

컴퓨터공학 종합설계 001분반

## 상세 설계서



컴퓨터공학 종합설계(202501-CSE4205-001)

팀명 : Team 890

12192077 박륜

12180557 신성혁

12201798 정우성

# 0. 목차

## 1. 개요

- 1-1. 설계 목적
- 1-2. 프로젝트 개요
- 1-3. 개발환경
- 1-4. 아키텍처

## 2. 데이터베이스 설계

- 2-1. ER 다이어그램
- 2-2. 테이블 상세 설계
  - 2-2-1. Team Table
  - 2-2-2. Player Table
  - 2-2-3. Season Table
  - 2-2-4. Match Table
  - 2-2-5. TeamStats Table
  - 2-2-6. PlayerStats Table
  - 2-2-7. PlayerSeasonStats Table

## 3. 승부 예측 모델

- 3-1. 개요
- 3-2. 데이터 전처리
  - 3-2-1. 결측치/이상치 필터링
  - 3-2-2. 불필요한 feature 삭제
  - 3-2-3. 범주형 변수 처리
- 3-3. 학습 데이터 구성

### 3-4. 모델 학습

#### 3-4-1. 모델별 학습

#### 3-4-2. 모델별 장/단점

### 3-5. 출력

### 3-6. 성능 평가

### 3-7. API 배포

## 4. 비정형 데이터 모듈

### 4-1. 개요

### 4-2. 비정형 데이터

#### 4-2-1. 전문가 예측

#### 4-2-2. key player 정보

## 5. 해설 모듈

## 6. 서비스 설계

### 6-1. 서버

### 6-2. API 명세

## 7. UI 설계

## 8. WBS

# 1. 개요

## 1-1. 설계 목적

본 문서는 <PREMO>의 상세요구사항명세서를 기반으로 작성되어 프로젝트를 진행함에 있어 필요한 모든 요소들을 직접적으로 제시하여 개발을 수월히 진행시킬 수 있도록 작성되었다.

## 1-2. 프로젝트 개요

<PREMO>는 footystat.org에서 구입한 데이터(EPL 2010/11 ~ 2020/21의 학습데이터 + 2021/22 ~ 2024/25의 검증 데이터)를 기반으로 경기의 승/무/패의 결과와 예상 스코어를 예측하고 공신력 높은 해외 기사, 트위터에서 수집한 비정형 데이터를 감성 분석후 통계를 내어 해당 경기에 대한 전문가들의 예측 결과와 key player 정보들을 보여주는 서비스이다.

### 1-2-1. 공신력 높은 기사 사이트 / 트위터 계정

#### (a) 기사 사이트

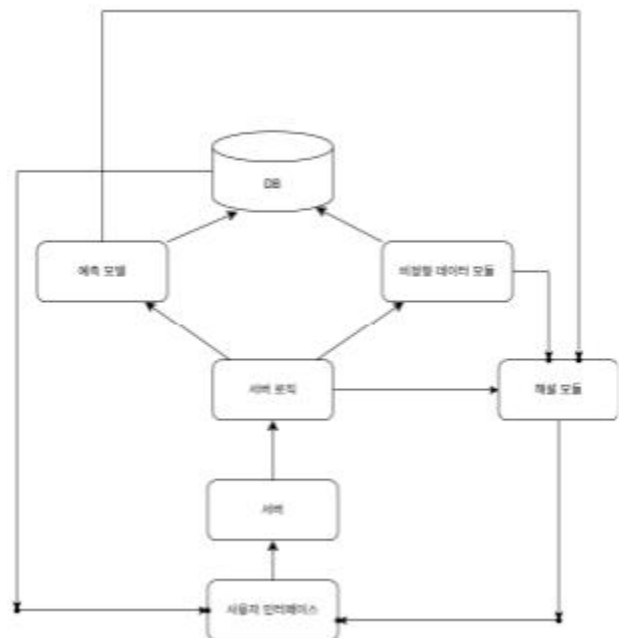
이름	설명	링크
BBC Sport	영국 공영방송. 가장 공신력 높음. 뉴스/인터뷰/속보 다양	<a href="https://www.bbc.com/sport">https://www.bbc.com/sport</a>
The Guardian Football	분석글, 전술해설, 칼럼 매우 훌륭함	<a href="https://www.theguardian.com/football">https://www.theguardian.com/football</a>
Sky Sports Football	빠르고 정확한 속보, 이적시장 소식 강점	<a href="https://www.skysports.com/football">https://www.skysports.com/football</a>
ESPN FC	전세계 축구 리그 다룸. 분석/영상 콘텐츠 많음	<a href="https://www.espn.com/soccer">https://www.espn.com/soccer</a>
The Athletic (Football)	유료지만 분석 깊고 정확함. 팀별 전문 기사 있음	<a href="https://theathletic.com/football">https://theathletic.com/football</a>
UEFA / FIFA	공식 경기 리포트 및 통계용으로 활용	<a href="https://www.uefa.com">https://www.uefa.com</a> <a href="https://www.fifa.com">https://www.fifa.com</a>

## (b) 트위터 계정

계정	설명	링크
@FabrizioRomano	이적시장의 제왕, "Here we go!"로 유명	<a href="https://twitter.com/FabrizioRomano">https://twitter.com/FabrizioRomano</a>
@BBCSport	BBC 스포츠 공식 트위터	<a href="https://twitter.com/BBCSport">https://twitter.com/BBCSport</a>
@SkySportsNews	실시간 이슈, 속보 에 강함	<a href="https://twitter.com/SkySportsNews">https://twitter.com/SkySportsNews</a>
@OptaJoe	Opta 공식, 통계 기 반 트윗	<a href="https://twitter.com/OptaJoe">https://twitter.com/OptaJoe</a>
@WhoScored	경기 후 선수 평가 와 분석, 데이터 기 반	<a href="https://twitter.com/WhoScored">https://twitter.com/WhoScored</a>
@TheAthleticFC	The Athletic 축구 전문 계정	<a href="https://twitter.com/TheAthleticFC">https://twitter.com/TheAthleticFC</a>

프로젝트의 컴포넌트 구성은 다음과 같다.

