PORTFOLIO

허지혜 포트폴리오

PROFILE

허지혜

프로필

이 름 허지혜
생 일 1995.
연 락 처
이 메 일
주 소 서울시 송파구 잠실동

개발능력

JAVA 4/5
Spring 4/5
Python 4/5
R 2/5

학력

2019.02
한국사학과 졸업
2024.02 서울강서폴리텍대학
데이터분석과 졸업예정

자격증 및 수상 경력

2021.10 정보처리기능사
2023.05 정보처리기사

PORTFOLIO
CONTENTS

Project 01.

아두이노를 활용한 신체 놀이학습 서비스

About project

한이음 ICT 공모전 참여

팀장

수상 : 입상

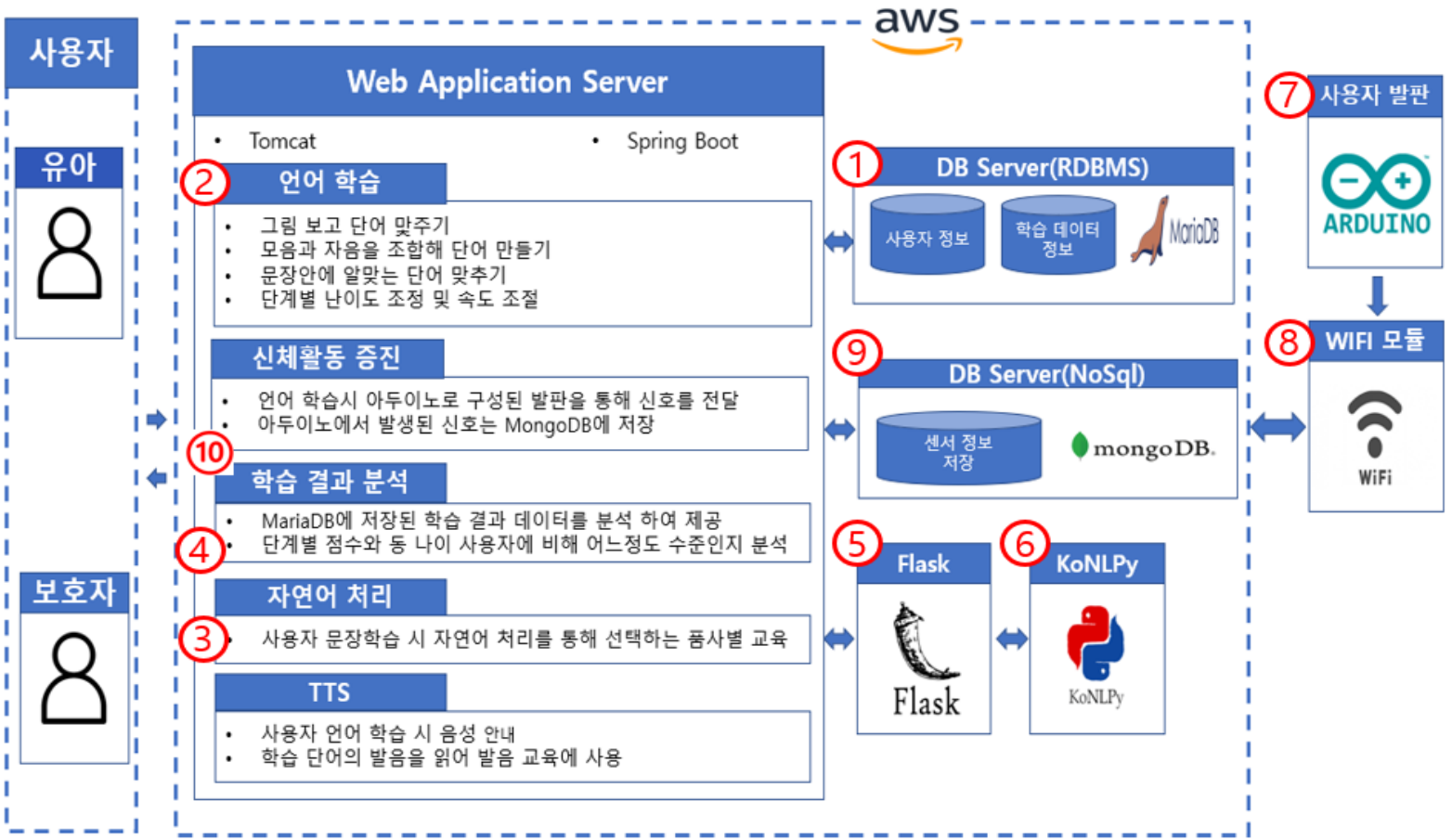
Project 01. 아두이노를 활용한 신체 놀이학습 서비스

Introduce project

작업 기간	2022. 03 ~ 2022. 11 (9개월)
인력 구성(기여도)	팀장
프로젝트 목적	한이음 ICT 공모전
프로젝트 내용	1) 영유아의 신체적, 언어적 발달 문제를 해결하기 위한 신체 놀이학습 서비스 2) 다양한 학습 데이터와 아두이노 센서를 이용한 인터렉티브 방식의 올인원 서비스 3) 아두이노 센서 발판을 단계별로 사용하는 신체학습
주요 업무 및 상세 역할	1) 회원가입 및 사용자 정보 기능 개발 2) Flask Framework를 활용한 API 서버 개발 3) KoNLPy를 활용한 문장 품사 태깅 및 문제 생성 4) MQTT를 상용한 센서 데이터 처리 5) AWS 배포 및 버그 수정
사용언어 및 개발 환경	JAVA, Python, C++, AWS, EC2, Python, HTML5, JAVASCRIPT

Project 01. 아두이노를 활용한 신체 놀이학습 서비스

Main work 서비스 구성도



Main work Okt 라이브러리를 활용한 문제 생성

```python
def nlp_tes(spring_sentence, study_type):
    sentence_res2 = [] # 자연어 처리 결과를 담을 변수
    for i in range(len(spring_sentence)):
        # spring에서 받은 학습 데이터 list에서 꺼내서 str로 변환
        voc = spring_sentence[i]
        # 형태소 분석을 위한 okt 라이브러리 객체 생성
        okt = Okt()
        # 형태소 분석 시작
        okt_res = okt.pos(voc)
        print(okt_res)
        # 학습 타입별 문제를 제공하기위한 반복
        for j in range(len(okt_res)):
            # 학습 타입과 같은 품사가 있을경우 처리 시작
            if study_type == okt_res[j][1]:
                # spring에서 받은 문장안에서 태깅된 품사 종류와 같은 단어를 변환
                voc = voc.replace(okt_res[j][0], 'a')
                # 변환된 문장과 정답을 dictionary 타입으로 변환
                dic_content = {'content': voc, 'blink_word': okt_res[j][0]}
                sentence_res2.append(dic_content) # spring으로 전달하기 위해 list 타입으로 변경
                break
```

- **Flask Server Python 라이브러리인 KoNLPy에 학습 데이터를 전달**
- **KoNLPy API는 전달받은 학습 데이터를 받아 Okt 라이브러리를 사용하여 형태소 분석을 진행**
- **선택한 품사별 문제를 생성하여 Spring Server로 반환**

Project 01. 아두이노를 활용한 신체 놀이학습 서비스

Main work 아두이노와 Spring 간 MQTT 통신

```
void reconnect() {
  // MQTT 연결
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // MQTT 연결 시도
    if (client.connect("arduinoClient", "admin", "1234")) {
      Serial.println("connected");
      // MQTT 발행
      client.publish("outTopic", "hello world");
      // MQTT 구독
      client.subscribe("inTopic");
    } else {
      //실패시 수행 5초 딜레이
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
}
```

```
private MqttClient client;
private MqttConnectOptions option;
private Consumer<HashMap<Object, Object>> FNC1 = null;  //메시지 도착 후 응답하는 함수
private Consumer<HashMap<Object, Object>> FNC2 = null;  // 커넥션이 끊긴 후 응답하는 함수
private Consumer<HashMap<Object, Object>> FNC3 = null;  //전송 완료 후 응답하는 함수

/* 옵션 객체에 접속 하기 위한 세팅
 * 파라미터로 4개 필요(사용자이름, 비밀번호, 주소, 접속 후 사용할 아이디)
 */
public MyMqttClient init(String userName, String password, String serverURI, String clientId){
    option = new MqttConnectOptions();
    option.setCleanSession(true);
    option.setKeepAliveInterval(30);
    option.setUserName(userName);
    option.setPassword(password.toCharArray());  //옵션 객체에 접속하기위한 세팅끝!
    try {
        client = new MqttClient(serverURI, clientId);
        client.setCallback(this);
        client.connect(option);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return this;
}
```

- 사용자가 아두이노 발판을 통해 정답을 선택하면 센서 신호가 발생되고 발생된 센서 데이터를 MQTT 통신을 통해 Spring Server로 전달 후 MongoDB에 저장
- 사용자가 문제의 정답을 맞췄을 경우 학습 시간과 정답 여부를 MariaDB에 저장 후 학습 종료

Project 02.

인공지능 시선추적 학습 모듈 기반 루게릭병(ALS) 환자 커뮤니케이션 플랫폼

About project

프로보노 공모전

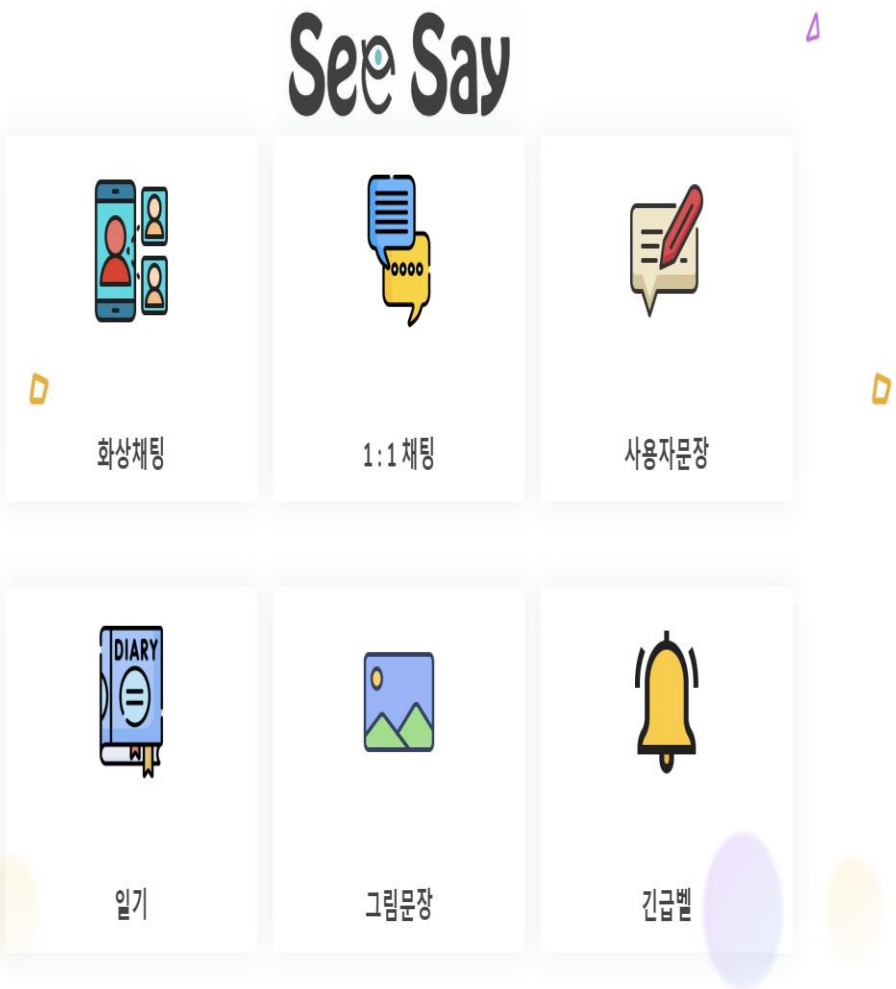
팀원

수상 : 동상

Project 02. 인공지능 시선추적 학습 모듈 기반 루게릭병(ALS) 환자 커뮤니케이션 플랫폼

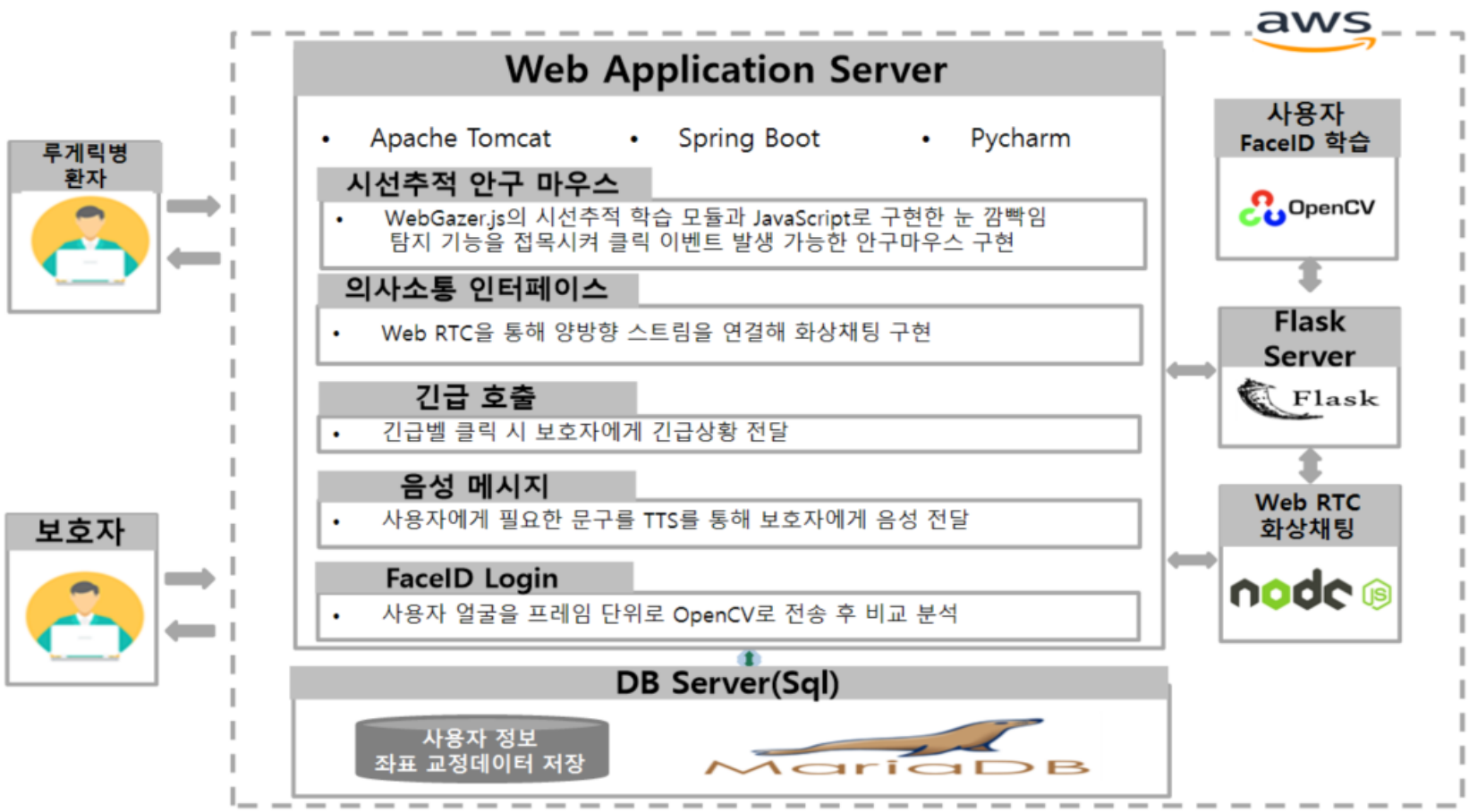
Introduce project

작업 기간	2022.09~2022.12
인력 구성(기여도)	팀원
프로젝트 목적	프로보노 공모전
프로젝트 내용	한국루게릭협회(KALSA)에 따르면 루게릭병 환자 수는 매년 약 28% 씩 증가하고 있으며, 환자의 75% 이상은 발병 이후 3~4년 안에 안구 근육을 제외한 전신 근육 마비 및 수축으로 보호자의 도움 없이 의사소통이 어려워 OpenCV를 활용한 안구 마우스를 사용해 의사소통에 도움을 주기 위해 제작
주요 업무 및 상세 역할	1) OpenCV 기반 WebGazer.js 최적화 및 안구 마우스 제작 2) ORM 방식의 JPA를 사용하여 사용자 데이터 저장 3) JavaScript를 사용하여 웹 페이지 안구 마우스 제작 4) 사용자 안구 Dot Point 최적화 5) 화상채팅 기능 구현 6) Face ID 로그인 7) 긴급상황 시 보호자 문자 전송
사용언어 및 개발 환경	JAVA, Node.js , AWS, EC2, WebRTC, HTML5, JavaScript



