서울과학기술대학교 SEOULTECH SEOUL NATIONAL UNIVERSITY OF SCIENCE & TECHNOLOGY
전자IT미디어공학과

OpenAI 및 머신러닝 기반 멀티모달 레시피 추천 앱

22110119 권영호
지도교수 박구만교수님

Content

01

1. 작품 제작 동기 & 차별성

작품 제작 동기

- 식품 낭비는 전 세계적으로 큰 문제로 꾸준히 대두되어 왔다.
- 코로나, 우크라이나 사태 이후로 식자재 물가 상승으로 전세계는 유례없는 인플레이션을 맞이하였다.
- 일반 가정에서 식재료를 효율적으로 관리하지 못해 자주 버리거나, 잊혀져 불필요한 식재료 구매가 이루어 지곤 한다.

- 식재료 관리와 이를 활용한 레시피에 대한 효율적 솔루션의 부재를 인지하였다.
- 보다 스마트한 식재료 관리, 활용을 할 수 있도록 하고 이를 기반으로 레시피를 추천하는 모바일 앱을 제작하게 되었다.

차별성

- 기존 비슷한 목적을 위한 여러 앱, 솔루션이 있었다.
- 대부분 수동으로 식재료 입력, 관리가 필요하며 이를 기반으로한 레시피 추천은 제한적이었다.
- 위 기능을 위한 서버, 데이터베이스의 구축은 필수적이었다.
- 이러한 수동적 접근은 사용자로 하여 많은 시간과 노력을 요구하는 단점이 있음과 동시에 확장 가능성과 다양성에 상당한 제약을 주었다.

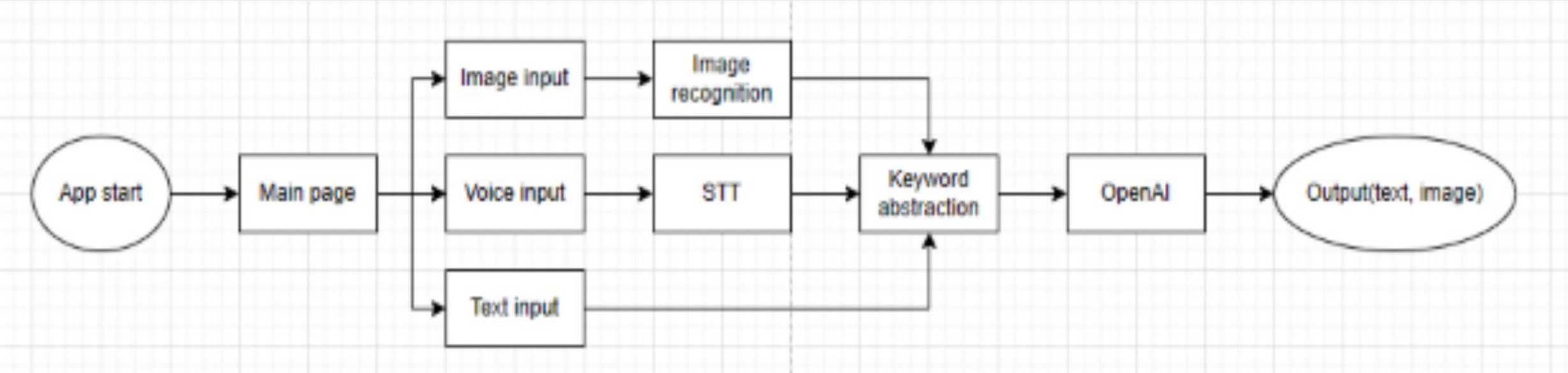
- 이미지 인식 기술과 STT를 활용하여 식재료 자동 분류 기능을 활용 한다.
- OpenAI를 활용하여 분류된 식재료 기반의 레시피와 그 예시 사진을 추천해 준다.
- 이렇게 함으로써 사용자의 편의성을 대폭 향상 시키고, 실시간으로 개인화된 레시피 추천을 하여 식재료 낭비를 줄인다.