- 예측 모델
- 비정형 데이터 모듈
- 해설 모듈
- Backend API
- Frontend API
- DB

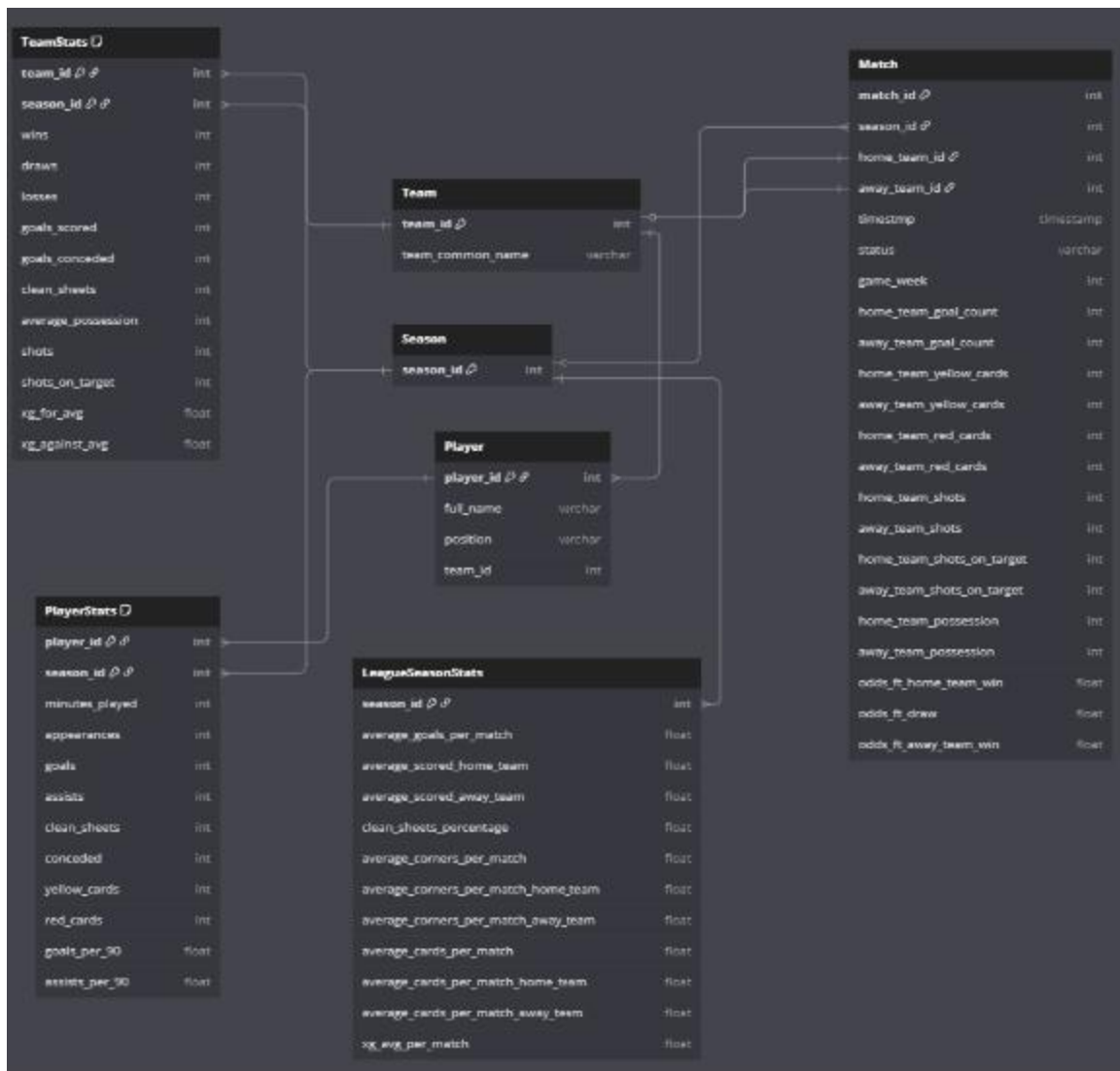


### 1-3. 개발환경

구성 요소	세부 항목	사용 도구 및 버전/라이브러리
프론트엔드	기술 스택	React, HTML, CSS, JavaScript, Axios, Redux, Recharts
백엔드	기술 스택	Node.js, Express, JWT, cookie-parser, mongoose
서버	호스팅	AWS EC2, AWS RDS, AWS EBS
데이터 저장소	데이터 저장	PostgreSQL, MongoDB
예측 모델	데이터 처리	pandas (1.5.3), numpy (1.24.4)
	전처리 및 분할	scikit-learn (1.3.2), LabelEncoder, train_test_split
	모델 학습	XGBoost (1.7.6), LightGBM (3.3.5), AWS SageMaker
	모델 해석	SHAP, confusion_matrix, f1_score
비정형 모듈	텍스트 크롤링	BeautifulSoup, snsrape
	텍스트 정제	re, nltk, pandas
	감성 분석	transformers, LLaMAA, Mistral

## 2. 데이터베이스 설계

### 2-1. ER 다이어그램



## 2-2. 테이블 상세 설계

### 2-2-1. Team Table

Team			
컬럼명	설명	타입	제약조건
team_id	팀 고유 ID	INT	PK, AUTO_INCREMENT
Team_common_name	팀 이름	VARCHAR(100)	NOT NULL

### 2-2-2. Player Table

Player			
컬럼명	설명	타입	제약조건
player_id	선수 고유 ID,	INT	PK, FK → Team.team_id
full_name	선수 이름	VARCHAR(100)	NOT NULL
position	포지션	VARCHAR(30)	NOT NULL
team_id	소속 팀 ID	INT	FK → Team.team_id

### 2-2-3. Season Table

Season			
컬럼명	설명	타입	제약조건
season_id	시즌 고유 ID	INT	PK, AUTO_INCREMENT

### 2-2-4. Match Table

Match			
컬럼명	설명	타입	제약조건
match_id	경기 고유ID	INT	PK, AUTO_INCREMENT
season_id	시즌 ID	INT	FK → Season.season_id
home_team_id	홈팀 ID	INT	FK → Team.team_id
away_team_id	원정팀 ID	INT	FK → Team.team_id
start_time	경기 시작 시간	TIMESTAMP	
status	경기 상태	VARCHAR(20)	
game_week	경기 주차	INT	
home_team_goal_count	홈팀 득점	INT	
away_team_goal_count	원정팀 득점	INT	
home_team_yellow_cards	홈팀 경고 수	INT	
away_team_yellow_cards	원정팀 경고 수	INT	
home_team_red_cards	홈팀 퇴장 수	INT	



away_team_red_cards	원정팀 퇴장 수	INT	
home_team_shots	홈팀 슈팅 수	INT	
away_team_shots	원정팀 슈팅 수	INT	
home_team_shots_on_target	홈팀 유효 슈팅 수	INT	
away_team_shots_on_target	원정팀 유효 슈팅 수	INT	
home_team_possession	홈팀 점유율(%)	INT	
away_team_possession	원정팀 점유율(%)	INT	
odds_ft_home_team_win	홈팀 승 배당률	FLOAT	
odds_ft_draw	무승부 배당률	FLOAT	
odds_ft_away_team_win	원정팀 승 배당률	FLOAT	
match_result	경기 결과 1: 홈 승, 0: 어웨이 승, 2: 무승부	INT	

### (match\_result 컬럼 생성)

footystat.org에서 제공해주는 dataset에는 경기의 결과를 따로 나타내는 column이 없기에 home\_team\_goal\_count와 away\_team\_goal\_count를 비교하여 홈팀 승 : 1, 어웨이팀 승 : 0, 무승부 : 2로 column을 생성해준다.

함수명	get_result(row)
입력값	row
출력값	1/0/2
설명	match dataset을 row로 입력받아 경기의 결과를 1: 홈팀 승, 0: 홈팀 패, 2: 무승부로 출력하여주는 함수이다.

## 2-2-5. TeamStats Table

TeamStats			
컬럼명	설명	타입	제약조건
team_id	팀 ID	INT	PK, FK → Team.team_id
season_id	시즌 ID	INT	PK, FK → Season.season_id
wins	승 수	INT	
draws	무승부 수	INT	
losses	패배 수	INT	
goals_scored	총 득점	INT	
goals_conceded	총 실점	INT	
clean_sheets	무실점 경기 수	INT	
average_possession	평균 점유율(%)	INT	
shots	총 슈팅 수	INT	
shots_on_target	유효 슈팅 수	INT	
xg_for_avg	평균 기대득점 (xG)	FLOAT	
xg_against_avg	평균 기대실점 (xGA)	FLOAT	

## 2-2-6. PlayerStats Table

PlayerStats			
컬럼명	설명	타입	제약조건
player_id	선수 ID	INT	PK, FK → Player.player_id
season_id	시즌 ID	INT	PK, FK → Season.season_id
minutes_played	출전 시간 (분)	INT	
appearances	출전 횟수	INT	
goals	득점 수	INT	
assists	어시스트 수	INT	
clean_sheets	무실점 경기 수	INT	
conceded	실점 수	INT	
yellow_cards	경고 수	INT	
red_cards	퇴장 수	INT	
goals_per_90	90분당 득점	FLOAT	
assists_per_90	90분당 어시스트	FLOAT	

