Ostbayerische Technische Hochschule Regensburg
Faculty for Mathematics and Computer Science

IMDB Software of Hollywood Actors and Actresses

Applied Data Science with Python

November 25, 2021

Author: Anna Breithaupt
MatNr. 3202199

Contents

1. Task 1 . . . . . **1**
1.1. Short Description of the Problem 1
1.2. General Approach 1
1.3. Tools 1
1.3.1. Documentation 2
1.3.2. Scraping, Storing and Processing the Data 2
1.3.3. User Interface 2
1.4. Algorithms and Data Structures 2
1.4.1. Scraping and Storing the Data 2
1.4.2. Processing und Evaluation 3
1.5. Modules 3

A. Project Structure **4**

References **5**

1. Task 1

1.1. Short Description of the Problem

The goal of the project is to create a user-friendly application that provides information about the top 50 actors and actresses.
The information about the movies can be accessed through the IMDb page of the 50 most popular actors (Top 50 Popular Hollywood Actors and Actresses | IMDb, 2013).

1.2. General Approach

To implement this task, I want to scrape the required information from the website, sort it and then store it in a database.

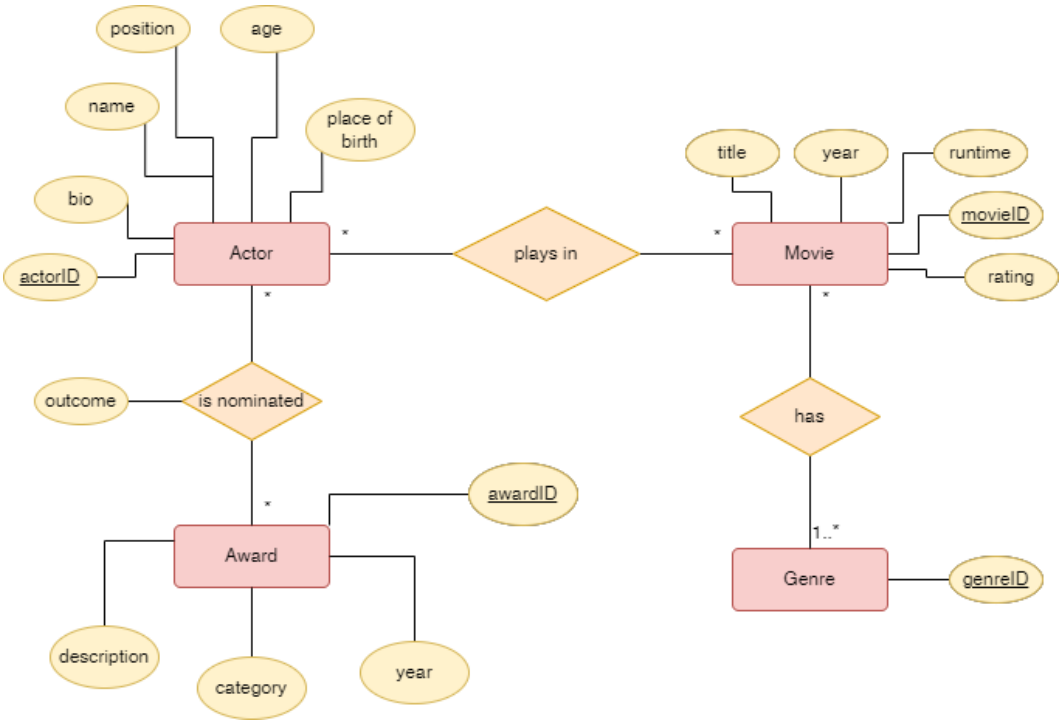

Figure 1: Entity Relationship Sketch

I considered the schema of the database for the beginning as seen in figure 1.
The stored data can then be used to retrieve specific information, statistics as well as to create graphs. The data obtained with this will be displayed to the user in a web application.

1.3. Tools

As development environment I will use **PyCharm**. Besides refactoring and debugging possibilities PyCharm also supports me in the organization and structuring of my project.
As versioning tool I will use **git**, whereby PyCharm also supports versioning with git.

1.3.1. Documentation

For documentation of the code I plan to use **Sphinx**, if more diagrams are needed for planning I will use **draw.io**. Additionally I will use **LaTeX** and **TeXstudio** to write the documentation.