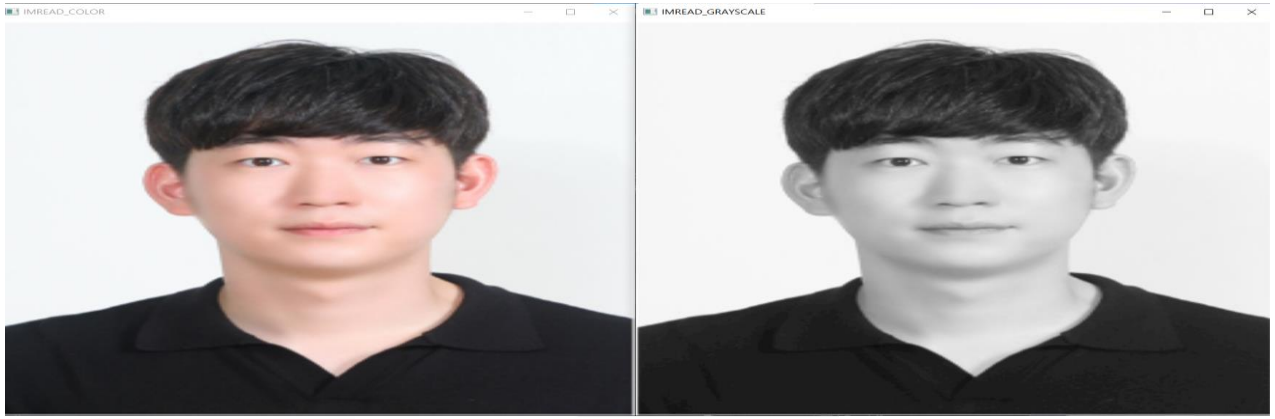
Project 02. 인공지능 시선추적 학습 모듈 기반 루게릭병(ALS) 환자 커뮤니케이션 플랫폼

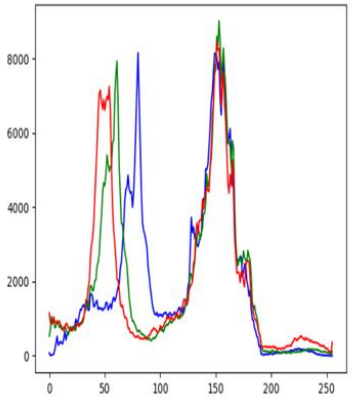
Main work 서비스 기능도



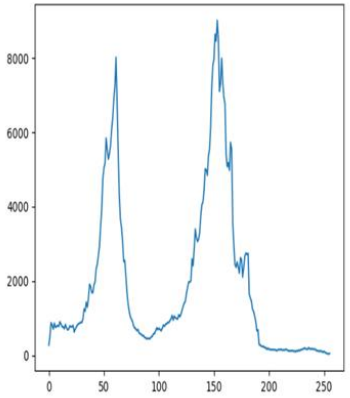
Project 02. 인공지능 시선추적 학습 모듈 기반 루게릭병(ALS) 환자 커뮤니케이션 플랫폼

Main work OpenCV를 활용한 Face ID 로그인 기능 개발

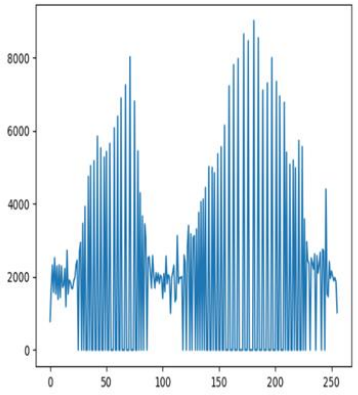




(a) 원본 이미지



(b) GrayScale



(c) 히스토그램 평활화

- OpenCV를 활용한 Face ID 로그인 개발
- 캠에서 인식된 사용자 얼굴을 100장의 사진으로 변환하여 저장하고 이미지 파일 경로를 MariaDB에 저장함
- 저장된 이미지 파일은 Spring Framework에서 URL을 통해 Flask server로 전송함
- Flask Server는 전송받은 이미지 파일을 Python Libary인 OpenCV로 전달함
- Deep Learning 된 이미지가 아닐경우 GrayScale로 변환 후 하르 분류기를 사용해 얼굴 검출 및 히스토그램 평활화
- 학습 데이터 생성

Main work WebGazer.js를 활용한 안구 마우스(1/2)

```
// EXTERNAL MODULE: ./node_modules/localforage/dist/localforage.js
var localforage = __webpack_require__(141);

// EXTERNAL MODULE: ./node_modules/@tensorflow-models/facemesh/dist/index.js
var dist = __webpack_require__(273);

// CONCATENATED MODULE: ./src/facemesh.mjs

/**
 * Constructor of TFFaceMesh object
 * @constructor
 * */
const TFFaceMesh = function() {
  //Backend options are webgl, wasm, and CPU.
  //For recent laptops WASM is better than WebGL.
  //TODO: This hack makes loading the model block the UI. We should fix that
  // this.model = (async () => { return await facemesh.load({"maxFaces":1}) })();
  this.model = dist.load({"maxFaces":1});
  this.predictionReady = false;
};

// Global variable for face landmark positions array
TFFaceMesh.prototype.positionsArray = null;
```

FaceMesh 얼굴 랜드마크 추정

```
/**
 * Initializes navigator.mediaDevices.getUserMedia
 * depending on the browser capabilities
 *
 * @return Promise
 */
function setUserMediaVariable(){

 if (navigator.mediaDevices === undefined) {
   navigator.mediaDevices = {};
 }

 if (navigator.mediaDevices.getUserMedia === undefined) {
   navigator.mediaDevices.getUserMedia = function(constraints) {

     // gets the alternative old getUserMedia is possible
     var getUserMedia = navigator.webkitGetUserMedia || navigator.mozGetUserMedia;

     // set an error message if browser doesn't support getUserMedia
     if (!getUserMedia) {
       return Promise.reject(new Error("Unfortunately, your browser does not support access to the webcam through
       Try to use the latest version of Google Chrome, Mozilla Firefox, Opera, or Microsoft Edge instead."));
     }

     // uses navigator.getUserMedia for older browsers
     return new Promise(function(resolve, reject) {
       getUserMedia.call(navigator, constraints, resolve, reject);
     });
   }
 }
```

Tracking.js 눈동자 움직임 추적

Main work WebGazer.js를 활용한 안구 마우스(2/2)

```
src_webgazer.setRegression = function(name) {
  if (regressionMap[name] === undefined) {
    console.log('Invalid regression selection');
    console.log('Options are: ');
    for (var reg in regressionMap) {
      console.log(reg);
    }
    return src_webgazer;
  }
  data = regs[0].getData();
  regs = [regressionMap[name]()];
  regs[0].setData(data);
  return src_webgazer;
};

/**
 * Adds a new tracker module so that it can be used by setTracker()
 * @param {String} name - the new name of the tracker
 * @param {Function} constructor - the constructor of the curTracker object
 * @return {webgazer} this
 */
src_webgazer.addTrackerModule = function(name, constructor) {
  curTrackerMap[name] = function() {
    return new constructor();
  };
};

/**
 * Adds a new regression module so that it can be used by setRegression() and addRegression()
 * @param {String} name - the new name of the regression
 * @param {Function} constructor - the constructor of the regression object
 */
src_webgazer.addRegressionModule = function(name, constructor) {
  regressionMap[name] = function() {
    return new constructor();
  };
};

/**
 * Adds a new regression module to the list of regression modules, seeding its data from the first regression module
 * @param {String} name - the string name of the regression module to add
 * @return {webgazer} this
 */
src_webgazer.addRegression = function(name) {
  var newReg = regressionMap[name]();
  data = regs[0].getData();
  newReg.setData(data);
  regs.push(newReg);
  return src_webgazer;
```

능선 회귀모델 시선 예측값 추출 및 학습

```
window.onload =  webgazer.setGazeListener(function (data, elapsedTime) {
    if (data == null) {
        return;
    }
    var xprediction = parseInt(data.x); //these x coordinates are relative to the viewport
    var yprediction = parseInt(data.y); //these y coordinates are relative to the viewport
    var blink = String(data.eyeFeatures.left.blink);

    const message_send = document.getElementById("draw");

    let message_top = message_send.getBoundingClientRect().top;
    let chg_top = parseInt(message_top);
    let message_bottom = message_send.getBoundingClientRect().bottom;
    let chg_bottom = parseInt(message_bottom);
    let message_left = message_send.getBoundingClientRect().left;
    let chg_left = parseInt(message_left);
    let message_right = message_send.getBoundingClientRect().right;
    let chg_right = parseInt(message_right);


    console.log("눈 깜빡임 ===== " + blink);


    if (chg_left <= xprediction && xprediction <= chg_right && chg_top <= yprediction && yprediction
        console.log("================================");
        location.href = "test.html";
    }
```

사용자 애플리 케이션 실행 후 모델 로드
div 영역 내 이벤트 발생 시 클릭

Project 03.

자연어 처리기반 사용자 여행지 추천 서비스

About project

Paas-TA 공모전

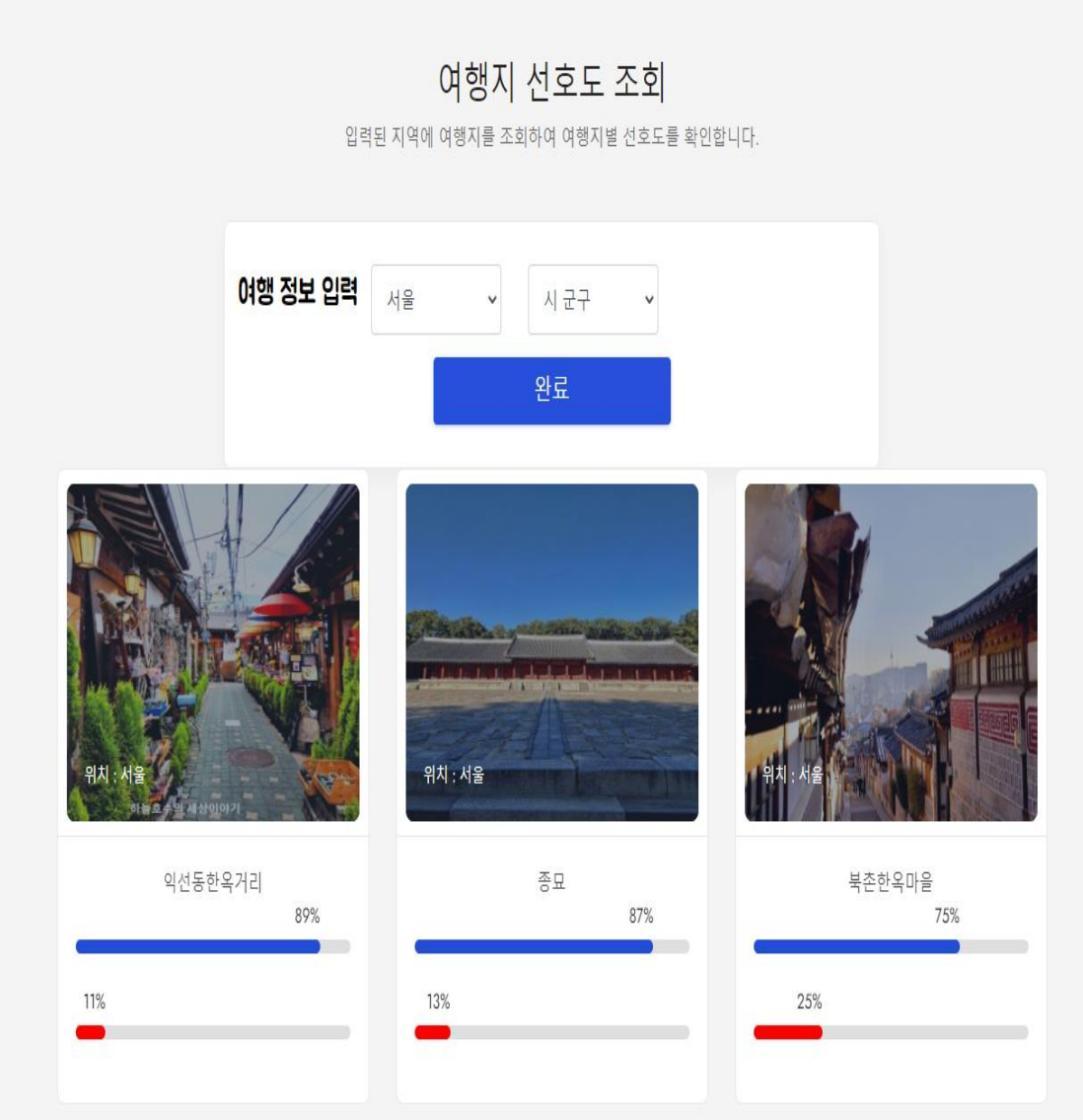
팀장

수상 : 과학기술정보통신부장관상

Project 03. 자연어 처리기반 사용자 여행지 추천 서비스

Introduce project

작업 기간	2022.08~2022.11
인력 구성(기여도)	팀장
프로젝트 목적	개방형 클라우드 플랫폼 Paas-Ta 공모전
프로젝트 내용	많은 여행객들은 여행 전 리뷰 데이터를 통해 여행 계획에 참고하지만 실제 여행객의 리뷰 데이터가 아닌 홍보성 리뷰 데이터도 존재 하기 때문에 여행지 추천 시 신뢰성을 높이고자 여행객들의 리뷰데이터를 분석하여 해당 여행지에 대한 긍정 부정을 통해 여행지를 추천함
주요 업무 및 상세 역할	1) 빅데이터 기반 국내 여행지 소개 2) 인공지능 자연어 처리 리뷰 데이터 분석 3) 캘린더 공유(여행일정 공유) 4) 관심정보 기반 여행지 추천 5) 사용자 보안을 위한 Security 적용 6) 로그인 인증 JWT Token 발행
사용언어 및 개발 환경	JAVA, Java Script, HTML5, CSS, Spring Boot, Python, Flask, PaaS, Spring Security, Jwt Token



