TEAM

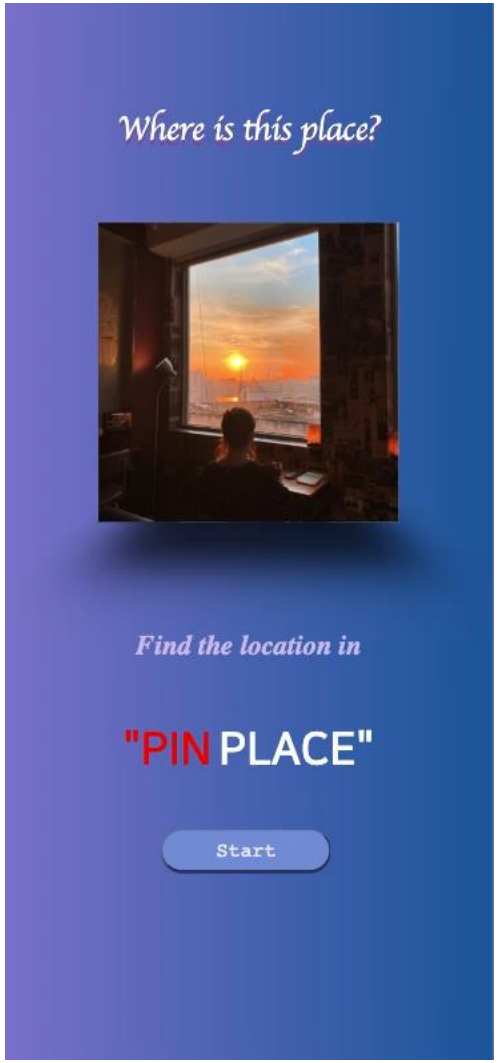
THE OUTSIDERS

“Capstone Project_Final Presentation”

Agenda

00. Objective
01. Project Progress
02. Final Design
03. Final Implementation
04. Challenges
05. Limitations
06. Demonstration

00. Objective

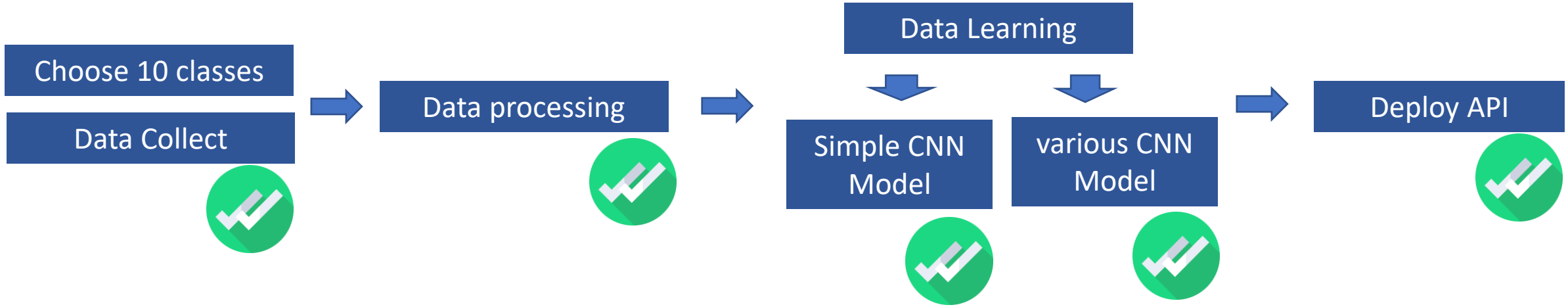


“CNN based place recognition web app”

1. Service of place recognition feature & SNS feature.
2. Collect data set & Build CNN models which have the best accuracy
3. Work on UI design & graphic Design
4. Apply CNN models on web app

01. Project Progress > CNN Part

#	Contents		1주차	2주차	3주차	4주차	5주차	6주차	7주차	8주차	9주차	10주차	11주차
			~9/26	~10/3	~10/10	~10/17	~10/24	~10/31	~11/7	~11/14	~11/21	~11/28	~12/4
1	Collect dataset(crawling)												
2	Build simple CNN model												
3	Various CNN models												
4	Improve accuracy	Data augmentaion											
		Select model(Resnet50)											
		Using confusion matrix											
5	Integration & Code Review												
6	QA Testing & Prepare Final Presentation												



01. Project Progress > Frontend Part

#	Contents		1주차	2주차	3주차	4주차	5주차	6주차	7주차	8주차	9주차	10주차	11주차
			~9/26	~10/3	~10/10	~10/17	~10/24	~10/31	~11/7	~11/14	~11/21	~11/28	~12/4
1	Initial Design(with figma)												
2	Final Design (included overall graphic design)												
3	Making web structure												
4	Implementation (10 Pages)	Cover Page											
		Start page											
		Guide Page											
		SignUp Page											
		Login Page											
		Find Location Page											
		Listup Page											
		Upload Picture Page											
		SNS Page											
		My Page											
5	Integration & Code Review												
6	QA Testing & Prepare Final Presentation												

All Completed ☺

Role Distribution

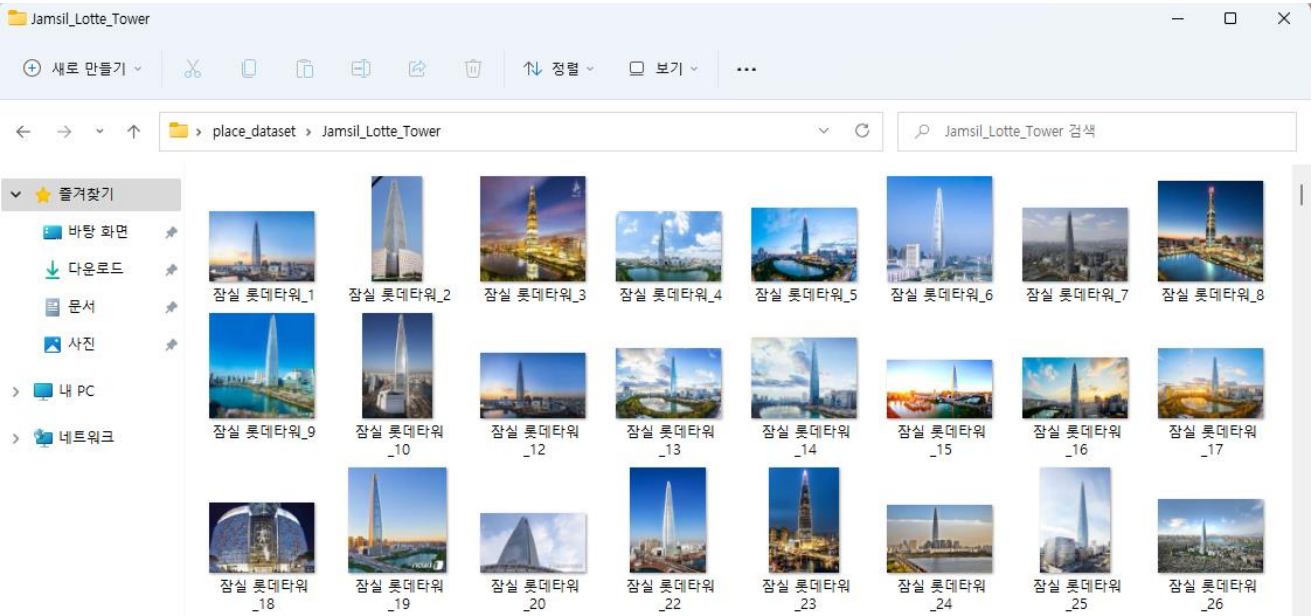
Person in charge	Work
Jung chae won	Making UI and web programing in main pages, cnn implementation pages and posting pages
Lee jisup	Web programing in hot place list pages and pages about collecting images from users
Chae seung yun	Image pretreatment, building cnn models and test them and comparing performance
Hong sung jun	Collecting images from google and Instagram by crawling and then searching cnn models and helping cnn building
Uhm ji yong	Backend such as establishing db and server, connecting cnn models into web app