02

2. 프로젝트 설계 및 구현

Flow chart

이 프로젝트는 Flutter앱 프론트엔드를 통해 이미지 인식, 음성인식(STT), 텍스트로 식재료를 입력 받아 OpenAI의 GPT-3, DALL-E를 활용하여 레시피를 제공한다. 이에 대한 Flow chart는 다음과 같다.



이미지 인식 : 데이터 수집, 라벨링

36가지 야채, 과일에 대한 이미지 데이터를 수집하고 라벨링을 진행하였다. 다음은 수집 방식 중 하나였던 Web Crawling에 대한 코드 및 결과에 대한 내용이다.

```python
import os
import shutil
from bing_image_downloader.bing_image_downloader import downloader

classes_list = ['apple', 'banana', 'beetroot', 'bell pepper', 'cabbage', 'capsicum', 'carrot', 'cauliflower',
                'chilli pepper', 'corn', 'cucumber', 'eggplant', 'garlic', 'ginger', 'grapes', 'jalepeno',
                'kiwi', 'lemon', 'lettuce',  'mango', 'onion', 'orange',  'paprika',
                'pear', 'peas', 'pineapple', 'pomegrante', 'potato', 'raddish', 'soy beans',
                'spinach', 'sweetcorn', 'sweetpotato', 'tomato', 'turnip', 'watermelon']

query_1 = ' as a ingredient'
query_2 = ' with white background'
query_list = [query_1, query_2]

for directory in directory_list:
    if not os.path.isdir(directory):
        os.makedirs(directory)

def dataset_split(query, train_cnt, val_cnt):
    for directory in directory_list:
        if not os.path.isdir(directory + '/' + query):
            os.makedirs(directory + '/' + query)
    cnt = 0
    for file_name in os.listdir(query):
        if cnt < train_cnt:
            print(f'[Train Dataset] {file_name}')
            shutil.move(query + '/' + file_name + query + '/' + file_name)
        elif (train_cnt + val_cnt) > cnt and cnt > val_cnt:
            print(f'[Validation Dataset] {file_name}')
            shutil.move(query + '/' + file_name + query + '/' + file_name)
        else:
            print(f'[Test Dataset] {file_name}')
            shutil.move(query + '/' + file_name + query + '/' + file_name)
        cnt += 1
    shutil.rmtree(query)

for keyword in classes_list:
    for c_query in query_list:
        query = keyword+c_query
        downloader.download(query, limit=60,  output_dir='./', adult_filter_off=True, force_replace=False, timeout=60)
        dataset_split(query, 100,10)
```

