GUESS IT! An Online Multiplayer Trivia

Project report submitted to the Amrita Vishwa Vidyapeetham in partial fulfilment of the requirement for the Degree of

BACHELOR of TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Varun Musunuru AM.EN.U4CSE19336
Sai Pavan Krishna Nagam AM.EN.U4CSE19347
Venkata Praneeth Villuri AM.EN.U4CSE19358
Gopikrishna Vasantha AM.EN.U4CSE19359

AMRITA SCHOOL OF COMPUTING

AMRITA VISHWA VIDYAPEETHAM
(Estd. U/S 3 of the UGC Act 1956)
AMRITAPURI CAMPUS

KOLLAM -690525

APRIL 2023

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

AMRITA VISHWA VIDYAPEETHAM
(Estd. U/S 3 of the UGC Act 1956)

Amritapuri Campus

Kollam -690525

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **"GUESS IT! An Online Multiplayer Trivia"** submitted by **Varun Musunuru (AM.EN.U4CSE19336), SaiPavanKrishnaNagam(AM.EN.U4CSE19347), VenkataPraneethVilluri(AM.EN.U4CSE19358) and GopikrishnaVasantha(AM.EN.U4CSE19359)**, in partial fulfillment of the requirements for the award of Degree of Bachelor of Technology in Computer Science and Engineering from Amrita Vishwa Vidyapeetham, is a bonafide record of the work carried out by them under my guidance and supervision at Amrita School of Computing, Amritapuri during Semester 8 of the academic year 2022-2023.

Dr.Thushara M.G
Project Guide

Dr.Thushara M.G
Project Coordinator

Dr. Prema Nedungadi
Chairperson
Dept. of Computer Science & Engineering

Reviewer

Place : Amritapuri
Date : 24 April 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

AMRITA VISHWA VIDYAPEETHAM
(Estd. U/S 3 of the UGC Act 1956)

Amritapuri Campus

Kollam -690525

DECLARATION

We, **VarunMusunuru(AM.EN.U4CSE19336),SaiPavanKrishnaNagam (AM.EN.U4CSE19347),VenkataPraneethVilluri(AM.EN.U4CSE19358) and GopikrishnaVasantha(AM.EN.U4CSE19359)** hereby declare that this project entitled **"GUESS IT! An Online Multiplayer Trivia"** is a record of the original work done by us under the guidance of **Dr.Thushara M.G**, Dept. of Computer Science and Engineering, Amrita Vishwa Vidyapeetham, that this work has not formed the basis for any degree/diploma/associationship/fellowship or similar awards to any candidate in any university to the best of our knowledge.

Place : Amritapuri

Date : 24 April 2023

Varun Musunuru
Sai Pavan Krishna Nagam
Venkata Praneeth Villuri
Gopikrishna Vasantha

Signature of the students

Signature of the Project Guide

Acknowledgements

We would like to express our deepest and sincere gratitude to our mentor, Dr. MG Thushara ,Assistant Professor (Sr. Grd.) Amrita School of Computing, for their invaluable guidance and support throughout the whole project. Their judicious feedback and expertise certainly helped us in making this project shape up in the right direction. We would like to take this opportunity to thank the Computing department for selecting our project to be in the department as a reference to our juniors and also for their constant support and management of all the projects of our fellow mates. We value the assistance and oversight provided by our project coordinators and panel members, Aiswarya S Kumar(Assistant Professor), and Sanjay S Kumar (Faculty Associate).Their cooperation and assistance were invaluable in helping us achieve our objectives.

Finally, we extend our gratitude to all those whose work and ideas have contributed to the development of this project. We are grateful for the knowledge and insights that we have gained through this experience.

Abstract

Any game is a good game as long as it serves its purpose. On that note, the purpose of this simple yet not so simple game is to be the rendezvous for a group of people to spend quality time together and even better brush up on their movie knowledge if they appear to be movie geeks. Several movie data have been collected through credible sites like IMDB, Rotten Tomatoes, etc. And important information like the movie Description, cast & crew, and other movie details have been thoroughly extracted using web scraping, and Data sets have been formed regarding that. Based on the movie plot, Keyword Extractors do their magic and as the name suggests extract the keywords from the data we give. The basic working of the web application is that users will be given 3-4 images of the words extracted from the keyword extractors from the movie plot. Based on these images, the user has to guess the image. Certain hints like the year of release and cast are also included which takes a toll on the score. After all the rounds, the user with the highest score wins. This is the gist and basic working of the web application.

This attempt isn't just a one-dimensional way to showcase the work done but rather a platform to demonstrate the journey each went through in their learning phase. Maximal care has been taken to ensure that this touches several fields instead of completely sticking to a single field.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

We started this project with several aims and goals in mind. One was to let the game stick to its intended purpose i.e be the rendezvous for a group of players and let them have a memorable time. we have seen the trend of multiplayer games dethroning the other games. A good game doesn't need to rely on heavy graphics for it to be successful. This point has been proven by an immensely successful game called **Among Us**, which took the whole gaming community by storm. Among Us is a popular online multiplayer game developed and published by the American game studio, InnerSloth.

The game features a simple, colorful, and easygoing design, and players can customize their characters' appearance and names. Communication is an essential aspect of the game, as players must discuss and debate among themselves to identify the Impostor(s) and vote them off the spaceship before it's too late.

Among Us has become a popular sensation worldwide, with millions of players and viewers on various online platforms. It has been praised for its simple yet addictive gameplay, easy-to-learn mechanics, and its ability to foster communication and social interaction among players. A considerable

part of its success has to be credited to the global pandemic, which made people restricted to their homes. Personally experiencing boredom, has made us seek a way to find joy while staying in our homes. This game exactly fulfilled that requirement. This has become the go-to spot for all groups of friends to play together while staying in their homes. The great success of this game doesn't lie in the story or the graphics of the game, It lies in the ability to let you play with your friends. The communication part is handled by another popular cross-platform app known as **Discord**. Users can create a server(group) and join the call, which acts similarly to a video call. This project includes several stages branching to different fields like web scraping, NLP, Web Application Development, Cloud Deployment, etc.

1.1 Web Scraping

Web scraping is the process of automatically extracting data from websites using software programs called web scrapers or crawlers. These programs access the website's HTML code, extract the desired information, and store it in a structured format, such as a database or spreadsheet.

Web scraping has been used to collect movie details from various credible sources like IMDB, Rotten Tomatoes, MyDramaList, etc. Relevant information such as the name, year of release, cast,synopsis has been scraped using web scraping

1.2 Keyword Extraction

NLP stands for Natural Language Processing. It is a field of computer science and artificial intelligence that focuses on the interaction between human

language and computers. NLP aims to enable computers to understand, interpret, and generate human language in a way that is meaningful and useful for humans.NLP involves a range of techniques and algorithms for processing and analyzing natural language data, including text and speech. Some common NLP tasks include text classification, sentiment analysis, machine translation, named entity recognition, and speech recognition.NLP has many real-world applications, such as chatbots, virtual assistants, language translation, and text summarization. It is also used in industries such as healthcare, finance, and customer service to automate and improve communication with customers and clients.

Natural Language Processing(NLP) has always been a rapidly advancing field. Keyword Extraction is a part of NLP which deals with texts. It is used to identify and extract important words or phrases from a piece of text. It can be applied in various domains, such as information retrieval, text classification, and content analysis. Several keyword Extractors have been thoroughly tested and run through the 10000 movie dataset, resulting in the keywords from different algorithms and their respective scores.

1.3 Web Application

A web application is a type of software application that is accessed through a web browser over a network, typically the internet. Unlike traditional software applications that are installed directly on a computer or mobile device, web applications are hosted on a remote server and accessed through a web browser. Web applications can be used for a wide range of purposes, including e-commerce, social networking, online banking, project management, and more. They are typically designed to be platform-independent, meaning

that they can be accessed from any device with a web browser, including computers, smartphones, and tablets.

Web applications can be built using a variety of programming languages and frameworks, including HTML, CSS, JavaScript, PHP, Ruby on Rails, and more. They can also be designed to interact with databases, allowing users to store and retrieve information online. One of the key benefits of web applications is that they can be accessed from anywhere with an internet connection, making them ideal for remote work and collaboration. They are also typically easier to update and maintain than traditional software applications, as updates can be made centrally on the server and do not need to be installed on individual devices. Every game, at least every trivia game needs a dataset from which it can obtain the questions.

1.4 Cloud Computing

Cloud computing refers to the delivery of computing services, such as servers, storage, databases, networking, software, analytics, and more, over the internet or "the cloud." Rather than owning and maintaining physical data centers and servers, companies can access these services on-demand and pay for them based on usage.Cloud computing enables organizations to scale their computing resources quickly and efficiently without the need for significant capital investment in hardware and infrastructure. It also allows for greater flexibility, as businesses can easily adjust their computing resources up or down as their needs change. Additionally, cloud computing offers enhanced security and disaster recovery options, as data is stored and protected on remote servers that are constantly monitored and backed up.

Chapter 2

Problem Definition

Games & movies have been a detour from the hectic life for the majority. Blending these two is how we ended up with our game **GUESS IT!**. Some games are better when played with friends, and this sure is one of them.

The end result is a web application/game similar to movie trivia where the player is presented with 3 or 4 movie plot-related images and the player has to guess the movie based on these clues. To properly show the movie in 4 images has become a hassle, so to tackle this, the dataset comprises all the movie details like movie name, year of release, cast & crew, genre, and synopsis of the movie, Keyword Extractors have been run through the plots of the movie. Giving us the top 5 keywords/ important words from the whole plot. Now using APIs, images related to these keywords will be accessed and shown to the player. The development of this application involves various stages, from the creation of the data set to deployment on the cloud. Even though, there are several datasets containing movie details, rather than directly using them, Our own datasets have been scraped from scratch and deployed for public use through Kaggle. So, Data collection has become an important step. Rather than a simple step, It easily became a

phase. Dataset Extraction and Collection is a huge part of this project. Based on these Datasets, a web application and Game Engine are developed that takes the data from these Datasets and prompts the user with three or four relevant images related to a random movie plot from the selected Dataset. And it prompts the user to guess the movie based on these images while getting many hints like the year of release, cast & crew, Language, etc. This is the basic working principle behind the application.

Chapter 3

Related Work

3.1 Keyword Extraction

Rose,et.al.[1] developed a very compact and compressed keyword Extractor that greatly decreases the processing time called **RAKE**. It works by analyzing the text and identifying the frequently occurring words or phrases, which are then ranked based on their frequency and relevance.

Campos,et.al.[2] designed a widely used keyword Extractor called **YAKE**. YAKE! is a lightweight unsupervised automatic keyword extraction method that rests on text statistical features extracted from single documents to select the most important keywords of a text.

M G Thushara, et.al.[1] clearly discussed the working of several widely used keyword Extractors. The working of keyword Extractors like RAKE, PostionRank, and TextRank has been observed in several research papers. This paper is a study of unsupervised keyword Extractors that doesn't exactly need training on corpus and don't need any predefined rules or dictio-

nary.

3.2 Movie Trivia

Movie Trivia has always been a major form of entertainment in all gatherings or meet-ups. And online movie Trivias take this one step further igniting the competitive spirit among peers.

- Buzzfeed Quizparty

BuzzFeed is a digital media company that produces and distributes news, entertainment, and lifestyle content across a variety of platforms.BuzzFeed Quiz Party is a feature on the BuzzFeed website that allows users to take quizzes with friends in a virtual party setting. It was launched in 2020 and has since become a popular way for people to socialize and have fun online.//

There have been many works that paved the way. Some of them shed a light on unexplored areas and some of them focused on improving the existing methods, assisting us to a certain extent.

3.3 Similarity Measures

"Keyword-based similarity measures for text document clustering" by L. M. Cabral et al. (DOI: 10.1007/978-3-540-74783-2_21) proposes several keyword-based similarity measures for clustering text documents. The paper evaluates the effectiveness of these measures in terms of clustering accuracy and computational efficiency.

"Keyword-based text document similarity" by S. S. Salankar et al. (DOI: 10.1109/ICACCI.2014.6968435) presents a novel keyword-based text document similarity measure that takes into account the frequency and position

of keywords in the documents. The paper compares the performance of the proposed measure with traditional measures, such as Cosine similarity, on a benchmark data set.

"A survey of similarity and distance measures used in graph-based clustering" by Sumaiya Iqbal et al. (DOI: 10.1109/ICMEET.2016.7587976) provides an overview of various similarity and distance measures, including Cosine similarity, Adamic Adar, Dice Coefficient, Overlap index, Jaccard coefficient, Manhattan distance, and Euclidean Distance, and their applications in graph-based clustering.

"A comparative study of similarity measures for graph clustering and classification" by H. B. Kekre et al. (DOI: 10.1109/ICCICCT.2015.7480005) compares the effectiveness of various similarity measures, including Cosine similarity, Adamic Adar, Dice Coefficient, Overlap index, Jaccard coefficient, Manhattan distance, and Euclidean Distance, in graph clustering and classification tasks.

Chapter 4

Requirements

The design of this project contains both hardware and software. The specifications are listed below.

4.1 Hardware

As this is a web application deployed on the cloud. Personal electronic devices like laptops, mobile, and computers with internet access can go to a browser and start the game. A high bandwidth internet is preferred for the smooth processing of the game.

4.2 Software required

Different software modules have come into play and emerged into making this a full-fledged web application.

Name	***Usage***
Python 3.9	Code
Python Packages	Code
NLTK	NLP Toolkit
Jupyter	Platform for web scraping & Keyword Extraction
bs4,selenium	Web scraping
Visual Studio Code	Platform to develop Web App
Scikit-multiflow	Data Visualisation
HTML 5	Structuring the content of Web App
CSS 3	Styling the content of Web App
Django	Web Framework
Heruko	Deployment Platform

Table 4.1: Software Requirements

Chapter 5

Proposed System

A proposed system is a new system that is being suggested or developed to address a particular problem or need. It is a system that has not yet been implemented or put into use but is still in the planning or development stage. In terms of software development, a proposed system refers to a new software application or system that is being developed to replace an existing system or to address a specific need.

The proposed system is important in software development because it serves as a blueprint or a roadmap for the development process. It helps to define the scope of the project and the requirements that the software should meet. By defining the proposed system, developers can identify the features and functionality that the software should have, along with the technical requirements needed to build the system. It also helps to ensure that the resulting software meets the needs and requirements of the stakeholders. Before starting the actual development process, it is important to have a clear understanding of what the system should do and how it should work. This is where the proposed system comes in. By carefully analyzing the proposed system, developers can identify potential problems or issues before

development begins. This helps to reduce the risk of errors and issues during the development process.

Furthermore, a well-defined proposed system can help to minimize the cost and time required for software development. By having a clear understanding of the scope of the project and the requirements of the software, developers can better estimate the resources needed for the project. This can help to ensure that the development process is focused, efficient, and effective.