02. Final Design > CNN Part

Choose 10 classes

- Dongdaemun_Design_Plaza
- Gyeongui_Line_Forest_Park
- Haebangchon
- Han_River_Sebitseom
- Ikseon_Dong_Hanok_Village
- Jamsil_Lotte_Tower
- Myeongdong_Cathedral
- Naksan_Park
- Namsan_Seoul_Tower
- The_Hyundai_Seoul_Mall

10 classses



Choose 10 hot places in Seoul where MZ generation likes

Various CNN models

"LeNet-5"

➢ Model spec

- convolution layer : 3
- sub-sampling layer(pooling layer) : 2
- fully-connected layer : 1
- Parameter : 60,000

"Alexnet"

➢ Model spec

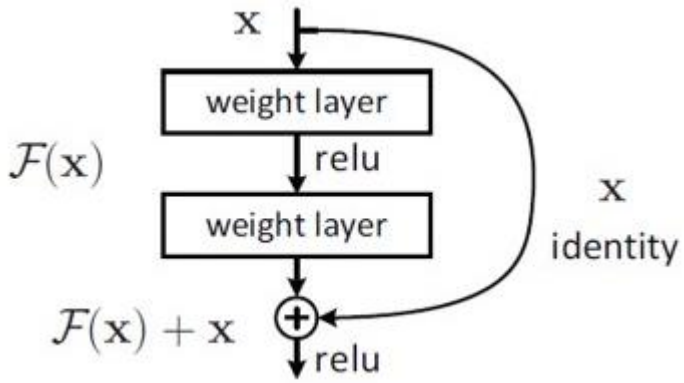
- convolution layer : 5
- sub-sampling layer(pooling layer) : 3
- Local Response Normalization layer : 2
- fully-connected layer : 3
- Parameter : 62,000,000

"VGGNet"

➢ Model spec

- convolution layer : 13
- sub-sampling layer(pooling layer) : 5
- fully-connected layer : 3
- Parameter : 1,380,00,000

"Resnet"



02. Final Design > CNN Part

Data Collect

main PINPLACE / crawling.py / <> Jump to

orioncsy Update crawling.py

1 contributor

2 lines (53 sloc) | 2 KB

```
#selenium을 이용하여 구글에서 이미지를 크롤링하는 코드
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
import time
ort os
rt urllib.request

def createFolder(directory):
    try:
        if not os.path.exists(directory):
            os.makedirs(directory)
SError:
nt ('Error: Creating directory. ' +  directory)
렬 파크'

r('C:/Users/user/Desktop/images/'+keyword+'_img_download')
r = 'C:/Users/user/Downloads/chromedriver.exe'
driver.Chrome(chromedriver)
icitly_wait(3)
```

Library

Selenium

Platform

jupyter

```
data_aug_gen = ImageDataGenerator(rescale=1./255,
                                  rotation_range=15,
                                  width_shift_range=0.1,
                                  height_shift_range=0.1,
                                  shear_range=0.5,
                                  zoom_range=[0.8, 2.0],
                                  horizontal_flip=True,
                                  vertical_flip=False,
                                  fill_mode='nearest')
```

- Collect image data by crawling. Using Python, selenium library.

  Collect 1000 images for each class

- Remove irrelevant images and duplicate data.

  After image processing, about 600 images remain for each class.

- Use data augmentation

Final CNN model design

➢ Model spec

- ResNet50 model is adopted.
- Total image data : 25,450
- Training & validation data: 17,815
- Input Size : 128 * 128
- Train set, Validation set, Test set : 5:2:3
- Classes : 10
- Batch size : 32 epoch : 80
- Optimizer : Nadam

"Our selected model"

conv5_block3_2_conv (Conv2D)	(None, 4, 4, 512)	2359296	conv5_block3_2_pad[0][0]
conv5_block3_2_bn (BatchNormali	(None, 4, 4, 512)	2048	conv5_block3_2_conv[0][0]
conv5_block3_2_relu (Activation	(None, 4, 4, 512)	0	conv5_block3_2_bn[0][0]
conv5_block3_3_conv (Conv2D)	(None, 4, 4, 2048)	1050624	conv5_block3_2_relu[0][0]
conv5_block3_out (Add)	(None, 4, 4, 2048)	0	conv5_block2_out[0][0] conv5_block3_3_conv[0][0]
post_bn (BatchNormalization)	(None, 4, 4, 2048)	8192	conv5_block3_out[0][0]
post_relu (Activation)	(None, 4, 4, 2048)	0	post_bn[0][0]
avg_pool (GlobalAveragePooling2	(None, 2048)	0	post_relu[0][0]
predictions (Dense)	(None, 10)	20490	avg_pool[0][0]

Total params: 23,585,290
Trainable params: 23,539,850
Non-trainable params: 45,440

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

```
history = model.fit(X_train, y_train, batch_size=32, epochs=80, validation_split=0.2)
```

```
model = ResNet50V2(include_top=True, weights=None, input_shape=(128,128,3), classes=10)
model.compile(loss='categorical_crossentropy', optimizer='Nadam', metrics=['accuracy'])
```