## 2-2-7. PlayerSeasonStats Table

PlayerSeasonStats			
컬럼명	설명	타입	제약조건
season_id	시즌 ID	INT	PK, FK → Season.season_id
average_goals_per_match	경기당 평균 득점 수	FLOAT	
average_scored_home_team	홈 팀 평균 득점	FLOAT	
average_scored_away_team	원정 팀 평균 득점	FLOAT	
clean_sheets_percentage	무실점 경기 비율 (%)	FLOAT	
average_corners_per_match	경기당 평균 코너킥 수	FLOAT	
average_corners_per_match_home_team	홈 팀 평균 코너킥	FLOAT	
average_corners_per_match_away_team	원정 팀 평균 코너킥	FLOAT	
average_cards_per_match	경기당 평균 카드 수	FLOAT	
average_cards_per_match_home_team	홈 팀 평균 카드 수	FLOAT	
average_cards_per_match_away_team	원정 팀 평균 카드 수	FLOAT	
xg_avg_per_match	경기당 평균 기대 득점 (xG)	FLOAT	

### 3. 승부 예측 모델

#### 3-1. 개요

##### (목표)

입력된 경기 데이터를 바탕으로 경기 결과 (승/무/패)와 예측 스코어를 XGBoost / LightGBM/양상블 모델을 통해 예측한다.

#### 3-2. 데이터 전처리

모델 학습 및 예측에 필요한 경기 데이터를 수집하고 전처리하여 사용 가능한 형태로 변환하는 것을 목적으로 한다.

##### 3-2-1. 결측치/이상치 필터링

footystat.org에서 받아온 데이터 중 값이 비어는 셀을 찾아 제거하거나 평균이나 중앙값으로 채워 결측치를 제거해준다. 그리고 분포를 벗어난 데이터가 발견된다면 제거해주는 방법으로 이상치를 필터링하여준다.

##### 예시 코드

```
import pandas as pd

# 데이터 불러오기
df = pd.read_csv('data.csv')

# 1. 결측치 처리
df = df.dropna()
df = df.fillna(df.mean()) # 평균으로 채움
df = df.fillna(df.median()) # 중앙값으로 채움

# 2. 이상치 처리 (IQR 방식)
Q1 = df.quantile(0.25)
Q3 = df.quantile(0.75)
IQR = Q3 - Q1
df = df[~((df < (Q1 - 1.5 * IQR)) | (df > (Q3 + 1.5 * IQR))).any(axis=1)]
```

##### 3-2-2. 불필요한 feature 삭제

학습 데이터의 절대적인 양에 비해 차원이 너무 높아지면 모델이 과적합될 가능성이 높아지므로, 이를 방지하기 위해 승부예측에 도움이 되지 않을 것이 자명한 feature(예 : 경기장 이름, 팀의 공식 이름, 선수의 나이 등)들은 수작업으로 삭제하여 준다.

(삭제 후의 feature)

	Removed column count	User column count	예시
league	58	13	<ul style="list-style-type: none"> <li>- btts_percentage</li> <li>- format</li> <li>- game_week</li> <li>- goals_min_0_to_10</li> <li>- goals_min_0_to_15</li> <li>- over_05_cards_percentage</li> <li>- over_05_percentage</li> <li>- over_105_corners_percentage</li> <li>- prediction_risk</li> <li>- progress</li> <li>- status</li> <li>- total_cards_for_season ...</li> </ul>
match	10	56	<ul style="list-style-type: none"> <li>- attendance</li> <li>- date_GMT</li> <li>- odds_btts_no</li> <li>- odds_btts_yes</li> <li>- odds_ft_over15</li> <li>- odds_ft_over25</li> <li>- odds_ft_over35</li> <li>- odds_ft_over45</li> <li>- referee</li> <li>- stadium_name</li> </ul>
player	39	38	<ul style="list-style-type: none"> <li>- accurate_crosses_per_90_overall</li> <li>- accurate_crosses_per_game_overall</li> <li>- accurate_crosses_total_overall</li> <li>- additional_info</li> <li>- birthday</li> <li>- birthday_GMT</li> <li>- dispossessed_per_game_overall</li> <li>-</li> <li>- distance_travelled_per_game_overall ..</li> <li>-</li> </ul>
team	550	178	<ul style="list-style-type: none"> <li>- BTTS_and_draw_away</li> <li>- BTTS_and_draw_home</li> <li>- BTTS_and_draw_overall</li> <li>- BTTS_and_draw_percentage_away</li> <li>-</li> <li>- BTTS_and_draw_percentage_home ...</li> </ul>

※학습에서 제외되는 feature는 프로젝트를 진행하면서 조정될 수 있다.

### 3-2-3. 범주형 변수 처리

범주형 변수들을 one-hot encoding을 통하여 encoding시켜준다. 단, 선수명과 같이 데이터의 개수가 많은 feature에 경우 one-hot encoding으로 encoding을 진행시키게 되면 차원의 수가 많아져 모델의 성능을 저하시킬 수 있기에 선수들에 각각 고유 식별자를 부여하여하는 방식을 사용하여 범주형 변수에서 제외시켜준다.

#### 예시 코드

```
import pandas as pd

# 예시 데이터
df = pd.DataFrame({
    'team': ['Arsenal', 'Chelsea', 'Arsenal', 'Man City'],
    'player_name': ['Saka', 'Sterling', 'Odegaard', 'Haaland']
})

# 1. One-hot encoding (차원 증가)
df_onehot = pd.get_dummies(df, columns=['team'])

# 2. 선수명에 ID 부여 (Label Encoding 방식)
df['player_id'] = df['player_name'].astype('category').cat.codes
```

### 3-3. 학습 데이터 구성

PL 2010/11 ~ 2024/25 총 15시즌의 데이터셋을 사용, 한 시즌의 승부를 예측하는 데 직전 5년간의 데이터를 input feature들로 학습하므로 유의미하게 예측할 수 있는 시즌은 총 10시즌이다. 이 중 6시즌은 Train set, 2시즌은 Validation set, 2시즌은 Test set으로 사용하여 6 : 2 : 2비율로 모델을 학습 및 검증한다.

※학습과정에서 지표결과에 따라 비율은 수정될 수 있다.

인풋 데이터 (train)	예측 시즌 (test)
2010/11 ~ 2014/15	2015/16
2011/12 ~ 2015/16	2016/17
2012/13 ~ 2016/17	2017/18
2013/14 ~ 2017/18	2018/19
2014/15 ~ 2018/19	2019/20
2015/16 ~ 2019/20	2020/21
2016/17 ~ 2020/21	2021/22
2017/18 ~ 2021/22	2022/23
2018/19 ~ 2022/23	2023/24
2019/20 ~ 2023/24	2024/25

※ Train set  
※ Validation set  
※ Test set

### 3-4. 모델 학습 (XGBoost / LightGBM / 앙상블)

#### 3-4-1. 모델별 학습

##### XGBoost 단일 모델

###### 예시 코드

```
from xgboost import XGBClassifier

xgb_model = XGBClassifier(use_label_encoder=False,
                           eval_metric='mlogloss')
xgb_model.fit(X_train, y_train)

pred_xgb = xgb_model.predict(X_test)

acc_xgb = accuracy_score(y_test, pred_xgb)
f1_xgb = f1_score(y_test, pred_xgb, average='macro')
```

##### LightGBM 단일 모델

###### 예시 코드

```
from lightgbm import LGBMClassifier

lgb_model = LGBMClassifier()
lgb_model.fit(X_train, y_train)

pred_lgb = lgb_model.predict(X_test)

acc_lgb = accuracy_score(y_test, pred_lgb)
f1_lgb = f1_score(y_test, pred_lgb, average='macro')
```

##### XGBoost + LightGBM 앙상블 모델

###### 예시 코드

```
from sklearn.ensemble import VotingClassifier

ensemble = VotingClassifier(estimators=[
    ('xgb', xgb_model),
    ('lgb', lgb_model)
], voting='soft')

ensemble.fit(X_train, y_train)
pred_ens = ensemble.predict(X_test)

acc_ens = accuracy_score(y_test, pred_ens)
f1_ens = f1_score(y_test, pred_ens, average='macro')
```

### 3-4-2. 모델별 장/단점

모델	장점	단점
XGBoost 단일	성능이 우수한 부스팅 모델이다.	학습속도는 LightGBM보다 느릴 수 있다.
LightGBM 단일	빠르고 대용량 데이터에 강하다.	하이퍼파라미터가 예민하여 민감할 수 있다.
XGBoost + LightGBM 앙상블	두 모델의 장점을 결합한 것으로 soft voting으로 예측 안정성을 증 가시킬 수 있다.	학습 속도가 가장 느릴 수 있다.

