# Code Quality Review report

# IFoundMovie

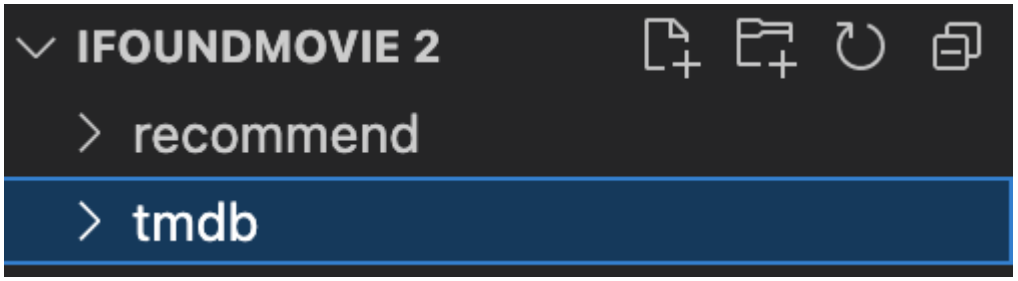
Xufeng Liu, Yixu Zhou

## 1. Introduction

The running of IFoundMovie is mainly divided into the front end and back end. The front end includes static UI templates and various interactive interfaces. The back end contains the database and API interface. The front end mainly uses HTML, CSS and PHP language, and the back end mainly uses Java and Python. This report carries out a quality assessment from the following aspects.

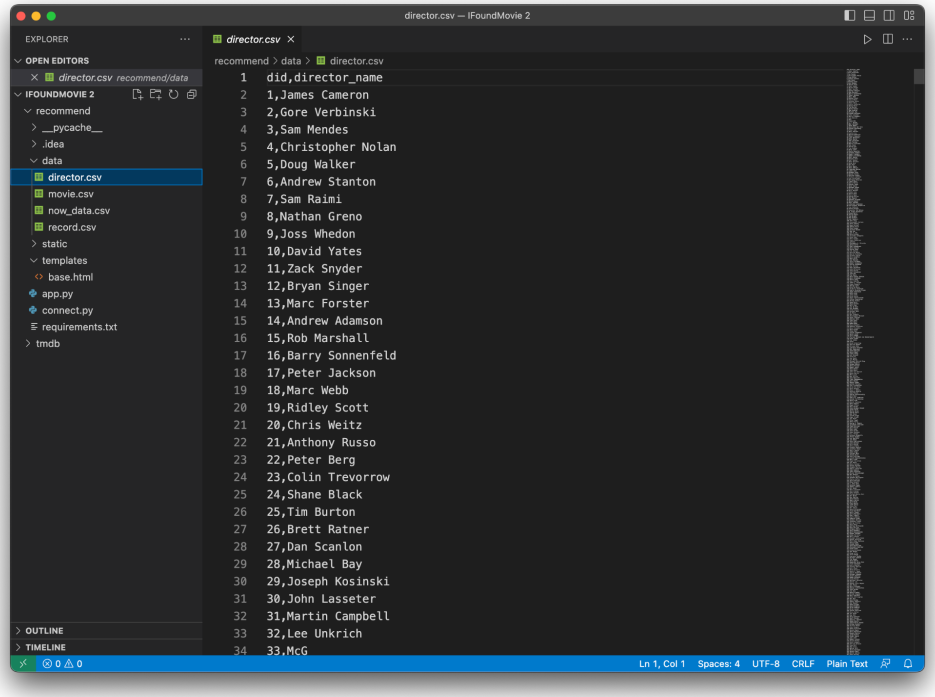
## 2. Specific Sections

### 2.1. Integral part




All of our code is divided into two sections: recommend, which mainly stores the AI algorithm section and various data sets, and tmdb, which mainly processes the data obtained from tmdb.

## 2.2. Data Set Part




In recommend/data, you can see that we have four CSV-type data sets that store director data, movie data, id data, match value data, and converted value. This approach combined with the code to our background data more convenient and quick management.

## 2.3. Code Style

```
        data = pd.DataFrame(nowData)
        pd.DataFrame(nowData).to_csv('./data/movie.csv', index=False, mode='a+', head

    if bl_record==False:
        count=1
        nowData = ['']
        nowData[0] =(did,mid,count)
        data = pd.DataFrame(nowData)
        pd.DataFrame(nowData).to_csv('./data/record.csv', index=False, mode='a+', hea
    else:
        recordList[now_index]=[did,mid,count]
        data = pd.DataFrame(data=recordList)
        pd.DataFrame(data).to_csv('./data/record.csv', index=False)
    return did

# Get the most similar director information
def topn_simusers(did, n,pd_users):
    users = pd_users.loc[did,:].sort_values(ascending = False)
    topn_users = users.iloc[:n,]
    topn_users = topn_users.rename('score')
    #print("Similar actor as actor:", did)
    return topn_users

# Build matrix
def createDataFrame():
    path = './data/now_data.csv'
    ratings = pd.read_csv(path, names=None)
    rp = ratings.pivot_table(columns='mid', index='did', values='matching')
    rp = rp.fillna(0)
    rp_mat = np.matrix(rp)
    m, n = rp.shape
    mat_users = np.zeros((m, m))  # Two dimensional array, put the small one here
    # Movie director similarity matrix (1- corrected cosine similarity = corrected co
    for i in range(m):
```