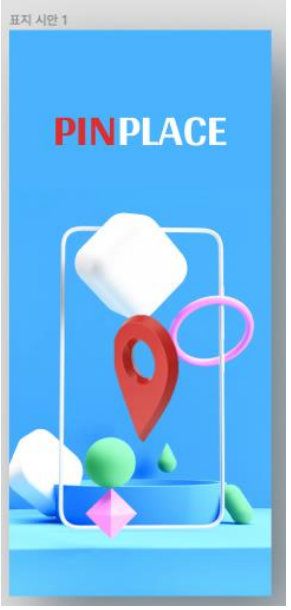
02. Final Design > Frontend Part

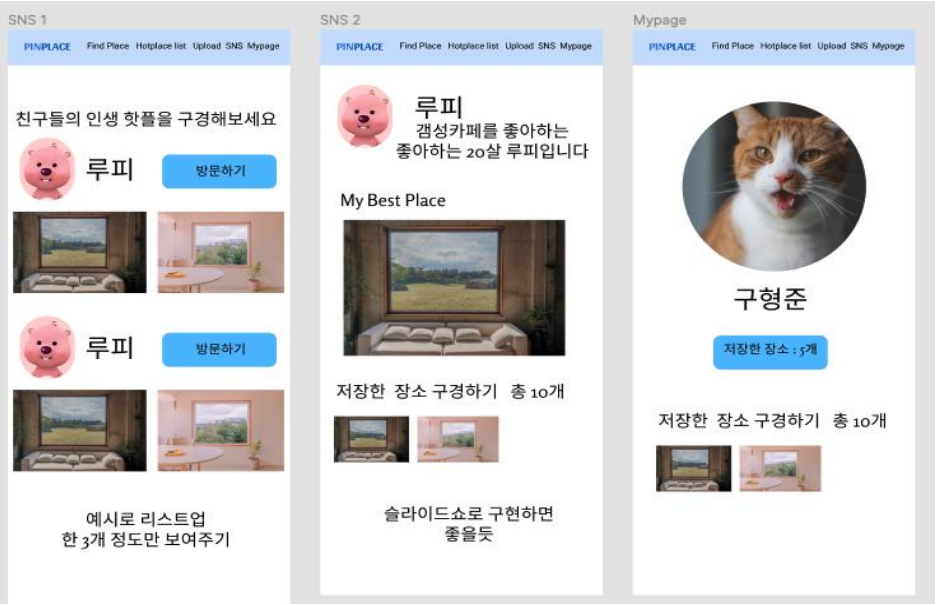
Front end

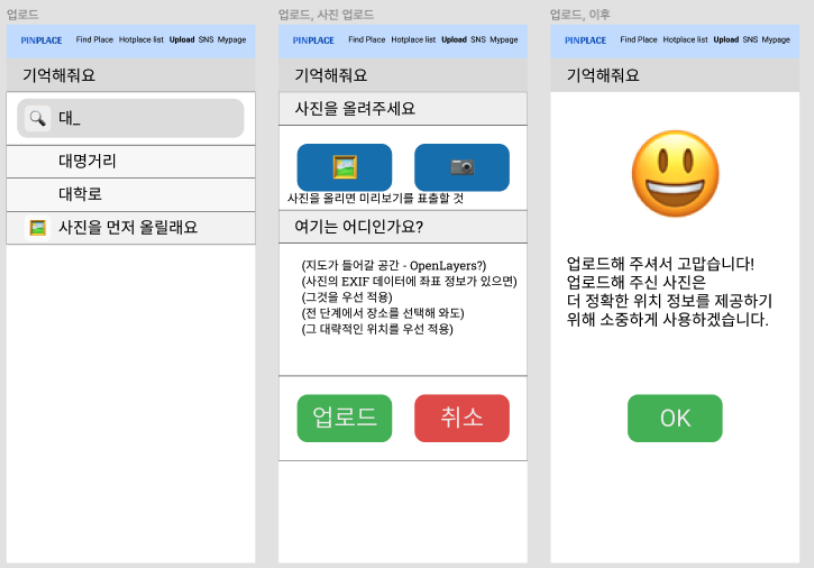
UI Design

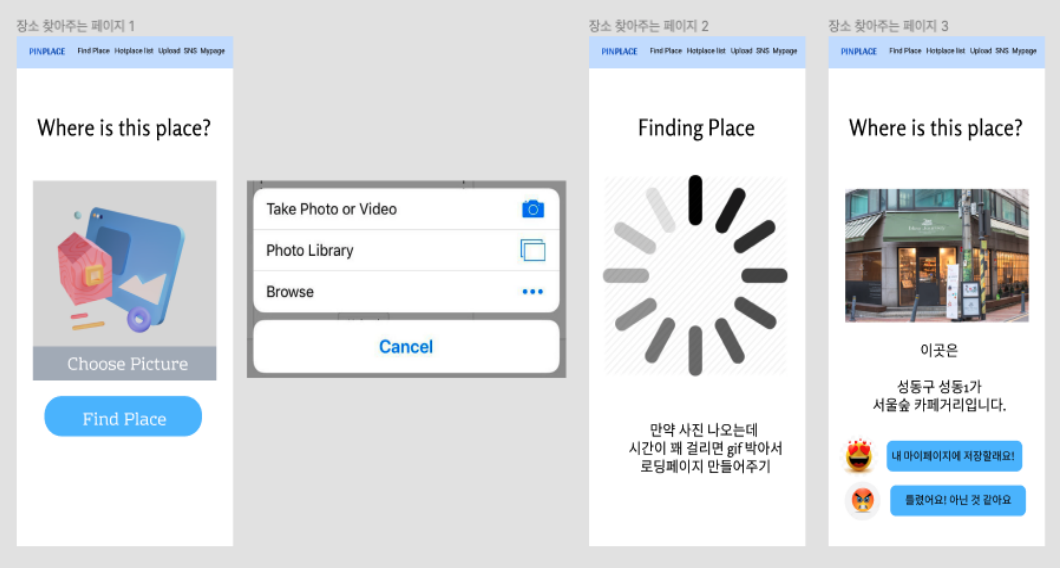
- **The initial design Was made using Figma**

10/10 Completed









02. Final Design > Frontend Part

10 Pages

- Cover Page — 10/17 Completed
- Start page — 10/24 Completed
- Guide Page — 11/2 Completed
- Find Location Page — 10/31 Completed
- List Up Page — 11/2 Completed
- Upload picture Page — 10/31 Completed
- SNS Page — 11/22 Completed
- My Page — 11/16 completed
- Sign Up Page — 11/27 completed
- Login Page — 11/27 completed

All Completed

How to develop?

