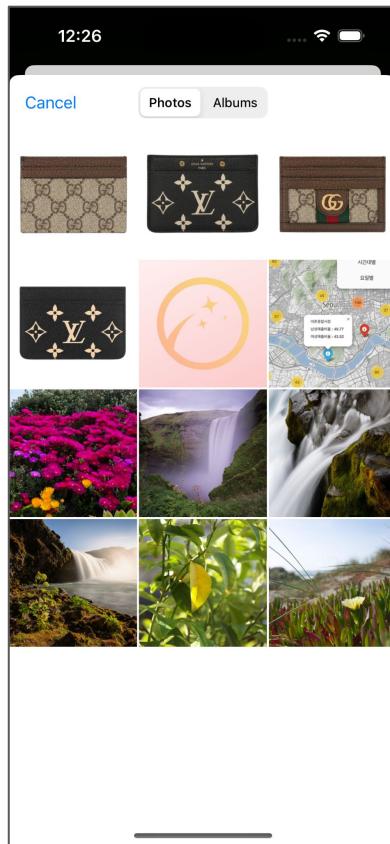
- 상품 수정하기를 선택하면 수정 페이지로 해당 상품의 정보를 넘겨준다.
- 입력 정보에 맞게 Firebase에 Update



## 삭제하기

- 삭제 완료를 하면 Firebase DataBase에서 영구적으로 삭제가 된다.

# 기능 구현 (Add)



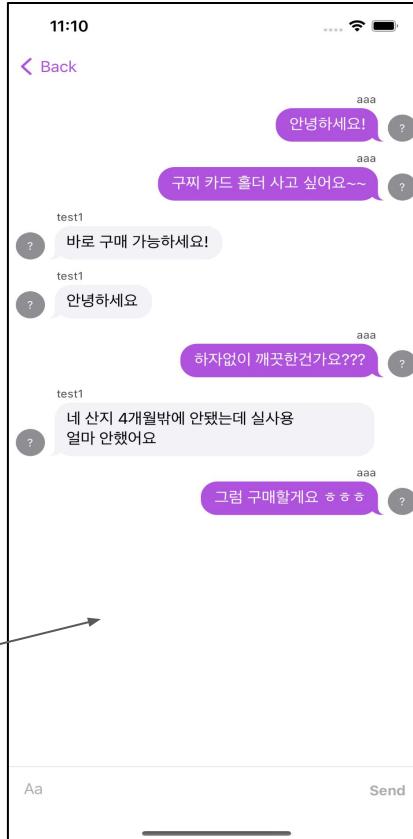
# 기능 구현(Chatting)