3개의 모델을 제작하여 학습을 진행시킨 후 정확도 등의 지표를 비교하여 가장 우수한 모델을 채택한 뒤 api로 배포시킨다.

### 3-5. 출력

모델은 다음 7개의 feature들을 예측 결과값으로 출력한다.

- home\_win\_rate : 홈 팀 승리 확률
- draw\_rate : 무승부 확률
- away\_win\_rate : 원정 팀 승리 확률
- home\_goal\_count : 홈 팀 득점 수
- away\_goal\_count : 원정 팀 득점 수

### 3-6. 성능 평가

다음 지표들의 목표치를 기준으로 모델의 성능을 판단한다.

지표	목표 값	지표로 채택된 근거
Accuracy	$\geq 70\%$	- EPL의 승/무/패는 대체로 4050% / 2030% / 20~30% 수준으로 분포하며, 단순 무작위 예측 시 정확도는 약 33% 수준이다. 따라서 Accuracy 70%는 랜덤의 2배 이상에 해당하는 성능으로, 통계적으로도 충분히 유의미한 성과로 볼 수 있다. - 기존의 통계 기반 또는 배당률 기반 예측은 약 55~60% 수준의 정확도를 보이며, 머신러닝 모델이 이보다 높은 성능인 70%를 목표로 하는 것은 합리적이다.
F1-score (macro)	$\geq 0.60$	- 0.60 이상이면 모든 클래스에서 균형 잡힌 성능을 내고 있다는 의미가 된다.

F1-score (weighted)	$\geq 0.65$	- Accuracy가 70%에 도달하면, weighted F1은 보통 0.65~0.7 정도로 따라 올라간다.
Log Loss	$\leq 0.90$	- 다중 클래스 분류에서 baseline 수준은 1.0 이상이고, 1.0 이하로 떨어뜨리면 의미 있는 신뢰도 개선이 이루어졌다고 본다. - 0.90 이하는 통계/머신러닝 기반 예측에서 실용적으로 관찰은 cutoff로 자주 쓰인다.
Top-2 Accuracy	$\geq 85\%$	- 축구처럼 이변이 많고 무승부 확률이 높은 게임에서는 정답을 딱 1순위로 못 맞춰도, Top 2 안에 포함시키는 게 더 현실적일 수 있다. - Accuracy가 70%면, 보통 Top-2 Accuracy는 그보다 10~15% 더 높게 나온다. 따라서 85%는 자연스럽고 달성 가능한 기준선이다.
Confusion Matrix	균형 잡힌 분포	- 특정 클래스(예: 무승부)에 너무 치우치지 않도록 점검하는 데 사용

목표치 미달 시 다음 대처 전략을 수행한다.

- Accuracy
  - Feature 엔지니어링을 추가한다. ( e.g., 선수 능력, 홈 이점 등)
  - 데이터 불균형을 확인한 후에 클래스를 조정한다.
  - 더 많은 학습 데이터를 확보한다.
  - 모델 구조를 변경하여준다. (단일 → 앙상블)
- F1-score(macro)
  - 데이터 불균형이 원인일 가능성이 크므로 SMOTE와 Class Weight를 적용시켜준다.
  - 모든 클래스에서 고르게 학습할 수 있도록 모델을 튜닝시켜준다.
  - 라벨의 기준을 재검토해본다. ( e.g., 무승부 정의 등)
- F1-score(weighted)
  - Accuracy가 높아도 소수 클래스가 무시될 수 있으므로 클래스별 성능을 점검하여준다.
  - 오버샘플링이나 클래스 가중치를 조절하여 균형을 잡아준다.
- Log Loss
  - 확률 예측 품질이 낮은 경우, predict\_proba 기반의 학습을 확인한다.



- 모델이 과도하게 확신하거나 확률을 왜곡할 경우 확률 calibration을 적용한다. ( e.g., Platt Scaling 등)
- Top-2 Accuracy
  - 결과가 애매하거나 무승부 빈도가 높은 경우 Top-N의 활용을 강화한다.
- Confusion Matrix
  - 특정 클래스(무승부 등)의 편중될 경우 클래스간의 불균형을 시각화하여 파악한다.
  - precision/recall도 함께 확인하여 보완 전략을 결정한다.

다음 기능들을 사용하여 모델의 예측 결과의 원인을 파악하여 모델의 정확도를 증가시킨다.

기능	설명
Feature Importance	예측에 영향을 준 feature 순위를 시각화 (bar chart)
SHAP 분석	개별 예측에 기여한 feature 기여도를 정량화
실패 사례 분석	예측 실패 경기 수집 및 오류 원인 분석

### 3-7. API 배포

모델의 API 배포는 AWS SageMaker를 사용하여 배포한다. 배포하는 과정은 다음과 같다.

(SageMaker API 배포 절차)

i. 학습된 모델을 .tar.gz로 압축하여 S3에 업로드

ii. SageMaker에 모델 등록

예시 코드
<pre>from sagemaker import Model  model = Model(     image_uri='xgboost 이미지 URI',     model_data='s3://your-model-path/model.tar.gz',     role='SageMakerExecutionRole', )</pre>

### iii. Endpoint로 배포

#### 예시 코드

```
predictor = model.deploy(  
    initial_instance_count=1,  
    instance_type='ml.m5.large',  
    endpoint_name='my-predict-endpoint'  
)
```

### iv. API 호출 예시

#### 예시 코드

```
result = predictor.predict([[feature1, feature2, feature3]])  
  
POST https://runtime.sagemaker.aws/.../endpoints/my-predict-  
endpoint/invocations  
Content-Type: application/json  
  
{  
    "instances": [[1.2, 2.3, 3.4]]  
}
```

## 4. 비정형 데이터 모듈

### 4-1. 개요

공신력 높은 기사나 트위터에서 비정형 데이터를 수집하고 가공한 후 해당 경기에 대한 전문가들의 예측과 해당 경기에서의 예상 key player를 출력시켜주는 모듈

### 4-2. 비정형 데이터

#### 4-2-1. 전문가 예측

##### (a) 기사/트위터 수집

공신력 높은 기사와 트위터를 수집한 뒤 데이터베이스에 저장한다.

ex) 트위터 내용 수집

##### 예시 코드

```
import snsrape.modules.twitter as sntwitter

query = '토트넘 맨시티 예측 since:2024-08-01 until:2024-08-05'
tweets = [tweet.content for i, tweet in
            enumerate(sntwitter.TwitterSearchScraper(query).get_items()) if i < 5]
```

##### (b) 텍스트 정제

수집한 텍스트에서 불필요한 요소들을 제거해주기위해 정제 과정을 거친다.