"Responsive Web based Application"

-Programming Language
: HTML,CSS, Javascript

-DEMO UI (The most optimal size)
: Iphone X (375 * 812)

03. Final Implementation > CNN Part

The result of various CNN models

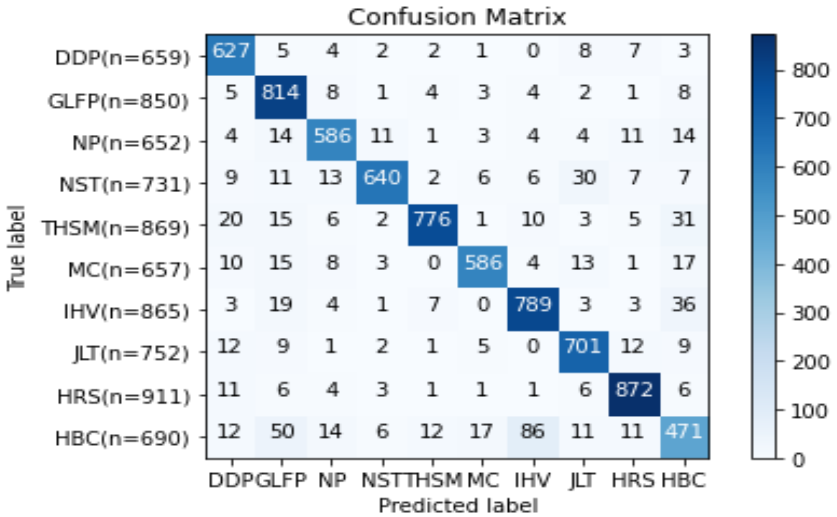
Model	Lenet-5	AlexNet	VGG16	ResNet50
Accuracy	66.3%	13.31%	12.79%	91.12%

➢ Select model – ResNet 50

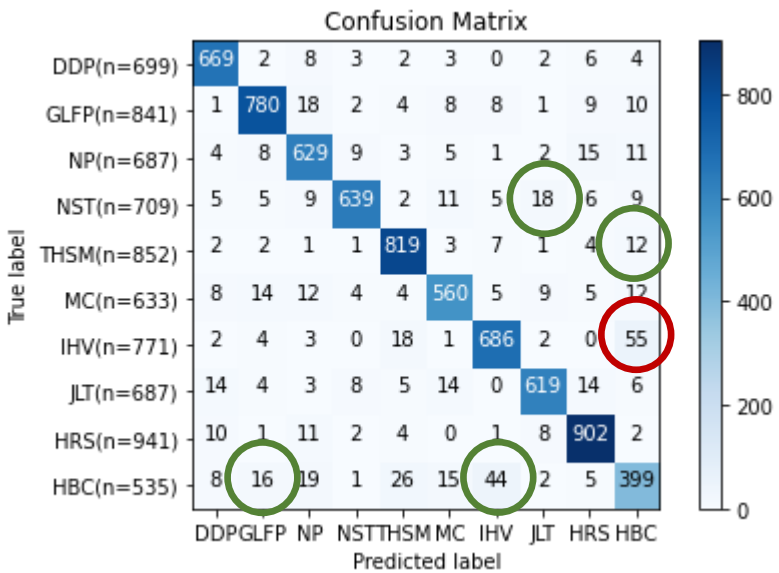
- **AlexNet and VGG16 models have remarkably low accuracy.**
- **Although dataset is not enough to run them, they have so deeper layers and so many parameters to learn.**
- **However, ResNet50 model has less parameters than other models and this model solve the gradient vanishing problem by skip connection.**
- **In this reason, this can have 50 layers which is deeper than other models and we decide to adopt this ResNet model.**

03. Final Implementation > CNN Part

Evaluation – confusion matrix



BEFORE



AFTER

- DDP : Dongdaemun Design Plaza
- GLFP : Gyeongui Line Forest Park
- NP : Naksan Park
- NST : Namsan Seoul Tower
- THSM : The Hyundai Seoul Mall
- MC : Myeongdong Cathedral
- IHV : Ikseon Dong Hanok Village
- JLT : Jamsil Lotte Tower
- HRS : Han River Sebitseom
- HBC : Haebangchon