Project 03. 자연어 처리기반 사용자 여행지 추천 서비스

Main work 감정 분석

```
# 학습모델 로딩하기
loaded_model = tf.keras.models.load_model("model/travel_model.h5")

stopwords = ['의','가','이','은','들','는','좀','잘','걍','과','도','를','으로','자','에','와','한','하다']

with open("model/tokenizer.pickle", "rb") as handle:
  tokenizer = pickle.load(handle)

okt = Okt()

max_len = 30

def sentiment_predict(new_sentence):
  new_sentence = re.sub(r'[^ㄱ-ㅎㅏ-ㅣ가-힣 ]','', new_sentence)
  new_sentence = okt.morphs(new_sentence, stem=True) # 토큰화
  new_sentence = [word for word in new_sentence if not word in stopwords] # 불용어 제거
  encoded = tokenizer.texts_to_sequences([new_sentence]) # 정수 인코딩
  pad_new = pad_sequences(encoded, maxlen = max_len) # 패딩

  score = float(loaded_model.predict(pad_new)) # 예측
  if(score > 0.5):
    print("{:.2f}% 확률로 긍정 리뷰입니다.\n".format(score * 100))
  else:
    print("{:.2f}% 확률로 부정 리뷰입니다.\n".format((1 - score) * 100))
```

- TensorFlow, Keras, LSTM을 활용한 사용자 여행지별 리뷰데이터 긍정, 부정 분석

- 리뷰 데이터 수집은 naver 영화 리뷰 데이터 수집
- 20만건의 데이터를 학습 데이터 15만건 테스트용 5만건으로 분류
- 중복 및 null 데이터 제거 및 불용어 설정
- 단어 토큰화

Project 03. 자연어 처리기반 사용자 여행지 추천 서비스

Main work KoNIpy를 활용한 사용자 게시물 오피니언 마이닝

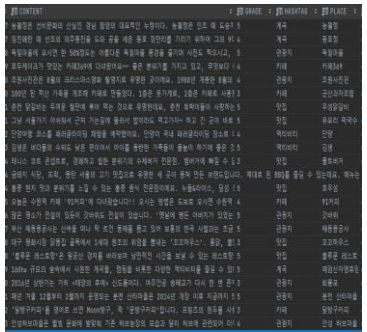
01 데이터 수집

02 데이터 정제

03 감성분석 모델

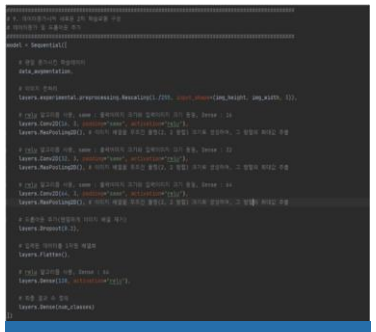
04 분석모델 평가

05 감성예측



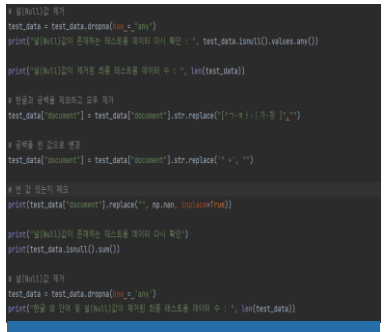
사용자 작성 게시물

사용자 작성 게시물을 통해
각 지역별 여행지 정보 수집



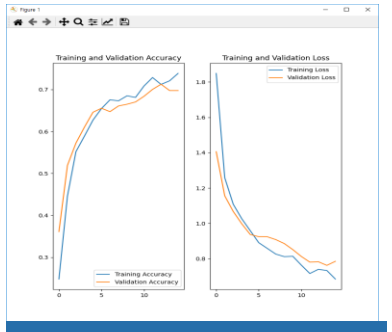
데이터 전처리

결측 값 제거
분석 불가능 문자 제거
Okt 형태소 분석



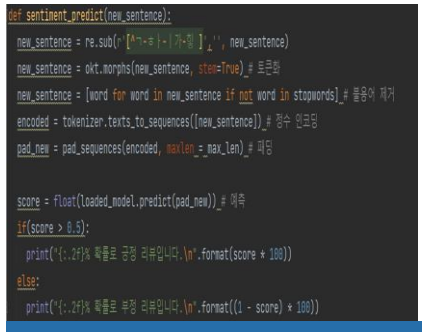
학습모델 생성

LSTM 분석 모델
최적 변수 도출
최적 모델 학습



분석모델 확인

분석모델 확인
감성예측



```
def sentiment_predict(new_sentence):
  new_sentence = re.sub(r'[^ㄱ-ㅎㅏ-ㅣ가-힣 ]','', new_sentence)
  new_sentence = okt.morphs(new_sentence, stem=True) # 토큰화
  new_sentence = [word for word in new_sentence if not word in stopwords] # 불용어 제거
  encoded = tokenizer.texts_to_sequences([new_sentence]) # 정수 인코딩
  pad_new = pad_sequences(encoded, maxlen = max_len) # 패딩

  score = float(loaded_model.predict(pad_new)) # 예측
  if(score > 0.5):
    print("{:.2f}% 확률로 긍정 리뷰입니다.\n".format(score * 100))
  else:
    print("{:.2f}% 확률로 부정 리뷰입니다.\n".format((1 - score) * 100))
```

사용자 데이터 감성 분석

결과 예측

Main work 데이터 정제 및 token 생성

```python
# document의 리뷰 중복인 내용이 있다면 중복 제거
train_data.drop_duplicates(subset=["document"], inplace=True)

print("중복 제거된 최종 학습용 데이터 수 : ", len(train_data))

# 라벨 값들의 리뷰의 수 확인
print(train_data.groupby("label").size().reset_index(name = "count"))

# 널(Null)값이 존재하는 학습용 데이터 확인
print("널(Null)값이 존재하는 학습용 데이터 확인 : ", train_data.isnull().values.any())

print("널(Null)값이 존재하는 학습용 데이터 수")
print(train_data.isnull().sum())

print("널(Null)값인 데이터 확인")
print(train_data.loc[train_data.document.isnull()])

# 널(Null)값 제거
train_data = train_data.dropna(how = "any")
print("널(Null)값이 존재하는 학습용 데이터 다시 확인 : ", train_data.isnull().values.any())

print("널(Null)값이 제거된 최종 학습용 데이터 수 : ", len(train_data))

# 한글과 공백을 제외하고 모두 제거
train_data["document"] = train_data["document"].str.replace("[^ㄱ-ㅎㅏ-ㅣ가-힣 ]","")

# 공백을 빈 값으로 변경
train_data["document"] = train_data["document"].str.replace('^ +', "")

# 널(Null)값 제거
train_data = train_data.dropna(how = 'any')
print("한글 외 단어 및 널(Null)값이 제거된 최종 학습용 데이터 수 : ", len(train_data))
```

```python
#불용어 사전
stopwords = ['의','가','이','은','들','는','좀','잘','강','과','도','를','으로','자','에','와','한','하다']

# 형태소 분석기 사용함
okt = Okt()

# 형태소 분석을 통한 학습용 데이터의 단어 추출하기
X_train = []
for sentence in tqdm(train_data["document"]):
    tokenized_sentence = okt.morphs(sentence, stem=True) # 토큰화
    stopwords_removed_sentence = [word for word in tokenized_sentence if not word in stopwords] # 불용어 제거
    X_train.append(stopwords_removed_sentence)

# 형태소 분석을 통한 테스트용 데이터의 단어 추출하기
X_test = []
for sentence in tqdm(test_data['document']):
    tokenized_sentence = okt.morphs(sentence, stem=True) # 토큰화
    stopwords_removed_sentence = [word for word in tokenized_sentence if not word in stopwords] # 불용어 제거
    X_test.append(stopwords_removed_sentence)

# 토근에 학습용 단어 저장하기
tokenizer = Tokenizer()
tokenizer.fit_on_texts(X_train)
```

Project 03. 자연어 처리기반 사용자 여행지 추천 서비스

Main work 빈도수 낮은 단어 정제 및 LSTM 알고리즘을 활용한 모델 생성

```
# 단어와 빈도수의 쌍(pair)을 key와 value로 받는다.
for key, value in tokenizer.word_counts.items():
    total_freq = total_freq + value

    # 단어의 등장 빈도수가 threshold보다 작으면
    if(value < threshold):
        rare_cnt = rare_cnt + 1
        rare_freq = rare_freq + value

print('단어 집합(vocabulary)의 크기 :',total_cnt)
print('등장 빈도가 %s번 이하인 희귀 단어의 수: %s'%(threshold - 1, rare_cnt))
print("단어 집합에서 희귀 단어의 비율:", (rare_cnt / total_cnt)*100)
print("전체 등장 빈도에서 희귀 단어 등장 빈도 비율:", (rare_freq / total_freq)*100)

# 전체 단어 개수 중 빈도수 2이하인 단어는 제거.
# 0번 패딩 토큰을 고려하여 + 1
vocab_size = total_cnt - rare_cnt + 1
print('단어 집합의 크기 :',vocab_size)

tokenizer = Tokenizer(vocab_size)
tokenizer.fit_on_texts(X_train)
X_train = tokenizer.texts_to_sequences(X_train)
X_test = tokenizer.texts_to_sequences(X_test)

# 단어별 인덱스를 파일로 저장
with open("model/naver_movie_tokenizer.pickle", "wb") as handle:
    pickle.dump(tokenizer, handle)

y_train = np.array(train_data["label"])
y_test = np.array(test_data["label"])

drop_train = [index for index, sentence in enumerate(X_train) if len(sentence) < 1]

X_train = np.delete(X_train, drop_train, axis=0)
y_train = np.delete(y_train, drop_train, axis=0)
```

```
def below_threshold_len(max_len, nested_list):
  count = 0
  for sentence in nested_list:
    if(len(sentence) <= max_len):
        count = count + 1
  print("전체 샘플 중 길이가 %s 이하인 샘플의 비율: %s" %(max_len, (count / len(nested_list))*100))

max_len = 30
below_threshold_len(max_len, X_train)

X_train = pad_sequences(X_train, maxlen = max_len)
X_test = pad_sequences(X_test, maxlen = max_len)

embedding_dim = 100
hidden_units = 128

model = Sequential()
model.add(Embedding(vocab_size, embedding_dim))
model.add(LSTM(hidden_units))
model.add(Dense(1, activation="sigmoid"))

es = EarlyStopping(monitor="val_loss", mode="min", verbose=1, patience=4)
mc = ModelCheckpoint("naver_movie.h5", monitor="val_acc", mode="max", verbose=1, save_best_only=True)

model.compile(optimizer="rmsprop", loss="binary_crossentropy", metrics=["acc"])
history = model.fit(X_train, y_train, epochs=15, callbacks=[es, mc], batch_size=64, validation_split=0.2)

loaded_model = load_model("model/naver_movie.h5")
print("\n 테스트 정확도: %.4f" % (loaded_model.evaluate(X_test, y_test)[1]))
```

Project 04.