DetailPage의 “채팅하기”  
Button Click 시  
Chatting Page로 이동

segue를 통해  
현재 로그인 유저의 UID,  
Nickname 정보와

게시글 작성자의  
UID, Nickname 정보를  
넘겨줌

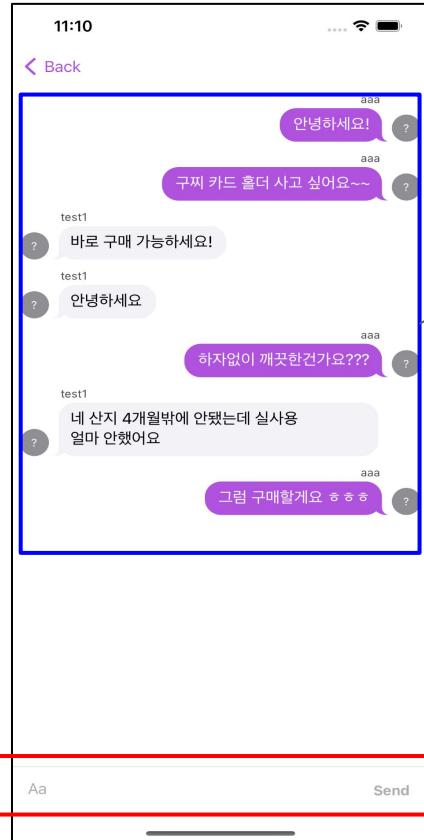


View가 Load될 때

넘겨받은 로그인 유저와  
게시글 작성자의  
정보를 FirebaseStore에서  
검색하여

주고 받은 Message  
정보를 가져옴