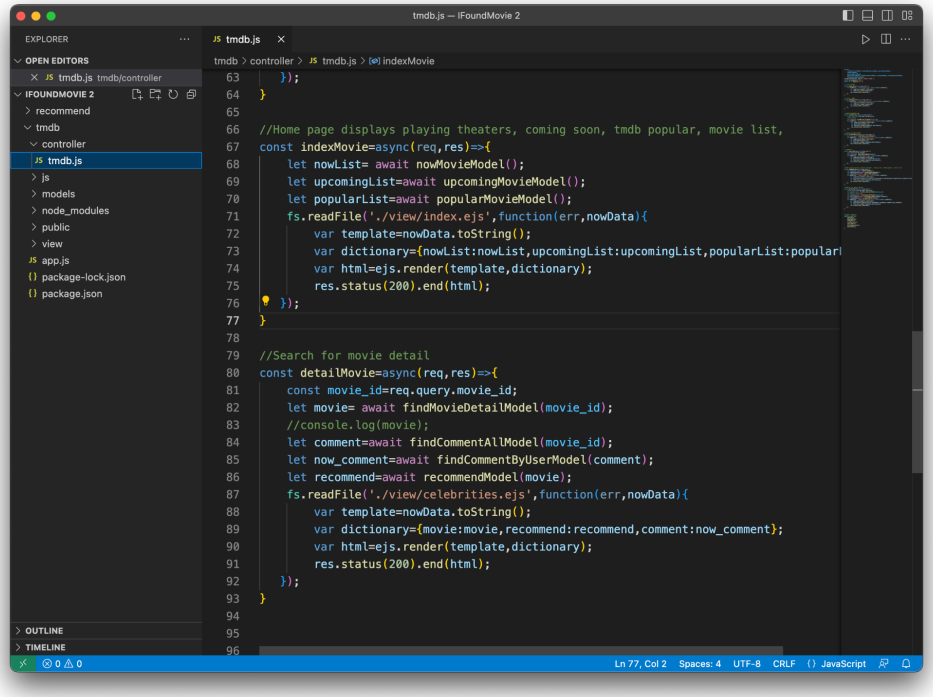
In our main code, we adopt the way of rich comments, which not only provides a supplementary description for a certain code but also facilitates our future modification and debugging.

## 2.4. Requirement

```
Flask==2.2.2
mysql-connector==2.2.9
numpy==1.23.4
pandas==1.4.4
PyJWT==2.6.0
requests==2.28.1
scipy==1.9.1
```

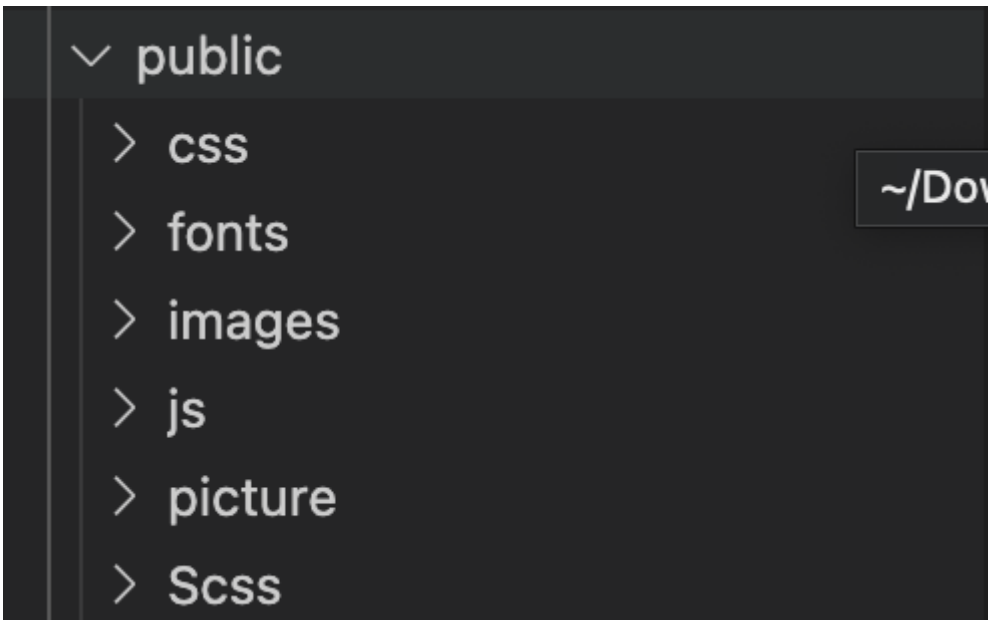
We also generate the requirement.txt to explain the version requirements for some of the necessary plug-ins to run.

## 2.5. TMDB Controller




In TMDB.js, we centrally manage the data obtained from tmdb to plan the relevant movie information displayed on each plate of the home page, which is more convenient for the front-end operation.

### 2.6. UI Design




In the public folder, we manage all the UI design and user interaction elements that are the first things that users come across when they use our site, so we pay a lot of attention to colour matching and styling.

## 3. Readability

Our code is almost always commented in important places to enhance readability, which makes it easier for us to manage and access, and also brings the viewer more detailed instructions. At the same time, the names of variables and equations are very specific and appropriate, which can be associated with their corresponding meanings and responsibilities at first sight. camelCase is also a standard we follow closely, which makes the code look clean and professional and readable.