머신러닝을 활용한 분리배출 방법과 자연어처리를 활용한 환경 트렌드 분석

About project

개방형 클라우드 플랫폼 Paas-Ta 공모전

팀원

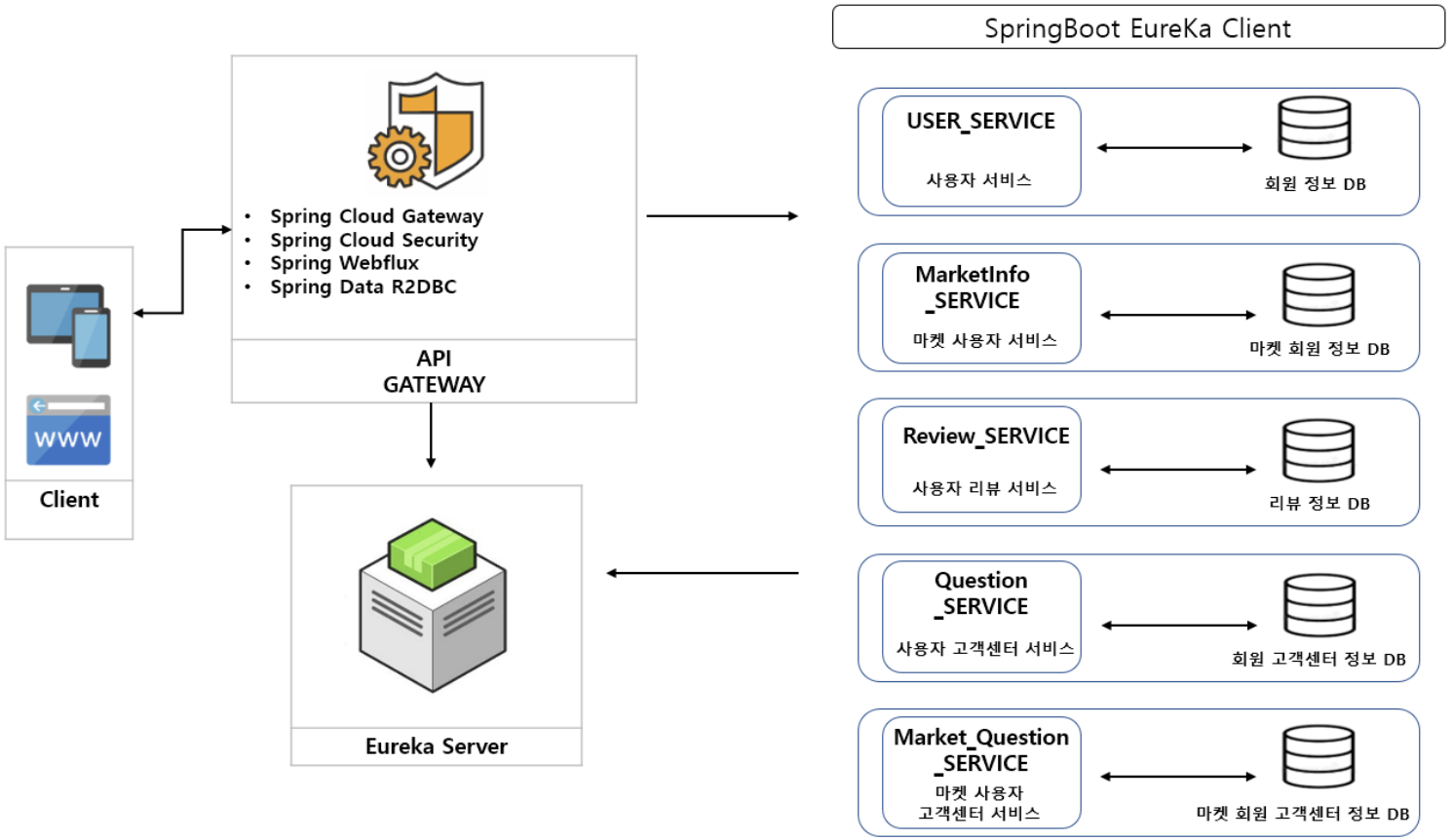
수상 : 한국지능정보사회진흥원장상(NIA) 특별상

Project 04. 머신러닝을 활용한 분리배출 방법과 자연어처리를 활용한 환경 트렌드 분석

Introduce project

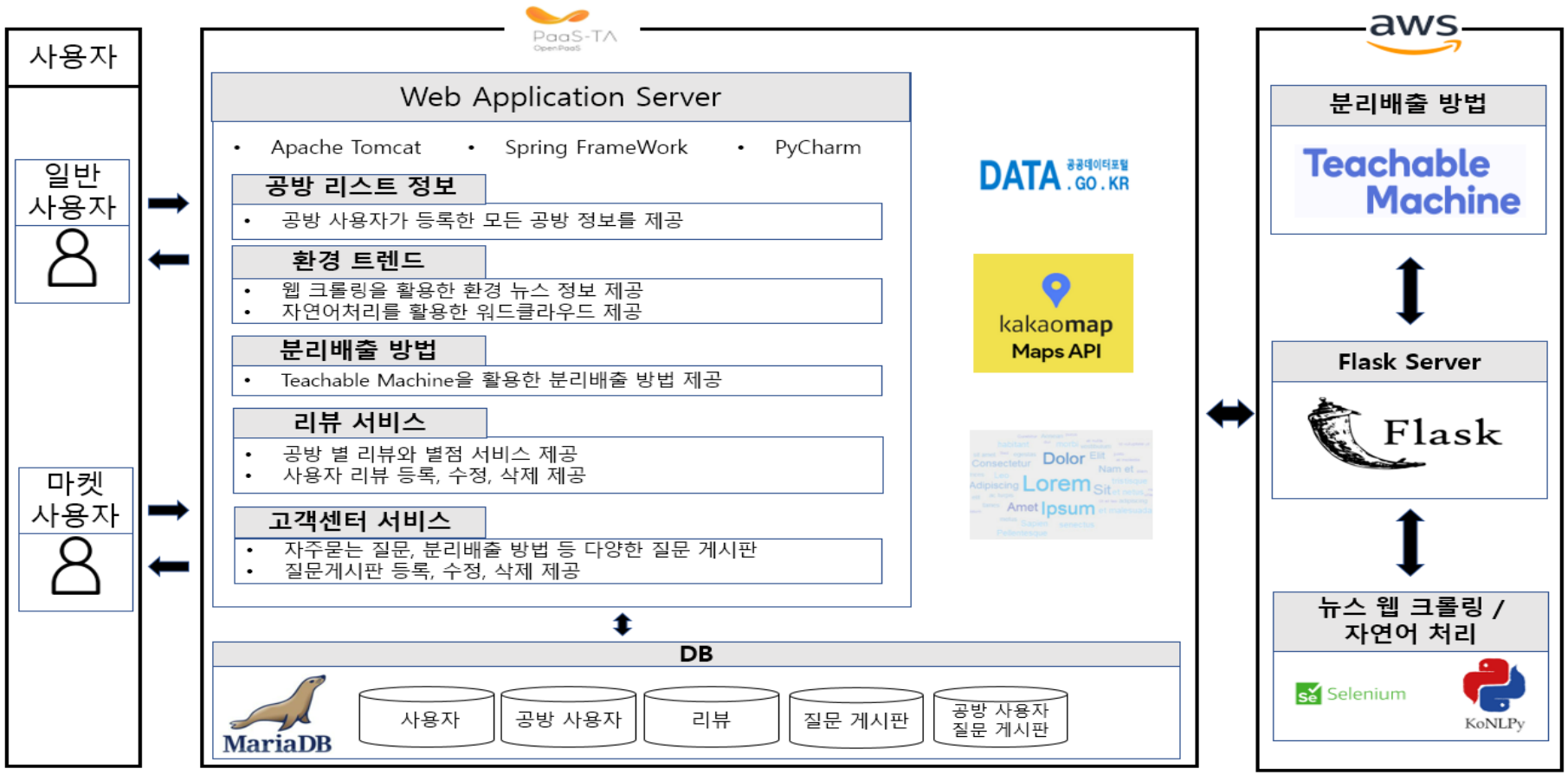
작업 기간	2022.09~2022.12
인력 구성(기여도)	팀원
프로젝트 목적	개방형 클라우드 플랫폼 Paas-Ta 공모전
프로젝트 내용	'다시, 새롭게' 라는 뜻으로 쓰레기에서 새로운 작품으로 태어나는 (업사이클링) 공방을 소개하는 서비스 - 머신러닝을 활용한 분리배출 방법과 자연어 처리를 활용한 환경 트렌드 분석-
주요 업무 및 상세 역할	1) Spring Cloud Gateway 적용 2) Spring Security 적용 3) 머신러닝 학습도구인 구글의 Teachable Machine을 이용하여 다양한 재활용품들의 이미지를 학습하여 학습 모델 생성 4) 생성된 학습 모델을 기반으로 재활용품 인식 및 인식된 재활용품에 대한 분리배출 방법 추천 5) Flask를 통한 학습 모델 서빙 및 REST 기반 인공지능 서비스 구현 6) 실시간 데이터 수집을 위해 웹 크롤링을 활용하여 사용자에 환경 뉴스 등 다양한 정보 제공 7) 최근 환경 정보를 기반으로 전처리 및 자연어처리를 수행하여 최근 트랜드를 이해하기 쉽게 워드클라우드를 활용하여 데이터 시각화 제공 8) 업사이클링 공방을 운영하는 사업자 등록을 위해 사업자등록 정보 진위확인 및 상태조회서비스 Open API 적용
사용언어 및 개발 환경	JAVA, Java Script, HTML5, CSS, Python, MSA, Spring Boot Framework, Paas

Main work MSA 서비스 기능도



Project 04. 머신러닝을 활용한 분리배출 방법과 자연어처리를 활용한 환경 트렌드 분석

Main work 작품 구성도



Project 04. 머신러닝을 활용한 분리배출 방법과 자연어처리를 활용한 환경 트렌드 분석

Main work Spring Cloud Gateway / Spring Security 적용

```
@Bean
public RouteLocator gatewayRoutes(RouteLocatorBuilder builder) {
    return builder.routes()
            .route(r -> r.path("/user/**") // 일반회원
                    // 라우터 등록
                    .filters(
                            // URL별 독립적으로 저장 항목을 추가할 경우 정의함
                            f -> f.addRequestHeader( headerName: "user-request",  headerValue: "From API Gateway!!")
                                    .addResponseHeader( headerName: "user-response",  headerValue: "From API Gateway!!")

                    )
                    .uri("http://localhost:12000") // 연결될 서버 주소

            ).route(r -> r.path("/jwtUser/**") // 일반회원
                    // 라우터 등록
                    .filters(
                            // URL별 독립적으로 저장 항목을 추가할 경우 정의함
                            f -> f.addRequestHeader( headerName: "user-request",  headerValue: "From API Gateway!!")
                                    .addResponseHeader( headerName: "user-response",  headerValue: "From API Gateway!!")

                    )
                    .uri("http://localhost:12000") // 연결될 서버 주소

            ).route(r -> r.path("/market/**") // 사업자
                    // 라우터 등록
                    .filters(
                            // URL별 독립적으로 저장 항목을 추가할 경우 정의함
                            f -> f.addRequestHeader( headerName: "market-request",  headerValue: "From API Gateway!!")
                                    .addResponseHeader( headerName: "market-response",  headerValue: "From API Gateway!!")
                    )
                    .uri("http://localhost:13000") // 연결될 서버 주소

            ).route(r -> r.path("/auth/**") // 사업자
                    // 라우터 등록
                    .filters(
                            // URL별 독립적으로 저장 항목을 추가할 경우 정의함
                            f -> f.addRequestHeader( headerName: "market-request",  headerValue: "From API Gateway!!")
                                    .addResponseHeader( headerName: "market-response",  headerValue: "From API Gateway!!")
```

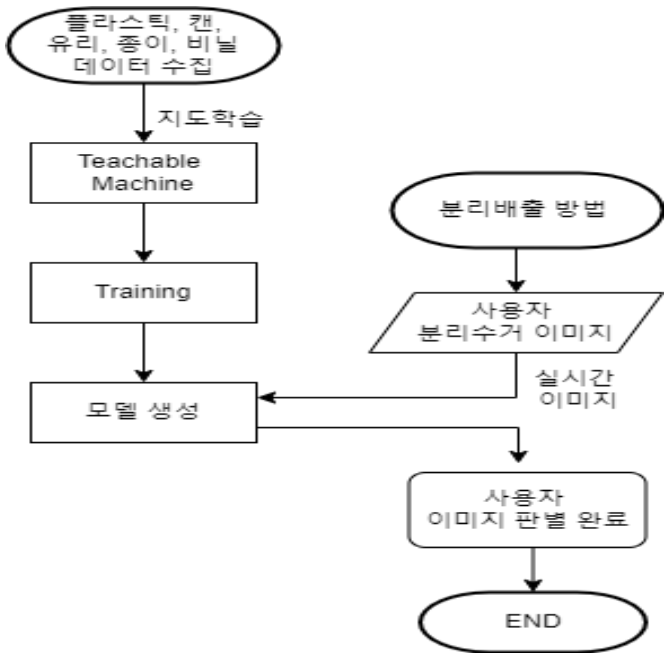
```
//isPresent() 함수는 Optional객체의 값이 존재하는지 체크
if (uEntity != null) {
    res = 2;
} else {

    //비밀번호 해시 알고리즘 암호화 저장
    uDTO.setUserPwd(passwordEncoder.encode(userPwd));

    //이메일 암호화
    uDTO.setUserEmail(EncryptUtil.encAES128CBC(userEmail));

    // 회원가입을 위해 Entity 생성 //반드시 Builder
    UserEntity sEntity = UserEntity.builder()
            .userId(uDTO.getUserId()).userName(uDTO.getUserName()).userPwd(uDTO.getUserPwd()).userEmail(uDTO.getUserEmail())
            .phoneNumber(uDTO.getPhoneNumber()).addr1(uDTO.getAddr1()).addr2(uDTO.getAddr2())
            .reg_id(uDTO.getUserId()).reg_dt(DateUtil.getDateTime( fm: "yyyy-MM-dd hh:mm:ss"))
            .chg_id(uDTO.getUserId()).chg_dt(DateUtil.getDateTime( fm: "yyyy-MM-dd hh:mm:ss"))
            .roles("ROLE_USER")
            .build();


    //DB에 저장
    userRepository.save(sEntity);
```

- API Gateway 구현을 위해 논 블로킹 기반 Spring Cloud Gateway 적용
- 각 서비스들의 접근 제어는 Spring Cloud Security 적용

- 회원가입 및 로그인 인증을 위해 Spring Security 적용
- 이전에는 SHA-256과 같은 단방향 해시를 적용해 비밀번호를 저장했지만 쉽게 해독될 수 있기 때문에 Spring Security는 Bcrypt 해시 함수 적용

Project 04. 머신러닝을 활용한 분리배출 방법과 자연어처리를 활용한 환경 트렌드 분석

Main work Teachable Machine을 이용한 분리수거 배출 방법




```
const URL = "https://teachablemachine.withgoogle.com/models/nvXPJOfT7/";

let model, webcam, labelContainer, maxPredictions;

// Load the image model and setup the webcam
async function init() { //init으로 모델을 실행
    const modelURL = URL + "model.json";
    const metadataURL = URL + "metadata.json";



    // load the model and metadata
    // Refer to tmImage.loadFromFiles() in the API to support files from a file picker
    // or files from your local hard drive
    // Note: the pose library adds "tmImage" object to your window (window.tmImage)
    model = await tmImage.load(modelURL, metadataURL);  //모델이 저장된 파일을 가져와서 load하고 model에 저장해줌
    maxPredictions = model.getTotalClasses();
    labelContainer = document.getElementById( elementId: "label-container");
}

//지정된 정보에 대해 찾을 때 그 설명에 대해 찾을 수 있도록 나오게 하기
// run the webcam image through the image model
async function predict() {
    let image = document.getElementById( elementId: "input-image") //id를 통해서 이미지를 가져오기
    console.log("들어온 파일은" + image)
    const prediction = await model.predict(image, false); //예측하기 위한 이미지를 넣음 두번째 인자는 flipped로 뒤집혔는지 안뒤집혔는지 확인

    if (prediction[0].className == "플라스틱" && prediction[0].probability.toFixed( fractionDigits: 2) == 1.00) {
```

- **Teachable Machine 학습**

- 분리배출 방법 기능을 구현하기 위해 플라스틱, 캔, 유리, 종이, 비닐로 분류해서 데이터 수집

- Teachable Machine 학습을 바탕으로 분리수거 배출 기능 학습 모델 생성

- **Teachable Machine 사용**

- 사용자는 실시간으로 분리수거 이미지를 올림

- 학습 모델을 통해 실시간 사용자 이미지 판별

Project 04. 머신러닝을 활용한 분리배출 방법과 자연어처리를 활용한 환경 트렌드 분석

Main work 파이썬을 활용한 실시간 뉴스 크롤링 및 자연어 처리

```python
Options = Options()
def funcNewsList():
    # 봇 차단을 막기 위한 우회
    Options.add_argument("user-agent=Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko")
    # 실행했을때 웹 브라우저를 띄우지 않는 headless chrome 옵션 적용
    # Options.add_argument("headless")

    # 크롬 드라이버 경로
    driver = webdriver.Chrome("webdriver/chromedriver.exe")

    result = []

    naverUrl = "https://news.naver.com/main/list.naver?mode=LS2D&mid=shm&sid1=102&sid2=252"

    driver.get(naverUrl)

    idx = 0
    while True:
        try:
            top_area = driver.find_element_by_xpath(
                '//*[@id="main_content"]/div[2]/ul[1]/li[' + str(idx + 1) + ']/dl/dt[2]/a')
            # 제목 가져오기
            top_title = top_area.text
            # url 가져오기
            top_url = top_area.get_attribute('href')
            # 타이틀 클릭해서 들어가기
            driver.find_element_by_xpath(
                '//*[@id="main_content"]/div[2]/ul[1]/li[' + str(idx + 1) + ']/dl/dt[2]/a').click()
            # 클릭해서 들어간 후에 본문 가져오기
            top_content = driver.find_element_by_id('dic_area').text
```

```python
    #Okt 객체 선언
okt = Okt()
def nlp_data(res):
    # data =[{"title": "서울시 \"경유차 운행제한 확대¨내연차 단계적 퇴출\"", "url": "https://n.news.naver.com/mnews/article/422/0
    rea ="" #뉴스 기사 본문 모아둘 곳
    dict_data=[]
    #뉴스 기사 본문만 뽑기
    for dict_data in res:
        rea += dict_data["content"]

    word_list=[]
    #okt 사용해서 기사 본문 명사만 뽑기
    j = okt.nouns(rea)
    #불용어 제거
    stopwords =['으로','하다','기자','위해','내년','뉴스','결과','발표','자료','제시','설명','대한','최고','기사','이번','대해','통해',
    j=[word for word in j if not word in stopwords]

    # 한글자인 명사 제외하기
    for i in range(len(j)):
        if len(j[i]) > 1:
            word_list.append(j[i])



    #불용어, 한글자 명사 제외 후 남은 명사 , 많이 사용된 30의 명사 추출하기
    count=Counter(word_list).most_common(n=30)
    print(count)
    #last_word에 튜플에서 명사만 뽑아서 넣기
    last_word=[x[0] for x in count]
```

- **파이썬을 활용한 실시간 뉴스 크롤링**

- 크롤링 한 데이터를 JSONArray 타입으로 변환 후 URL에 담아 Flask Server로 전송

- Flask Server에서 Python 라이브러리인 KoNLPy에 크롤링 한 데이터 전달

- **Okt 라이브러리를 사용한 형태소 분석**

- KoNLPy API는 전달받은 데이터를 받아 Okt 라이브러리를 사용하여 형태소 분석을 진행

- 품사 태깅이 완료되면 Spring으로 전달 후 JSON 데이터 파싱

Project 04. 머신러닝을 활용한 분리배출 방법과 자연어처리를 활용한 환경 트렌드 분석

Main work JqCloud 워드클라우드

네이버 환경뉴스	환경부 보도
종로구, 길고양이 서식지에 따뜻한 겨울집 마련	버려지거나 방치된 야생동물, 국립생태원에서 새로운 삶
정부 "에너지 위기에도 올겨울 전력 공급은 안정적"	일회용컵 보증금제 참여, 친환경 노력을 지원한다

```
<!--jqcloud-->
<link rel="stylesheet" type="text/css" href="/assets/jqcloud/jqcloud.css"/>
<script src="/assets/js/jquery-3.6.0.js"></script>
<script type="text/javascript" src="/assets/jqcloud/jqcloud.js"></script>
<script type="text/javascript">

let words = [
<% for(int i =0; i<30; i++){%>
{text: '<%=nlpDTOList.get(i).getWord()%>', weight:<%=30 - i%>},
<%}%>

];

console.log(words);

$(function () {
$("#go").jQCloud(words);
});

</script>
```