##### 예시 코드

```
import re

💡 적용 예시
python
복사
편집
cleaned_tweets = [clean_text(t) for t in tweets]
article_text = " ".join(cleaned_tweets)
```

함수명	clean_text(text)
코드	<pre>def clean_text(text):     # 개행 제거 및 공백 정리     text = text.replace("\n", " ").strip()     # 링크 제거     text = re.sub(r"http\S+ www.\S+", "", text)     # 멘션 제거     text = re.sub(r"@w+", "", text)     # 해시태그 제거 (필요 시)     text = re.sub(r"#w+", "", text)     # 이모지 제거     text = re.sub(r"[^\w\s,.!?가-힣A-Za-z]", "", text)     # 중복 공백 제거     text = re.sub(r"\s+", " ", text)     return text.strip()</pre>
입력값	크롤링해온 기사나 트윗의 text
출력값	정제된 text
설명	크롤링하여 수집한 기사나 트윗의 text에서 HTML 태그, 특수 기호, 여러 줄 개행, 공백 정리 등의 과정을 거친 후 정제된 text를 반환한다.

### (c) 감성 분석

HuggingFace Transformers 기반으로 정제된 txt에 대한 감성 분석을 진행한다.

예시 코드
<pre>from transformers import AutoTokenizer, AutoModelForCausalLM, pipeline  # 모델 불러오기 model_id = "NousResearch/Llama-2-7b-chat-hf" tokenizer = AutoTokenizer.from_pretrained(model_id) model = AutoModelForCausalLM.from_pretrained(model_id)  pipe = pipeline("text-generation", model=model, tokenizer=tokenizer)  # 프롬프트 설계 prompt = f""" 아래 기사를 종합적으로 읽고, 전문가들의 의견이 '홈팀 승', '원정팀 승', '무승부' 중 어디에 가까운지 판단해줘.  응답은 반드시 아래 형식만 사용해줘. 예측: [홈팀 승 / 원정팀 승 / 무승부] 이유: [한 문장 요약]  기사: \`\`\` {article_text}</pre>

```
\\\"\\\"\\\"\\\"  
\"\"\"
```

```
output = pipe(prompt, max_new_tokens=100, do_sample=False)  
print(output[0][\"generated_text\"])
```

#### (d) 결과 파싱

##### 예시 코드

```
# 단순 파싱 예시  
response = output[0][\"generated_text\"]  
if \"홈팀 승\" in response:  
    result = \"home\"  
elif \"원정팀 승\" in response:  
    result = \"away\"  
elif \"무승부\" in response:  
    result = \"draw\"  
else:  
    result = \"unknown\"
```

#### (e) 여러 기사 결과를 통계화

##### 예시 코드

```
from collections import Counter  
  
results = [\"home\", \"home\", \"draw\", \"away\", \"home\"] # 여러 기사에서 나온 결  
과라고 가정  
count = Counter(results)  
  
# 비율 계산  
total = sum(count.values())  
stats = {  
    \"home_win_ratio\": round(count[\"home\"] / total, 2),  
    \"draw_ratio\": round(count[\"draw\"] / total, 2),  
    \"away_win_ratio\": round(count[\"away\"] / total, 2)  
}  
print(\"전문가 예측 감성 통계:\", stats)
```

## 4-2-2. key player 정보

특정 경기에서 가장 영향력 있을 것으로 예상되는 key player를 최근 경기 평점과 긍정 기사 비율을 기반으로 정량적으로 선정한다.

- key player 산정 공식

$\text{key\_score} = \text{최근 } n\text{경기 평점 평균} + \alpha \times \text{긍정 기사 비율}$

( $n$ 값과  $\alpha$ 값은 예측 결과의 정확도에 따라 조정될 수 있다.)

### (a) 최근 $n$ 경기 출전 선수별 평점 평균 계산

선수의 최근 경기 폼을 수치화하기 위해 선수별 최근  $n$ 경기에 대한 평점 데이터를 기반으로 평점 평균을 계산한다.

#### 예시 코드

```
def get_recent_avg_rating(player_name, rating_db, n=3):
    ratings = rating_db.get(player_name, [])
    return round(sum(ratings[:n]) / len(ratings[:n]), 2) if ratings else 0.0
```

### (b) key player 후보군

감성 분석에 대상으로 삼을 핵심 선수 후보군을 평점 평균을 기준으로 상위  $n$ 명을 추천한다.

#### 예시 코드

```
top_candidates = sorted(players, key=lambda x: x["avg_rating"],
reverse=True)[:n]
```

### (c) 후보 선수별 + 해당 경기 관련 기사/트윗 수집

선수 이름 + 경기 키워드 기반 기사/트위터를 수집하여 감성 분석에 활용할 txt 데이터를 확보한다.

#### 예시 코드

```
query = f'{{player_name}} {{home_team}} {{away_team}} 경기'
# → 뉴스 크롤링 or snsrape로 트윗 수집
```

#### (d) LLaMA를 활용한 선수별 감성 분석

해당 선수에 대한 전반적인 평가가 긍정인지 부정인지 LLaMA모델을 활용하여 감성 분석을 진행한다. 이때 기사 원문을 통으로 넣어 해당 기사에서의 선수에 대한 감성 분석을 진행한 후 감성 분석 결과를 점수로 변환한다.

긍정:1.0, 부정:0.0, 중립/모름:0.5

##### 예시 코드

```
prompt = f"""
다음 기사에서 '{player_name}'에 대한 평가가 긍정인지 부정인지 판단해줘.
응답: {{'{player_name}': "긍정"}} 또는 {{'{player_name}': "부정"}}
기사: \\\\" {article_text} \\\\"
"""
```

#### (e) key player score 계산한다.

처음 설정하였던 key player 공식을 사용하여 해당 선수의 key player score를 계산한다.

##### 예시 코드

```
key_score = avg_rating +  $\alpha$  × sentiment_score
```

#### (f) key\_score 기준 정렬 → 최종 key player 선정

key player로 출력할 n명을 최종으로 선정한다.

##### 예시 코드

```
top_key_players = sorted(key_players, key=lambda x: x["key_score"],
reverse=True)[:n]
```

## 5. 해설 모듈

gpt 모델을 통해 해당 경기에 대한 예측 결과값에 대한 근거, 전문가의 예측 결과, key player를 자연어로 소비자에게 설명해준다.

### 5-1. 과정

#### (a) 예측 결과 도출

예측 모델에서 출력값으로 내놓은 클래스와 예측 확률을 추출한다.

##### 예시 코드

```
proba = model.predict(X_input)[0] # 예: [0.73, 0.2, 0.07]
prediction_label = classes[np.argmax(proba)] # 예: 'home'
prediction_prob = max(proba) # 예: 0.73
```

#### (b) 중요 feature 상위 k개 추출

SHAP 값을 기반으로 예측 모델에 예측에 가장 큰 영향을 준 상위 k개의 feature를 추출한다.

##### 예시 코드

```
explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X_input)[np.argmax(proba)]
top_k_idx = np.argsort(np.abs(shap_values))[:-1][:k]
top_k_features = [(feature_names[i], X_input[0][i]) for i in top_k_idx]
```

#### (c) 전문가 예측 결과 + key player 입력으로 받음

##### 예시 코드

```
expert_sentiment = {
    "home": 0.65,
    "draw": 0.25,
    "away": 0.10
}

key_players = {
    "home": [{"name": "손흥민", "score": 9.3}, {"name": "케인", "score": 8.8}],
    "away": [{"name": "더브라워너", "score": 9.0}, {"name": "홀란드", "score": 8.7}]
}
```



#### (d) GPT 입력 생성 함수

별도 가공 없이 그대로 key\_players 딕셔너리를 GPT 입력 포맷에 포함시키면 된다.