5.1 System Architecture

This project has been an accumulation of several fields that needs proper integration so that nothing feels out of place. Right from Web Scraping to Web Application development there has been a collective goal to achieve this.

5.1.1 System Design

The structure of this web application has always been a crucial point as it depicts the direction and the process of how a web application should work. Overall, the system design involves defining models, views, templates, forms, authentication, security, and database configuration. With these components in place, you can create a fully functional quiz app that allows users to create game rooms, and start the game.

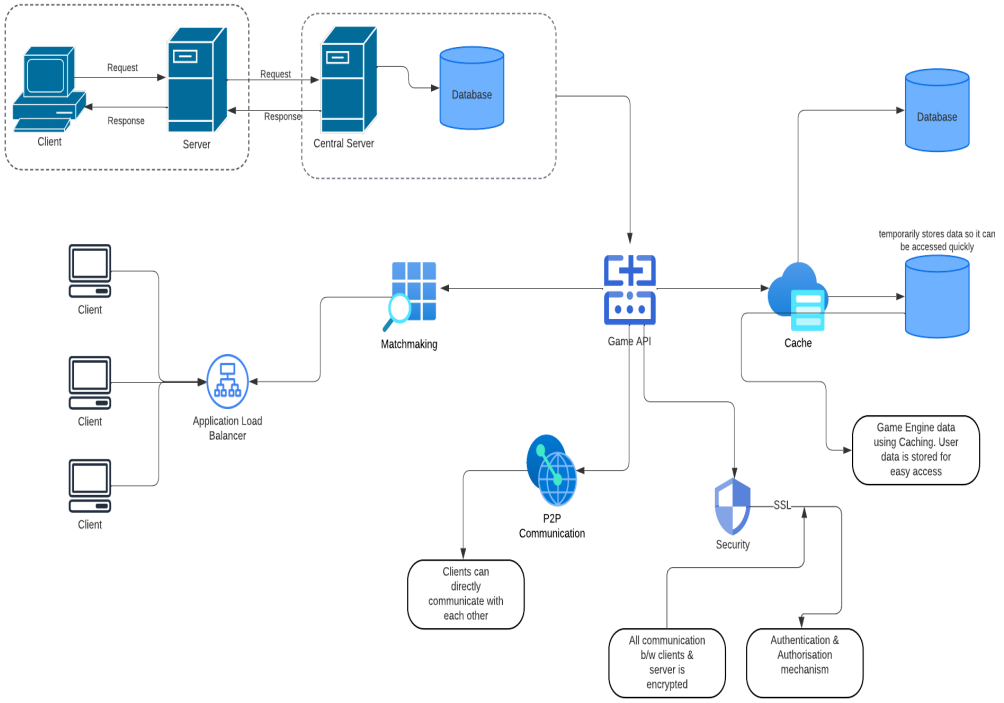

Figure 5.1: System Design

Right from when the player starts the game, the client makes a request from the local server. Then, the local server again requests the main/central server that has access to the game database. This database has all the information stored regarding the game like the score of the player, the timer, and importantly the questions. The questions are curated based on the options selected by the player. If the player selects an easy difficulty, there are different questions than when he selects another difficulty.

Now, the Game API, based on this, the entire application is running. This has access to every nook and cranny of the system. Right from the database, from which, the game engine access the movie details to curate the questions to the match-making part. Widely, match-making has been used to match players based on the likes of interests, age, and history of online

behavior – both in competitive and non-competitive or social online games and experiences. But here, we used it just to match the players who are joining the game using the shared link or code by the first player/quiz master.

Coming to the Caching, it stores all the necessary information or the information that has to be accessed very frequently like the scores which have to be updated for every question they answer. For instance, if the player somehow closes the game, this caching makes sure to save the progress and can be accessed immediately.

Authentication is done when the player wants to join a game and a player needs to be logged in if he wants to know the stats of all the games he played. And, the communication between the client and the servers is always encrypted.

Peer-to-Peer communication comes into play when there is a team match. Each team will be assigned a separate chat room to discuss their answers.

5.1.2 Process Flow

The whole project has been divided into 2 phases based on the areas of work. Phase 1 is completely devoted to Data Collection and Dataset making as the datasets are deployed for public usage. Ample care has been taken in the data collection state as well as the pre-processing state.

This diagram assists us in getting a clearer picture of phase 1 i.e Data Collection.

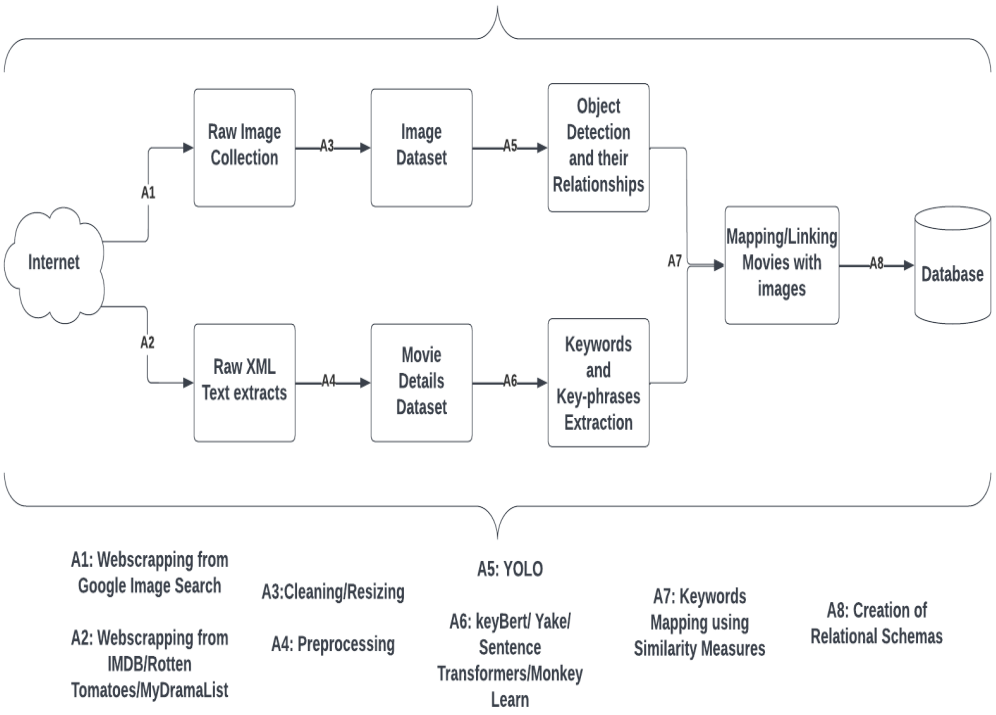

Figure 5.2: Data Collection & Dataset making

Starting from web scraping, Using beautiful Soup, an NLP tool, and Selenium, a Python automation tool to extract data from websites, Both the movie details and raw images that are royalty-free are extracted from the web. After the extraction, pre-processing is done to curate the datasets. All the empty and inaccessible fields have been erased or replaced. Datasets have been created using these pre-processed data. Several keyword extractors are run through the whole movie dataset giving us the top keywords from every movie plot. And, Object Detection is done on the image dataset to know the objects present in the images and relatively map the images to the keywords extracted. After mapping all the images to the existing keywords, the final Dataset which will be used for the game is originated.

Chapter 6

Data Collection

Data collection is the process of gathering and measuring information on variables of interest, in an organized and systematic fashion. It is an essential part of research and analysis, and is used to build an understanding of a particular phenomenon or situation. Data collection can be done through various methods, such as surveys, interviews, observations, and experiments.

The first step in data collection is to define the research question or hypothesis. This helps to determine what data needs to be collected, and what methods are best suited for the study. The data collection method should be selected based on the type of information needed, the population being studied, and the available resources. Once the data is collected, it must be organized and analyzed. This involves cleaning and processing the data, and then applying statistical methods to identify patterns and relationships. The results can then be used to draw conclusions and make decisions.

Web scraping can be a useful tool for data collection, as it allows for the efficient collection of large amounts of data from various sources.

6.1 Web-scraping

Web scraping is the process of automatically collecting data from websites. It involves writing software code to retrieve the HTML or XML code of a web page, and then parsing that code to extract specific data that can be used for various purposes such as analysis, research, or even machine learning.

Web scraping can be used to gather a wide range of data, it can be in the form of text, images, videos, or any other type of content that is available on a website. Web scraping can be useful for various purposes such as data analysis, research, marketing, and price comparison.

To perform web scraping, one usually uses programming languages such as Python, Ruby, or JavaScript, along with specialized libraries and frameworks. The process typically involves sending requests to a website's server, parsing the HTML or XML response, and extracting the desired information using techniques such as regular expressions or XPath queries.

There are different methods for web scraping, including:

- Manual web scraping: This involves manually copying and pasting data from a website into a spreadsheet or database.
- Automated web scraping: This involves using software tools or libraries such as BeautifulSoup, Scrapy, or Selenium to extract data from websites automatically.
- API access: Some websites offer APIs (Application Programming Interfaces) that allow developers to access their data in a structured and organized way.

6.1.1 Movie Description Extraction

To extract keywords that are used for the image representation in the game we are in need of the movie descriptions. For this, we created an automated web scrapper using BeautifulSoup to extract movie details from different movie sites like IMDB, Rotten Tomatoes, MyDramaList. What is Beautiful Soup? Beautiful Soup is a Python library used for web scraping purposes to extract data from HTML and XML files. It provides a simple way to navigate, search, and modify the parse tree of an HTML or XML document. Beautiful Soup creates a parse tree from the HTML or XML file that can be used to extract relevant data from the document. Beautiful Soup is widely used in the Python community for web scraping tasks, and its ease of use and flexibility make it a popular choice for scraping web data.

6.1.2 Image Scraping

Image scraping is the process of automatically extracting images from websites or other sources on the internet using software or tools. This can be done for various reasons such as data collection, research, or even for creating a dataset for machine learning or computer vision applications. There are different tools and techniques for image scraping, including web scraping libraries like Beautiful Soup or Scrapy, which can be used to extract images from HTML pages. Additionally, there are specialized image-scraping tools that can extract images from social media platforms or search engines. It's important to note that scraping images from websites without the owner's permission or violating their terms of service can be illegal and unethical. Therefore, it's essential to always obtain permission before scraping images and to use the images only for lawful purposes.

Why did we use Image Scraping? Image collection: Image scraping can be used to collect a large amount of image data for analysis or research purposes. Content Collection: Image scraping can be used to generate content for websites or social media platforms by automatically fetching and posting images related to a specific topic or keyword. Machine Learning: Image scraping is often used to create datasets for machine learning and computer vision applications. So, for creating a poster dataset we want to use these requirements so Image scraping is selected.

How did we use this Image Scraping? We have used the Beautiful Soup and Selenium techniques for extracting images from a website named IMDB.

What is Selenium? Selenium is a popular open-source software suite used for automating web browsers. Selenium is particularly useful for testing web applications that have complex user interfaces, dynamic content, or require interaction with different web elements. With Selenium, one can write automated test scripts that mimic real user behavior and help detect bugs or issues early in the development cycle. In addition to its testing capabilities, Selenium is also used for web scraping and data extraction. By automating the process of navigating through web pages and extracting data, one can save time and effort in collecting and processing large amounts of data from the web.

We have used Python programming language for this Image Scraping. This Python code uses Selenium and requests libraries to scrape and download a single image from Google Images based on a user-specified search term. Here's a brief summary of what the code does: The main thing is we will take posters for the top 10000 movies on IMDB. For that we set a loop starting at 0 and ending at 10001, so every movie will do the following thing i.e We will call the function *fetch_image*. The *fetch_image* function is

defined to perform a Google Images search for a given search term('movie') using a web driver('wd'). The first image in the search results is clicked on, and the script waits for up to 20 seconds for the full-sized image to load. If the image loads successfully, it is saved using the *persist_image* function. The *persist_image* function is defined to download and save an image from a URL to a specified folder path with a given file name. If the download is successful, a success message is printed to the console. If an error occurs during the download or saving of the image, an error message is printed. So, After the completion of this loop, all the downloaded posters are available in the folder which we have specified in our code.

6.2 Preprocessing

Data preprocessing is an essential step in data analysis and machine learning. It involves the transformation of raw data into a format that can be easily understood and analyzed by a computer. Data preprocessing includes a range of tasks such as data cleaning, data transformation, data reduction, and data normalization.

Data cleaning is the process of identifying and correcting or removing errors and inconsistencies in the data. This includes handling missing values, correcting typos, and removing duplicates.

Data transformation involves converting data from one format to another. For example, converting categorical data into numerical data, or normalizing data to ensure that it is on the same scale.

Data reduction involves reducing the amount of data to be analyzed, while still maintaining the most important information. This can be achieved through techniques such as feature selection, which identifies the most rel-

evant features for analysis, or dimensionality reduction, which reduces the number of variables that need to be analyzed.

Data normalization involves scaling data to ensure that it is on the same scale, which is important for machine learning algorithms that rely on distance measures. This can involve techniques such as min-max scaling, z-score normalization, or logarithmic scaling.

Web scraping involves extracting data from websites, and the data obtained can be messy and inconsistent. Prepossessing includes tasks like identifying and correcting or removing any errors or inconsistencies in the scraped data. There are several techniques that can be used for data cleaning in web scraping. One common technique is to remove any HTML tags or other formatting elements from the data. This can be done using regular expressions or parsing libraries.

Another technique is to remove any duplicates or irrelevant information from the data. For example, if the website contains ads or navigation menus, these can be removed from the scraped data. Similarly, if the website contains multiple versions of the same data (e.g., due to pagination), these can be deduplicated.

In addition, data cleaning may involve transforming or normalizing the data to a common format. For example, dates may need to be converted to a standard format, or numeric values may need to be rounded or truncated.

Overall, data cleaning is an essential step in web scraping to ensure that the data obtained is accurate, consistent, and usable for analysis or other purposes.

	movie_title	synopsis
0	A Splash of Love	Chloe Turner, a Ph.D. candidate in Marine Mamm...
1	The Grey Man	Kevin Dodds is a browbeaten deputy bank manage...
2	Descendants	Ben, son of Belle and the once selfish Beast, ...
3	Teen Wolf: The Movie	A full moon rises in Beacon Hills, and with it...
4	High School Musical	Troy Bolton and Gabriella Montez are two total...
...	...	...
9741	Hyperland	A mother afflicted with depression comes into ...
9742	The Brass Ring	When a wealthy businessman is murdered by a hi...
9743	One Hot Summer Night	A look at Yellowstone National Park's wildlife...
9744	Yellowstone in Four Seasons	Biopic of sorts about Agatha Christie, the fam...
9745	Agatha Christie: A Life in Pictures	It looks like we don't have any Plot Summaries...

	movie_title	synopsis
0	A Splash of Love	chloe turner a phd candidate in marine mammalo...
1	The Grey Man	kevin dodds is a browbeaten deputy bank manage...
2	Descendants	ben son of belle and the once selfish beast is...
3	Teen Wolf: The Movie	a full moon rises in beacon hills and with it ...
4	High School Musical	troy bolton and gabriella montez are two total...
...	...	...
9741	Hyperland	a mother afflicted with depression comes into ...
9742	The Brass Ring	when a wealthy businessman is murdered by a hi...
9743	One Hot Summer Night	a look at yellowstone national parks wildlife ...
9744	Yellowstone in Four Seasons	biopic of sorts about agatha christie the fame...
9745	Agatha Christie: A Life in Pictures	it looks like we dont have any plot summaries ...