- **JqCloud를 활용한 실시간 환경 트렌드 분석**
- 크롤링한 뉴스에서 형태소분석을 통해 많이 사용된 명사 30개 추출
- JqCloud를 통해 실시간 환경 인기 단어를 시각적으로 제공

Main work 사업자등록정보 진위확인 및 상태조회 서비스 API

```
function marketCheck() {
    let b_no = document.getElementById("b_no").value;
    let start_dt = document.getElementById("start_dt").value;
    let p_nm = document.getElementById("p_nm").value;

    let data = {
        "businesses": [
            {
                "b_no": b_no,
                "start_dt": start_dt,
                "p_nm": p_nm,
            }
        ]
    };
    $.ajax({
        url: "http://api.odcloud.kr/api/nts-businessman/v1/validate?serviceKey=ióxokwKsdvlwERvOibTsILóz%2Fx7
        type: "POST",
        data: JSON.stringify(data), // json 을 string으로 변환하여 전송
        dataType: "JSON",
        contentType: "application/json",
        accept: "application/json",
        success: function (result) {
            console.log(result.data[0].valid);
            if(result.data[0].valid =='01') {
                swal('정상적인 사업자입니다.', "확인눌러 회원가입을 진행해주세요.", 'success')
                    .then(function(){
                        location.href="/market/MarketRegForm";
                    });
            }else {
                swal('사업자를 다시 확인해주세요', "확인눌러 다시 진행해주세요", 'error')

            }},


        error: function (result) {
            console.log(result.responseText); //responseText의 에러메세지 확인
            if(result.data[0].valid =='02'){
```

- **사업자 등록 정보 진위확인 및 상태조회**
- 국세청 API를 활용하여 사업자 확인 및 중복 제거

Main work 지도 API를 활용한 공방 위치 정보 제공

```
var map = new daum.maps.Map(mapContainer, mapOption);

var geocoder = new daum.maps.services.Geocoder();
var listData = [

  '<%=CmmUtil.nvl(rDTO.getAddr1())%>',

];

var listname = [

  '<%=CmmUtil.nvl(rDTO.getMarketName())%>',

];
let coords;
listData.forEach(function(addr, index) {
  geocoder.addressSearch(addr, function(result, status) {
    if (status === daum.maps.services.Status.OK) {
      coords = new daum.maps.LatLng(result[0].y, result[0].x);

      var marker = new daum.maps.Marker({
        map: map,
        position: coords
      });
      var infowindow = new daum.maps.InfoWindow({
        content: '<div style="width:150px;text-align:center;padding:6px 0;">' + listname[index] + '</div>',
        disableAutoPan: true
      });
      infowindow.open(map, marker);
      map.setCenter(coords);
    }
  });
});

//-----------------------------------------
```

- **Kakao Map을 활용한 공방 위치 정보 제공**

- 공방 주소 위도, 경도 변환 후 지도 제공

Project 05.

국가위기 데이터 활용 파이프라인 실증

About project

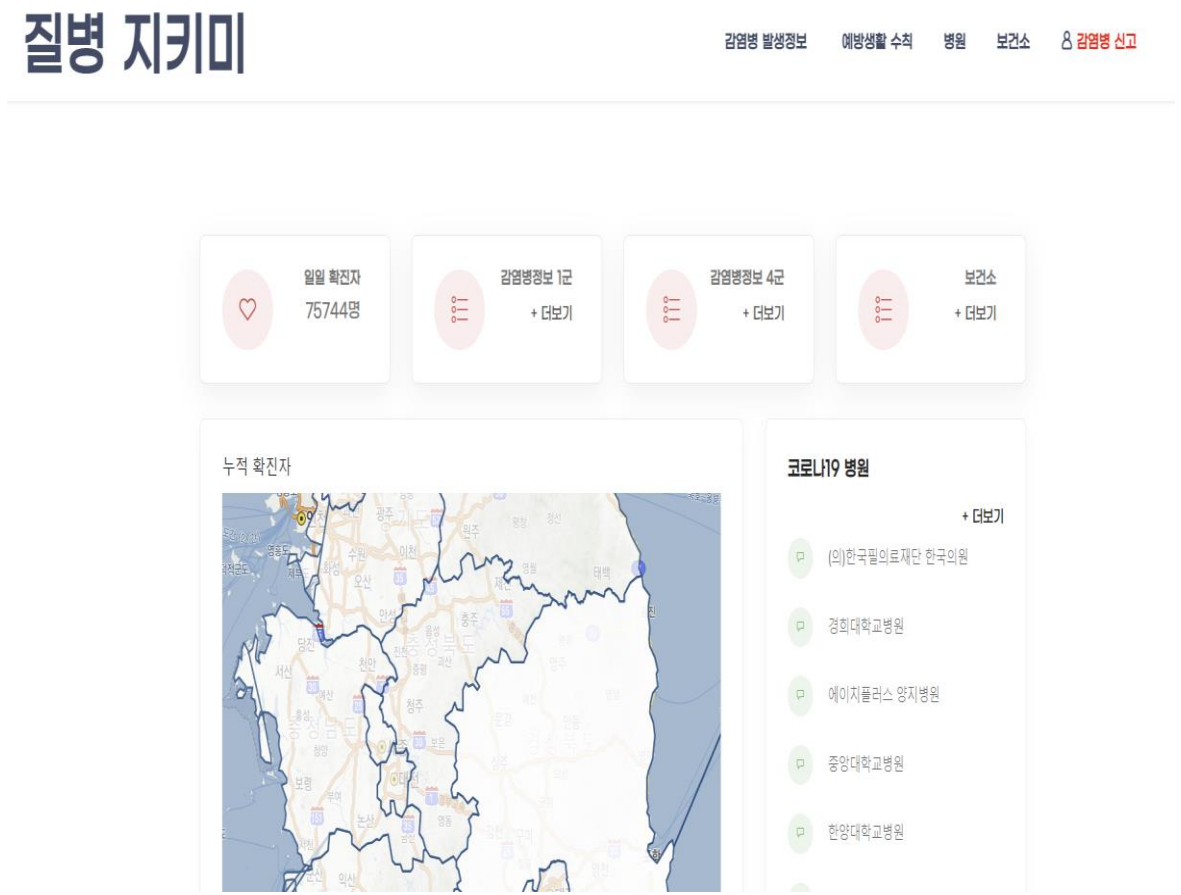
2022년 Nia 민간협력기반 위기대응 프로젝트 참여

개인

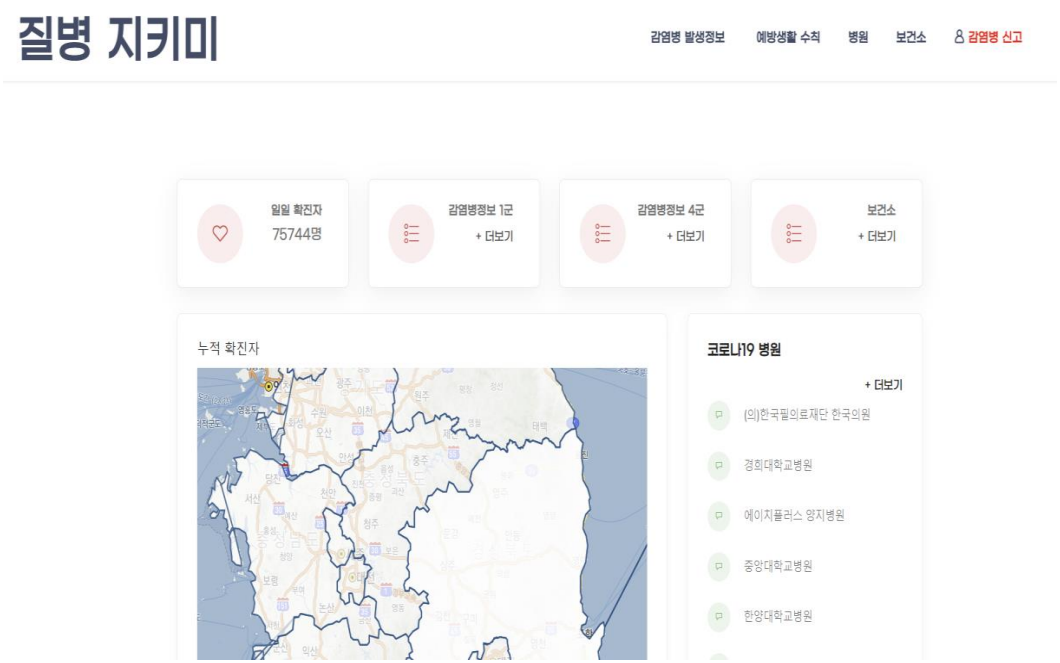
Project 05. 국가위기 데이터 활용 파이프라인 실증

Introduce project

작업 기간	2022.12.19~2022.12.23
인력 구성(기여도)	개인
프로젝트 목적	국가위기 데이터 활용 파이프라인 실증 프로젝트 참여
프로젝트 내용	1) 국가위기 발생 시 원활한 데이터 활용 지원을 위한 데이터 파이프라인에 대해 위기대응 시나리오 기반으로 검증, 정비 추진 2) 시빅해커 및 개발자 커뮤니티 등 개발자를 참여시켜 위기대응 시나리오 주제에 관련된 대국민 서비스 개발 및 의견 수렴
주요 업무 및 상세 역할	1) 코로나 일일 확진자 현황 제공 2) 카카오 맵을 통한 지역별 코로나 누적 확진자 현황 제공 3) 감염병 1.4군 기간, 지역별로 발생 수, 발생률, 사망 수 현황제공 4) 예방생활 수칙 제공 5) 코로나 19 병원 정보 제공 6) 코로나 19 병원 길찾기 제공 7) 보건소 정보 제공 8) 보건소 길찾기 제공 9) 감염병 신고 제공
사용언어 및 개발 환경	JAVA, Java Script, HTML5, CSS, Spring Boot Framework



Main work API를 활용한 코로나 발생 현황




```java
1 usage  huhjihye
@Override
public String getCovidNum() throws ParserConfigurationException, IOException, SAXException {
    log.info(this.getClass().getName() + ".getCovidNum Start !!");
    String url = "getCovid19CurrentStatusConfirmationsJson";
    String []keyArr = {"numOfRows","pageNo"};
    String []valueArr = {"100", "1"};

    String todayCovidNum = "";
    try {
        JSONObject jsonObject = PasingUtil.jsonPasing(url, keyArr, valueArr);
        jsonObject =(JSONObject)jsonObject.get("response");
        JSONArray jsonArray =(JSONArray)jsonObject.get("result");
        jsonObject = (JSONObject) jsonArray.get(0);
        todayCovidNum = jsonObject.get("cnt7").toString();
    } catch (NullPointerException e) {
        todayCovidNum = "0";
    }
    log.info(this.getClass().getName() + ".getCovidNum End !!");
    return todayCovidNum;
}
```

- **질병관리청_코로나19_국내 발생현황 조회 API**
  - 코로나 일일확진자 수 제공
- **보건복지부_코로나19 시도 발생현황 API**
  - API와 카카오 맵을 활용해 전국 단위 코로나 누적확진자 수 제공

Project 05. 국가위기 데이터 활용 파이프라인 실증

Main work API를 활용한 감염병 발생 수 및 사망자 수 현황

	발생 수	발생률	사망수
콜레라	0	0%	0명
장티푸스	3	0.2%	0명
파라티푸스	2	0.14%	0명
세균성이질	3	0.2%	0명
장출혈성대장균감염증	14	0.95%	0명
A형간염	82	5.59%	0명

```
1 usage  huhjihye
@Override
public List<No1IftDTO> getNo1Ift(String year, String dsvd) throws ParserConfigurationException, IOException, SAXException {
    log.info("start!!");
    String url = "getCallStat01Api";
    String []keyArr = {"numOfRows","pageNo", "year", "dvsd"};
    String []valueArr = {"500", "1", year, URLEncoder.encode(dsvd, enc: "UTF-8")};

    Document document = PasingUtil.xmlPasing(url, keyArr, valueArr);
    List<No1IftDTO> no1IftList = AllSetDTOUtil.allSetNo1Ift(document);

    return no1IftList;
}
```

- **보건복지부_보건복지현황_제1군 감염병 발생 수 및 사망자 수 API**
- **보건복지부_보건복지현황_제4군 감염병 발생 수 및 사망자 수 API**

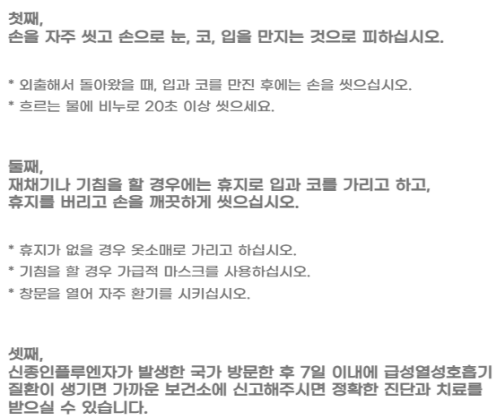
- 조회 연도와 지역을 조건 검색하여 발생 수, 발생률, 사망 수 제공

Project 05. 국가위기 데이터 활용 파이프라인 실증

Main work API를 활용한 감염병 예방 및 관리지침

신종인플루엔자 Q & A

신종인플루엔자 예방수칙



- **질병관리청_신종인플루엔자 A(H1N1) 예방 및 관리지침 API**

- API를 활용해 신종인플루엔자 예방수칙 및 Q & A 제공

Project 05. 국가위기 데이터 활용 파이프라인 실증

Main work API를 활용한 병원 및 보건진료소 제공

병원 조회

기관명	시도명	시군구명	전화번호	구분
(의)한국필의료재단 한국의원	서울	강동구	02-517-1728	코로나검사 실시기관
의료법인 신원의료재단 신원의원	경기	광명시	1899-1510	코로나검사 실시기관
에이치플러스 양지병원	서울	관악구	02-1877-8875	코로나검사 실시기관

```java
huhjihye
@Override
public List<HospitalDTO> getHospital() throws IOException, ParserConfigurationException, SAXException {
    log.info(this.getClass().getName() + " getHospital Start !!");
    String url = "getpubReliefHospList";
    String []keyArr = {"numOfRows","pageNo"};
    String []valueArr = {"500", "1"};

    List<HospitalDTO> hospitalList = new ArrayList<>();

    Document document = PasingUtil.xmlPasing(url, keyArr, valueArr);
    document.getDocumentElement().normalize();
    NodeList nList = document.getElementsByTagName("item");

    for(int temp=0; temp<nList.getLength(); temp++) {
        Node nNode = nList.item(temp);
        Element eElement = (Element) nNode;
        HospitalDTO hospitalDTO = new HospitalDTO();

        hospitalDTO.setType(PasingUtil.getTagValue( tag: "spclAdmTyCd", eElement));
        hospitalDTO.setName(PasingUtil.getTagValue( tag: "yadmNm", eElement));
        hospitalDTO.setLocation(PasingUtil.getTagValue( tag: "sidoNm", eElement));
        hospitalDTO.setPhoneNumber(PasingUtil.getTagValue( tag: "telno", eElement));
        hospitalDTO.setDetailLocation(PasingUtil.getTagValue( tag: "sggu Nm", eElement));

        hospitalList.add(hospitalDTO);
    }

    log.info(this.getClass().getName() + " getHospital End !!");
    return hospitalList;
}
```

- **건강보험심사평가원_코로나19병원정보 API**
- **보건진료소 API**
- **Kakao Map API**
  - 시설 클릭 시 Kakao Map을 활용하여 길찾기 제공

Project 05. 국가위기 데이터 활용 파이프라인 실증

Main work API를 활용한 감염병 신고

- **질병관리청_신종인플루엔자 A(H1N1) 예방 및 관리지침 API**

- 감염병 신고 기능 제공

Project 06.