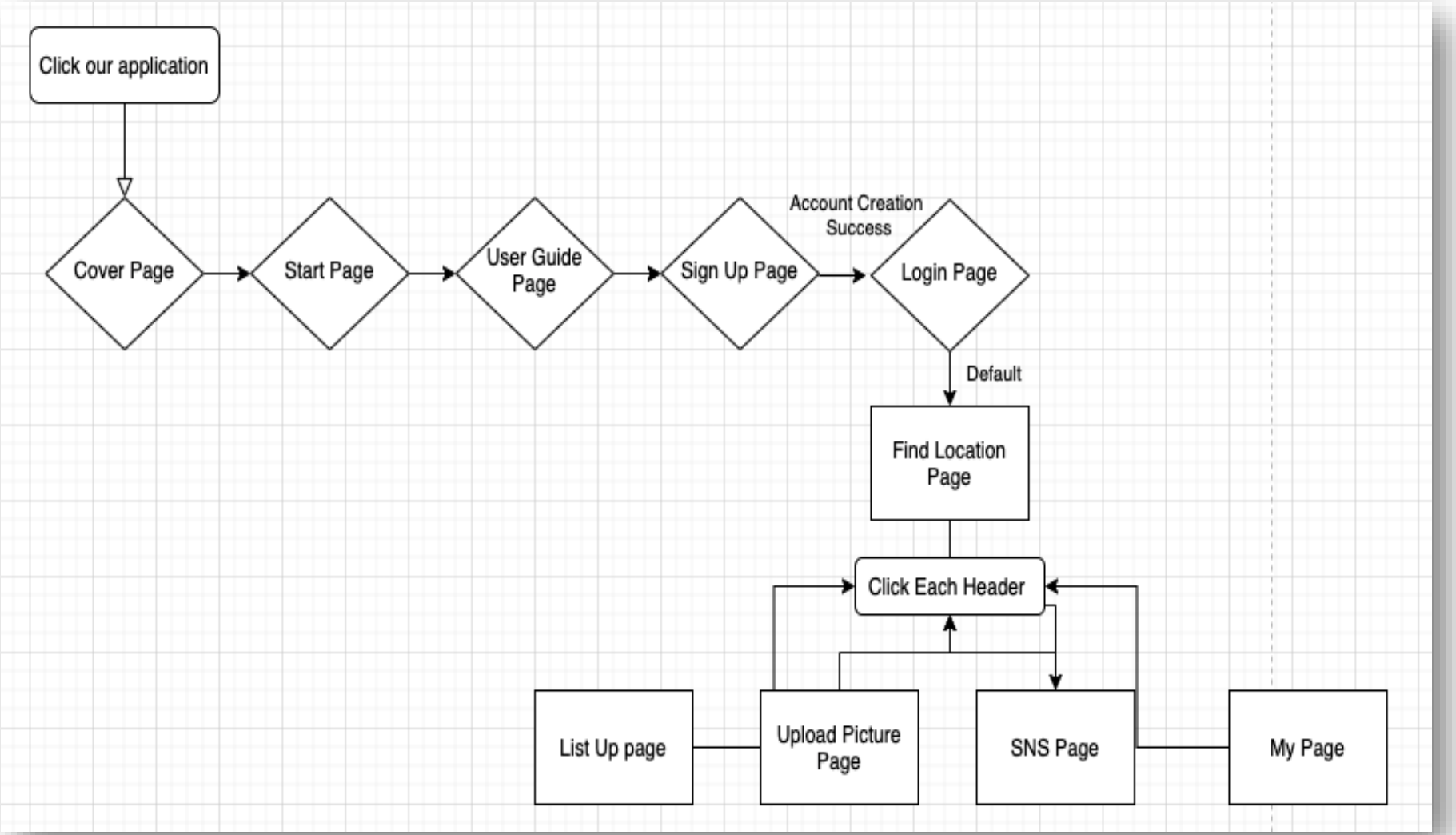
➢ Improved performance

- We remove ambiguous images
- Then, we can get 91% accuracy as same condition
- Also, we get improved result in confusion matrix

```
#모델 정확도 출력
print("정확도 : %.4f" % (model.evaluate(X_test, y_test)[1]))
```

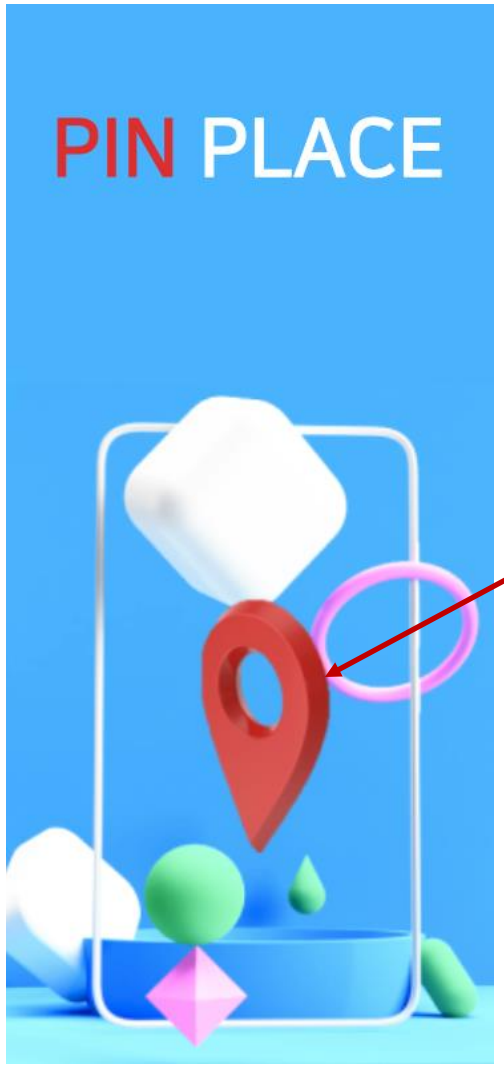
```
230/230 [==============================] - 7s 31ms/step - loss: 0.4801 - accuracy: 0.9112
정확도 : 0.9112
```

- ## User Flow


Click our application
Cover Page
Start Page
User Guide Page
Sign Up Page
Account Creation Success
Login Page
Default
Find Location Page
Click Each Header
List Up page
Upload Picture Page
SNS Page
My Page

03. Final Implementation > Frontend Part

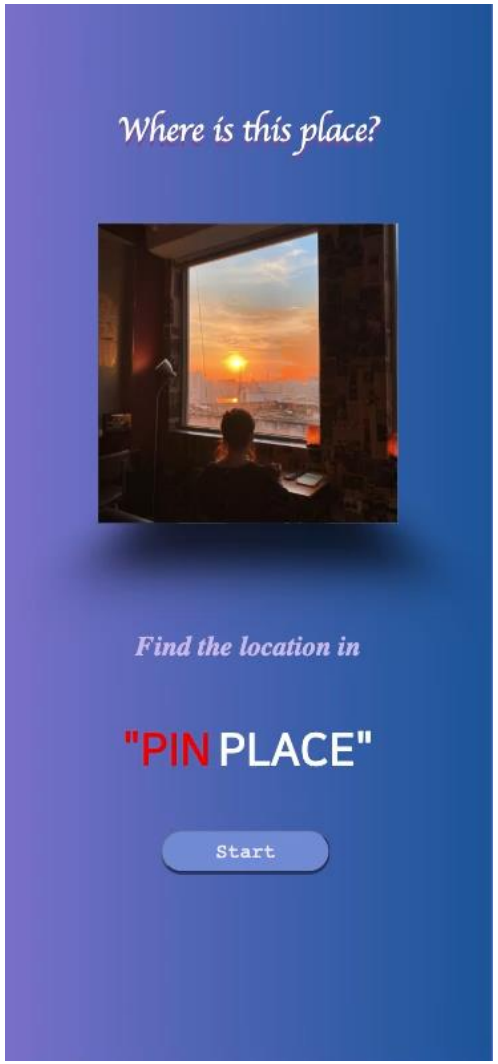
▪ Cover Page



➢ Instruction

- Found 3d graphic asset and placed it myself.
- Top -> ”PIN PLACE”
  : Our service’s Logo
- Location Pin 3d Graphic
  : included in ”Onclick Function” that could move next page
  (onclick="location.href= ' ./start.html’”)

▪ Start Page



➢ Instruction

- Top Layer -> 3D Rotated Cube (with diverse place’s pictures)
- Implemented by setting x,y, z- axis angles with css
- Bottom Layer
  -> Start Button (with onclick function)