# 기능 구현(Chatting)

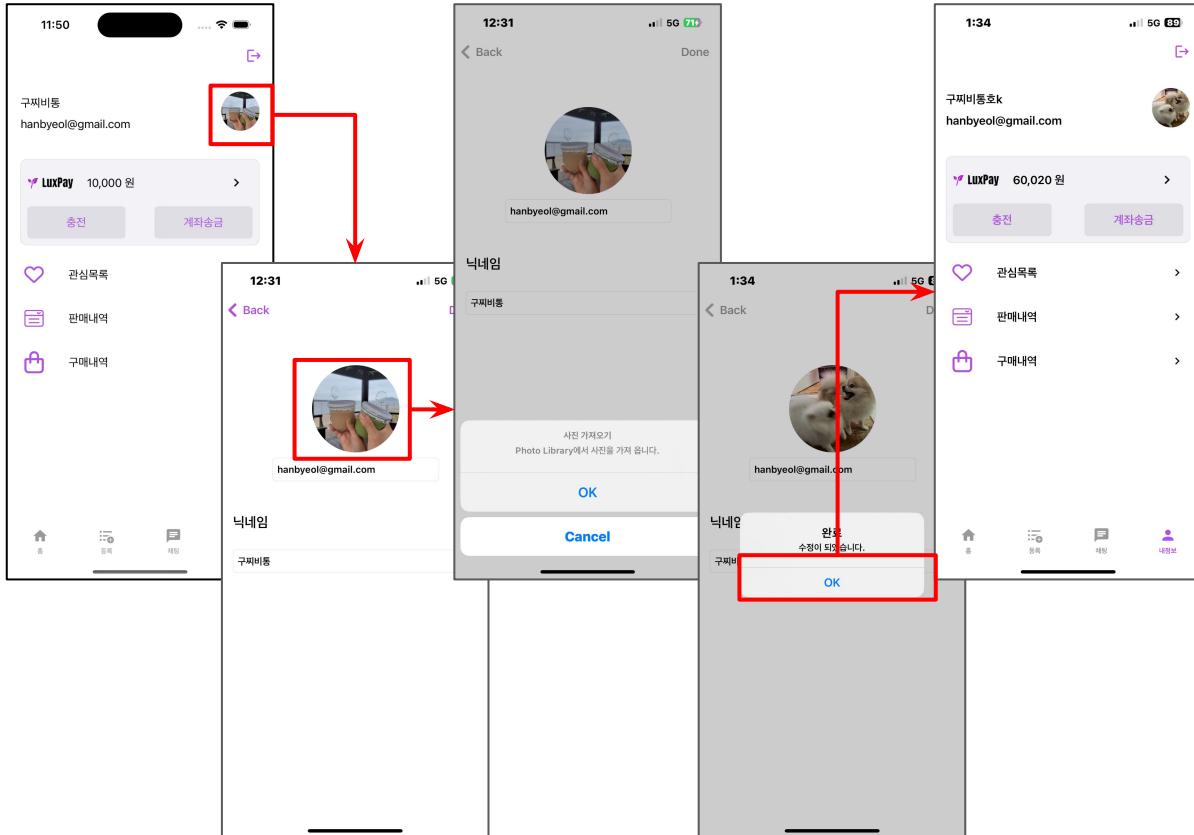


Message[indexPath.section]을  
통해 Sender를 구분하여 Message Bubble 출력

MessageInputBar의 Textfield에  
보낼 메세지를 입력하고 Send Button 클릭 시,

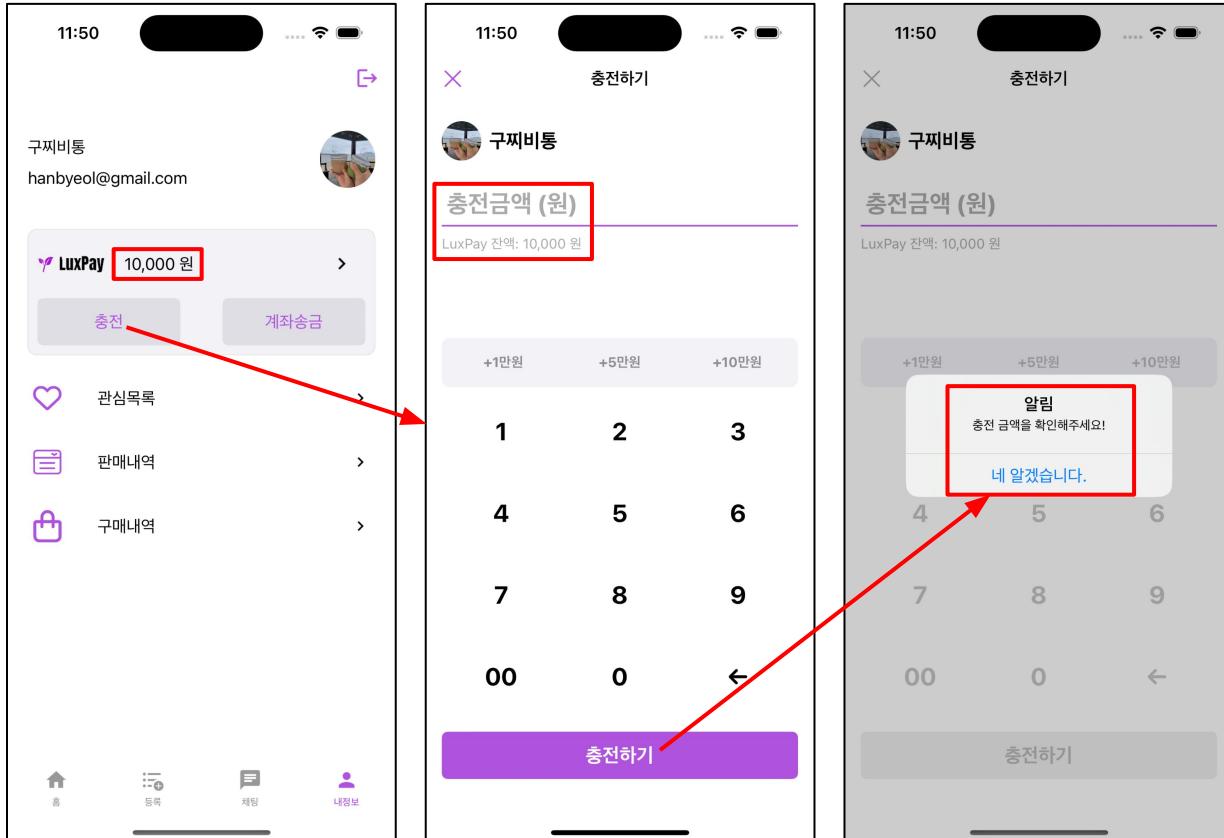
FirebaseStore Database의 chatrooms Collection 내  
other Collection 내 Messages Collection 안에  
Document 생성