##### 예시 코드

```
def make_gpt_input(
    prediction_label: str,
    prediction_prob: float,
    shap_top_features: list,
    expert_sentiment: dict,
    key_players: dict
) -> dict:
    ...

gpt_input = make_gpt_input(prediction_label, prediction_prob,
top_k_features, expert_sentiment, key_players)
```

#### (e) GPT 요청 prompt

##### 예시 코드

```
def explain_prediction_with_gpt(input_dict, api_key):
    formatted = json.dumps(input_dict, indent=2, ensure_ascii=False)
    prompt = f"""
다음은 축구 경기 예측 결과입니다. 아래 데이터를 기반으로,
모델이 왜 이런 예측을 했는지를 자연스럽게 설명해줘. 핵심만 3~5문장으로
요약해줘.

{formatted}
"""
    response = openai.ChatCompletion.create(
        model="gpt-4-turbo",
        messages=[{"role": "user", "content": prompt}],
        temperature=0.7,
        max_tokens=500
    )
    return response['choices'][0]['message']['content']
```

#### (출력 예시)

모델은 홈팀 승리를 예측했습니다 (확률: 73%). 이는 홈팀 평균 평점이 높고, 전문가 의견의 65%가 홈팀 승리를 점쳤기 때문입니다. 또한 손흥민과 케인 같은 핵심 선수들이 포함되어 있어 전력 우세가 예상됩니다. 이러한 요소들이 종합적으로 반영되어 홈팀 승리 확률이 높게 예측되었습니다.

## 6. 웹 서비스 설계

다음과 같은 이유로 <PREMO>를 web으로 구현한다.

근거	설명
접근성	별도의 앱 설치 없이, 브라우저에서 링크만으로 바로 서비스 이용의 편의를 향상시키기 위해 web으로 구현한다.
개발 리소스	Android, iOS 각각 따로 앱을 만들 필요 없이 하나의 코드베이스로 전체 사용자를 커버할 수 있기 때문이다.
검색 엔진과의 연동	SEO 최적화 가능, 사용자 확보 측면 효과적 ex) 구글과 같은 검색 엔진을 통해 '맨시티 vs 리버풀' 같은 키워드로 유입 가능

### 6-1. 서버

#### 6-1-1. 서버 사양 및 환경

항목	내용
EIP	추후 기입
DNS	추후 기입
OS	ubuntu 20.04 LTS
Instance	AWS EC2 t3.medium
EBS 크기	8 GB
웹 서버	Nginx 1.18
배포	Docker / Docker Compose or PM2
Key Pair	추후 기입

#### 6-1-2. Port Information

포트 번호	설명
22	프론트엔드 작업 포트
80	백엔드 작업 포트

## 6-2. API 명세

### 6-2-1. Server Information

항목	내용
Base URL	추후 기입
Method	REST API
Auth 방식	jwt

### 6-2-2. API Description

설명	경기 일정 정보 제공 API		
Page	Main Page	End Point	/matchSchedule
Method	GET		
param	{ matchDate: string // "YYYY-MM-DD" }		
response	{ teamName: string matchTime: string // "HH:MM" teamImage: image league: string }		

설명	특정 경기 정보 제공 API		
Page	Main Info Page	End Point	/matchInfo
Method	GET		
param	{ matchId: number }		
response	{ teamName: string matchTime: string // "HH:MM" matchVenue: string teamImage: image league: string winProbability: number // 홈팀 기준 betting: number // 홈팀 기준 recentMatches: { opponent: string result: string score: string // "N-N" } }		

설명	경기 정보 요약 및 인사이트 제공 API		
Page	Main Preview Insight Page	End Point	/matchInsight
Method	GET		
param	{ matchId: number }		
response	{ teamName: string matchTime: string // "HH:MM" matchVenue: string teamImage: image league: string homeInsight: string awayInsight: string newsSummary: string }		

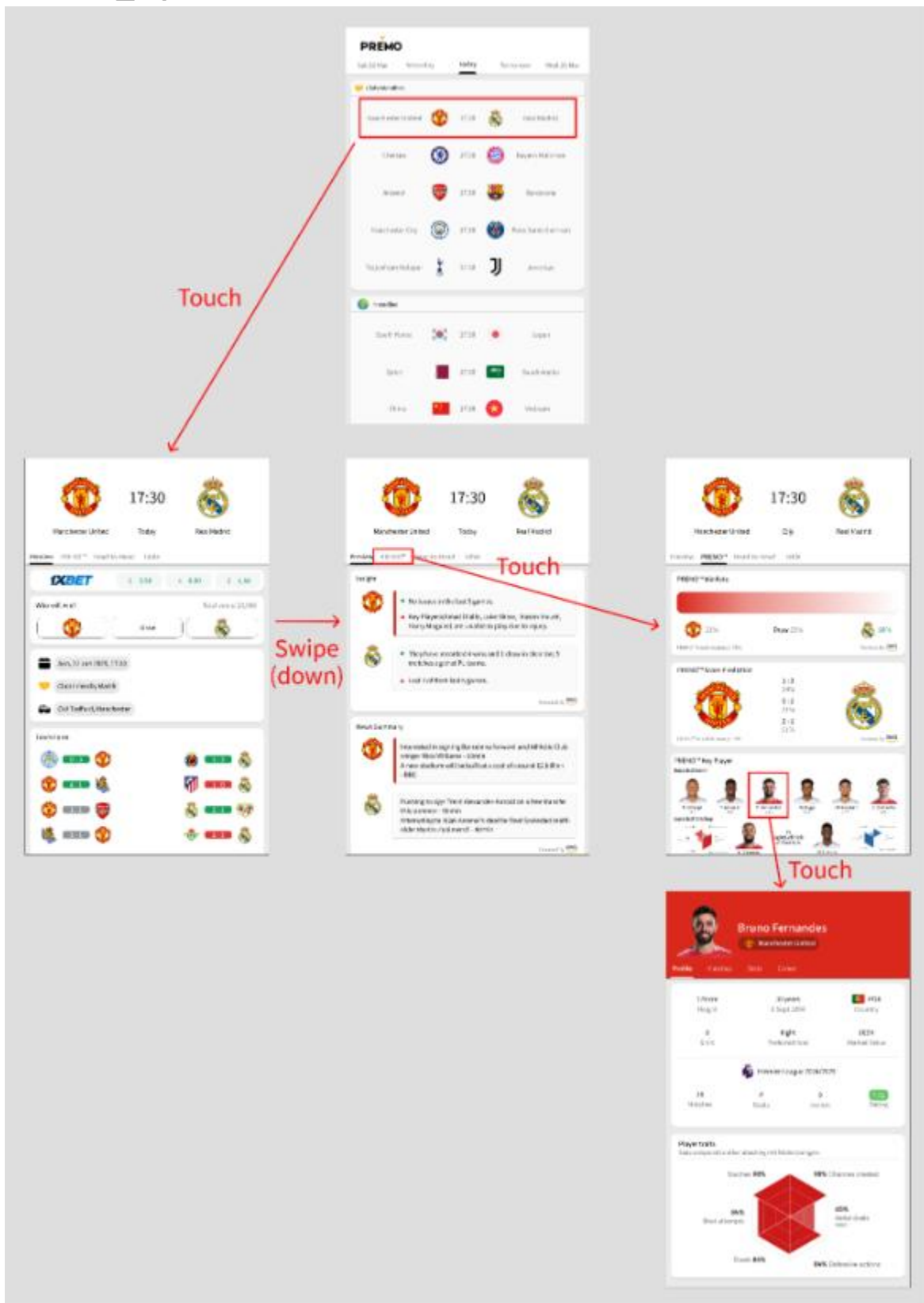
설명	경기 세부정보, 예측, 팀 데이터 분석 정보 제공 API		
Page	Main Detail Page	End Point	/matchDetail
Method	GET		
param	{ matchId: number }		
response	{ teamName: string matchTime: string // "HH:MM" matchVenue: string teamImage: image league: string winProbability: number // 홈팀 기준 predictedScore: string // "N-N" matchHistory: { result: string score: string // "N-N" } keyPlayer: { goal: string touches: number shotAttempts: number chancesCreated: number defensiveActions: number } }		