Figure 6.1: Dataset cleaning before & after

Chapter 7

Game Dataset

After pre-processing is done on the whole dataset, which helped us clean the data as it contained some empty fields. As we are scraping the data from the web, it is inevitable to miss some of the data fields. Missing minor information such as the side cast or year of the movie has been neglected as all the important data is still present. If a movie misses the plot, the movie will be skipped without us scraping the information. The dataset is created after Data cleaning. And the keyword Extractors have been run on the dataset.

7.1 Keyword Extraction

Keyword extraction is a critical task in natural language processing that involves identifying and extracting the most significant and relevant words in a text. These keywords are essential for several applications, such as text classification, document summarization, and search engine optimization. The process of keyword extraction requires a tool or an algorithm that can accurately recognize and extract these words from the text. This tool is known

as a keyword extractor, which is designed to analyze the text and identify the most important words based on their frequency, context, and relevance to the overall document. Keyword extractors use various techniques, such as statistical analysis, machine learning, and natural language processing, to perform this task automatically. As the volume of textual data continues to grow exponentially, keyword extraction has become an increasingly important task for businesses, researchers, and professionals alike.
Keyword extraction is the process of identifying and extracting the most relevant words or phrases from a piece of text, which can then be used to categorize, summarize or index the document.

There are several types of keyword extractors, each with its own approach and methodology. Here are some of the most common types:

Statistical-based keyword extraction: Statistical methods are commonly considered to be simple and direct ways of extracting keywords. These methods involve computing statistics for individual words or phrases, such as word frequency, collocation, and co-occurrence, and then using these statistics to determine the importance or relevance of each keyword. There are more complex statistical methods for extracting keywords, such as **TF-IDF** (term frequency-inverse document frequency), which calculates a score based on how often a word appears in a document relative to its frequency across all documents, and **YAKE!** (Yet Another Keyword Extractor), which combines statistical measures with machine learning techniques to identify important phrases in a text.

Graph-based keyword extraction: Graph-based keyword extraction is a method that uses graph theory to identify the most important words or phrases in a document. In this approach, the document is represented as a graph, with each word or phrase as a node and their relationships as

edges. The graph can be created using various criteria, such as co-occurrence, dependency, or similarity measures.

Once the graph is constructed, graph-based algorithms such as PageRank, TextRank, or TopicRank can be applied to rank the nodes based on their importance. The nodes with the highest scores are considered to be the most important keywords in the document.

Linguistic keyword extraction: This approach uses linguistic analysis techniques to identify the most important words or phrases in a document based on their grammatical structure, part of speech, and other linguistic features.

Hybrid keyword extraction: This approach combines multiple techniques, such as frequency-based, statistical, and linguistic approaches, to identify the most important keywords.

Keyword extractors work by analyzing the text and identifying the most important words or phrases based on the selected approach. The extracted keywords can then be used for various purposes such as indexing, summarization, classification, and information retrieval.
Keyword extraction has a wide range of uses, including:

- Search engine optimization: Keyword extraction can be used to identify the most relevant keywords for a web page or document, which can help improve its search engine ranking.

- Document summarization: Keyword extraction can be used to generate summaries of documents by identifying the most important keywords and using them to create a condensed version of the text.

- Text classification: can be used to classify documents based on their content by identifying the most relevant keywords for each category.

- Sentiment analysis: can be used to identify the most important words or phrases that indicate the sentiment of a piece of text, such as positive or negative sentiment

7.1.1 Keyword Extractors

There are different techniques and algorithms to extract keywords from a text. They can be statistical, graph-based, or hybrid, but the function & working of each keyword extractor hugely differ.

In a statistical-based approach, the basic idea is to derive a score based on several statistics and extract keywords based on that score. Different approaches employ different ways of calculating scores for n-grams as well. Some of the statistical-based approaches are:

- TF-IDF (Term Frequency-Inverse Document Frequency): It is a commonly used technique in keyword extraction. It calculates the importance of a keyword in a document based on how frequently it appears in the document and how rarely it appears in other documents.TF-IDF is calculated by multiplying two values: the term frequency (TF) and the inverse document frequency (IDF). The term frequency is simply the number of times a keyword appears in a document, while the inverse document frequency is calculated as the logarithm of the total number of documents divided by the number of documents that contain the keyword. By combining these two values, TF-IDF is able to identify the most important and relevant keywords in a document.

- YAKE (Yet Another Keyword Extractor) is a machine learning-based keyword extraction tool that is designed to identify important keywords

in a document based on a combination of text features and statistical measures. Unlike some other keyword extraction tools that rely solely on frequency analysis, YAKE takes into account a wider range of factors to identify the most relevant keywords.

YAKE uses an unsupervised learning approach to identify keywords, which means that it does not require any training data or prior knowledge of the document's topic. Instead, it uses a combination of statistical measures such as TF-IDF, co-occurrence, and position information, as well as linguistic features such as the length and parts of speech of words, to identify important keywords.

- RAKE (Rapid Automatic Keyword Extraction) is a keyword extraction algorithm that is designed to identify important keywords in a document based on the frequency of co-occurrence of words. It was developed to address the limitations of traditional keyword extraction methods that rely solely on frequency analysis, such as the fact that they often produce long lists of uninformative keywords.

  RAKE identifies keywords by analyzing the distribution and frequency of co-occurring words in a document. It uses a set of predefined stop words (words that are commonly used in the language but do not carry much meaning, such as "the", "and", "or") to split the text into candidate phrases, and then scores each phrase based on its frequency and distribution in the document.

  The algorithm assigns a score to each candidate phrase based on the frequency of its constituent words and the number of times it appears in the document. It then identifies the top-scoring phrases as the most relevant keywords.

One of the advantages of RAKE is that it is able to identify multi-word phrases, or "key phrases", which can be more informative and meaningful than individual words. It also allows for the customization of the algorithm by allowing users to specify their own list of stop words and to adjust other parameters, such as the minimum length of the candidate phrases.

Overall, RAKE is a simple yet effective algorithm for keyword extraction that can be applied to a wide range of text data. It is particularly useful for identifying key phrases and has been shown to produce more accurate results than traditional frequency-based methods in some cases.

- TextRank is a graph-based algorithm that works by treating a document as a graph of interconnected words or phrases. It calculates a score for each word or phrase based on its co-occurrence with other words or phrases in the graph. The algorithm identifies the top-scoring words or phrases as the most relevant keywords.

  PositionRank: PositionRank is a variant of TextRank that takes into account the position of words or phrases in a document. It assigns higher weight to words or phrases that appear at the beginning or end of a sentence, as these are more likely to be important. PositionRank also takes into account other factors such as the length of the phrase and its grammatical role in the sentence.

  SingleRank: SingleRank is another graph-based algorithm that works by representing a document as a graph of interconnected words or phrases. It calculates a score for each word or phrase based on its similarity to other words or phrases in the graph. The algorithm iden-

tifies the top-scoring words or phrases as the most relevant keywords.

- Sentence Transformers work by encoding each sentence in a document into a vector and then clustering the vectors to identify the most representative sentences in the document. These representative sentences can then be used to extract keywords or summarize the content of the document.

  One of the advantages of using sentence transformers for keyword extraction is that they can capture the semantic meaning of sentences, rather than just their surface-level features. This allows them to identify important keywords that may not be obvious from a simple frequency analysis of the text.

7.1.2 Algorithms

Several keyword Extractors have been defined and used. Some of the algorithms we used are YAKE, RAKE,keyBERT, and Sentence Transformers. And, some of the graph-based keyword Extractors such as TextRank, PositionRank, SingleRank, etc also have been used to provide a dimension to the keywords. Even though all the algorithms are pre-defined, curating them according to our preferences has become a tedious task. Some of the algorithms have their scores and a different metric on how they select their keywords, and mapping them all to our dataset gave us the final product.

These algorithms along with a few others like TextRank, SingleRank, etc have been run on the whole dataset, giving us keywords from the plot respectively. After pre-processing, another dataset that includes all the keywords has been derived.

7.1.3 Image Dataset

There are some licensed and free stock image sites such as Unsplash, pixabay, etc. They were understanding enough to lend us their free stock image dataset which is about 40GB when we requested it officially and they obliged it. As the Image dataset is huge, we got access to the official API which is used to retrieve the images based on the keywords we extracted.

Extracting keywords from images: There has always been scrutiny of the accuracy in the field of object detection. As this is just a mapping of extracted keywords from the dataset to the keywords from the images. A simple object detection algorithm that can classify and segregate the objects is enough. So, we are going with a curated and scaled version of the YOLOv4 (You Only Live Once Version 4) algorithm. Using darknet repository that has most of the dependencies required for object detection. Several functions have been defined to run object detection on the input image. In the darknet repository, there are some predefined classes and trained models that are able to detect about 40+ regular objects such as a person, bench, mobile, etc.

Figure 7.1: Object Detection on a movie poster

7.2 Keyword Mapping

Keyword sets mapping is a technique used to identify the similarity between different sets of keywords and group them together based on their semantic meaning. This is typically done using similarity measures, which compare the distance or similarity between two or more sets of keywords based on a set of metrics or attributes.

One common approach to keyword set mapping is to use the Jaccard coefficient. This measure calculates the similarity between two sets of keywords based on their intersection and union. In the context of keyword sets mapping, the Jaccard coefficient is used to compare the overlap between different sets of keywords and group together those that have the highest degree of overlap. Another approach to keyword sets mapping is to use the Cosine similarity measure. This measure calculates the cosine of the angle between two vectors in a high-dimensional space. In the context of keyword set mapping, the vectors represent the frequency of occurrence of each set of keywords in a set of documents or texts. The Cosine similarity measure is then used to compare the relative frequency of occurrence of each set of keywords, and group together those that are most similar.

Keyword sets mapping using similarity measures can be used for a variety of applications, such as text classification, information retrieval, and content recommendation. By grouping together similar sets of keywords, it is possible to create more accurate and relevant search results, improve the performance of recommendation systems, and gain insights into patterns of behavior and preferences among users.

7.2.1 Similarity Measures

Similarity measures are techniques used to compare the similarity between two or more objects or data points. They are widely used across a range of fields, including natural language processing, data clustering, and recommendation systems.

In natural language processing, similarity measures are used to compare the similarity between two texts, such as two documents or two sentences. This is important for tasks such as document classification, information retrieval, and text summarization. Common similarity measures used in this field include Cosine similarity, the Jaccard coefficient, and Euclidean distance.

In data clustering, similarity measures are used to group together similar data points based on a set of features or attributes. The goal is to create clusters that are internally homogeneous and externally heterogeneous. Common similarity measures used in this field include Manhattan distance, Euclidean distance, and Pearson correlation coefficient.

In recommendation systems, similarity measures are used to recommend items to users based on their past behavior or preferences. The goal is to identify items that are similar to those that the user has already shown an interest in. Common similarity measures used in this field include Cosine similarity, Pearson correlation coefficient, and Jaccard coefficient. For Keyword mapping, we will be using these term-based similarity measures in our project.

7.2.2 Image Rankings

Now, we need to find out the best image that is more relevant to the given movie/series description. We have a set of keywords extracted from the description for each movie/series and we also have a set of keywords associated with each image. For each set of keywords from descriptions, we need to identify the most similar keyword set from image keyword sets.

Now let's discuss the above statements more formally and in the mathematical sense. Let's represent each keyword set as a vector. Now we apply similarity measures over these vectors as discussed in the literature review. For simplicity let's assume the vectors obtained from movie/series description keywords are set to be A and the vectors from image keywords are set to be B. Our task is to identify vectors from B that are most similar to each vector in A. For this project, we take the top three similar vectors.

As mentioned earlier before, we consider term-based similarity measures under string-based similarity. That is, there are no neighborhood relationships among characters or the terms. We just consider the bags of words and measure how similar they are. We employed six similarities and calculated and mapped the top three images to each movie/series. They are Block distance or Manhattan distance, Cosine similarity, Dice's coefficient, Euclidean Distance, Jaccard's coefficient, and Overlap Coefficient. Once we obtain the top three mappings, we create a dataset with movie/series name with image id and its corresponding rank.

Chapter 8

Web Application

8.1 Game Overview

The fast-paced lifestyle that many of us lead can be overwhelming and stressful. To cope with this, people often look for ways to unwind and relax. One of the most popular methods of doing so is by playing games or watching movies. These forms of entertainment allow people to escape their daily routines and immerse themselves in a different world for a while.

At times, playing games or watching movies can become monotonous, and individuals may seek out new experiences. Combining games and movies is a creative way to offer something fresh and innovative to the audience. This is precisely how our game, GUESS IT!, came into being.

GUESS IT! is a game that blends the excitement of movies with the thrill of guessing games. One of the greatest joys of gaming is the ability to play with friends. There's something about the social aspect of games that adds an extra layer of fun and excitement. GUESS IT! is no exception. It's a game that's best played with friends, as you try to guess the movie titles based on the clues provided while bringing the people together.

At times, we may think that a game's worth is determined by its graphics, complexity, or other factors. However, the point we like to make is that any game can be enjoyable as long as it serves its purpose. In other words, a game does not need to have heavy graphics or be complicated to be entertaining.

GUESS IT! is a testament to this philosophy. It is a simple game that anyone can play and enjoy, regardless of their age or gaming experience. It offers a fun and engaging way to spend time with loved ones, making memories that will last a lifetime.

What is this game?

Trivia is a universally beloved game that has entertained people all over the globe. If you're a movie buff, a trivia game about films is sure to pique your interest. To play, simply compile a wide range of questions and strive to identify the correct answers. The advent of online trivia has taken this already-entertaining pastime to new heights of amusement.

Single Player Game:

If you're feeling bored and fatigued from aimlessly browsing the web, there's no need to worry. You can now engage in a solo game of trivia where the system presents you with questions to answer. You have the option to choose from various categories such as genres, actors, and filmographies and attempt to guess the movie. This is just one of several ways to enjoy the game.

Multi player:

1 vs 1:

A straightforward way to play the game is in a one-on-one format. Both players will receive an identical set of questions, and the player who correctly guesses the answers with the least amount of time remaining will earn points. Once all rounds are completed, the winner will be determined based on their

total score.