교통약자를 위한 대중교통 소통 플랫폼

About project

서울시 열린데이터광장 경진대회

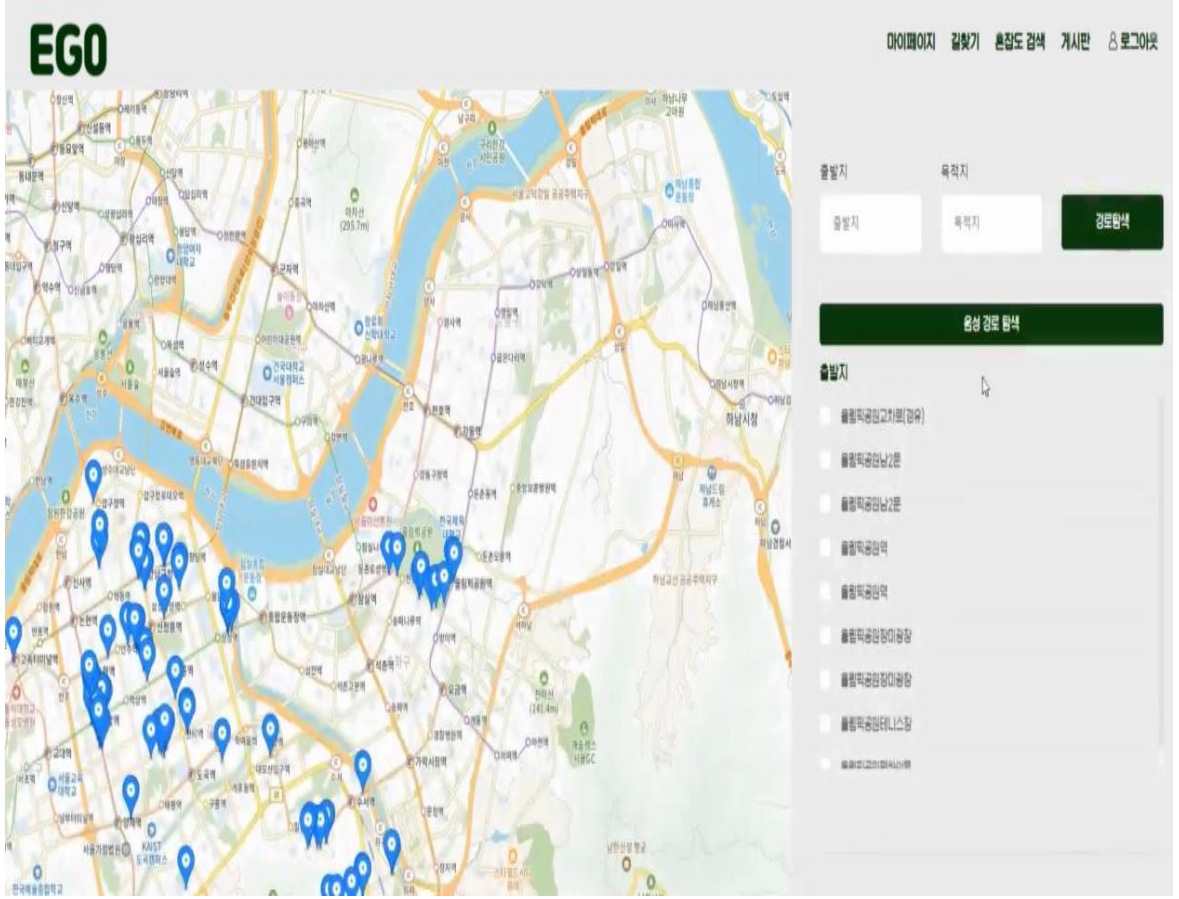
팀장

수상 : 대상 (서울특별시장상)

Project 06. 교통약자를 위한 대중교통 소통 플랫폼

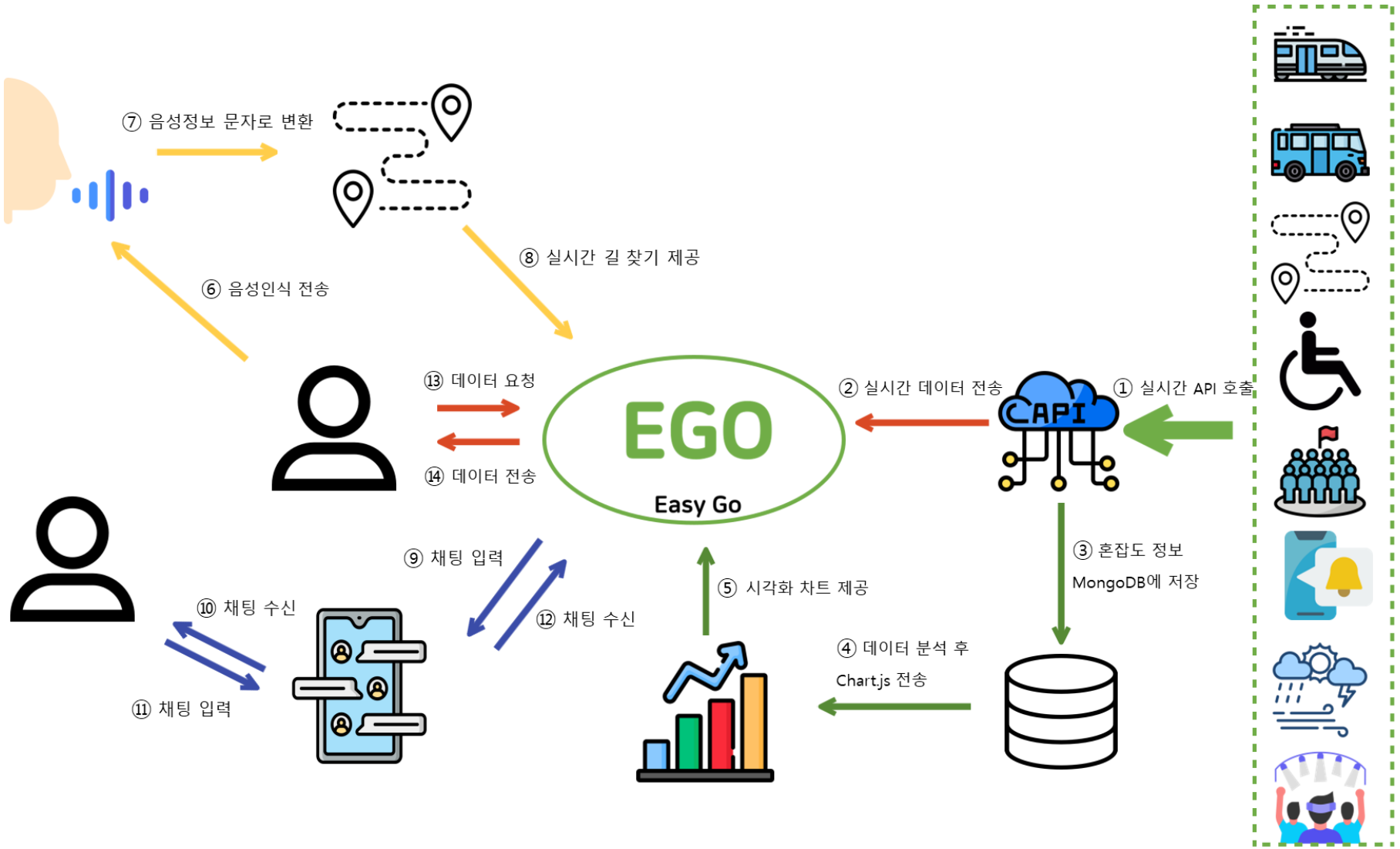
Introduce project

작업 기간	2023.03~2023.06
인력 구성(기여도)	팀장
프로젝트 목적	서울시 열린데이터광장 경진대회
프로젝트 내용	디지털 약자를 위한 '맞춤형' 대중교통 및 이동권 보장 올인원 웹앱
주요 업무 및 상세 역할	1) Spring Cloud Gateway 적용 2) 음성인식 기술 STT를 활용해 실시간 대중교통 길 찾기 제공 3) 기상정보 및 재난 정보, 문화행사 정보 제공 4) 지하철 및 버스 실시간 교통 정보 제공 5) 실시간 혼잡도 제공 및 혼잡도 분석 시각화 서비스 제공 6) 실시간 지역별 채팅 제공
사용언어 및 개발 환경	JAVA, Java Script, HTML5, CSS, MSA, Spring Boot Framework



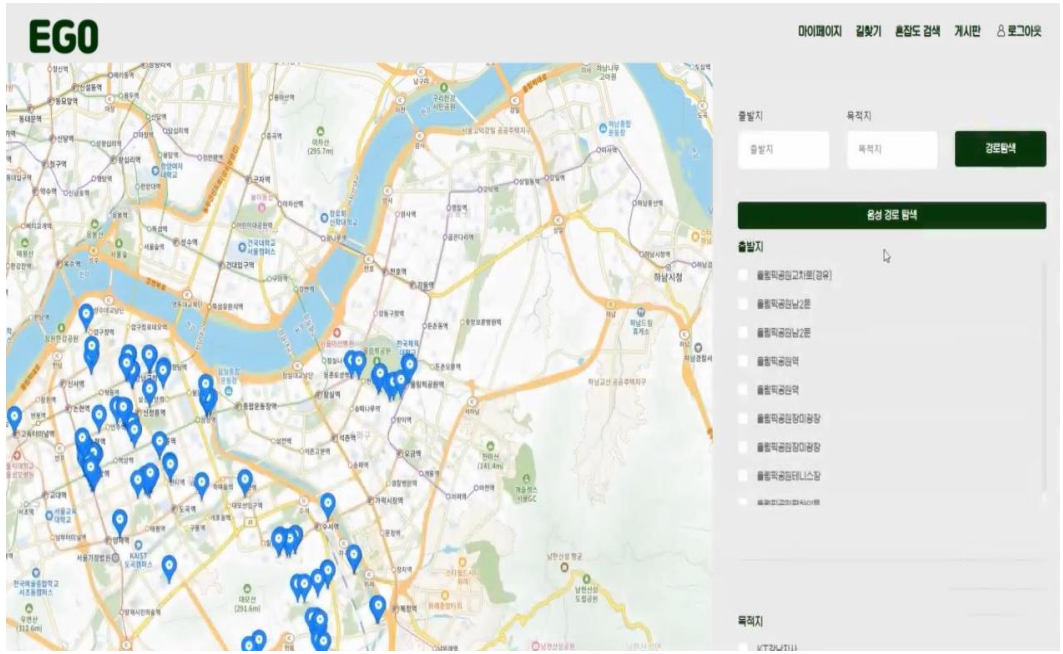
Project 06. 교통약자를 위한 대중교통 소통 플랫폼

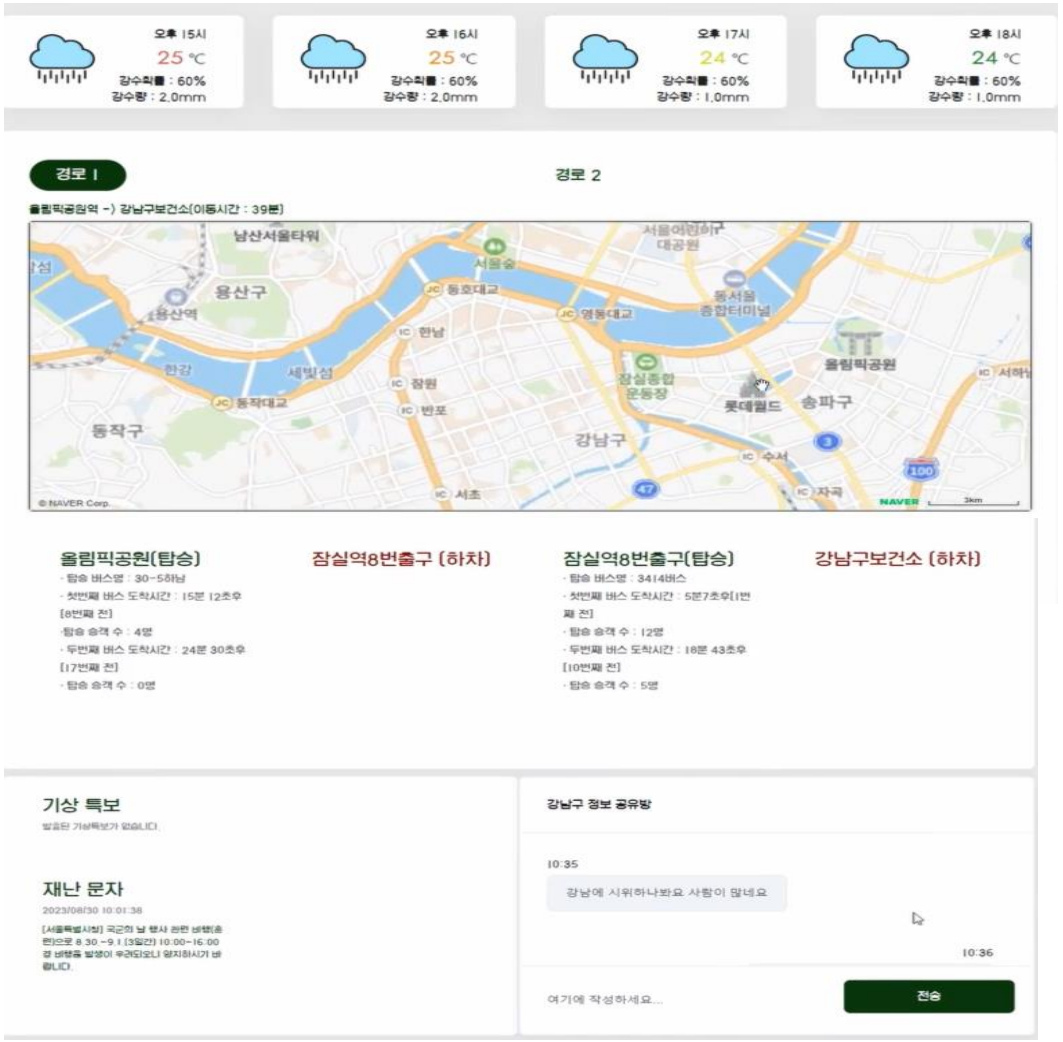
Main work 서비스 흐름도



Project 06. 교통약자를 위한 대중교통 소통 플랫폼

Main work 음성인식 기술 STT를 활용한 실시간 대중교통 길찾기

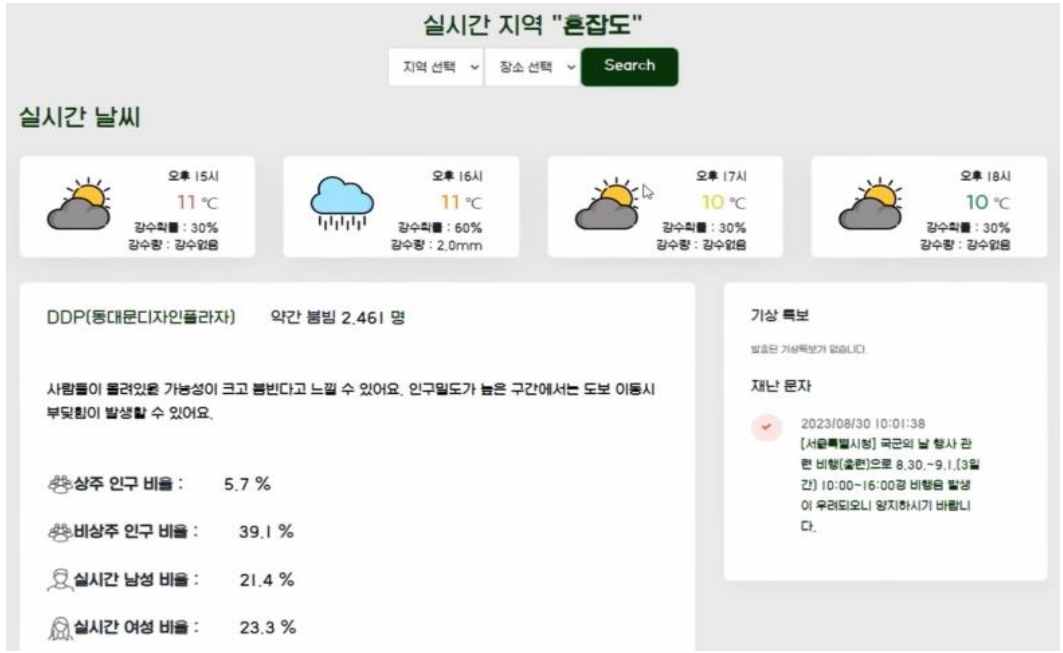




- 안양 라이브러리를 활용한 STT 음성인식 길찾기 제공
- 도착지 날씨, 기상특보, 재난문자, 대중교통 혼잡도 제공
- 실시간 지역별 채팅 제공