이미지 인식 : 데이터 전처리, 모델 학습

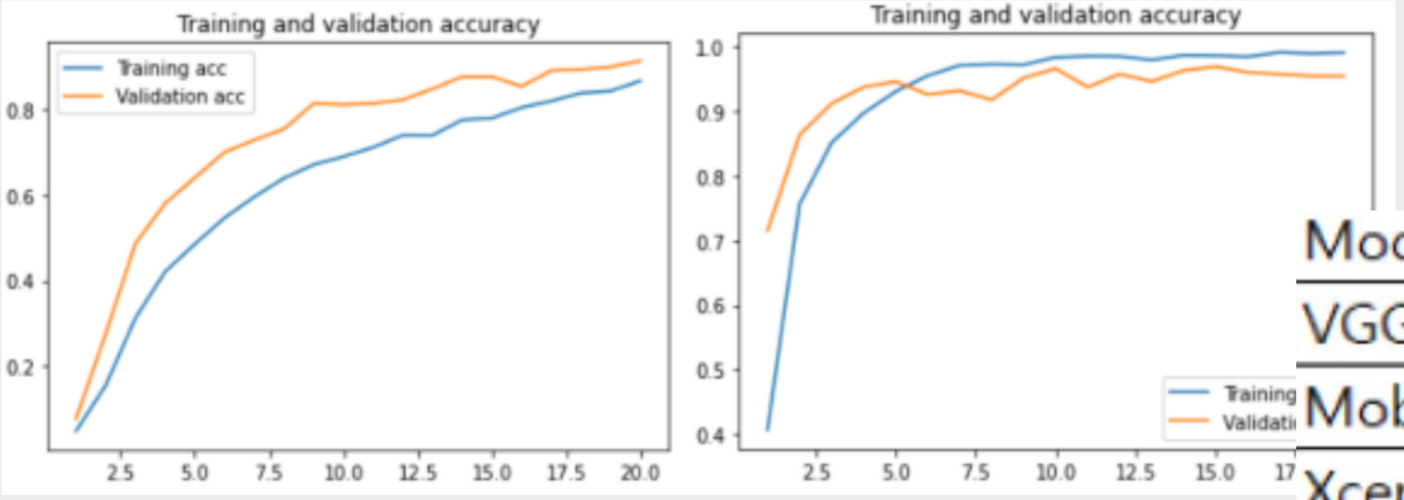
```
batch_size = 32
#image_size = 256
target_size = (256,256)
input_shape = (256, 256, 3)

#Fetching train data and validation data and processing the data
train_datagen = ImageDataGenerator(rescale = 1.00 / 255.0)
val_datagen = ImageDataGenerator(rescale = 1.00 / 255.0)
test_datagen = ImageDataGenerator(rescale = 1.00 / 255.0)

train_dir = ''
test_dir = ''
val_dir = ''

train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size = target_size,
    batch_size = batch_size

)

validation_generator = val_datagen.flow_from_directory(
    val_dir,
    target_size = target_size,
    batch_size = batch_size
)

test_generator = test_datagen.flow_from_directory(
    test_dir,
    target_size = target_size,
    batch_size = batch_size
)
```

```
# add new classifier layers
flat1 = model_mobilenetV2.layers[-1].output
flat1 = GlobalAveragePooling2D()(flat1)
class1 = layers.Dense(1024, activation='relu')(flat1)
class1 = layers.Dense(512, activation='relu')(class1)
class1 = layers.Dense(256, activation='relu')(class1)
# class1 = layers.Dense(128, activation='relu')(class1)
# class1 = layers.Dense(64, activation='relu')(class1)
output = layers.Dense(36, activation='sigmoid')(class1)
# define new model
modelMobileNetV2 = Model(inputs=model_mobilenetV2.inputs, outputs=output)
modelMobileNetV2.compile(
  loss = "binary_crossentropy",
  optimizer="adam",
  metrics = ["acc"]
)
# summarize
modelMobileNetV2.summary()
```

```
#Training the model with train data and judging this training with validation data
es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=5)
historyMNV2 = modelMobileNetV2.fit(
    train_generator,
    batch_size=batch_size,
    epochs = 25,
    validation_data = validation_generator,
    callbacks=[es],
    verbose=1)
```

이미지 인식 : 정확도 비교, 모델 채택

위의 데이터 및 모델 구조 조건으로 VGG19, MobileNet, Xception, ResNet50, MobileNetV2 모델들의 학습을 하였고 정확도를 평가 지표로 하여 비교하였다. 그 결과 MobileNetV2를 이 프로젝트의 이미지 분류 모델로써 활용하기로 결정하였다.



Model		loss	acc
VGG19		0.0161	0.9136
MobileNet		**0.0105**	**0.9554**
Xception		0.0087	0.9471
ResNet50		0.0963	0.2284
MobileNetV2		**0.0059**	**0.9747**

Tensorflow lite : 개요

- TensorFlowLite(TFLite)를 사용하여 훈련된 모델을 모바일 환경에서 실행 가능한 형태로 변환하였다.
- TFLite는 개발자가 모바일, 내장형기기, IoT기기에서 모델을 실행할 수 있도록 지원하여 기기 내 머신러닝을 사용할 수 있도록 하는 도구 모음이다.
- TFLite는 TFLite Converter와 TFLite Interpreter로 구성되어있다.
- TFLite Converter는 TensorFlow 모델을 Interpreter가 사용할 수 있도록 최적화하는 역할을 하며 효율적이고, 적은 용량, 성능은 유지하도록 한다.
- TFLite Interpreter은 최적화된 모델을 다양한 하드웨어에서 구동 될 수 있도록 도와주는 역할로 모바일폰, 임베디드 리눅스 디바이스, 마이크로 컨트롤러 등에서 동작한다.