# 기능 구현 (MyPage - Profile)



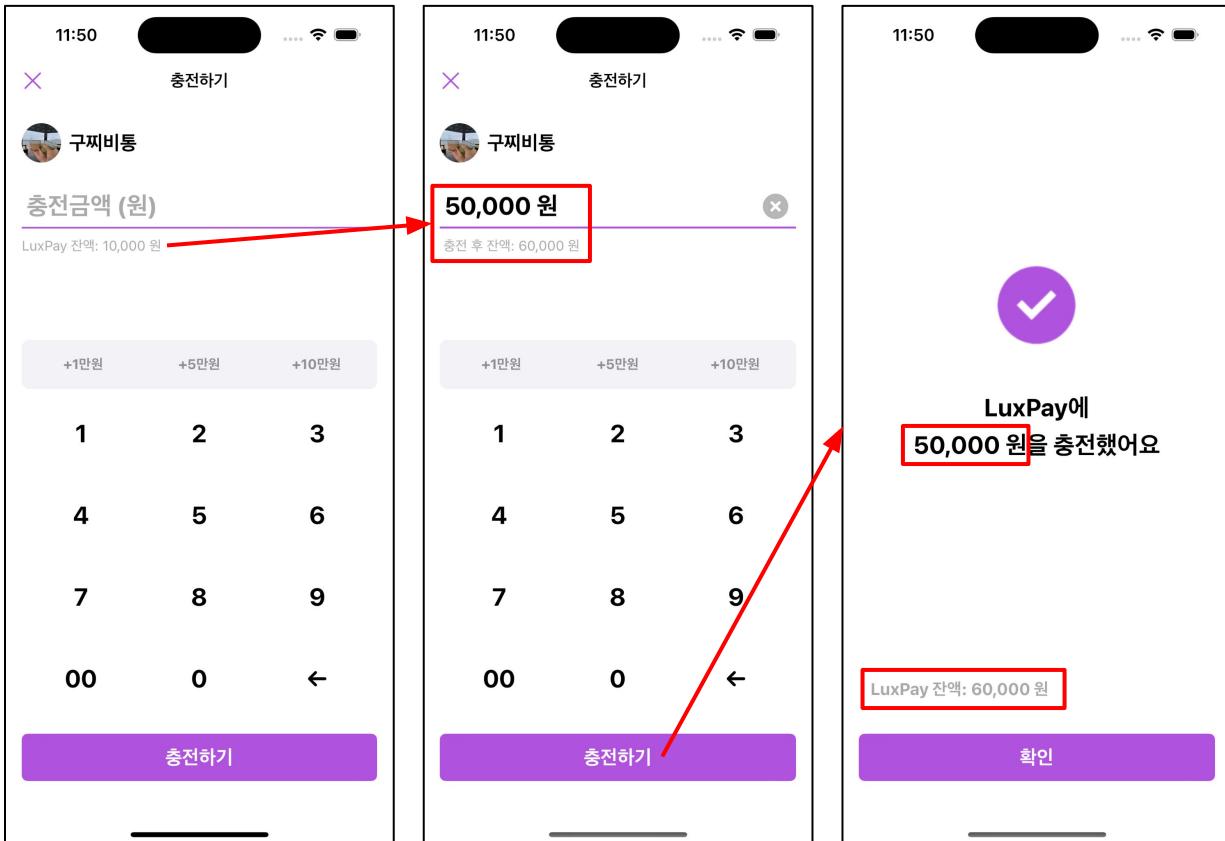
- 마이페이지에서 프로필 사진 클릭시 프로필 변경 페이지로 이동
- 프로필 변경 페이지에서 이미지 클릭시 갤러리와 연결
- 이메일은 변경이 불가하며, 프로필 이미지와 닉네임만 변경 가능

# 기능 구현 (MyPage - Pay)



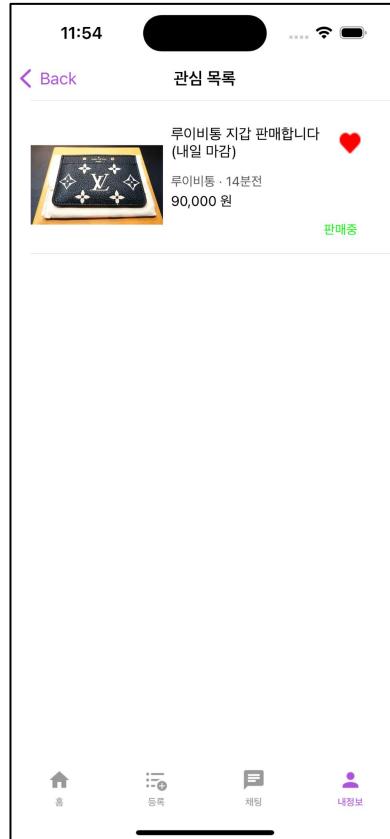
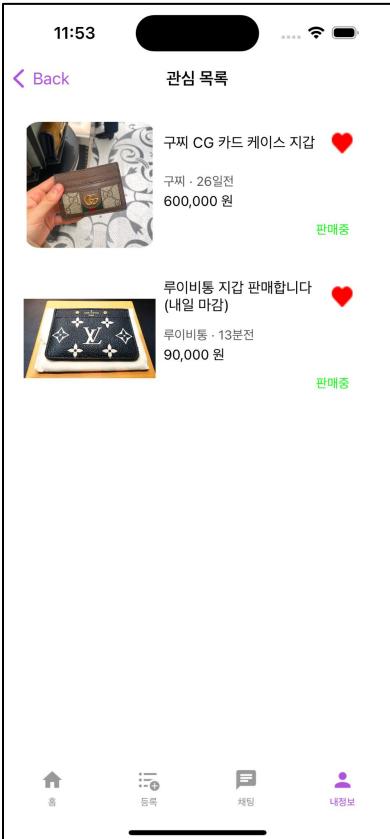
- 마이페이지에서 현재 가지고 있는 LuxPay잔액 표시
- 충전버튼 클릭 시 충전페이지로 이동
- 충전금액을 입력하지 않거나 0원일 경우 Alert로 경고 표시