Team:

The game can accommodate a maximum of four players on each team, resulting in a 4 vs. 4 format. The players may either be randomly assigned or selected. A designated host is appointed to manage the questions. Private chat rooms are allotted to each team or Discord may be used for improved communication. Before the time limit expires, each team must submit their answer, and points will be assigned based on the time left. After completing all rounds, the team with the highest score will be declared the victor.

8.1.1 Rules

- Filters are available to allow players to select the language of the movies.
- The player has the option to select the level of difficulty, with two options available: easy and hard.
- Easy mode is a buzzer round where the player will be given 5 questions and the images are based on the title of the movie.

8.2 Design

Design is the first and important step in any development. And the best way to design is through pictorial representations. Design diagrams are important because they provide a visual representation of a system or process, making it easier to understand and communicate complex information. Here are some reasons why design diagrams are important:

Enhance communication: Design diagrams help to enhance communication between stakeholders, allowing everyone to share a common understanding of the system or process being designed. This is especially important when working with technical details or complex data.

Identify problems: Design diagrams can help to identify potential problems or bottlenecks in a system or process. By visualizing the flow of information or resources, it is easier to identify areas where improvements can be made.

Improve efficiency: Design diagrams can help to improve the efficiency of a system or process. By identifying areas where resources are being wasted or duplicated, it is possible to streamline processes and reduce costs.

Facilitate design iterations: Design diagrams can help to facilitate design iterations, allowing designers to quickly iterate and improve upon designs. By visualizing different design options, it is easier to make informed decisions about the best approach to take.

Aid in training: Design diagrams can be used as a training tool, helping new team members or stakeholders to understand the system or process being designed. This is especially important when working with complex systems or processes that may be difficult to understand.

Some of the examples include ER Diagram, Process-flow diagram, Class Diagram, Use-case etc. We drew Use-case, ER Diagram and Class Diagram to help us building our web app GuessIt.

8.2.1 Usecase Diagram

Use case diagrams are a type of diagram in the Unified Modeling Language (UML) that provide a visual representation of how users interact with a system. They are easy to understand and help to identify system requirements,

boundaries, and potential problems. Use case diagrams are an important tool for facilitating communication between stakeholders, including developers, designers, and users. They provide a common language for discussing system requirements and ensure that everyone is on the same page. In summary, use case diagrams are an important aspect of system design that help to ensure that systems are designed to meet the needs of their users, and that stakeholders have a clear understanding of the system's requirements.

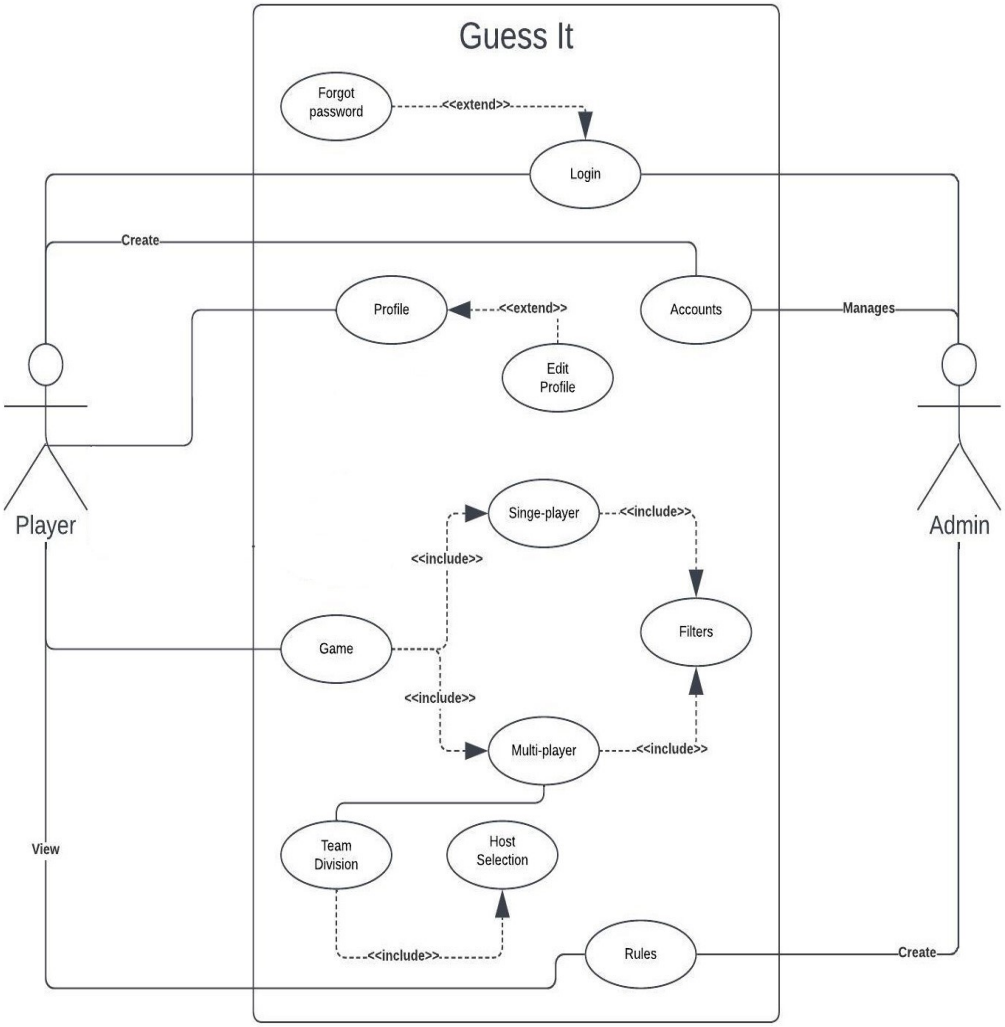

Figure 8.1: Usecase Diagram

As you can see in the diagram, We have two actors in the system, the Admin and the Player. Both actors are able to log in to the system and perform their respective tasks. As an Admin, they have the ability to manage user accounts, modify game rules, and perform other administrative tasks. As a Player, they can create an account, update their profile information, and play the games. The games available in the system can be either single-player or multiplayer, and include various filters such as difficulty level, movie languages, team divisions for multiplayer games, and more. Overall, the system provides a comprehensive gaming experience for both Admins and Players, with features that cater to the needs of both types of users.

Entity-Relationship Diagram

An ER (Entity-Relationship) diagram is a type of diagram used in database design that depicts the relationships between entities (objects) in a database. It consists of three main components:

Entities: An entity is a real-world object or concept that can be uniquely identified and represented in a database. Examples of entities include customers, orders, products, and employees.

Attributes: An attribute is a property or characteristic of an entity. For example, a customer entity may have attributes such as name, address, and phone number.

Relationships: A relationship is a connection between two or more entities. It describes how the entities are related to each other. For example, a customer may place multiple orders, so there is a relationship between the customer and order entities.

ER diagrams are typically represented using boxes to represent entities, ovals to represent attributes and lines to represent relationships. The lines

can be labeled with the type of relationship, such as one-to-one, one-to-many, or many-to-many. ER diagrams are useful for database design because they help to visualize the structure of the database and the relationships between different entities. They can also help to identify potential issues with the design, such as redundant or incomplete data. Overall, ER diagrams are an important tool for designing efficient and effective databases.

In our application, we deal with entities like Player, Game, Stat, and Movie. We can establish relationships like the Player plays the Game, the Player has Stat, the Game uses Movie, etc. A detailed representation is given in the diagram.

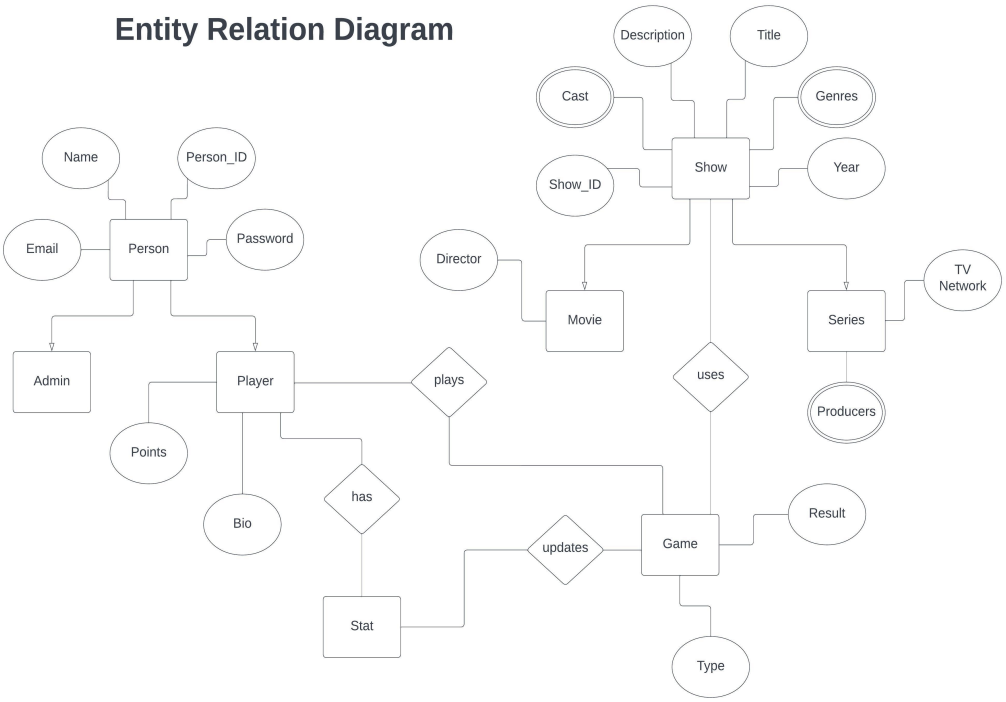

Figure 8.2: ER Diagram

Class Diagram

A class diagram is a type of diagram used in object-oriented programming to depict the classes and objects in a system and the relationships between them. It provides a high-level view of the system's structure and is useful for designing, documenting, and communicating the system's architecture.

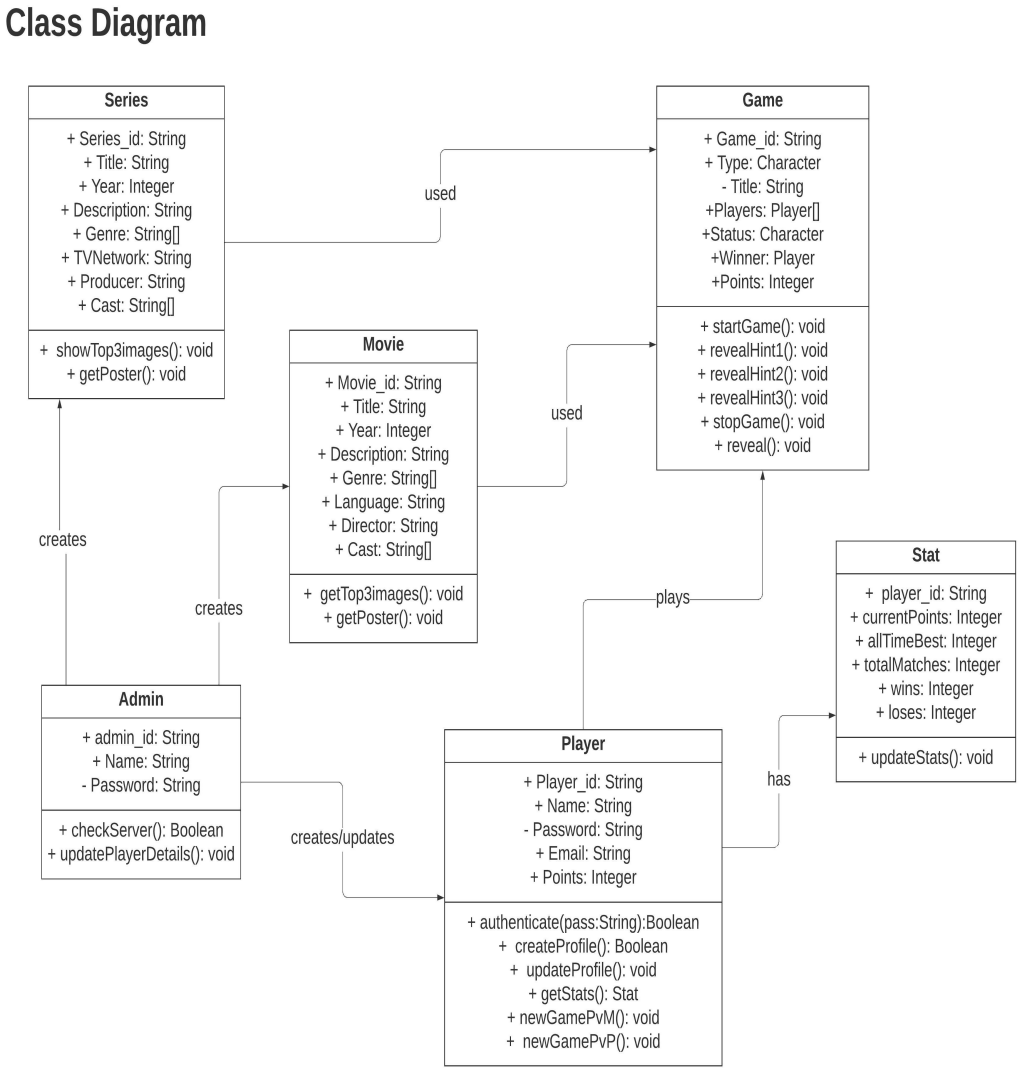

Figure 8.3: Class Diagram

A class diagram consists of classes, interfaces, associations, attributes,

and operations. Here's what each of these components means:

Class: A class is a blueprint or template for creating objects. It defines the properties and behaviors that objects of that class will have.

Interface: An interface is a type of class that defines a set of methods that a class can implement. It specifies what a class must do, but not how it should do it.

Association: An association is a relationship between two classes. It specifies how the two classes are related and can be one-to-one, one-to-many, or many-to-many.

Attribute: An attribute is a property or characteristic of a class. It defines the data that objects of that class will store.

Operation: An operation is a method or function that a class can perform. It defines the behavior that objects of that class will have.

Class Diagram helps in all phases of designing, modeling, implementation and documentation. Our app include four main classes namely, User, Game, Stat and Movie. The relationships among these can be seen in the diagram.

8.3 Key Functionalities

Every Application has it's own set of functionalities that are determined by the requirements of that application. It is important to find out what functionalities your website needs before you begin development. This will help you choose an appropriate web developer and also guide them through the process of building your site. The key functionalities of your application should be considered and planned for when building your site. If you are designing a website for an online store, it would be important to consider the shopping cart functionality and payment gateway integration. For other

applications such as blogs or portfolios, it might be important to provide social media sharing buttons so users can easily share their content on their favourite networks. For our web application, which is a game, we define the following key functionalities:

User Authentication: User authentication functionality is a process that verifies the identity of a user attempting to access a system or application. This process helps to ensure that only authorized users can access sensitive or confidential information. It helps to ensure that only authorized users are able to access the information they need, while also preventing unauthorized users from accessing the information. Implementing strong user authentication functionality is a critical aspect of system security and can help to prevent data breaches and other security incidents.