Tensorflow lite : 구현

- TFLite의 개발 진행 순서는 다음과 같이 정리 된다.

1. 모델 훈련
2. 모델 변환
3. 모델 통합
4. 모델 추론
5. 하드웨어 가속

Tensorflow lite : 모델 변환

```python
import tensorflow as tf

# Load the model
model = tf.keras.models.load_model('./models/modelMobileNetV2.h5')

# Convert the model.
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()

# Save the model.
with open('model.tflite', 'wb') as f:
  f.write(tflite_model)
```

[13] ✓ 28.1s Python

```
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_conv
INFO:tensorflow:Assets written to: C:\Users\YOUNGH~1\AppData\Local\Temp\tmp_xd3axr4\assets
INFO:tensorflow:Assets written to: C:\Users\YOUNGH~1\AppData\Local\Temp\tmp_xd3axr4\assets
```

Tensorflow lite : 모델 통합

```dart
Future<void> processImage() async {
  if (imagePath != null) {
    // Read image bytes from file
    final imageData = File(imagePath!).readAsBytesSync();//List<int>
    // print(imageData.shape);

    // Decode image using package:image/image.dart (https://pub.dev/i
    image = img.decodeImage(imageData);
    // print(image);
    setState(() {});

    // Resize image for model input (Mobilenet use [256, 256])
    final imageInput = img.copyResize(
      image!,
      width: 256,
      height: 256,
    );

    // Get image matrix representation [256, 256, 3]
    final imageMatrix = List.generate(
      imageInput.height,
      (y) => List.generate(
        imageInput.width,
        (x) {
          final pixel = imageInput.getPixel(x, y);
          return [pixel.r, pixel.g, pixel.b];
        },
```

1. 갤러리, 카메라를 통한 이미지 입력
2. 이미지의 픽셀 읽음 및 디코딩
3. 모델에 입력하기 위한 resize(256, 256)
4. (1, 256, 256, 3) 크기의 matrix 생성

Tensorflow lite : 모델 추론

```dart
// Run inference
Future<void> runInference(List<List<List<num>>> imageMatrix) async {
  // Set tensor input [1, 256, 256, 3]
  final input = [imageMatrix];

  // Set tensor output [1, 36]
  final output = [List<double>.filled(36, 0)];

  // Run inference
  interpreter.run(input, output);

  // Get first output tensor
  // result : [probabilities]
  final result = output.first;

  // argmax result
  // resultIndex : The biggest probability Index
  final resultIndex = argmax(result);

  // label : classified ingredient
  final label = labels[resultIndex];
  showConfirmationDialog(File(imagePath!), label);
  setState(() {});
}
```

```dart
int argmax(List<dynamic> X) {
  int idx = 0;
  int l = X.length;
  for (int i = 0; i < l; i++) {
    idx = X[i] > X[idx] ? i : idx;
  }
  return idx;
}
```

Tensorflow lite : 하드웨어 가속

```dart
// Load model
Future<void> loadModel() async {
  final options = InterpreterOptions();

  // Use XNNPACK Delegate
  if (Platform.isAndroid) {
    options.addDelegate(XNNPackDelegate());
  }

  // Use GPU Delegate
  // doesn't work on emulator
  if (Platform.isAndroid) {
    options.addDelegate(GpuDelegateV2());
  }

  // Use Metal Delegate
  if (Platform.isIOS) {
    options.addDelegate(GpuDelegate());
  }
```

TFLite는 특정 장치에서 하드웨어 가속을 지원한다. 이는 장치가 머신러닝 작업을 가속화할 수 있는 하드웨어(예: GPU 또는 Neural Processing Unit)를 가지고 있다면, TFLite가 이를 활용해 추론을 더 빠르게 실행할 수 있다는 것을 의미한다.