# 기능 구현 (MyPage - Pay)



- 충전금액을 입력하면 소수점 3자리마다 ","와 뒤에 원(₩) 표시
- 충전금액을 입력 시 현재 잔액에서 더한 값을 보여줌
- 충전하기 버튼 클릭 시 충전완료페이지로 이동 및 사용자의 충전금액 표시

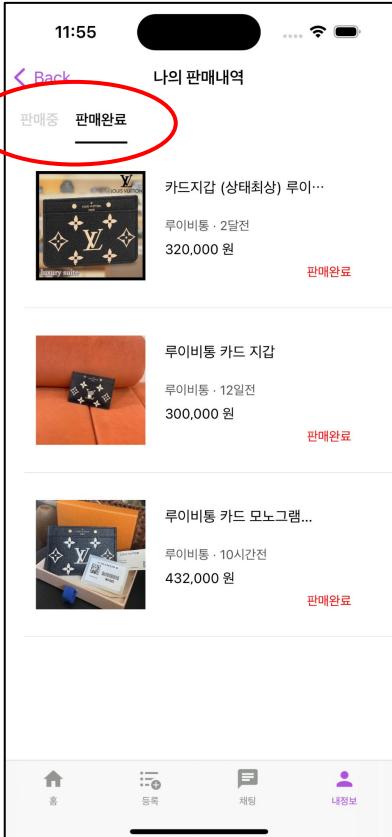
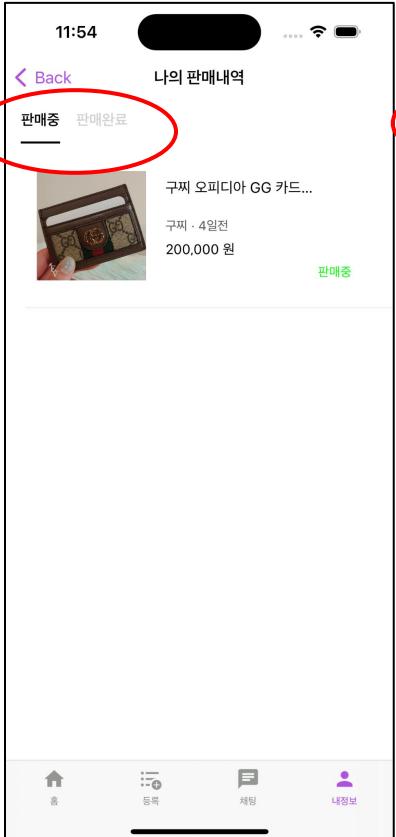
# 기능 구현 (MyPage - LikeList)



## 관심목록 리스트

- 내가 찜한 관심목록 리스트만 출력
- 해당 제품 누르면 디테일 페이지로 넘어간 후 하트 다시 누르면 하트 unclick

# 기능 구현 (MyPage - SaleList)



- 내가 올린 판매내역 리스트
- TabBar 사용하여 판매중인 상품과 판매 완료 상품을 각각 다른 페이지에 따로 분리하여 출력
- cell 누르면 디테일 페이지로 넘어감

# 향후 계획

## [chatting]

- 채팅 목록 TableView로 구성 후 채팅
- 채팅방 내에서 구매완료 처리
- 유저 프로필 이미지 띄우기
- 이미지 주고 받기
- Real-Time Database로 실시간 채팅 진행

## [Pay]

- 계좌연동 가능시 계좌 송금 및 충전 연동

## [MyPage - PurchaseList]

- 로그인시 이메일과 동일한 이메일이 들어간 상품 가져오기
- 관심목록: 디테일 페이지가 아닌 관심목록 페이지에서도 짐 unclick 가능하게 하기

감사합니다.