Project 06. 교통약자를 위한 대중교통 소통 플랫폼

Main work 실시간 지역 혼잡도 및 혼잡도 예측



- 대용량의 비정형 혼잡도 데이터를 수집 및 처리하기 위해 스프링 배치 모델 설계 및 구현
- 스프링 배치를 통해 수집한 빅데이터 MongoDB에 저재 서비스 설계 및 구현
- 수집된 빅데이터는 데이터 가공 및 처리를 통해 사용자에게 Chart.js를 통해 시각화된 정보로 제공

Project 07.

장애부모를 위한 감정분석 기반 인공지능 이미지 생성을 통한 감정 소통 플랫폼

About project

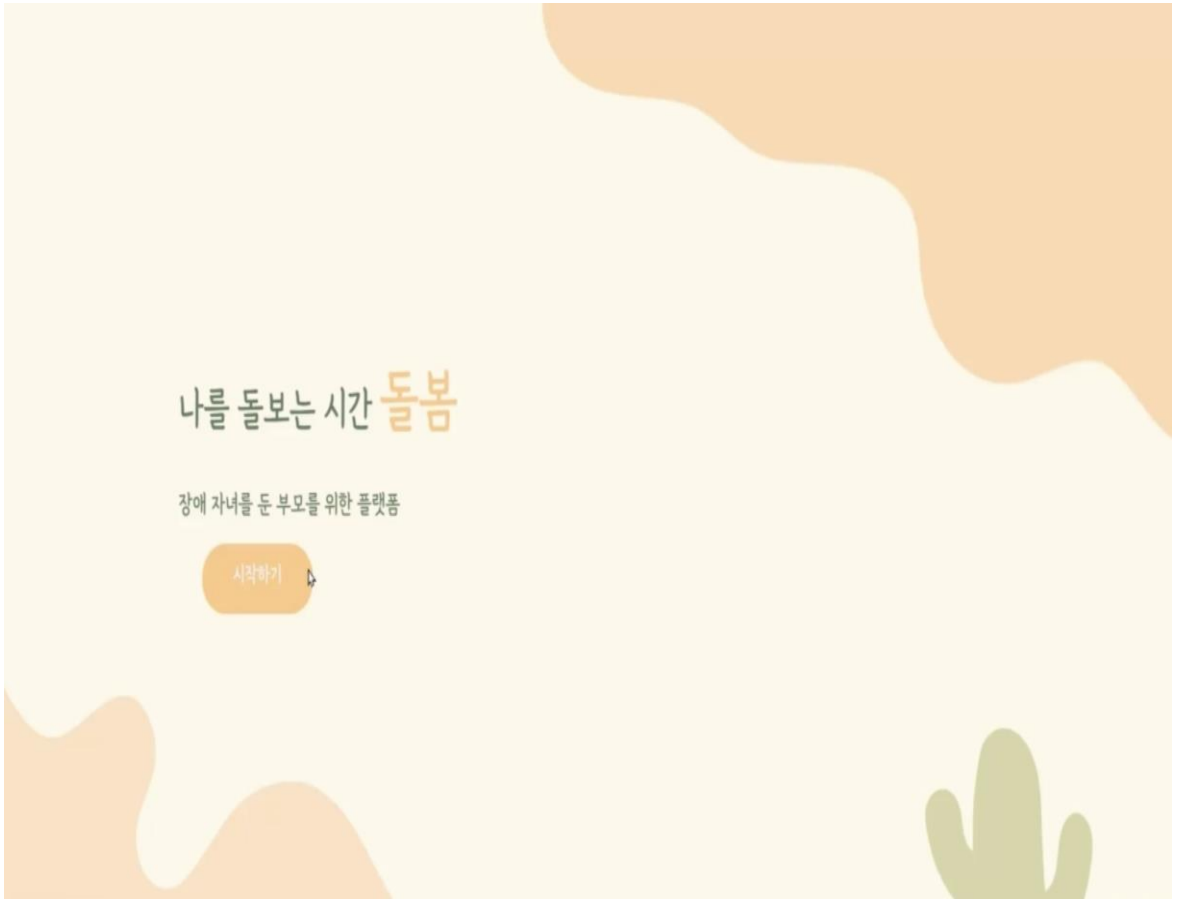
프로보노 ICT 공모전

팀장

Project 07. 장애부모를 위한 감정분석 기반 인공지능 이미지 생성을 통한 감정 소통 플랫폼

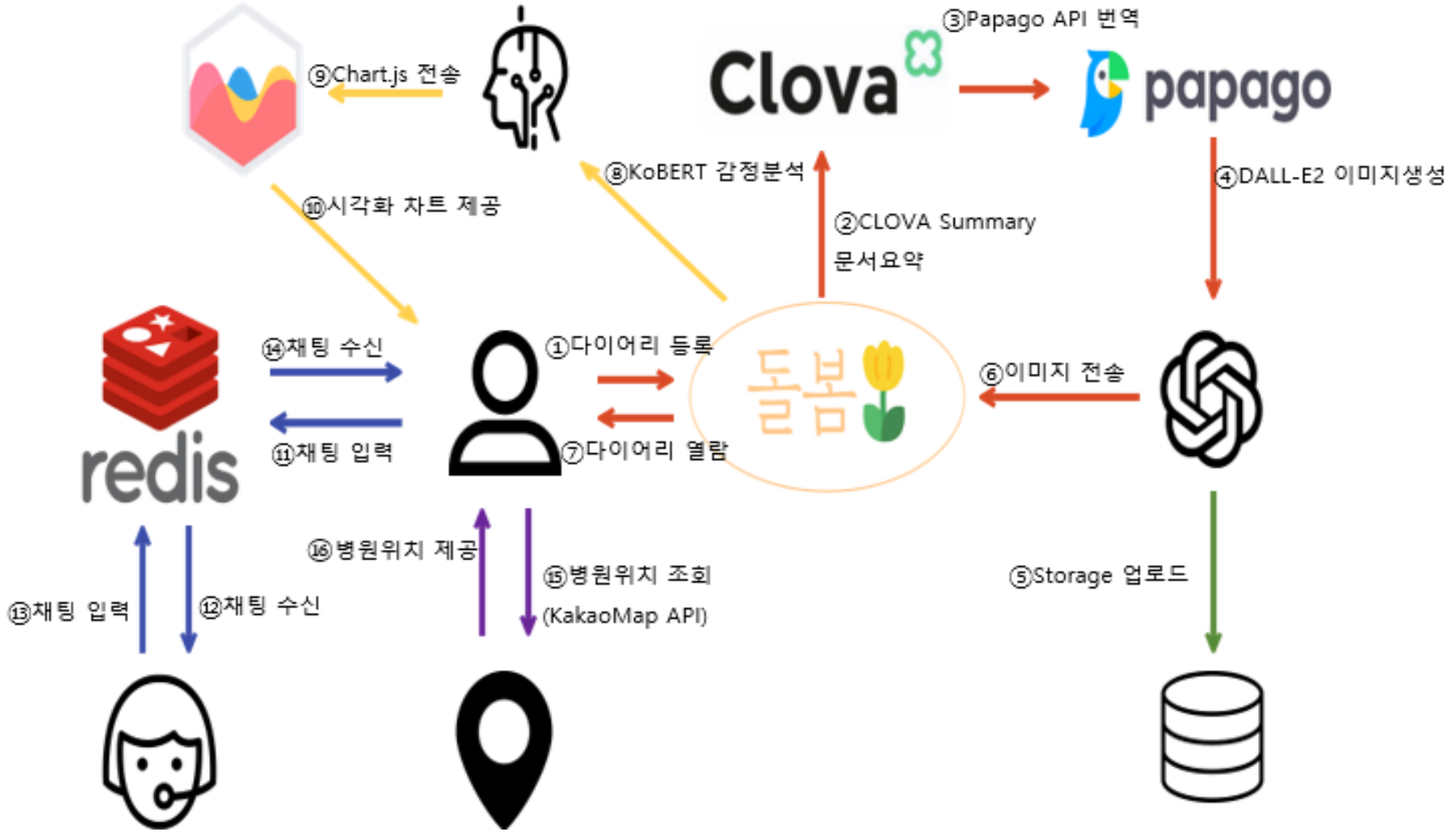
Introduce project

작업 기간	2023.03.02~
인력 구성(기여도)	팀장
프로젝트 목적	프로보노 ICT 공모전
프로젝트 내용	1) 장애부모의 심리적 문제를 해결하고자 감정분석 기반 인공지능 이미지 생성을 통해 감정표현을 돕고 장애부모들끼리의 감정 소통을 도움 2) 복지정보 사각지대를 해소하기 위해 장애부모들에게 복지센터와 복지정보를 가독성 있게 알려줌
주요 업무 및 상세 역할	1) KoBERT를 활용한 자연어처리로 다이어리 속 감정 분석 2) Chart.js를 활용해 그날 감정을 시각화하여 제공 3) webSocket과 Redis를 통해 감정분석 결과가 비슷한 보호자끼리 소통할 수 있는 채팅 제공 4) AI 이미지 생성 모델 DALL-E를 활용하여 다이어리 이미지 생성 5) 다양한 복지 정보 크롤링 6) 관련 병원 API를 활용한 현재 위치 기반 반경 1KM 이내 병원 정보 제공 7) SNS 기능 제공
사용언어 및 개발 환경	JAVA, Java Script, HTML5, CSS, Spring Boot Framework, Python, colab



Project 07. 장애부모를 위한 감정분석 기반 인공지능 이미지 생성을 통한 감정 소통 플랫폼

Main work 서비스 흐름도



Main work 성능평가

Parameters	Value
Max length	64
Classes	7
Batch size	64
Warmup ratio	0.1
Num epochs	5
Max grad norm	1
Learing rate	5e-5
Optimizer	AdamW

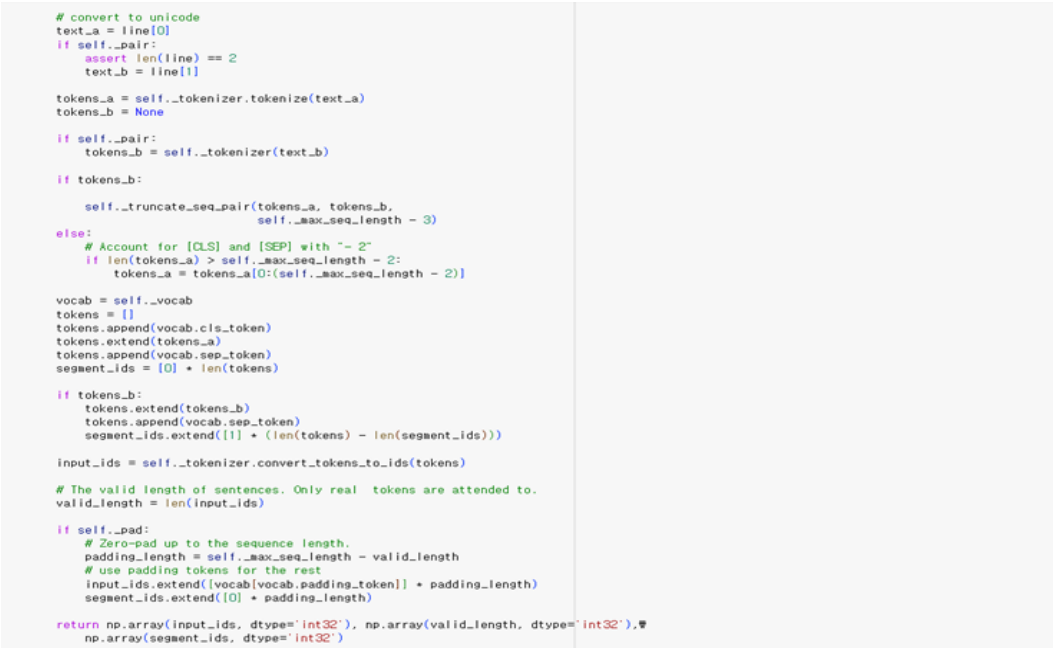
지표 / 모델	Accuracy	Precision	Recall	F1-Score
Bi-LSTM	0.68	0.62	0.68	0.65
LSTM	0.71	0.71	0.71	0.68
GRU	0.71	0.72	0.71	0.69
KoBERT	0.96	0.96	0.93	0.95

- LSTM, BI-LSTM, GRU, KoBERT 모델 별 성능평가 진행
- 7개의 감정으로 분류 후, 총 37,212개의 텍스트 데이터 사용
- 4:1로 학습 데이터와 테스트 데이터로 나누어 학습 및 테스트 진행

- KoBERT 모델은 정확도 0.92, LSTM 대비 0.21, Bi-LSTM 대비 0.24, GRU 대비 0.21 정도 성능이 높게 나옴
- KoBERT 모델 선정 후 다이어리 감정분석 진행