Speech To Text(STT)

- Flutter와 호환되는 STT 라이브러리를 선별하여 앱에 적용하였다.

```
Text(_text),
const SizedBox(height: 40,),
FloatingActionButton(
  onPressed: _listen,
  child: Icon(_isListening ? Icons.stop : Icons.mic),
),  // FloatingActionButton
```

```
void _listen() async {
  if (!_isListening) {
    bool available = await _speech.initialize(
      onStatus: (val) => print('onStatus: $val'),
      onError: (val) => print('onError: $val'),
    );
    if (available) {
      setState(() => _isListening = true);
      _speech.listen(
        onResult: (val) => setState(() {
          _text = val.recognizedWords;
          // ingredients.add(_text);
          splitTextBySpace();
        }),
        localeId: localeId
      );
    }
  } else {
    setState(() => _isListening = false);
    _speech.stop();
  }
}
```

OpenAI : GPT3 API

```
class ApiService {
  //request for openAI GPT
  static Future<List<String>> postRecipes(List<String> ingredients) async {
    try {
      var response = await http.post(Uri.parse(API_URL),
          headers: {
            'Authorization': 'Bearer $API_KEY',
            "Content-Type": "application/json"
          },
          body: jsonEncode({
            "model": "text-davinci-003",
            // "model": "gpt-3.5-turbo",
            "prompt": "suggest 3 recipes base on those:\n\n$ingredients",
            "temperature": 0.83,
            "max_tokens": 997,
            "top_p": 1,
            "frequency_p
            "presence_pe
          }));
      Map jsonResponse = jsonDecode(response.body);
      String recipes = jsonResponse["choices"][0]["text"];
      String recipe1 = recipes.substring(recipes.indexOf('1. '), recipes.indexOf('2. '));
      String recipe2 = recipes.substring(recipes.indexOf('2. '), recipes.indexOf('3. '));
      String recipe3 = recipes.substring(recipes.indexOf('3. '));
      return [recipe1, recipe2, recipe3];
    } catch (error) {
      print("error-postRecipes-response: $error");
      return ['recipe request failed'];
    }
  }
```

- OpenAI API reference를 참조하여 구현하였다. GPT-3와 DALL-E를 사용하였다. GPT-3에 원하는 질의를 하기 위해 request body에 prompt문을 작성해야한다. suggest 3 recipes base on those + $ingredients(분류된 라벨 List)의 요청문을 만들어 사용하였다. 받게 될 response에는 세 가지 레시피가 1. 2. 3. 으로 시작하게 된다. 이를 기준으로 response를 자르고 반환하도록 하였다.

OpenAI : DALL-E API

```
// api request for Image Generation Model
static Future<String> postImage(String recipe1) async {
  try {
    var response = await http.post(Uri.parse(API_IMG_URL),
        headers: {
          'Authorization': 'Bearer $API_KEY',
          "Content-Type": "application/json"
        },
        body: jsonEncode({"prompt": recipe1, "n": 1, "size": "256x256"}));
    Map jsonResponse = jsonDecode(response.body);
    // print("getImage: $jsonResponse");
    String img_url = jsonResponse["data"][0]["url"];
    // print("img url: $img_url");
    return img_url;
  } catch (error) {
    print("error-postImage-response: $error");
    return 'image request failed';
  }
}
```

- DALL-E의 경우 역시 request body에 prompt문을 사용하여 생성할 이미지를 요청한다.