설명	선수 세부정보 제공 API		
Page	Match Player Page	End Point	/matchPlayer
Method	GET		
param	<pre>{   matchId: number   playerId: number }</pre>		
response	<pre>{   playerName: string   teamName: string   playerImage: Image   playerBirth: string // "YYYY-MM-DD"   playerNationality: string   playerHeight: number   playerPreferredFoot: string   playerBackNumber: number   appearances: number // 경기수   goals: number   assists: number   averageRating: number }</pre>		

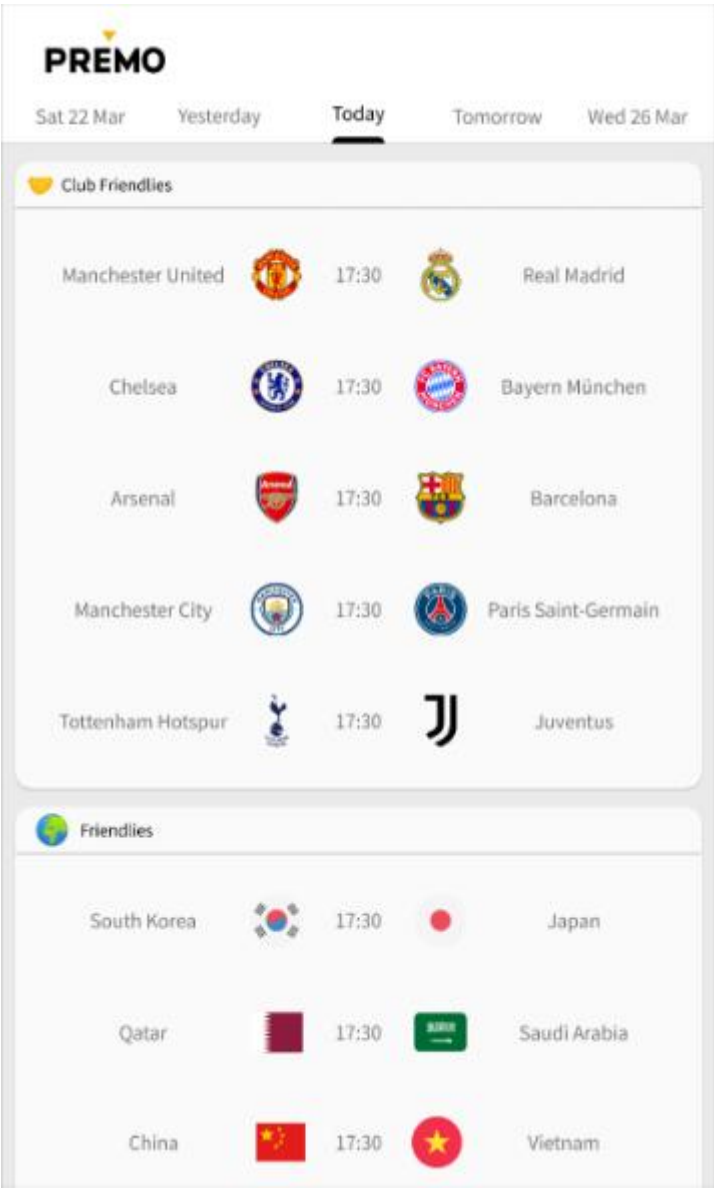
설명	역대 상대전적 분석 정보 제공 API		
Page	Match Head to Head Page	End Point	/matchHeadToHead
Method	GET		
param	<pre>{   matchId: number }</pre>		
response	<pre>{   teamName: string   matchDate: string // "YYYY-MM-DD"   matchVenue: string   teamImage: image   league: string   score: string // "N-N"   home: string   away: string }</pre>		

설명	리그 정보 제공 API		
Page	League Standings Page	End Point	/leagueInfo
Method	GET		
param	{ leagueId: number }		
response	{ league: string leagueInfo: { rank: number rank: number teamName: string wins: number draws: number losses: number goalsFor: number goalsAgainst: number goalDifference: number } }		
























## 6-3. UI 설계




### 6-3-1. 컴포넌트

페이지 이름	Match Selection Page(Main Page)
페이지 설명	날짜 별, 리그 별 경기 확인 페이지
	
기능 상세	
<ul style="list-style-type: none"> <li>- 날짜별 경기 탐색 기능 제공.</li> <li>- 하단 스와이프를 통해 추가 경기 load.</li> <li>- 경기 리스트 카테고리별 섹션 구분.</li> </ul>	




페이지 이름	Match Info Page
페이지 설명	해당 경기 팀, 날짜, 위치 정보 전달
<div><div><div><div><div></div><div>Manchester United</div></div><div><div>17:30</div><div>Today</div></div><div><div></div><div>Real Madrid</div></div></div><div><div>Preview</div><div>PREMO™</div><div>Head-to-Head</div><div>Table</div></div><div><div>1XBET</div><div>1 3.50</div><div>x 6.00</div><div>2 1.50</div></div><div><div>Who will win?</div><div>Total votes: 20,086</div><div><div></div><div>Draw</div><div></div></div></div><div><div> Sun, 22 Jun 2025, 17:30</div><div> Club Friendly Match</div><div> Old Trafford, Manchester</div></div><div><div>Team Form</div><div><div><div> 0-3 </div><div> 4-1 </div><div> 1-1 </div><div> 1-1 </div></div><div><div> 1-2 </div><div> 1-0 </div><div> 2-1 </div><div> 2-1 </div></div></div></div></div></div>	
기능 상세	
<ul style="list-style-type: none"><li>- 기본 경기 정보 제공. (팀, 경기 시간, 경기 날짜).</li><li>- 배당률 정보 제공. (제공 업체, 1/X/2 방식, 배당률 표시).</li><li>- 유저 투표 기능 제공. (투표 후 결과 비율 시각화)</li><li>- 경기 참여 팀 최근 경기 결과 정보 제공. (4~5 경기 결과 나열, 경기 클릭 시 경기 상세 정보 페이지로 이동)</li></ul>	

페이지 이름	Match Preview Insight Page
페이지 설명	해당 경기 정보 요약 및 인사이트 제공 페이지




17:30




Manchester UnitedTodayReal Madrid

[Preview](#)
[PREMO™](#)
[Head-to-Head](#)
[Table](#)


### Insight




- No losses in the last 5 games.
- Key Players(Amad Diallo, Luke Shaw, Mason Mount, Harry Maguire) are Unable to play due to injury.




- They have recorded 4 wins and 1 draw in their last 5 matches against PL teams.
- Lost 2 of their last 5 games.