1.3.2. Scraping, Storing and Processing the Data

To get the data from the website I plan to use **Beautiful Soup**. In order to find the right information I will also use the **Google Inspector Tool** provided by Chrome to locate the HTML tags and classes.
The data then will be stored in a database, for which I will use **MySQL**.
To process the data, I will access **Numpy** and **Pandas**. Since I also want to plot, I need **matplotlib.pyplot** as well.

1.3.3. User Interface

For the user to view the data in a descriptive manner, I will use a web interface. For this I utilize the web framework **Django**, **HTML5** and for the graphical elaboration **CSS** and **Bootstrap**.

1.4. Algorithms and Data Structures

1.4.1. Scraping and Storing the Data

The IMDb page lists the top 50 actors (Top 50 Popular Hollywood Actors and Actresses | IMDb, 2013). For each actor a detail page with further information is linked, where query parameters are used. To get the information I need, I thought of the following algorithm. After I have found the first actor, I search for the corresponding query parameter. This will also be used later in my database as ID for the actor. The query takes me to the details page to get all further information about the actor and his awards. On the details page all movies of the actor are listed. The movies are linked again and I can repeat the procedure for the movies. So if the movie is not yet saved, it will be persisted in the database. The movie now can be linked to the actor. The procedure is repeated for all other actors subsequently.
I will mostly use the **find** and **find_all** function of Beautiful Soup.

The information I find with **JSON** formats can be stored in a **dictionary**. The rest of the information can be added to the dictionary and then simply stored in the database using a function. In addition to the lector materials I also used realpython.com as source of information (Beautiful Soup: Build a Web Scraper With Python | Real Python, 2021).

1.4.2. Processing und Evaluation

For the evaluation of the data I would like to use **diagrams**, among other things. To display the awards of the actors I would like to use a **histogram**. So the difference between nominations and wins can be shown well. A **scatter plot** can be used for the average rating of the films in the respective years. The genres of the movies could be shown well in a pie chart or in a **wordcloud mask**. For this I can use different methods from the matplotlib.pyplot.

For returning all actors or movies I will use **lists**, which will then be sorted with a **QuickSort algorithm**.

1.5. Modules

In order to think of suitable modules, I first thought about how to structure the project. For this I roughly fall back on the three layer model, where an application layer, a persistence layer and a presentation layer is implemented. Additionally I was inspired by Hitchhiker's Guide to Python(Structuring Your Project | Hitchhiker's Guide to Python).
In appendix A you can see a rough logic, how I plan to structure the project with corresponding modules.

A. Project Structure

```
imdb/
|
├── bin/
|
├── webapplikation/
|      ├──app/
|      |      ├── __init__.py
|      |      ├── admin.py
|      |      ├── apps.py
|      |      |
|      |      ├── migrations/
|      |      |      └── __init__.py
|      |      |
|      |      ├── models.py
|      |      ├── tests.py
|      |      └── views.py
|      |
|      ├──docs/
|      |
|      ├──project/
|      |      ├── __init__.py
|      |      ├── settings.py
|      |      ├── urls.py
|      |      └── wsgi.py
|      |
|      ├── static/
|      |      └── style.css
|      |
|      ├── templates/
|      └──   └── base.html
|
|
├── application/
|      ├── __init__.py
|      ├── runner.py
|      ├── scrape/
|      |      ├── __init__.py
|      |      ├── scrape.py
|      |      └── save_information.py
|      |
|      ├── processinformation/
|      |      ├── __init__.py
|      |      ├── process_information.py
|      |      └── get_information.py
|      |
|      ├── tests/
|      |      ├── scraping_tests.py
|      |      └── information_processing_tests.py
|      |
|      ├── docs/
|      |      ├── scrape.md
|      └──    └── processinformation.md
|
├── data/
|      ├── __init__.py
|      ├── create_databases.py
|      ├── save_to_database.py
|      ├── load_from_database.py
|      ├── docs/
|      |      └── . . .
|      ├── tests/
|      └──    └── . . .
|
├── .gitignore
├── LICENSE
└── README.md
```

References

[Rea21a] Real Python: *Beautiful Soup: Build a Web Scraper With Python*. https://realpython.com/beautiful-soup-web-scraper-python/. Version: 11 2021

[Rea21b] Real Python: *Get Started With Django Part 1: Build a Portfolio App*. https://realpython.com/get-started-with-django-1/. Version: 11 2021

[unk13] *Top 50 Popular Hollywood Actors and Actresses*. https://www.imdb.com/list/ls053501318/. Version: 05 2013

[unk21] *Structuring Your Project — The Hitchhiker's Guide to Python*. https://docs.python-guide.org/writing/structure/. Version: 2021