03

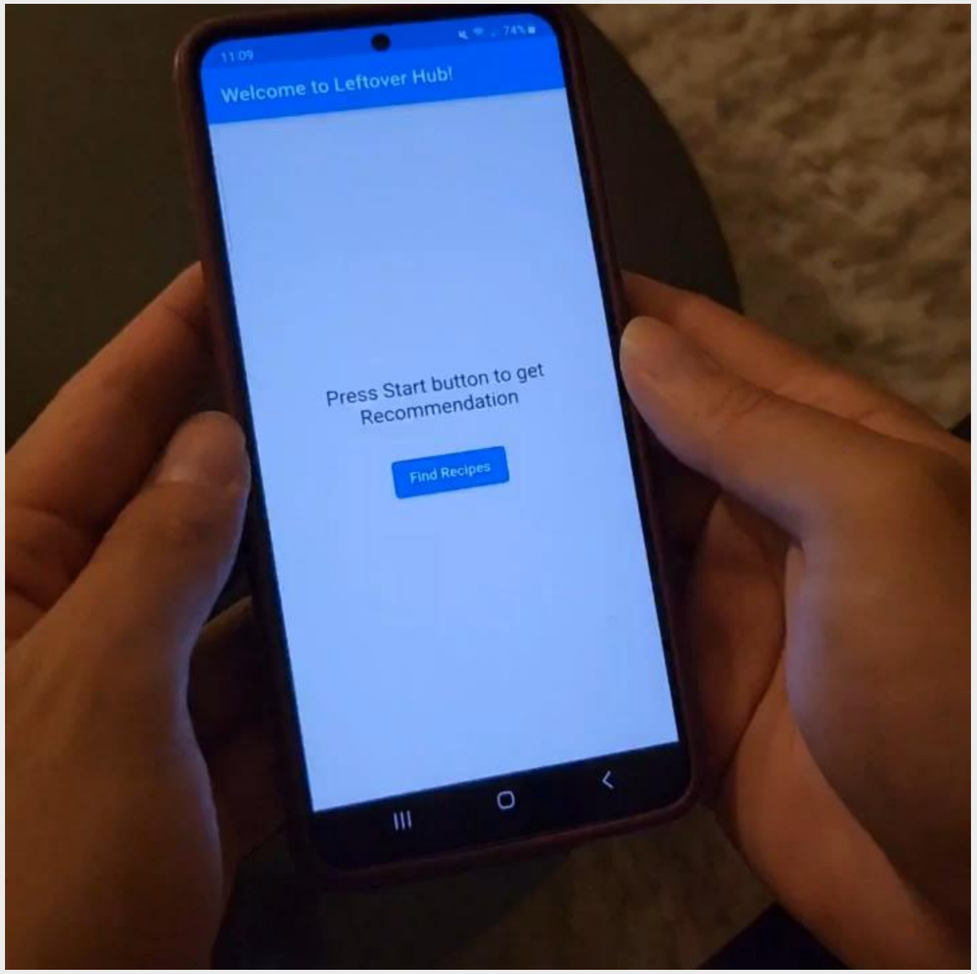
3. 결론 및 데모영상

결론

- 본 프로젝트에서는 식재료 인식을 기반으로 레시피를 추천하는 모바일 앱을 성공적으로 개발하였다. 이를 위해 다양한 기술, 논문, 그리고 도구를 종합적으로 연구하고 적용하였다. 특히, TensorFlow와 MobileNetV2를 활용한 이미지 인식 모델은 약 97%의 높은 정확도를 보였다. 또한, 사용자의 음성을 텍스트로 변환하는 STT 기술, 그리고 OpenAI의 GPT-3와 DALL-E를 이용한 레시피 생성 및 시각적 피드백도 원활하게 구현할 수 있었다.
- 향후 개선 및 확장 계획으로는 다음과 같은 사항들이 있다:

1. 더 다양한 종류의 식재료를 인식할 수 있도록 모델을 업그레이드 할 계획이다.
2. 사용자의 특정 건강 상태나 질병에 따른 식습관 가이드라인을 제공하는 기능을 추가할 예정이다.
3. 다이어트를 위한 칼로리 계산 기능을 통해 사용자에게 더 많은 정보를 제공할 계획이다.

데모 영상


11:09
Welcome to Leftover Hub!
Press Start button to get Recommendation
Find Recipes