Game Modes: This functionality includes starting new game, selecting what kind of difficulty they want to play on and save the user's progress when they exit the game. In addition, we also want to provide a high score table so users can see their ranking and compete against others. We have a quick-game mode where there's no need to login and it's unranked. There are Ranked matches where the user has to login and the points obtained will be considered to be shown on leaderboard.

Information Retrieval: Information retrieval from a database is a critical functionality for many web applications. When a user interacts with a web application, they often request information that is stored in a database. The web application must retrieve this information from the database and display it to the user in a way that is easy to understand. Databases can become slow and unresponsive if they are not properly optimized, so developers must carefully consider the queries they use and the indexing strategies they employ to ensure that information retrieval is fast and efficient.

8.4 Front-End

The process of developing a website is commonly divided into two primary sections: frontend and backend. Each section is responsible for different aspects of a website's functionality and user experience. The frontend handles the design and presentation of a website, while the backend manages its functionality and data.

The frontend is the part of a website that users directly interact with, making it the most visible and visually appealing aspect. It utilizes various technologies, such as HTML, CSS, and JavaScript, to create a website's visual design and functionality. HTML creates a website's structure and content, while CSS styles and arranges that content on HTML pages. It uses various properties, rules, and prefixes to define a website's look and feel, including size, color, font, and layout. Additionally, CSS creates animations and responsive design elements to ensure that a website is compatible with different devices and browsers. JavaScript adds interactive and dynamic behavior, such as user input validation, pop-ups, and animations, by creating interactive forms, buttons, and menus.

Other subcategories within frontend development include responsive web design, user experience (UX) design, and accessibility. Responsive web design is critical for creating websites that can adapt to different screen sizes and resolutions on various devices, while UX design focuses on creating intuitive navigation, clear calls to action, and user-friendly interfaces to provide a positive user experience. Accessibility ensures that websites are inclusive and accessible to users with disabilities.

8.4.1 Technologies and Frameworks

The Guess It web application's frontend primarily uses HTML, CSS, and JavaScript. HTML tags create the structure and content of the webpage, while CSS properties add styles to that content. Apart from the common HTML tags the web application also uses SVG tag or a.k.a. Scalable Vector Graphics, which is used to create graphics and images that can be scaled without losing quality. The "viewBox" attribute is one of the most important attributes in SVG, as it defines the position and size of the graphic or image within the SVG viewport. This attribute takes four values: x, y, width, and height, separated by spaces where the values of x and y define the top-left corner of the SVG image or graphic, while width and height define the dimensions of the image. These values are defined in user units, which are typically pixels, although they can also be specified in other units, such as inches or centimeters. The viewBox attribute is particularly useful when working with responsive web design, as it allows us to create scalable graphics that can adapt to different screen sizes and resolutions. By defining the viewBox, one can ensure that the graphic or image is always displayed in the correct position and size, regardless of the size of the viewport or the device on which it is viewed.

The other most powerful feature of SVG is the ability to define complex shapes using the "path" element. This element is used to create vector paths that can be used to draw lines, curves, and shapes. The path is defined using a series of commands and coordinates that specify the shape and position of the path. These commands include "moveTo", "lineTo", "arc", "bezierCurveTo", and "closePath", among others. Using these commands, one can create complex shapes and designs that would be difficult or impossible to

create using other image formats. The path element also allows us to define fill and stroke properties for the shape, giving us complete control over its appearance. We have specific websites to create path elements by designing the respective vector path in these websites and converting it into HTML tags. The Guess It web app also used these SVG elements to create different animations and paths.

While coming to CSS styling some of the common CSS properties used in the Guess It web application are background-image, height, width, position, display, margin, padding, background, color, and fontsize. CSS clip-path is a property that allows us to create complex shapes for HTML elements by defining a clipping path has also been used. This clipping path is used to define the visible area of an element, effectively masking off the parts of the element that fall outside of the path. This also allows us to create visually interesting designs that would be difficult or impossible to create using standard CSS properties. One of the most interesting uses of clip-path is in creating responsive designs. This allows one to create designs that look great on both desktop and mobile devices, without having to create separate layouts for each.

Additionally, CSS rules such as keyframes to create animations for the CSS styles, and media rules which apply media queries to a page to ensure website compatibility with various devices have been used. Keyframes allow us to define a series of style changes over time, which can be used to create animations and transitions for HTML elements. By defining keyframes at specific points in an animation, one can create complex and visually interesting effects that can help to draw user attention and engagement. Media

queries, on the other hand, allow us to apply CSS styles based on the characteristics of the device or browser that is being used to view the website. This can be useful for ensuring that the website is optimized for different screen sizes, resolutions, and orientations. By using media queries, one can create responsive designs that adapt to the user's device and ensure that the website looks great on all devices, from desktops to mobile phones.

Each browser has its own way of interpreting CSS styles, and some browsers may not support certain styles or properties. By using vendor prefixes, we can ensure that our CSS styles are correctly interpreted by different browsers and that the website looks and functions correctly on all devices. So to ensure that the Guess It web application is compatible with various browsers, a range of CSS prefixes such as -o for old versions of Opera, -ms for before Chromium versions of Microsoft Edge and Internet Explorer, -moz for old Mozilla, and -webkit for Chrome, Safari, newer versions of Opera and Edge, almost all iOS browsers including Firefox for iOS have been employed in the development of the web application.

Different functionalities have been added using JavaScript. While CSS can be used to create animations and graphics, some more complex effects may require the use of JavaScript to achieve. For example, CSS can be used to create basic animations such as transitions, transforms, and keyframes, but more complex animations like the SVG may require the use of JavaScript to control the animation's timing, direction, and other properties. So in the development of Guess It web app, JS also played a major role to make the client-end highly interactive and view appealing.

8.5 Back-End

What makes a car a supercar, is it's looks? No! It's the performance. And most of this performance isn't something you can see—it's under the hood stuff that powers these automobiles. The current web applications work the same way, And the backend powers the application.

The backend of a website or application is the foundation upon which everything else is built. It is responsible for handling the behind-the-scenes processing that enables the frontend to function properly. The importance of the backend cannot be overstated, as it directly impacts the user experience.

There's lot of functionalities that backend does, and it varies from application to application based on the use case. But there are few functionalities that we can see that are very basic yet powerful. They include handling data, managing users, core logic of the application, security and computation power. Data Handling includes sending and receiving data from the server. It also includes storing, updating and deleting data. Managing Users is very important in any application because it helps to keep track of all users. Core Logic of the application refers to the business logic that runs behind every functionality of your application. Security is also very important as it helps to prevent unauthorised users from accessing your application or any sensitive information like passwords etc.

Let's now discuss how the backend implemented from the perspective of our web app GuessIt. We start with discussing the design, tools and technologies employed, implementing game logic, storing data, communication, deployment and finally maintenance.

8.5.1 Technologies and Frameworks

Modern-day technologies, tools, libraries, and frameworks have made working with the backend much easier than it was in the past. These tools provide a structure and set of guidelines for building applications. It is a pre-written code that developers can use to build their applications, rather than starting from scratch. Frameworks are designed to make software development easier, faster, and more efficient by providing a set of pre-defined rules, functions, and libraries that can be used to build complex applications. In almost all web-apps we see in todays world have some or the other similar key functionalities like user authentication and authorisation, data storage, database management, etc. There are many frameworks available to help build your backend in a more efficient manner. Doing so, you can concentrate on engineering your application than hard coding every functionality.

Frameworks can be used for a variety of purposes, including web development, mobile application development, and desktop application development. They often provide features like database access, user authentication, session management, and other commonly-used functions that can be used to build complex applications. They can also help ensure that applications are built in a consistent and scalable manner. They provide a set of guidelines and best practices that can help developers build applications that are easy to maintain, update, and extend over time.

Some of the examples of web development frameworks include Ruby on Rails, Django and Express. These frameworks provide a set of components that can be used to build complex web applications. They are designed to be flexible, allowing developers to create applications that meet their specific needs. For building our web app we used Django.

Django

Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. It is built by experienced developers and takes care of much of the hassle of web development so that you can focus on writing your app without needing to reinvent the wheel.

Django is known to be a great choice for web development due to its advantages, such as its ability to handle traffic and mobile app API usage of more than 400 million users, helping maximize scalability and minimize web hosting costs. It also comes with a lot of features out of the box, which saves you from writing your own code, and allows you to import the packages that you want to use. Furthermore, Django has a large feature set with more than 10,000 packages that cover virtually anything you'll need a web application to do. Additionally, the scalability benefits of the Django framework extend to what you're building, as it is flexible enough to handle content management systems, social networks, machine learning, and complicated data analysis.

jQuery

jQuery is a popular JavaScript library that aims to simplify the process of using JavaScript on websites. At its core, jQuery is used to interact with HTML elements on a webpage through the Document Object Model (DOM). jQuery takes many common JavaScript tasks that can be complex and time-consuming to implement and simplifies them into methods that can be called with a single line of code, making it easier for developers to write efficient and effective code. One of the key advantages of jQuery is its popularity and large community of users and contributors who help to maintain and update the library. It also normalizes differences between web browsers, making it easier to develop cross-browser compatible web applications.

AJAX

AJAX stands for Asynchronous JavaScript and XML. It is a technique used in web development that enables web pages to reload content dynamically without having to reload the entire page. This can improve the user experience by making web pages more responsive and interactive. AJAX is based on a combination of several technologies, including HTML, CSS, JavaScript, and XML.

One of the key features of AJAX is its ability to send and receive data asynchronously from the server, which means that the user can continue to interact with the page while the data is being retrieved. This is different from traditional web applications, where the user has to wait for the entire page to reload before seeing new content.

Another important feature of AJAX is its ability to update parts of a web page without reloading the entire page. This is achieved by using JavaScript to manipulate the Document Object Model (DOM) of the web page. This allows developers to create more responsive and interactive user interfaces without having to reload the entire page.

AJAX also provides a way for web applications to communicate with web servers without having to reload the entire page. This is done using the XMLHttpRequest object, which allows JavaScript to make HTTP requests to the server in the background. This can be used to retrieve data from the server, submit data to the server, or update data on the server without having to reload the entire page.

JSON

JSON stands for JavaScript Object Notation. It is a lightweight data interchange format that is easy for humans to read and write and easy for ma-

chines to parse and generate. JSON is based on a subset of the JavaScript programming language and it is often used to transmit data between a server and a web application as an alternative to XML.

One of the key advantages of JSON is its simplicity. It uses a minimal set of syntax rules that make it easy to read and write, and it can represent a wide variety of data structures, including objects, arrays, strings, numbers, and boolean values. Additionally, JSON is supported by most modern programming languages, which makes it a popular choice for building web applications and web services.

Another important advantage of JSON is its smaller file size compared to XML. This makes it faster to transfer data over the internet, which can improve the performance of web applications. Additionally, JSON is easier to parse than XML, which makes it faster and more efficient for web browsers and web servers to process.

8.5.2 Implementation

Now that we had the plan and the tools, let's get down to cooking. We have divided the whole backend development process into building game engine with required functionalities of our application, setting up database with the datasets as mentioned in earlier chapters, linking with GUI and establishing structured communication using APIs.

Game Server

The game engine is the heart of the application. It is a collection of code that provides all the necessary functionalities for us to run our application. This includes handling user input, displaying graphics on screen and managing data in memory. As mentioned in the design phase, we start implementing

classes described in class diagram that is User, Game, Stat and Movie. We mapped all the required relationships. Once we have our classes intact, we implement the functionalities discussed in the key functionalities in earlier sections.

Django views can be written as functions or classes. Function-based views are simpler and easier to understand, while class-based views provide more flexibility and can be reused across multiple URLs. Views can also be decorated with various decorators, such as the login-required decorator, which ensures that a user is logged in before accessing a particular view. Django also provides generic views, which are pre-built views for common use cases, such as displaying a list of objects or creating a new object.

We implemented the login, logout, home, game, about, quick-game, stat-display, leaderboard functionalities using django views. When a user makes a request to a Django web application, the URL dispatcher maps the requested URL to a corresponding view. The view then processes the request and returns an HTTP response, which can be a simple HTML page, a JSON response, or any other format that the application supports.

Database

Django is a popular web framework that uses an object-relational mapping (ORM) system to interact with databases. One of the key components of Django's ORM is the concept of models. In Django, models are Python classes that define the structure of a database table. Each attribute of the class represents a field in the table, and the class itself represents the table. Models can be used to define relationships between tables and to enforce data validation rules.

Django comes with a default database engine called SQLite, which is a

lightweight, file-based database that is suitable for small to medium-sized applications. However, Django also supports a variety of other databases, such as MySQL, PostgreSQL, Oracle, and Microsoft SQL Server. Django's ORM is designed to be database-agnostic, which means that you can write code that works with multiple databases without having to make significant changes to your code. It also makes it easy to perform CRUD (create, read, update, delete) operations on your data.

Django's ORM is not a NoSQL database, but it does support some NoSQL-like features, such as denormalization and document storage. For example, you can use Django's JSONField to store JSON documents in your database, and you can use denormalization to store redundant data for performance reasons.

Performing CRUD operations in Django is relatively simple. To create a new object, you create a new instance of the model class and then call the save method. To read objects from the database, you use the model's manager to execute queries. To update an object, you modify its attributes and then call the save method. To delete an object, you call its delete method.

We created 4 simple Modal classes which are used to store our four classes User, Game, Stat and Movie. There are default modals that are created along with these which stores information about user-logins, user-groups, user-permissions, version history about changes in the database.

Linking with Front-end

Linking the Django-based backend with the frontend involves creating views that render templates and serve HTTP responses. Here is a step-by-step process for linking Django backend with the frontend:

Define URL patterns: Define URL patterns in the urls.py file for the

frontend views. URL patterns map the URL of the frontend view to the corresponding Django view function.

Create HTML templates: Create HTML templates for the frontend views. Django templates use the Django templating language to display dynamic data. The templates can use loops, conditionals, and other logic to render dynamic content.

Serve static files: Serve static files like CSS, JavaScript, and images from the Django backend. Django provides a built-in static file serving mechanism that can be used to serve static files. Alternatively, static files can be served by a web server like Nginx or Apache.

Use AJAX: Use AJAX to update parts of the page without a full page refresh. AJAX requests can be made to Django views that return JSON or other data. The frontend can then use this data to update the page content dynamically.

Use forms: Use HTML forms to collect data from the user and send it to the Django backend. Django provides built-in form handling mechanisms that can be used to validate and process form data.

APIs

APIs, or Application Programming Interfaces, are a way for two or more software systems to communicate with each other. An API defines the rules and protocols for how these systems can interact with each other, allowing developers to build complex applications that integrate with other services.