03. Final Implementation > Frontend Part

▪ Guide Page

➢ Instruction

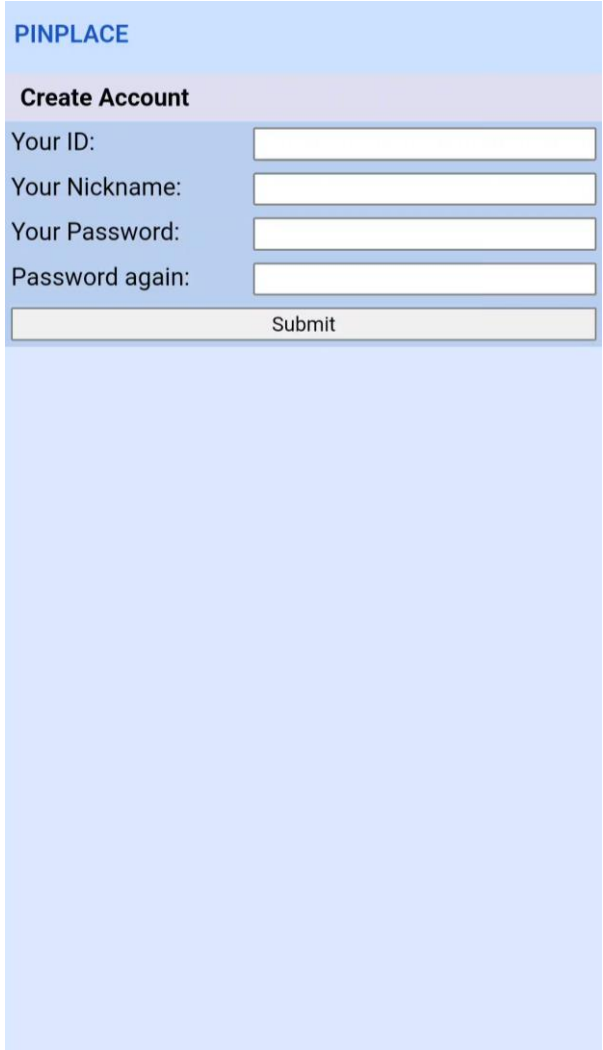
- For optimal UX, We made User Guide Page with card UI
- Every time user turn the page, the content and design are designed to be different
- Implement 'PREV, NEXT, FINISH'

-PREV : onclick function that move previous card

-NEXT : onclick function that move next card

-Start : onclick function that connect login page

▪ Sign Up Page

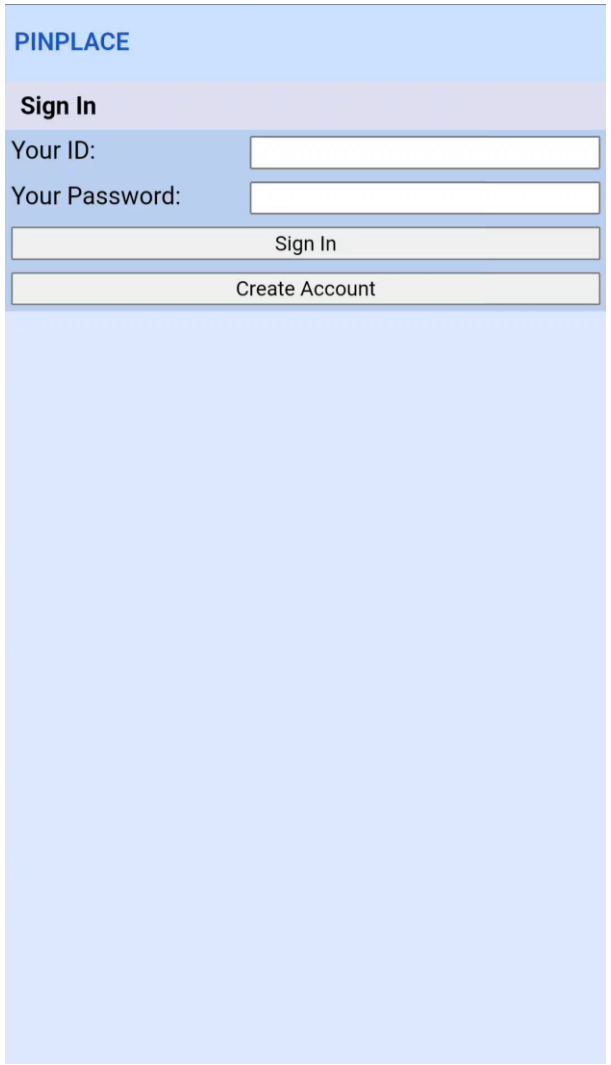


➢ Instruction

- Form with four text/password boxes
- Two password boxes in order to prevent mistyped password
  - "Not equal" alert if two password don't match
- "Account created" page for successful creation

03. Final Implementation > Frontend Part

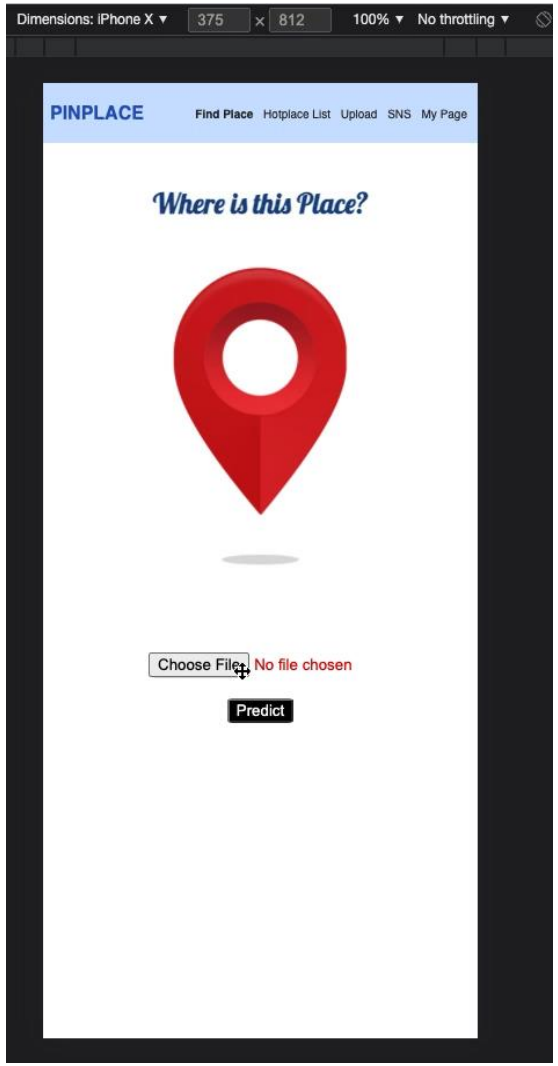
▪ Log-in Page



➢ Instruction

- Form with a text box for ID, a password box, and “sign in” and “create account” buttons
- “Invalid” alert for unsuccessful log-in attempt
- “Create Account” button for someone who doesn’t have an account for this service