Powered By 


### News Summary



Interested in signing Barcelona forward and Athletic Club winger Nico Williams - 90min  
A new stadium will be built at a cost of around £2 billion - BBC



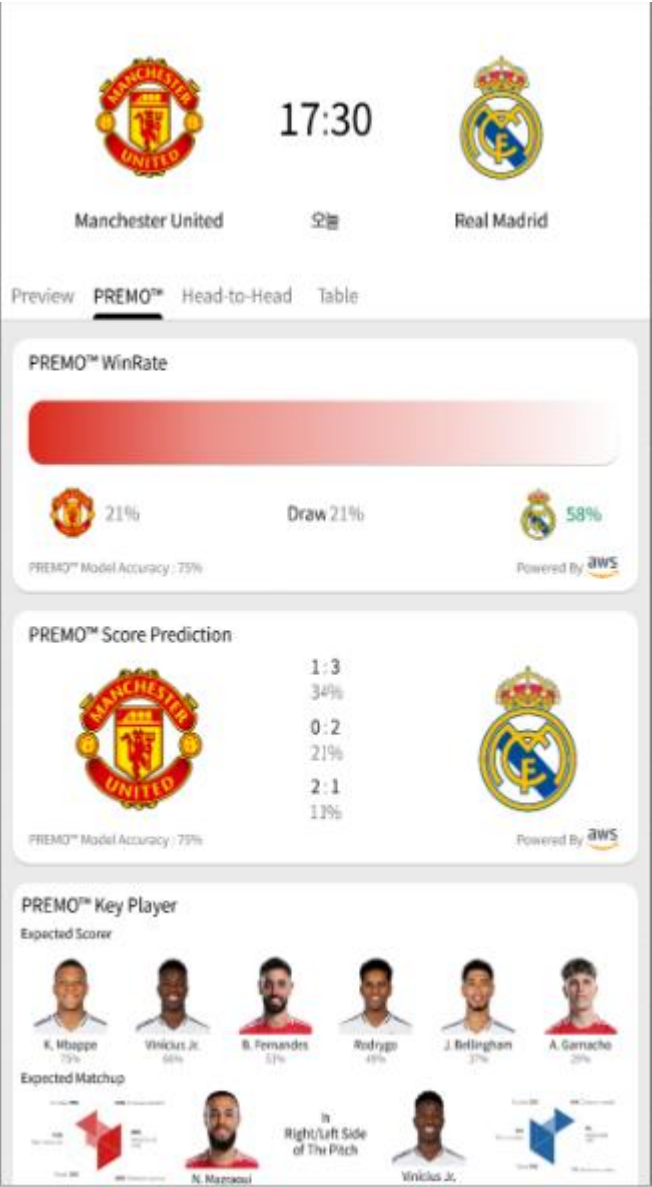
Pushing to sign Trent Alexander-Arnold on a free transfer this summer - 90min  
Attempting to hijak Arsenal's deal for Real Sociedad midfielder Martin Zubimendi - 90min

Powered By 

## 기능 상세

- 기본 경기 정보 제공. (팀, 경기 시간, 경기 날짜).
- Match Fact 카드 제공. (최근 경기 흐름, 리스크 정보, 전력 비교)
- 양 팀 관련 최신 이슈 주요 뉴스 정보 요약 제공.
- 뉴스 정보 클릭 시 원문 기사 하이퍼링크 연결

페이지 이름	Match Detail Page
페이지 설명	경기 예측 및 팀 데이터 분석 정보 전달



### 기능 상세

- 기본 경기 정보 제공. (팀, 경기 시간, 경기 날짜).
- 승부 확률 예측 정보 제공. (승리, 무승부 확률, 모델 정확도)
- 점수 예측 정보 제공. (예측 스코어 후보, 확률 수치, 모델 정확도)
- Key Player 분석 정보 제공. (예상 득점자, 기여도 수치)
- 경기 내 주요 포지션 매치업 예측 및 선수 능력 비교 레이더 차트 제공.

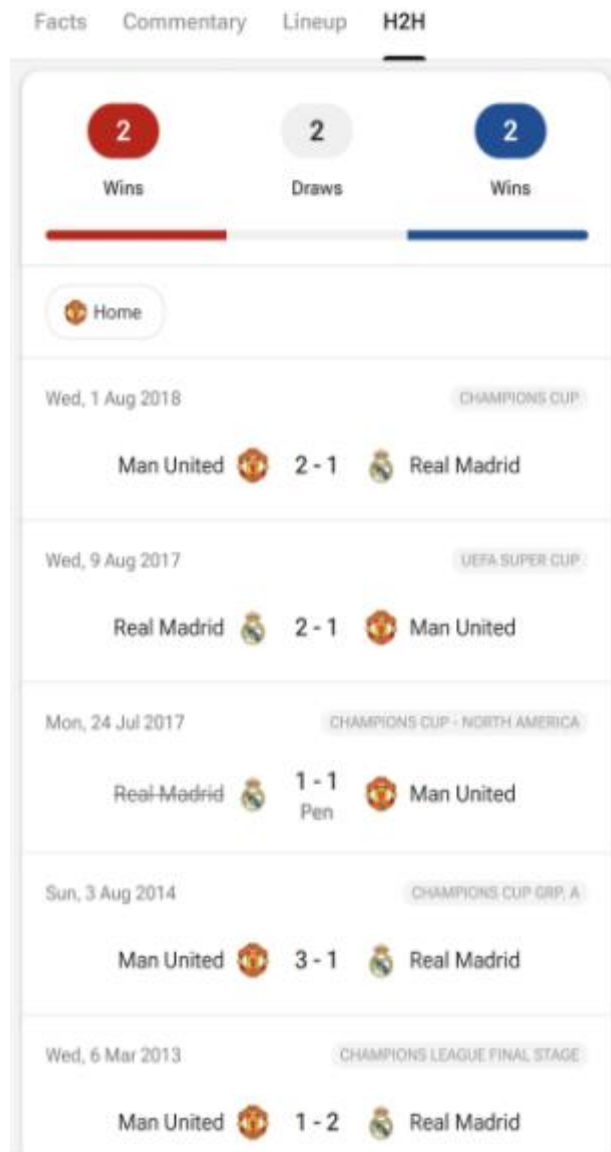
- Score prediction 클릭 시 역대 상대전적 분석 정보 페이지 이동.	
페이지 이름	Match Player Page
페이지 설명	선수 세부정보 전달 페이지



기능 상세

- 선수 프로필 정보 제공. (선수 이름, 소속 팀, 사진, 생년월일, 국적, 신체정보, 시장가치)
- 시즌 기록 정보 제공. (경기 수, 골 수, 어시스트 수, 평균 평점)
- 리그 로고 클릭 시 드롭다운을 통해 시즌 별 스탯 정보 선택 가능.
- 포지션 대비 선수 능력치 차트 백분위 표시 제공.(볼 터치 빈도, 기회 창출 수, 슈팅 횟수, 득점률, 공중볼 경합, 수비 관여도)
- 마우스오버 시 수치 설명 툴팁 가능.

페이지 이름	Match Head to Head Page
페이지 설명	역대 상대전적 분석 정보 페이지




### 기능 상세


- 전적 요약 상단 표시.
- 경기별 H2H 리스트 표시 (경기 날짜, 대회명, 홈/원정 팀 이름과 로고, 스코어 및 결과, 승부차기 결과 표시 여부.)
- 전적 클릭 시 경기 상세 페이지 이동.

페이지 이름	League Standings Page
페이지 설명	리그 순위 정보 페이지

←



04:00




⋮














Preview

Table

H2H



Premier League

#	Team	Pl	W	D	L	+/-	GD	Pts
1	 Liverpool	29	21	7	1	69-27	+42	70
2	 Arsenal	29	16	10	3	53-24	+29	58
3	 Nottm Forest	29	16	6	7	49-35	+14	54
4	 Chelsea	29	14	7	8	53-37	+16	49
5	 Man City	29	14	6	9	55-40	+15	48
6	 Newcastle	28	14	5	9	47-38	+9	47
7	 Brighton	29	12	11	6	48-42	+6	47
8	 Fulham	29	12	9	8	43-38	+5	45
9	 Aston Villa	29	12	9	8	41-45	-4	45
10	 Bournemouth	29	12	8	9	48-36	+12	44
11	 Brentford	29	12	5	12	50-45	+5	41
12	 Crystal Palace	28	10	9	9	36-33	+3	39
13	 Man United	29	10	7	12	37-40	-3	37

기능 상세

- 해당 경기 참여 팀 음영 처리.
- 하단 스와이프 시 하위 순위 팀 load.
- 리그 클릭 시 상단 드롭다운을 통해 리그 선택 가능.

## 7. WBS

[illegible]