In Django, APIs are typically implemented using the Django REST framework, which provides a powerful set of tools for building web APIs. Here are the main steps involved in implementing and using APIs in Django:

Define the API endpoints: Define the endpoints that your API will ex-

pose, including the HTTP methods that will be supported (e.g. GET, POST, PUT, etc.). This is typically done using the Django REST framework's @api-view decorator.

Serialize data: Serialize the data that will be returned by the API endpoints into a format that can be transmitted over the network. This is typically done using the Django REST framework's serializers.

Define permissions: Define the permissions that will be required to access each API endpoint. This is typically done using the Django REST framework's permission classes.

Implement views: Implement the views that will handle the API requests and return the serialized data. This is typically done using the Django REST framework's view classes.

Test the API: Test the API by making requests to the API endpoints and verifying that the expected data is returned. This can be done using the Django REST framework's built-in testing tools or using external tools like Postman.

Once the API is implemented, it can be used by other software systems to interact with your Django application. This might include mobile apps, external services, or other web applications. Users can access the API by making HTTP requests to the API endpoints, and the API will return the serialized data in the format specified by the API endpoint. APIs are a powerful tool for building complex applications that integrate with other services, and Django's REST framework provides a powerful set of tools for building and using APIs in a Django application.

8.6 Deploy

As the name suggests, web-app, the application supposed to be available to everyone across the internet. One way is to have dedicated server and connect it to internet. These physical servers can be a good option for some businesses, but they come with several significant drawbacks that are driving many businesses to consider cloud-based alternatives.

8.6.1 Cloud Deployment

Cloud deployment is the process of deploying an application to a cloud infrastructure instead of a traditional on-premise server. Cloud deployment offers several benefits, including scalability, flexibility, and cost-effectiveness. Here are some of the key benefits of cloud deployment:

Scalability: Cloud infrastructure can easily scale up or down to meet the changing demands of an application. This means that you can handle fluctuations in traffic without having to worry about hardware limitations.

Flexibility: Cloud deployment allows you to easily switch between different cloud providers or even different regions within a single provider. This provides flexibility in terms of where your application is hosted and how it is managed.

Cost-effectiveness: Cloud deployment can be more cost-effective than traditional on-premise deployment because you only pay for the resources you use. This means that you can avoid the upfront costs of purchasing and maintaining hardware.

8.6.2 Types of Deployment

Deployment types as a service refer to the different ways in which cloud services are deployed, managed, and delivered to customers. There are three main types of deployment models for cloud services as a service:

Public cloud deployment: In this type of deployment, cloud services are delivered over the internet and are managed by a third-party cloud provider. The resources are shared among multiple users, and customers pay for only the resources they use. Public cloud deployments are typically highly scalable and can be used for web hosting, software development, and testing.

Private cloud deployment: In this type of deployment, cloud services are deployed within an organization's own data center or on-premises infrastructure. The infrastructure is managed by the organization's IT department, and the resources are not shared with other organizations. Private cloud deployments offer greater control and security than public clouds but can be more expensive to operate.

Hybrid cloud deployment: In this type of deployment, cloud services are a combination of public and private clouds. Some resources are hosted in a public cloud, while others are hosted in a private cloud. A hybrid cloud allows organizations to take advantage of the scalability and cost-effectiveness of public clouds while maintaining greater control and security over sensitive data in a private cloud.

Deployment types as a service also include different ways in which cloud services are managed, such as:

Infrastructure as a Service (IaaS): IaaS is a cloud computing service model in which a third-party provider hosts and manages computing infrastructure, including servers, storage, and networking components. Ex: AWS EC2, Azure VMs, GC Compute Engine.

Platform as a Service (PaaS): PaaS is a cloud computing service model in which a third-party provider hosts and manages a computing platform, including operating system, programming language, and development tools. Ex: Heroku, Azure App Service, Google App Engine.

Software as a Service (SaaS): SaaS is a cloud computing service model in which a third-party provider hosts and manages software applications that are accessed by customers over the internet. Ex: Salesforce, Google Workspace, Microsoft 365.

Backend as a Service (BaaS): BaaS s a cloud-based platform that provides developers with a way to manage and access backend services. This includes services such as databases, search engine optimization, user authentication, analytics, notifications, and more. BaaS makes it easier for developers to create applications without the need for managing their own infrastructure. Ex: Firebase, Kinvey, AWS Amplify.

Database as a Service (DBaaS): This refers to the delivery of database management services over the internet. This includes the storage and management of data, as well as the ability to access and analyze data using a variety of tools. Ex: AWS RDS, Google Cloud SQL, Azure SQL databse.

8.7 Maintenance

Maintenance is an important aspect of web development that involves the ongoing support and upkeep of a website or web application after it has been launched. Website maintenance includes tasks such as updating content, fixing broken links, optimizing performance, ensuring security, and updating software and plugins. Some reasons why maintenance is important in web development, ensures website reliability, improves website security, increases

website performance, helps to identify and fix issues and keeps website content up-to-date. It is important to have a maintenance plan in place to ensure that your website remains up-to-date and functions properly over time. Web development is a constantly evolving field. The technologies used to build websites and web applications are always changing, so it's important to stay on top of the latest trends. Maintenance is also a good way to ensure that your website will be as functional and secure as possible.

We deployed our web application on Heroku which is a platform as a service (PaaS) provider. Heroku provides us with an easy way to deploy our application onto the Internet. Heroku has a free plan that allows us to host our application for free. The only downside of the free plan is that it limits the number of custom domains that we can have. We can upgrade by paying for more custom domains but it will cost money.

Heroku allows us to deploy our application with the push of a button. It also provides us with an easy way to manage our application by giving us access to Git, which is a version control system. We can use these tools to manage and update our code when necessary. Heroku also provides us with a dashboard that we can use to manage our application. This dashboard allows us to see how many users are currently on the site, what they are doing and how much bandwidth our application is using.

Chapter 9

Result and Analysis

As the whole project is branched to many fields, we inevitably end up with numerous results from different works. Starting from web scraping, dataset making, keyword Extraction, Performance Analysis of those Keyword Extractors, and Developing and deploying a Web application.

9.1 Web Scraping

Web scraping has been a predominant field in NLP. Even though there are many movie datasets present, we wanted to create our own dataset so that it can be flexible in all fields of our projects, whether it be keyword Extraction or the database for the Web Application. Also, In order to stay with the trends, live data has been scraped from movie sites like IMDB & Rotten Tomatoes. We extracted movie details such as title, year, director, rating, cast, and plot summary. We collected data for a total of 10,000 movies that are the top movies at that time. The data was cleaned and processed to create a dataset in CSV format, which can be used for further analysis and modeling. Real-time data has been scraped from the web so that instead of

sticking to the dataset made. We have the flexibility to update the dataset or create a new dataset in preference. Different genres & language movies have also been scraped to provide the player with more options when they start a game. With this, we have the option to play the game based on Language and genre. Not restricting this to only movies, we decided to create a dataset for K-dramas too.

Overall, our web scraping and dataset making of movie details from IMDB provides a useful resource for anyone interested in analyzing and understanding the trends and patterns in the movie industry.

	movie_title	year	genre	synopsis	cast
0	A Splash of Love	2022	Comedy, Romance, Back to top	Chloe Turner, a Ph.D. candidate in Marine Mamm...	Heather Hawthorn Doyle (dir.), Rhiannon Fish, ...
1	The Grey Man	2007	Biography, Crime, Drama, Back to top	Kevin Dodds is a browbeaten deputy bank manage...	Declan O'Dwyer (dir.), Daniel Ryan, Nitin Ganatra
2	Descendants	2015	Comedy, Family, Fantasy, Back to top	Ben, son of Belle and the once selfish Beast, ...	Kenny Ortega (dir.), Dove Cameron, Cameron Boyce
3	Teen Wolf: The Movie	n	Action, Comedy, Drama, Back to top	A full moon rises in Beacon Hills, and with it...	Russell Mulcahy (dir.), Melissa Ponzio, Linden...
4	High School Musical	2006	Comedy, Drama, Family, Back to top	Troy Bolton and Gabriella Montez are two total...	Kenny Ortega (dir.), Zac Efron, Vanessa Hudgens

Figure 9.1: Dataset after web scraping

All the datasets have been deployed and are available for public usage through Kaggle, a subsidiary of Google LLC, is an online community of data scientists and machine learning practitioners. Kaggle allows users to find and publish data sets, and explore and build models.

We have successfully received the DOIs for our respective Datasets after successfully publishing our Datasets. DOIs are attached below for further

reference.

- IMDB Movies Dataset
  DOI=10.34740/KAGGLE/DSV/4853944
- Rotten Tomatoes Film Dataset
  DOI=10.34740/KAGGLE/DSV/4854359
- Rotten Tomatoes Series Dataset-2
  DOI=10.34740/KAGGLE/DSV/4854461
- MyDramaList Movies & Drama's
  DOI=10.34740/KAGGLE/DSV/4854461

9.2 Keyword Extraction

The keyword datasets along with the TF-IDF score and cosine similarity are kept exclusive. We will be deploying those datasets along with the survey paper that we are currently working on. The main dataset consists of the keywords extracted from all the algorithms we used. Snippets of those Datasets are attached for reference.

	movie_title	year	genre	synopsis	cast	Key-Bert	Yake	Sentence_transformers
0	A Splash of Love	2022	Comedy, Romance, Back to top	Chloe Turner, a Ph.D. candidate in Marine Mamm...	Heather Hawthorn Doyle (dir.), Rhiannon Fish, ...	['whale', 'miami', 'fish', 'pacific', 'marine']	['Miami Central College', 'Central College', '...	['chloe', 'turner', 'help', 'northwest', 'loca...
1	The Grey Man	2007	Biography, Crime, Drama, Back to top	Kevin Dodds is a browbeaten deputy bank manage...	Declan O'Dwyer (dir.), Daniel Ryan, Nitin Ganatra	['rob', 'manager', 'bank', 'novel', 'devises']	['Kevin Dodds', 'browbeaten deputy bank', 'dep...	['bank', 'mcnab']
2	Descendants	2015	Comedy, Family, Fantasy, Back to top	Ben, son of Belle and the once selfish Beast, ...	Kenny Ortega (dir.), Dove Cameron, Cameron Boyce	['son', 'selfish', 'villain', 'beast', 'lost']	['attend Auradon Prep', 'Auradon Prep', 'son o...	['ben']
3	Teen Wolf: The Movie	n	Action, Comedy, Drama, Back to top	A full moon rises in Beacon Hills, and with it...	Russell Mulcahy (dir.), Melissa Ponzio, Linden...	['werewolf', 'wolves', 'howling', 'deadliest',...	['full moon rises', 'Beacon Hills', 'rises in ...	['moon rises', 'trusted', 'werecoyotes']
4	High School Musical	2006	Comedy, Drama, Family, Back to top	Troy Bolton and Gabriella Montez are two total...	Kenny Ortega (dir.), Zac Efron, Vanessa Hudgens	['decathlon', 'karaoke', 'mexico', 'party', 'a...	['Year Eve', 'Gabriella Montez', 'Troy Bolton'...	['troy', 'gabriella', 'hate', 'bolton', 'week'...

Figure 9.2: Dataset with keywords

9.3 Performance Analysis

It has become a hassle to select a particular keyword extractor algorithm to determine the whole dataset, As most of the keyword Extractors have their own score. Let us take YAKE for example, It considers several factors like the term frequency, positional importance & word co-occurrence, TF-IDF, etc.

Let us take a look at this example with a random movie plot.After considering several factors that are above mentioned, YAKE assigns a particular score to the possible n-grams present, and the keywords with the lowest scores are selected.

```
t="A full moon rises in Beacon Hills, and with it a
terrifying evil has emerged. The wolves are howling
once again, calling for the return of Banshees,
Werecoyotes, Hellhounds, Kitsunes, and every other
shapeshifter in the night. But only a werewolf like
Scott McCall, no longer a teenager yet still an Alpha
, can gather both new allies and reunite trusted friends
to fight back against what could be the most powerful
and deadliest enemy they've ever faced."
x=yake_func(t)
x
```

```
[('full moon rises', 0.0027080509450578723),
 ('Beacon Hills', 0.003754403924906832),
 ('rises in Beacon', 0.00961004627146306),
 ('evil has emerged', 0.013784437454882689),
 ('full moon', 0.01911893411113409),
 ('moon rises', 0.01911893411113409),
 ('terrifying evil', 0.01911893411113409),
 ('Hills', 0.053822159537056186),
 ('return of Banshees', 0.0622698075188547),
 ('Beacon', 0.06949481646017519)]
```

Figure 9.3: Working of YAKE

RAKE on the other hand follows a different way. Among the words of the candidate keywords, the algorithm looks at how many times each word is occurring and how many times it co-occurs with other words. Each word gets a score which is the ratio of the word degree (how many times it co-occurs with other words) to the word frequency a RAKE score for the full candidate keyword is calculated by summing up the scores of each of the words which define the candidate keyword.

Instead of sticking to the one-dimensional standard of selecting a particular keyword Extractor, different metrics have been considered to select the proper keyword extractor. Apart from their individual scores, another straightforward metric to select the keywords is **TF-IDF**.

TF (Term Frequency) measures how often a particular word or phrase appears in a document. If a word appears frequently, it is likely to be important to the document's meaning.

IDF (Inverse Document Frequency) measures how rare a word is across a corpus (i.e., a collection of documents).

Together, TF-IDF creates a score for each word that reflects its importance in a particular document relative to its importance in the corpus. The higher the TF-IDF score for a word, the more important it is to that document.

First, All the English stopwords such as"the", "a" etc along with the punctuation marks have been cleaned. Then, every word present in the corpus is vectorized with some random value using a count vectorizer, Both the term frequency and the inverse document frequency are calculated for all the plots in the dataset. Multiplying both the TF & IDF scores, you get the TF-IDF score, on whose basis the keywords are selected. Here are the keywords extracted based on the TF-IDF score.

Having different metrics for all the keyword Extractors, and a definite

	title	synopsis_cleaned	top_keywords
0	A Splash of Love	chloe turner a phd candidate in marine mammalo...	[cove, 0.262, cable, 0.244, population, 0.225,...
1	The Grey Man	kevin dodds is a browbeaten deputy bank manage...	[bank, 0.413, dodds, 0.336, browbeaten, mcnab,...
2	Descendants	ben son of belle and the once selfish beast is...	[auradon, 0.315, poised, 0.302, isle, prep, 0....
3	Teen Wolf: The Movie	a full moon rises in beacon hills and with it ...	[shapeshifter, 0.241, wolves, 0.224, howling, ...
4	High School Musical	troy bolton and gabriella montez are two total...	[gabriella, 0.477, troy, 0.419, new, 0.204, tr...

Figure 9.4: Keywords with TF-IDF score

reason for selecting these particular keyword extractors is needed. This metric has to be compatible with the results of all the keyword Extractors and should flow well with them. Considering all these reasons, Cosine similarity has been selected.