Project 07. 장애부모를 위한 감정분석 기반 인공지능 이미지 생성을 통한 감정 소통 플랫폼

Main work KoBERT 기반 사용자 다이어리 감정분석



```
# convert to unicode
text_a = line[0]
if self._pair:
    assert len(line) == 2
    text_b = line[1]

tokens_a = self._tokenizer.tokenize(text_a)
tokens_b = None

if self._pair:
    tokens_b = self._tokenizer(text_b)

if tokens_b:

    self._truncate_seq_pair(tokens_a, tokens_b,
                            self._max_seq_length - 3)
else:
    # Account for [CLS] and [SEP] with "- 2"
    if len(tokens_a) > self._max_seq_length - 2:
        tokens_a = tokens_a[0:(self._max_seq_length - 2)]

vocab = self._vocab
tokens = []
tokens.append(vocab.cls_token)
tokens.extend(tokens_a)
tokens.append(vocab.sep_token)
segment_ids = [0] * len(tokens)

if tokens_b:
    tokens.extend(tokens_b)
    tokens.append(vocab.sep_token)
    segment_ids.extend([1] * (len(tokens) - len(segment_ids)))

input_ids = self._tokenizer.convert_tokens_to_ids(tokens)

# The valid length of sentences. Only real  tokens are attended to.
valid_length = len(input_ids)

if self._pad:
    # Zero-pad up to the sequence length.
    padding_length = self._max_seq_length - valid_length
    # use padding tokens for the rest
    input_ids.extend([vocab[vocab.padding_token]] * padding_length)
    segment_ids.extend([0] * padding_length)

return np.array(input_ids, dtype='int32'), np.array(valid_length, dtype='int32'),
    np.array(segment_ids, dtype='int32')
```

- 학습 시킨 BERT 기반 모델에 접합한 형식으로 문장 변환

```
@app.route('/emotion')
def predict(predict_sentence):
    data = [predict_sentence, '0']
    dataset_another = [data]
    print(data)
    print(dataset_another)
    print(tokenizer)

    another_test = BERTDataset(dataset_another, 0, 1, tokenizer, vocab, max_len, True, False)
    test_dataloader = torch.utils.data.DataLoader(another_test, batch_size=batch_size, num_workers=5)

    model.eval()

    for batch_id, (token_ids, valid_length, segment_ids, label) in enumerate(test_dataloader):
        token_ids = token_ids.long().to(device)
        segment_ids = segment_ids.long().to(device)

        valid_length = valid_length
        label = label.long().to(device)

        out = model(token_ids, valid_length, segment_ids)

        test_eval = []
        for i in out:
            logits = i
            logits = logits.detach().cpu().numpy()

            if np.argmax(logits) == 0:
                test_eval.append("공포가")
            elif np.argmax(logits) == 1:
                test_eval.append("놀람이")
            elif np.argmax(logits) == 2:
                test_eval.append("분노가")
            elif np.argmax(logits) == 3:
                test_eval.append("슬픔이")
            elif np.argmax(logits) == 4:
                test_eval.append("중립이")
            elif np.argmax(logits) == 5:
                test_eval.append("행복이")
            elif np.argmax(logits) == 6:
                test_eval.append("혐오가")

        print(">> 입력하신 내용에서 " + test_eval[0] + " 느껴집니다.")

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=8605)
```

- 학습시킨 KoBERT를 사용한 감정분석 진행

Project 07. 장애부모를 위한 감정분석 기반 인공지능 이미지 생성을 통한 감정 소통 플랫폼

Main work 이미지 생성 모델 DALL-E2를 사용한 이미지 생성

```java
@Override
public String summary(String prompt) throws Exception {
    log.info(this.getClass().getName() + ".summary Start(다이어리 요약 시작) !!");

    // Document Object (문서 제목,본문)
    Map<String,String> document = new HashMap<>();
    document.put( k: "content", prompt); // 문서의 내용

    // Option Object (요약 옵션)
    Map<String, Object> option = new HashMap<>();
    option.put( k: "language", v: "ko"); // 문서의 언어 ex) ko = 한국어, jp = 일본어
    option.put( k: "model", v: "general"); // 요약에 사용될 모델 *general = 일반 문서요약, *news = 뉴스요약
    option.put( k: "tone", v: 0); // 요약된 문서의 어투 * 0 -> 원문의 어투를 유지
    option.put( k: "summaryCount", v: "2"); // 요약된 문서의 문장 수

    // 요약 문서와 옵션을 DTO에 담기
    ClovaDTO pDTO = new ClovaDTO();
    pDTO.setDocument(document);
    pDTO.setOption(option);

    log.info("DocumentObject : " + document);
    log.info("OptionObject : " + option);

    // 요청 헤더 생성
    HttpHeaders headers = new HttpHeaders();
    headers.add( headerName: "X-NCP-APIGW-API-KEY-ID",apiKeyId);
    headers.add( headerName: "X-NCP-APIGW-API-KEY",apiKey);
    headers.setContentType(MediaType.APPLICATION_JSON);

    // DTO와 헤더 담기
    HttpEntity<ClovaDTO> entity = new HttpEntity<>(pDTO,headers);

    // CLOVA SUMMARY에 요약 요청
    ResponseEntity<Map> response = restTemplate.exchange( url: "https://naveropenapi.apigw.ntruss.com/text-summary/v1/summarize", HttpMethod.POST, entity, Map.class);

    // 요청 결과를 Map 형태로 꺼내기
    Map<String,String> responseMap = response.getBody();

    // 결과 Map에서 요약결과 꺼내기
```

- CLOVA Summary를 활용하여 사용자의 다이어리 요약
- Papago API를 사용하여 영미 어로 번역

```java
@Override
public ImageDTO uploadImage(String url) throws Exception {
    // 이미지 저장할 경로와 저장될 이미지 이름
    String outputFilePath = "C:\\OpenAi-Image\\" + DateUtil.getDateTime( fm: "24hhmmss") + ".png";
    // Storage URL
    String FilePath = "";
    // 확장자를 담기위해..
    ImageDTO qDTO = new ImageDTO();
    // 생성된 URL을 읽고 로컬에 다운로드
    try {
        URL imageUrl = new URL(url);
        InputStream inputStream = imageUrl.openStream(); // URL의 이미지데이터를 읽을 입력스트림 열기
        FileOutputStream outputStream = new FileOutputStream(outputFilePath); // 해당 경로에 대한 이미지 데이터를 파일에 작성한다.
        // 이미지 다운로드
        byte[] buffer = new byte[1024];
        int bytesRead;
        while ((bytesRead = inputStream.read(buffer)) != -1) {
            outputStream.write(buffer, off: 0, bytesRead);
        }
        outputStream.close(); // 출력스트림 종료
        inputStream.close(); // 입력스트림 종료
        System.out.println("이미지 다운로드가 완료되었습니다. 파일 경로: " + outputFilePath);
        File image = loadImage(outputFilePath);
        log.info("이미지의 이름 : " + image.getName());
        log.info("이미지의 확장자 : " + image.getName().substring( beginIndex: image.getName().lastIndexOf( ch: '.') + 1));
        // NCP Object Storage에 저장
        String filePath = CmmUtil.nvl(storageService.upload(image,image.getName()));
        // 결과 url 가공
        String path1 = filePath.replace( target: "uddoro.project.", replacement: "");
        log.info("Path 1 : " + path1);
        String path2 = path1.substring(0,35);
        log.info("Path 2 : " + path2);
        String path3 = "uddoro.project";
        String path4 = path1.substring(36,path1.length());
        log.info("Path 4 : " + path4);
        FilePath = path2+ "/"+path3+ "/" +path4;
        log.info("FilePath : " + FilePath);
        // 반환할 데이터를 DTO에 담기
        qDTO.setFileExt(image.getName().substring( beginIndex: image.getName().lastIndexOf( ch: '.') + 1).toLowerCase()); // 확장자
        qDTO.setSeverFileName(image.getName()); // 저장될 이름
```

- AI 이미지 생성 모델 DALL-E2를 사용한 이미지 생성

Project 08.

OCR을 활용한 세무 업무 도움 플랫폼

About project

한이음 ICT 공모전

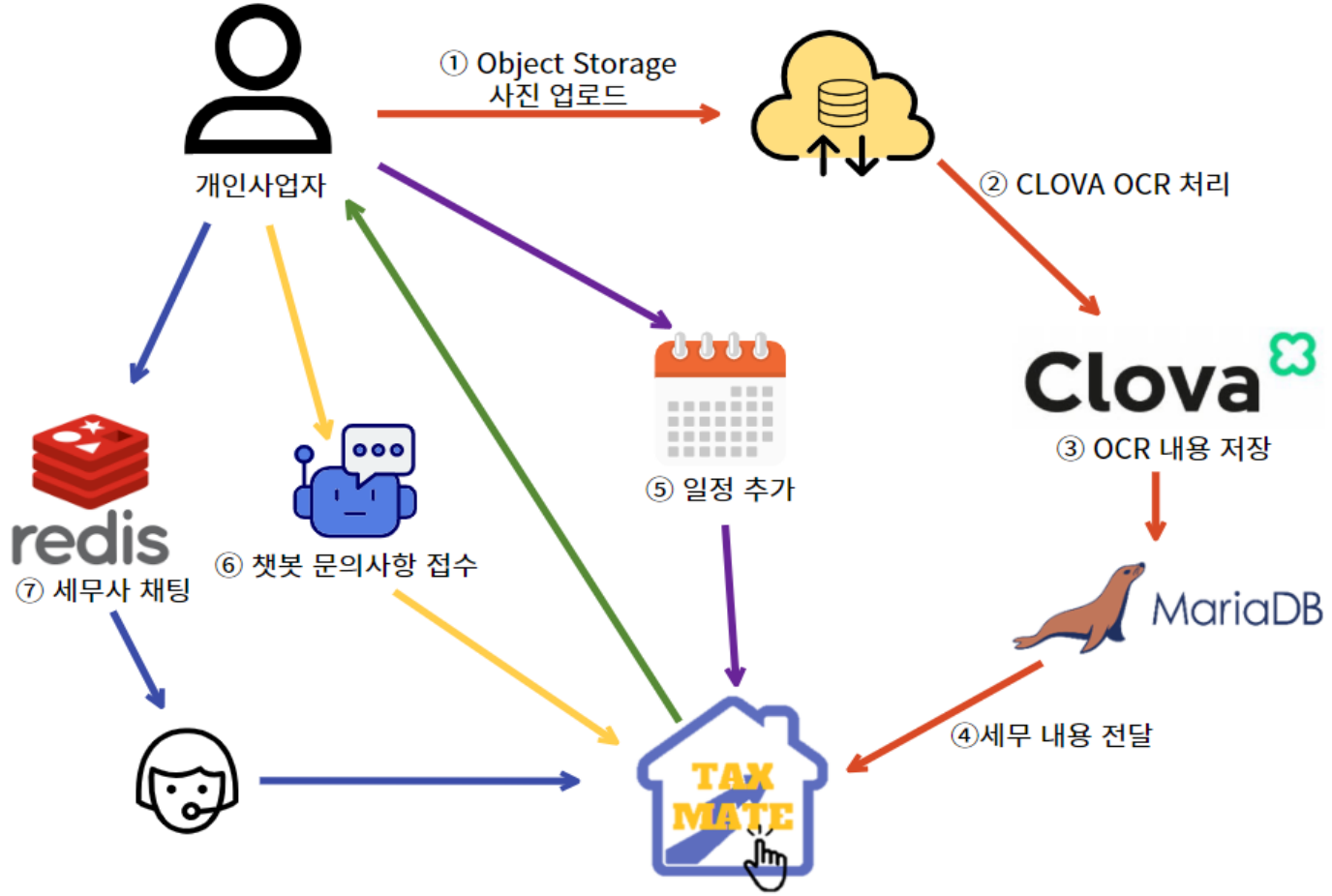
팀원

Project 08. OCR을 활용한 세무 업무 도움 플랫폼

Introduce project

작업 기간	2023.03.02~
인력 구성(기여도)	팀원
프로젝트 목적	2023년 한이음 ICT 공모전
프로젝트 내용	1) 세무에 대한 정보와 관련 지식이 부족한 개인 사업자들을 대상으로 세무 업무를 돕는 서비스 2) 주요 기능은 광학 문자 인식 기술(OCR)을 활용한 세금 고지서 스캔 기능
주요 업무 및 상세 역할	1) CLOVA OCR을 활용해 고지서를 스캔하여 업로드 후 사용자에게 필요한 정보 추출 2) Object Storage를 활용하여 업로드되는 세금 납부서 이미지 형태로 저장 3) FullCalender.js를 통해서 세무정 관련 국세청 API를 가독성 있게 달력 형식으로 제공 4) CLOVA Chatbot을 통한 간편 문의 및 세법 정보 제공 5) webSocket과 Redis를 통한 세무사 채팅 서비스 제공
사용언어 및 개발 환경	JAVA, Java Script, HTML5, CSS, Spring Boot Framework

Main work 서비스 흐름도



Project 08. OCR을 활용한 세무 업무 도움 플랫폼

Main work CLOVA OCR을 활용한 세금 고지서 분석


```java
//  CLOVA OCR API와 기본적인 통신준비
URL url = new URL(apiURL); //요청 URL 설정
HttpURLConnection con = (HttpURLConnection)url.openConnection();
con.setUseCaches(false); // 캐시 비활성화
con.setDoInput(true);// 입력 활성화
con.setDoOutput(true);// 출력 활성화
con.setRequestMethod("POST");// POST 전송방식 설정
con.setRequestProperty("X-OCR-SECRET", secretKey);// 인증키

// JSON 객체 생성하고 OCR 버전, 이미지 정보등을 담아서 저장
JSONObject json = new JSONObject();// json 객체 생성
json.put( s: "version",  o: "V2"); // OCR 버전 설정
json.put( s: "requestId", UUID.randomUUID().toString()); //요청 ID
json.put( s: "timestamp", System.currentTimeMillis()); // 타임스탬프 설정
JSONObject image = new JSONObject(); // 이미지 정보를 담은 JSON 객체 생성
image.put( s: "format",  o: "jpg"); // 이미지 형식
image.put( s: "url", oDTO.getImageURL()); // 이미지 URL
image.put( s: "name",  o: "tax"); // 이미지 이름
json.put( s: "image", image); //전송할 json 객체에 이미지파일 추가
String postParams = json.toString(); // JSON 데이터를 문자열로 변환

//DataOutputStream 스트림을 사용하여 json 객체를 CLOVA API에 전송
DataOutputStream wr = new DataOutputStream(con.getOutputStream());
wr.writeBytes(postParams); //postParams 문자열을 담아서 전송
wr.flush(); // 버퍼에 남은 데이터 모두 전송
wr.close(); // 스트림닫기
```

- 세금 납부서의 이미지를 Object Storage에 저장
- CLOVA OCR API와의 통신
- JSON 객체로 변환하여 image 키에 해당하는 값 배열로 저장
- MariaDB에 고지서 스캔 결과 저장 및 제공

Project 08. OCR을 활용한 세무 업무 도움 플랫폼

Main work FullCalendar를 활용한 세무 일정


```
<script type="text/javascript">

    $(function () {
        $(".close-modal-btn").on("click", function () {  // modal의 추가 버튼 클릭 시
            console.log(".close-modal-btn click!!!");
            $("#calendarModalEdit").modal("hide");
            $("#calendarModal").modal("hide");
        });
    });

    /**
     * Intl.DateTimeFormat(Intl, options) <<< options 에 사용됨
     * @type {{month: string, year: string, day: string}}
     */
    const options = {
        year: 'numeric',    // 년 (4자리)
        month: '2-digit',   // 월 (2자리)
        day: '2-digit',     // 일 (2자리)
    };
    // Intl.DateTim
    const dateFormatter = new Intl.DateTimeFormat('ko-KR', options);
    const dateFormat = myDate => {
        console.log("dateformat() start!");
        let formattedDate = formatter.format(date);

        let parts = formattedDate.split('. ');
        let formattedWithDash = parts.join('-');

        console.log(formattedWithDash); // 예: "2023-08-29"

    }

    function formatDate(date) {
        const year = date.getFullYear();
        const month = String(date.getMonth() + 1).padStart(2, '0');
        const day = String(date.getDate()).padStart(2, '0');
```

- **FullCalendar.js를 활용해 세무일정 달력 형식으로 제공**
- **개인의 세무일정 캘린더에 작성 가능**

End of Document