▪ Find Location Page



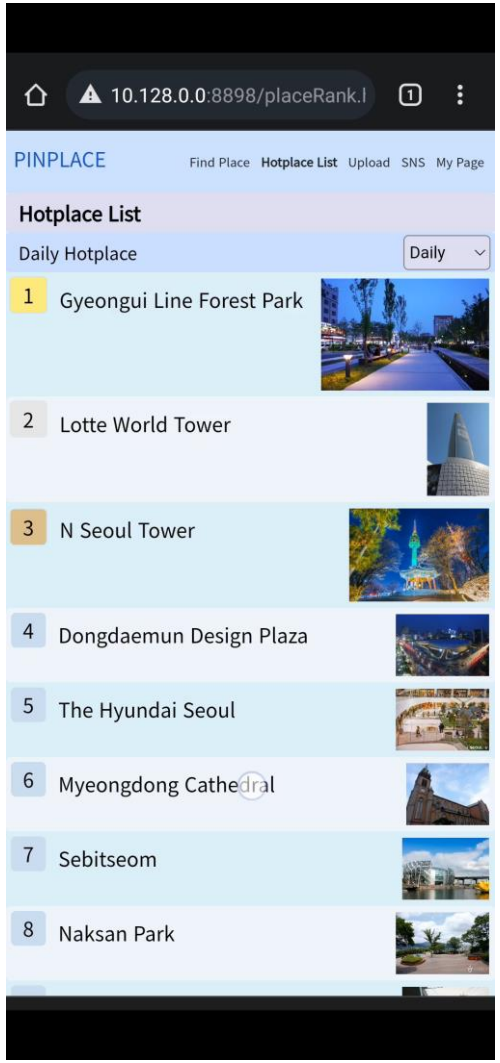
➢ Instruction

- Core Function Page
- Connect with CNN Model that we made ourselves
- User Flow

  Click Choose File Button
  -> Put Input File
  (regardless of file’s extension)
  -> Click Predict Button
  -> Appear Output(location)

03. Final Implementation > Frontend Part

List Up Page



Instruction

- Sort by popularity
  - Daily, Weekly, Monthly…
  - Actual measurement not implemented
- Page for each place:
  - ‘ejs’ Node.js module for automated generation
  - Images sorted by uploaded time

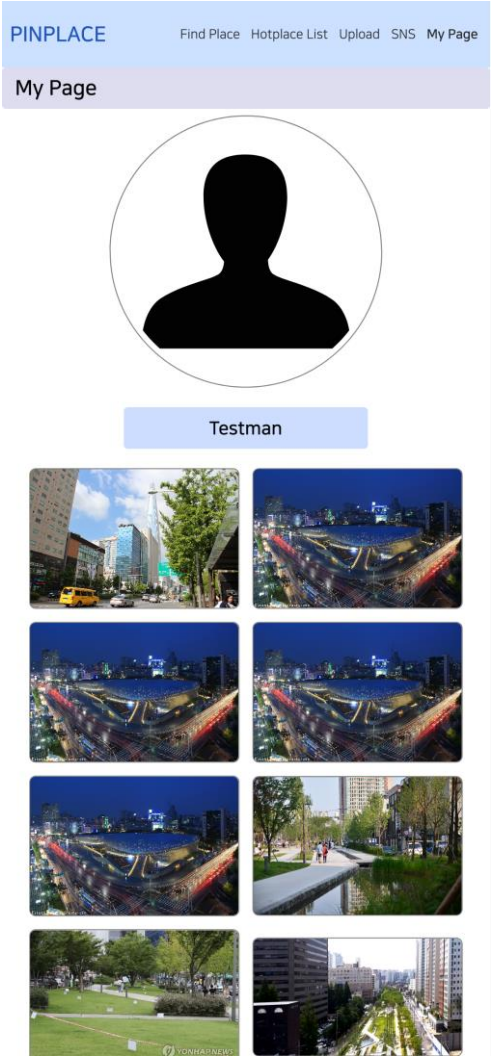
Upload Picture

Instruction

- With predefined locations
- Image preview right after file selection
- Submitted files are stored in a server with location identifiers

03. Final Implementation > Frontend Part

▪ My Page



➢ Instruction

- Photographs uploaded by a user in the session are shown
- Place for profile photo and user nickname
- ‘ejs’ Node.js module for automated generation

▪ SNS Page

➢ Instruction

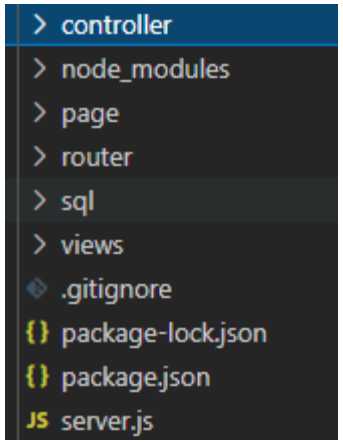
- Top Layer -> Image Slider that moves automatically (with js code)
- Implement ‘PREV, NEXT BTN’

  -PREV(<) : onclick function that move previous card

  -NEXT(>) : onclick function that move next card

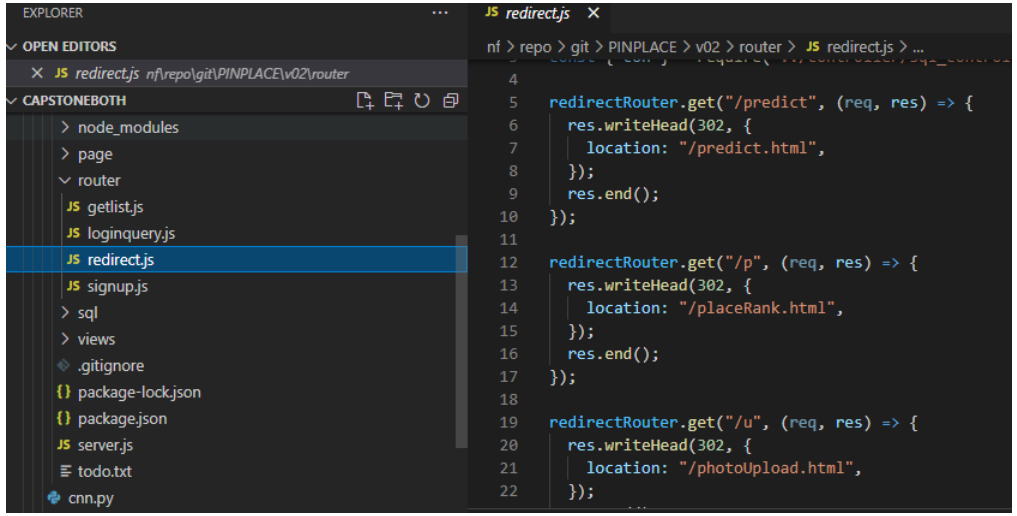
- Bottom Layer -> Posting function to recommend a location

General Layout



Main Server Code

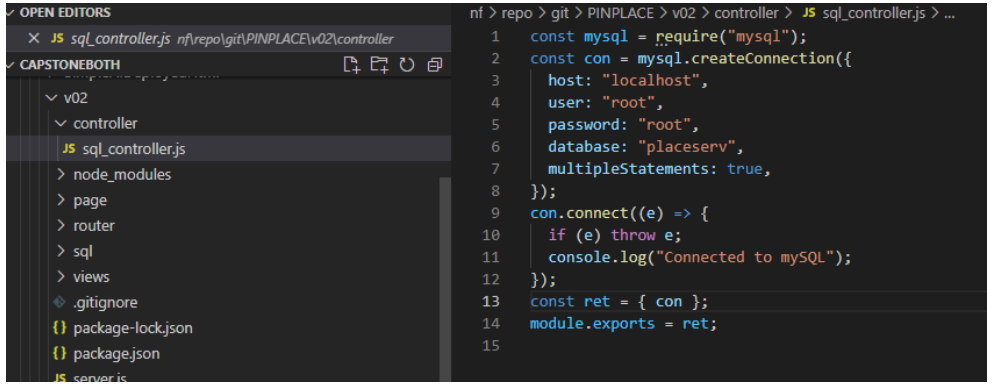
- Opens http server using express
- Routes signals into router or handle signals using callback function.