Cosine similarity is a measure of similarity between two vectors in a multi-dimensional space. In the case of keyword extractors, each vector represents a document, and the dimensions of the vector represent the frequency of each word in the document.

To calculate cosine similarity between two documents, we first represent each document as a vector. We then calculate the cosine of the angle between the two vectors. If the two vectors are identical (i.e., the documents have exactly the same set of words and word frequencies), the cosine similarity will be 1. If the two vectors are completely dissimilar (i.e., the documents have no words in common), the cosine similarity will be 0.

Now, this cosine similarity is run on all the keywords extracted using different keyword extractors. One on One cosine similarity has been calculated on each keyword Extractor. The keywords with the highest cosine similarity are selected and they are passed for mapping images. Based on the score of cosine similarity, the two better-performing keyword Extractors have been selected.

movie_title	year	genre	synopsis	cast	Key-Bert	Yake	Sentence_transformers	rake
Persuasion	2007	Action, Drama, Family, Back to top	Royal Navy captain Wentworth was haughtily tur...	Adrian Shergold (dir.), Sally Hawkins, Alice K...	['baronet', 'lovers', 'daughter', 'brother', '...	['Sir Walter Elliot', 'Elliot daughter Anne', ...	['royal', 'navy captain', 'wentworth', 'pompou...	[(96.5, 'pompous baronet sir walter elliots da...
Z-O-M-B-I-E-S	2018	Family, Musical, Romance, Back to top	Disney's "ZOMBIES" is a music and dance filled...	Paul Hoen (dir.), Milo Manheim, Meg Donnelly	['zombietown', 'zombie', 'zombies', 'apocalyps...	['dance filled story', 'filled story set', 'co...	['zombies', 'zombie', 'community', 'addison', ...	[(23.666666666666668, 'play football meets fre...

bert_yake_sim	bert_st_sim	bert_rake_sim	yake_st_sim	yake_rake_sim	st_rake_sim
0.000000	0.000000	0.036579	0.000000	0.000000	0.052033
0.305137	0.194314	0.298972	0.175786	0.425867	0.410315
0.126315	0.000000	0.277396	0.000000	0.132711	0.000000
0.000000	0.000000	0.183258	0.350976	0.000000	0.130400
0.000000	0.000000	0.040022	0.000000	0.108788	0.228132

Figure 9.6: Cosine similarity of all keyword Extractors

This is just to showcase the similarity of keywords from different algorithms. As we can infer, RAKE is one of the most consistent followed by the sentence Transformers. RAKE scores have been consistent throughout the whole dataset, making it evident. All the keyword Extractors have been put through different tests in order to select the most efficient one.

After the synopsis is cleaned by removing all the stopwords such as "In", "the", "is", "an" etc., and punctuation marks. Term Frequency(TF) is calculated for all the words present in the corpus. The IDF score measures how rare or common a word is across all documents in the corpus.Both scores are

calculated on the whole corpus. By combining these two metrics, TF-IDF can effectively identify important and relevant words within a document, while downplaying common or unimportant words.

title	synopsis_cleaned	top_keywords
A Splash of Love	chloe turner a phd candidate in marine mammalo...	[cove, 0.262, cable, 0.244, population, 0.225,...
The Grey Man	kevin dodds is a browbeaten deputy bank manage...	[bank, 0.413, dodds, 0.336, browbeaten, mcnab,...
Descendants	ben son of belle and the once selfish beast is...	[auradon, 0.315, poised, 0.302, isle, prep, 0....
Teen Wolf: The Movie	a full moon rises in beacon hills and with it ...	[shapeshifter, 0.241, wolves, 0.224, howling, ...
High School Musical	troy bolton and gabriella montez are two total...	[gabriella, 0.477, troy, 0.419, new, 0.204, tr...
...	...	...
Hyperland	a mother afflicted with depression comes into ...	[afflicted, 0.416, mother, 0.347, depression, ...
The Brass Ring	when a wealthy businessman is murdered by a hi...	[assassin, 0.388, abused, 0.356, suspicion, 0....
One Hot Summer Night	a look at yellowstone national parks wildlife ...	[solopiano, 0.363, minimal, 0.347, yellowstone...
Yellowstone in Four Seasons	biopic of sorts about agatha christie the fame...	[christie, 0.346, 1926, 0.244, psychiatrist, 0...
gatha Christie: A Life in Pictures	it looks like we dont have any plot summaries ...	[page, 0.428, plot, 0.401, summaries, 0.216, c...

Figure 9.7: Dataset with TF-IDF

9.4 Benchmark

Here is the benchmark function of all the keyword Extractors used. The working, time consumed, accuracy, etc have been keenly observed and noted. Working on Keywords Extractors for so long, the goal slightly changed to finding the best keyword Extractor that is most suitable for our project in addition to high performance. As the dataset is quite large, there might be a shift in the paradigm of keyword Extractors.

So, several well-known keyword Extractors are put to test. The keyword Extractors are defined in accordance with the dataset to extract the top 5 keywords. Each extractor takes in as an argument the synopsis from the dataset, from which we want to extract keywords, and returns a list of key-

words, from the best to the worse according to their weighing technique. This is a straightforward approach to counter the extra disturbance in the data. As we can see from the previous figures, some of the keywords are n-grams rather than a single word. We decided to keep it that way because some of the single-word keywords are ambiguous. To counter the n-grams that don't have a definite meaning, We are using pos tagging to restrict the accepted grammar patterns so that the n-gram keywords have a definite meaning.

Now, the keywords extractors are run on the whole dataset of 10000 movie corpus. Another function is written to capture the information passed by these algorithms and some other useful information like the time taken to execute the task. These keywords are passed through a match function that checks whether they are sensical or not. Spacy comes in handy with the Matcher object. We'll define a match function that takes in a keyword and returns True or False if the defined patterns match.

Finally, the benchmark function is called which has all the extractors and takes in the whole corpus as the input. For each extractor, it extracts the keywords and puts them in a dictionary for easy access.
For each algorithm in the list, we compute:

- average number of extracted keywords
- average number of matched keywords
- compute a score that takes into account the average number of matches found divided by how much time it took to perform the operation

And here are the results:

	algorithm	elapsed_time	avg_keywords_per_document	avg_matched_keywords_per_document	avg_percentage_matched_keywords	performance_score
0	rake_extractor	00:00:06	4.941617	3.728914	0.75	0.61
1	yake_extractor	00:03:41	4.995280	3.495691	0.70	0.02
2	topic_rank_extractor	04:01:42	4.802893	1.939668	0.40	0.00
3	position_rank_extractor	04:01:46	4.873281	3.058999	0.63	0.00
4	single_rank_extractor	03:51:37	4.906731	3.590088	0.73	0.00
5	multipartite_rank_extractor	04:08:42	4.868767	1.977221	0.41	0.00
6	keybert_extractor	01:26:25	4.984404	4.863534	0.98	0.00
7	st	03:39:30	4.906731	3.590088	0.73	0.00

Figure 9.8: Benchmark

As we can infer, several metrics have been calculated such as the total time taken to extract the keywords by each extractor, the average number of keywords extracted for each movie, the average number of keywords matched to the defined pos tagging, performance score, etc.

And a bar graph to easily visualize the data

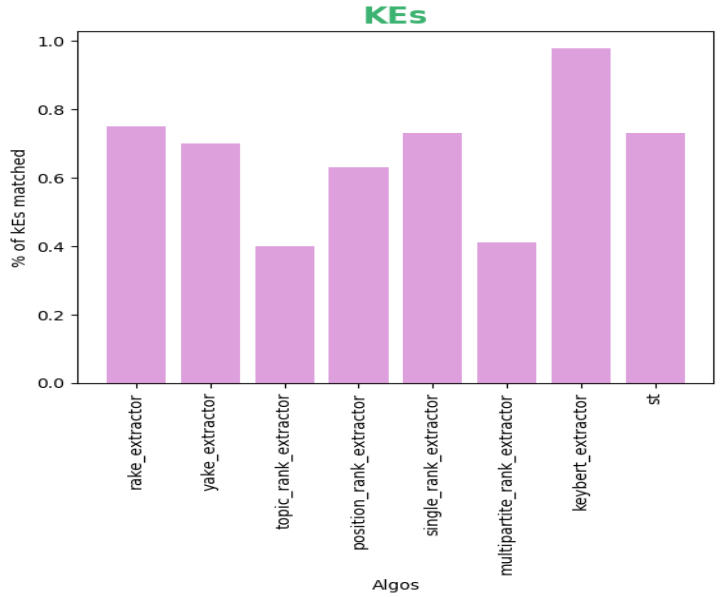

Figure 9.9: Average percentage of matched keywords

As we can see, **keyBERT** has been the most consistent across the whole dataset when compared to others with an enormous accuracy of 98%, followed by **RAKE** & **Sentence Transformers** with an accuracy of 75% &

73% respectively.Other keyword Extractors cannot be ignored as the accuracy is relatively close.Definite selection of a particular keyword extractor cannot be done based entirely on this.

But, when we take the time elapsed for the extraction across the dataset into consideration, keyBERT has taken precisely 1 hr 26 min 25 sec, whereas RAKE only took a mere 6 seconds. As the performance score takes into account accuracy over time, we can a dip in almost all of the keyword Extractors.

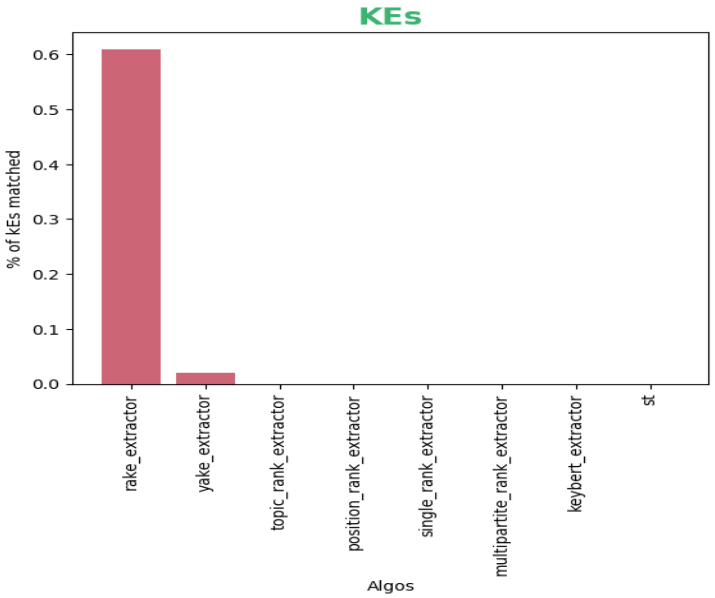

Figure 9.10: Performace score

The time taken for each algorithm for running on the entire corpus is shocking because RAKE, which is a compact keyword Extractor, only took 6 seconds to run, followed by YAKE which took 3 minutes & 41 seconds followed by keyBERT & Sentence Transformers respectively. Graph-based methods such as TextRAnk, PositionRank, etc have taken more than 4 hours. As we are considering the time taken, the performance score of most of the keywords dipped because, they took much more time compared to RAKE which only took 6 seconds, whereas some of them took 4 hours.

RAKE performed consistently in both cases making it the first keyword Extractor to consider.

This is the final part of Data Visualisation. It visualizes all the data we calculated till now using the benchmark function. i.e time taken, the average number of keywords & percentage of matched keywords, and performance score.

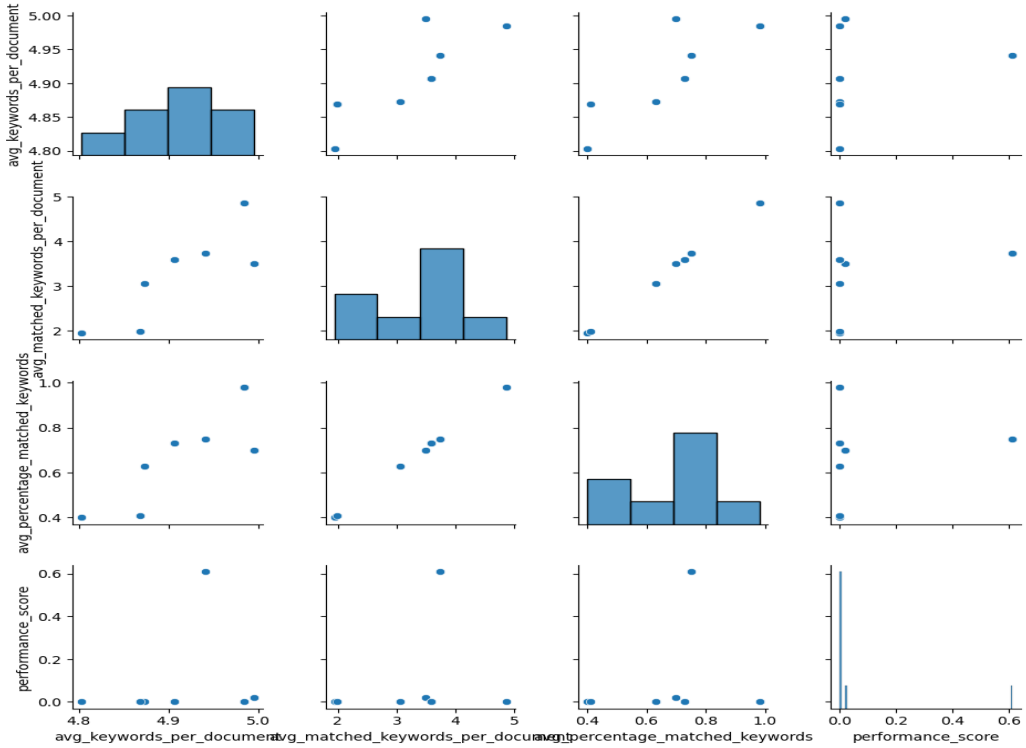

Figure 9.11: Dataset with keywords

Chapter 10

Conclusion

This project branched into many fields and gave us the flexibility to learn different things and implement them in our project. Right from the get-go, Datasets have been created from scratch using web scraping. Movie details have been scraped from various credible sources like IMDB, Rotten Tomatoes, etc. Datasets have been published through Kaggle for public usage, as this is a chance for us to help them build their projects with the help of these Datasets.

And Keyword Extraction helped us understand how deep the field of Natural Language Processing (NLP) actually is and this is just the tip of the iceberg. The working of several keyword Extractors has been carefully studied and the performance analysis of these keyword Extractors is keenly done. From the benchmarks, we can conclude that RAKE is the most consistent keyword Extractor among them considering several factors like the time taken, accuracy, cosine similarity, etc. It has been decided that RAKE & keyBERT will be used for our project, which scored the best in different metrics.

Image processing has been done on the images, We used a scaled version

of YOLOv4 which is modified for our project. The scaled version performed better than the original YOLOv4. As YOLO is a huge algorithm many of the functions will go unused. So, it has been toned down.