Router basic structure

- Handles routed signals
- Improves code readability
- Allows additional functionalities to be added.

General Layout



Controller

- Exports functionalities that would be used by multiple files
- Ex) SQL code

Packages

- Display dependency
- Allow upload to github without saving all the packages and modules.

"Challenges in CNN building"

Small size of image dataset

- we can get total 4,684 images
- So, we tried k-fold cross validation. However, this can not help to improve the accuracy
- So, we used data augmentation by ImageDataGenerator on keras library.
- Then we can get total 25,450 images

model confuse similar images

- Use confusion matrix to find what is predicted answer and what is the real answer.
- Then remove some ambiguous image data between classes.

04. Challenges – Front end & Back end

“Challenges in Front end”

User-friendly desing

- Designing is hard for non-professionals
- Made simple design to overcome

“Challenges in Back end”

Implementing Image Storage

- Image is saved into server static page folder and SQL stores the path
- Original Intension was to save Image Blob into SQL
- During development this seemed inefficient

“Limitation in current project”

Accuracy

- Although we do our best, 91.12% accuracy is not enough to users.
- If more complicate model is used, there might be better models which has accuracy more than 91.12%.

Range of places

- There are wide range of classes among our 10 classes.
- "Haebangchon", "Ikseon_Dong_Hanok_Village", "Gyeongui_Line_Forest_Park" have wide range of geographical place.
- Users can take photos at different places and then CNN model can not easily predict right places.

“Limitation in current project”

Distance of objects

- If the picture is taken far from the tower, the shapes of towers look so similar.
- Actually, our CNN model can confuse the "Namsan_Seoul_Tower" and "Jamsil_Lotte_Tower" which are taken far from towers.

"Limitation in current project"

Lack of user info change page

- Cannot change profile picture, password etc. user side

No proper social network functionality

- Only able to see photos as place pages are primitive

Measurement for place popularity not implemented

- Nearly impossible with little users and short time span
- Replaced by arbitrary numbers internally

“Limitation in current project”

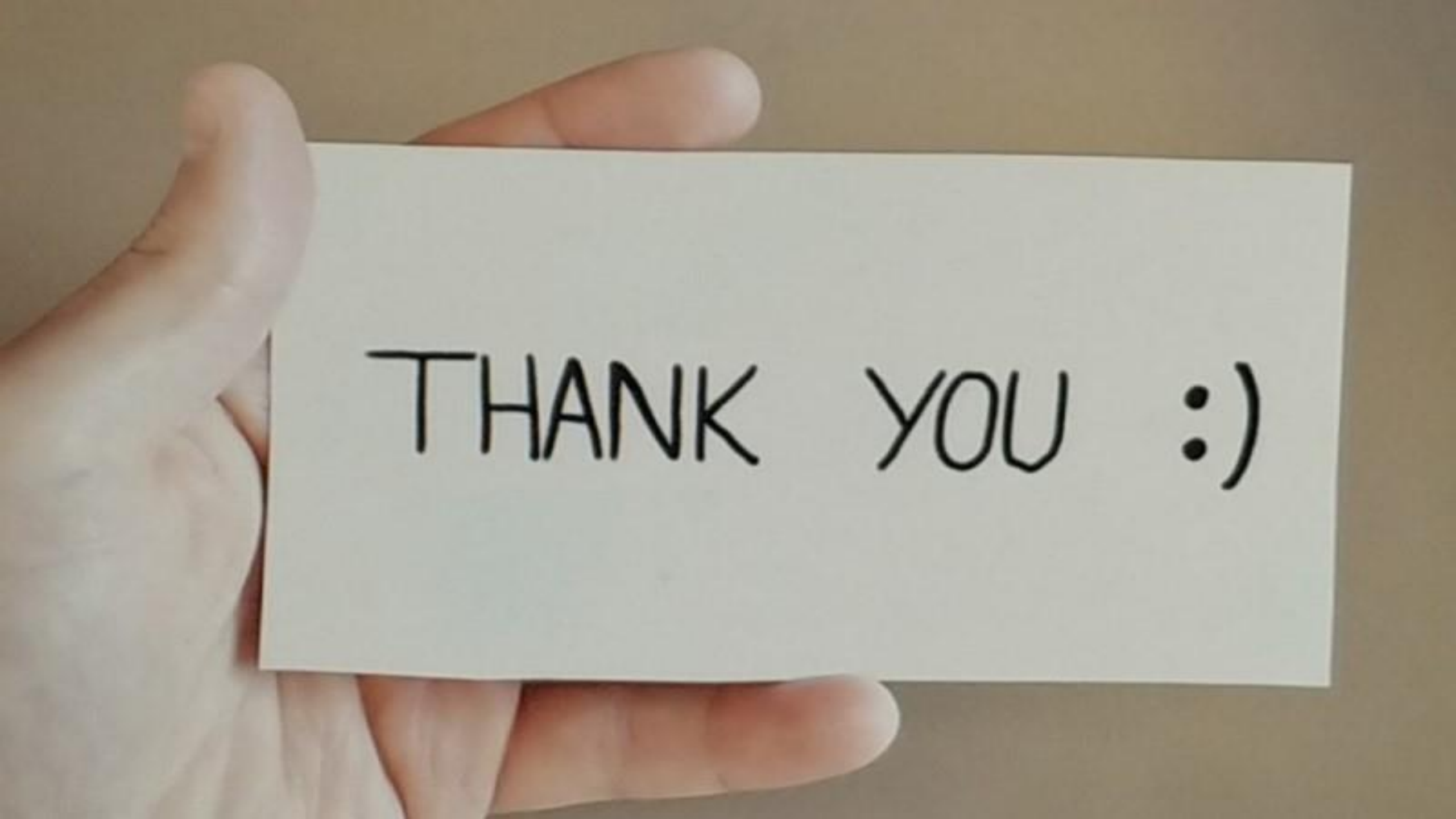
Security

- Does Hash the password however HTML pages use express static function to redirect.
- This allow user to access pages without login

Login Feature incomplete

- User login data is saved as cookie.
- This cookie is not hashed and is an integer value of user id in SQL Table.

Demo


THANK YOU :)