10.1 Contribution

We believe our datasets have the capability to support many other projects such as Movie recommendation systems, Movie rating and review analysis, Box office prediction and analysis, Genre and theme analysis, etc. This has been one of the main reasons for us to publicize our Datasets.
Keyword Extraction is greatly studied in the context of general text. Even though, keyword Extraction is a huge topic, little work has been done in the movies section using keywords. the working of Keyword Extractors in the context of movies has been highlighted through this project. Because the normal working of keyword Extractors doesn't exactly comply with the movies especially if we want the keywords to be the important parts of the plot of the movie. This certainly gives a new perspective to the working of keyword Extractors.

Coming to the web application, I think this contributes in a very different way compared to the earlier. This surely will be a memorable time spent with your group of friends. And, Improving the quality of time spent is the ultimate goal and we believe what we felt during the development stage and hope the users reciprocate the feeling. There have been some obstacles in the way, which killed a lot of resources & time for us and we want others to recognize those hurdles and tackle them in their own way.

10.2 Summary of Findings

10.2.1 Web Scraping

The use of beautiful soup for web scraping has been proven optimal because of its simplicity. we know most websites are built with HTML tags, Using beautiful soup we extract the data present in those tags, which made our work easy albeit with some pre-processing & cleaning. And, selenium has automated the whole extraction of movie details and relevant images from the web.

10.2.2 Keyword Extraction

The performance analysis of the keyword Extractors clearly shows us the consistency of different keyword Extractors when working with different environments. We clearly understand why RAKE has been one of the widely used Keyword extractors. Both because of the accuracy & the compact structure of the algorithm. Looking at the time taken for the extractors would present us with a clearer picture. For the TopicRank algorithm to run on the whole dataset that comprises 10000 movie details, It took almost 4hrs whereas RAKE only took just 6 seconds. which is 2400 times faster when compared. Imagine the time difference for even larger datasets than this.

10.2.3 Similarity Measures

We observed pplying similarity measures on large data can have several disadvantages, including:

Computational complexity: As the data size increases, the computational complexity of similarity measures also increases. This means that the time

and resources required to compute similarity measures on large data can be significant, leading to slower processing times and increased costs.

Memory requirements: Similarity measures often require large amounts of memory to store the data being compared. As the data size increases, the memory requirements for similarity measures also increase, which can lead to issues with memory management and performance.

Sensitivity to noise: Similarity measures can be sensitive to noise and outliers in the data, especially when dealing with large datasets. This can lead to inaccurate results and a loss of accuracy in the similarity measures.

Difficulty in interpretation: As the data size grows, the number of comparisons and similarities being calculated also increases, making it more difficult to interpret the results and draw meaningful insights from the data.

Need for feature reduction: When dealing with large data, it is often necessary to reduce the number of features or attributes being considered in order to improve the performance and accuracy of similarity measures. This can be a time-consuming and complex process, requiring significant expertise in data science and machine learning.

Overall, while similarity measures can be a powerful tool for data analysis, they do have limitations when applied to large datasets. It is important to carefully consider the computational and memory requirements, and to take steps to reduce noise and simplify the data where possible in order to obtain accurate and meaningful results.

10.3 Future Work

We strongly believe that this project has got great potential and could be expanded to many more future works and development. Starting from datasets,

the main dataset only contains Hollywood movies as of now. Work has been going on to create datasets in popular & native languages to make the user feel more included. Genre-based Data collection can be done so, we can provide the user with a plethora of options to choose from, whether he wants to play only Telugu movies or Malayalam movies, or only Romance movies. Not restricting ourselves to this, Datasets on anime would be a huge deal considering the number of people that are avid watchers of anime. There are abundant ways to improve the user experience.

Working of keyword Extractors can be optimized by training the model, if they are supervised or curating the model if the extractors are unsupervised. A minute difference in accuracy can make an enormous change.

The structure of the web application can be greatly improved and we are thinking of implementing a multiplayer option very soon. Where the players can join the room and are randomly assigned or chosen in a team. Each team gets its own chat room to discuss the answer. As we worked with a time constraint and with no prior knowledge, we will certainly try to make it better.

10.4 Work in Progress

We are implementing multi-player mode in our application with synchronised communication between team members.
A survey paper based on the working of the keyword Extractors is underway, and we promise to complete it and get acceptance immediately.
Talks with our project Guide have been going on regarding a research paper based on the performance analysis of the keyword Extractors.

References

[1] M. G. Thushara, T. Mownika and R. Mangamuru, "A Comparative Study on different Keyword Extraction Algorithms," 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 2019, pp. 969-973, doi: 10.1109/ICCMC.2019.8819630.

[2] Rose, Stuart & Engel, Dave & Cramer, Nick & Cowley, Wendy. (2010). Automatic Keyword Extraction from Individual Documents. 10.1002/9780470689646.ch1.

[3] Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C. and Jatowt, A. (2020).YAKE! Keyword Extraction from Single Documents using Multiple Local Features. In Information Sciences Journal. Elsevier, Vol 509, pp 257-289, ISSN 0020-0255.

[4] Reimers, Nils & Gurevych, Iryna. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. 3973-3983. 10.18653/v1/D19-1410.

[5] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In Proceedings of the 2004 Conference on Empirical Methods in

Natural Language Processing, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

[6] Corina Florescu and Cornelia Caragea. 2017. PositionRank: An Unsupervised Approach to Keyphrase Extraction from Scholarly Documents. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1105–1115, Vancouver, Canada. Association for Computational Linguistics.

[7] Košút, M., Šimko, M. (2016). Improving Keyword Extraction from Movie Subtitles by Utilizing Temporal Properties. In: Freivalds, R., Engels, G., Catania, B. (eds) SOFSEM 2016: Theory and Practice of Computer Science. SOFSEM 2016. Lecture Notes in Computer Science(), vol 9587. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-49192-8_44

[8] Holovaty, A., & Kaplan-Moss, J. (2006). The Django web framework. Proceedings of the 2006 Python Conference (pp. 1-9).

[9] Jabbar, S. M., & Naveed, N. (2017). Development of a web-based student information system using Django framework. Journal of Computer Science and Information Technology, 5(1), 1-10.

[10] Bhattacharya, A. (2019). An overview of Django framework in Python. International Journal of Innovative Technology and Exploring Engineering, 8(10S2), 271-276.

[11] Das, Debashis & Sahoo, Laxman & Datta, Sujoy. (2017). A Survey on Recommendation System. International Journal of Computer Applications. 160. 6-10. 10.5120/ijca2017913081

[12] M K, Vijaymeena K, Kavitha. (2016). A Survey on Similarity Measures in Text Mining. Machine Learning and Applications: An International Journal. 3. 19-28. 10.5121/mlaij.2016.3103

[13] Cabral, L. M., Ribeiro-Neto, B., Cristo, M. (2007). Keyword-based similarity measures for text document clustering. In Advances in Information Retrieval (pp. 115-126). Springer Berlin Heidelberg. DOI: 10.1007/978-3-540-74783-2_21

[14] S. S. Salankar, P. J. Kulkarni, and S. P. Narote, "Keyword-based text document similarity," 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), New Delhi, 2014, pp. 1979-1983, doi: 10.1109/ICACCI.2014.6968435.

[15] Sumaiya Iqbal et al. "A survey of similarity and distance measures used in graph-based clustering." 2016 International Conference on Microelectronics, Electromagnetics and Telecommunications (ICMEET), pp. 1-6, 2016, doi: 10.1109/ICMEET.2016.7587976.

[16] H. B. Kekre, S. S. Natu, S. A. Bharadi, and T. S. Chavan, "A comparative study of similarity measures for graph clustering and classification," 2015 International Conference on Computational Intelligence, Communication and Computer Technologies (ICCICCT), pp. 644-649, doi: 10.1109/ICCICCT.2015.7480005.

Appendix A

Source code

A.1 Dataset

Datasets have been created from scratch using web scraping, with emphasis on the plural of Datasets. Several Datasets have been created, the main one being the real-time top 10000 movies from IMDB. The rest of the datasets have been scraped from Rotten Tomatoes & MyDramaList. All the datasets have been published for public usage through Kaggle. DOIs have been mentioned before.

IMDB dataset contains movie details like the name of the movie, Year, Genre, Synopsis, Cast & Crew which are required for the web application. And the keyword Extractors are run on the cleaned dataset resulting in different sets of keywords from different algorithms.
Here is the snippet of the web scraping:

```
def extract(url):
    response = requests.get(url)
    soup = BeautifulSoup(response.text, "html.parser")
    mov_list = soup.find_all("span", {"class": "lister-item-header"})

    for i in mov_list:
        # Movie Title Extraction
        titles.append(i.find("a").getText())

        # Year Extraction
        year=(i.find("span",{"class":"lister-item-year text-muted unbold"}).getText())[1:5]
        if year.isnumeric()==True: years.append(str(year))
        else: years.append("NA")

        # Cast
        s=i.find_all("span")
        cast.append(str(s[1].attrs.get('title')))

        # Link
        link=i.find("a").attrs.get('href')

        # Synopsis Extraction
        site = pre+link+post
        resp = requests.get(site)
        extract = BeautifulSoup(resp.text, "html.parser")
        q_tags = extract.find_all('li',class_="ipl-zebra-list__item")
        s=[]
        for j in q_tags:
            s.append([j.text.strip().split('\n')][0])
        if len(s)>2: summary.append(s[1][0])
        else: summary.append(s[0][0])

        # Genre Extraction
        site=pre+link
        resp = requests.get(site)
        extract = BeautifulSoup(resp.text, "html.parser")
        q_tags = extract.find_all('span',class_="ipc-chip__text")
        genre=""
        for j in q_tags:
            genre+=str(j.text.strip().split('\n'))[2:-2]+", "
        genres.append(genre[:-2])
```

Figure A.1: Web Scraping top 10000 movies from IMDB

A.1.1 Pre-Processing

Scraped data has been cleaned, empty data fields are replaced or deleted. Removed all the whitespaces and newlines. And, Stopwords & punctuations are deleted from the synopsis.

```python
def get_stopwords_list(stop_file_path):
    """load stop words """

    with open(stop_file_path, 'r', encoding="utf-8") as f:
        stopwords = f.readlines()
        stop_set = set(m.strip() for m in stopwords)
        return list(frozenset(stop_set))
```

```python
def clean_text(text):
    """Doc cleaning"""

    # Lowering text
    text = text.lower()

    # Removing punctuation
    text = "".join([c for c in text if c not in PUNCTUATION])

    # Removing whitespace and newlines
    text = re.sub('\s+',' ',text)

    return text
```

```python
# Constants
PUNCTUATION = """!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~"""
TOP_K_KEYWORDS = 10 # top k number of keywords to retrieve in a ranked document
STOPWORD_PATH = "stopwords.txt"
PAPERS_PATH = "title_synopsis.csv"
```

Figure A.2: Dataset cleaning

A.1.2 Keyword Extraction & Benchmark

All the keyword Extractor algorithms have been defined and modified according to the project. Pre-processing is also done before running.

KE.py

```
# initiate BERT outside of functions
bert = KeyBERT()
# 1. RAKE
def rake_extractor(text):
    """
    Uses Rake to extract the top 5 keywords from a text
    Arguments: text (str)
    Returns: list of keywords (list)
    """
    r = Rake()
    r.extract_keywords_from_text(text)
    return r.get_ranked_phrases()[:5]
# 2. YAKE
def yake_extractor(text):
    keywords = yake.KeywordExtractor(lan="en", n=3, windowsSize=3,
top=5).extract_keywords(text)
    results = []
    for scored_keywords in keywords:
        for keyword in scored_keywords:
            if isinstance(keyword, str):
                results.append(keyword)
    return results
# KeyBERT
def keybert_extractor(text):
    keywords = bert.extract_keywords(text, keyphrase_ngram_range=(3,
5), stop_words="english", top_n=5)
    results = []
    for scored_keywords in keywords:
        for keyword in scored_keywords:
            if isinstance(keyword, str):
                results.append(keyword)
    return results
# Sentence Transformers Algo
from summa import keywords
def st(summary):
    st_keywords = keywords.keywords(summary, scores=False)
    return st_keywords
```

Figure A.3: Defining of Keyword Extractors

This is the final & important metric of selecting the keyword Extractors from the bunch. All the algorithms are run on the whole dataset and the time taken for it, average keywords & performance of the keyword Extractor has been calculated in accordance with the time taken. From the results we can infer that RAKE & keyBERT are the best options for our project.
Here is the benchmark code for the reference:

```
def benchmark(corpus, shuffle=True):
    logging.info("Starting benchmark...\n")

    # Shuffle the corpus
    if shuffle:
        random.shuffle(corpus)
```

```
# extract keywords from corpus
results = []
extractors = [
    rake_extractor,
    yake_extractor,
    topic_rank_extractor,
    position_rank_extractor,
    single_rank_extractor,
    multipartite_rank_extractor,
    keybert_extractor,
    st,


]
for extractor in extractors:
    result = extract_keywords_from_corpus(
                extractor, corpus)
    df99=pd.DataFrame(result)
    df99.to_csv('GFG.csv')
    results.append(result)

for result in results:
    len_of_kw_list = []
    for kws in result["corpus_kws"].values():
        len_of_kw_list.append(len(kws))
    result["avg_keywords_per_document"] = np.mean(
```

```
        len_of_kw_list)

    # match keywords
    for result in results:
        for idx, kws in result["corpus_kws"].items():
            match_results = []
            for kw in kws:
                match_results.append(match(kw))
                result["corpus_kws"][idx] = match_results

    # compute average number of matched keywords
    for result in results:
        len_of_matching_kws_list = []
        for idx, kws in result["corpus_kws"].items():
            len_of_matching_kws_list.append
            (len([kw for kw in kws if kw]))
        result["avg_matched_keywords_per_document"] =
        np.mean(len_of_matching_kws_list)
        result["avg_percentage_matched_keywords"] =
        round(result["avg_matched_keywords_per_document"]
              / result["avg_keywords_per_document"], 2)

    for result in results:
        elapsed_seconds = get_sec(
                          result["elapsed_time"]) + 0.1
        # weigh the score based on the time elapsed
        result["performance_score"] =
```

```
            round(result["avg_matched_keywords_per_document"]
                    / elapsed_seconds, 2)

    # delete corpus_kw
    for result in results:
        del result["corpus_kws"]

    # create results dataframe
    df = pd.DataFrame(results)
    df.to_csv("results.csv", index=False)
    logging.info(
        "Benchmark finished. Results saved to results.csv")
    return df
```

Your Appendix here

A.2 Web app

A.2.1 Code

views.py

```
from django.shortcuts import render, redirect
from .forms import RegistrationForm, AuthenticationForm
from django.contrib.auth import login,logout
from random import randint,sample,shuffle
from .models import Movie,User,Game,Stat
from django.http import JsonResponse,HttpResponseBadRequest
```

```
import json
from django.views.decorators.csrf import csrf_exempt

@csrf_exempt
def login_view(request):
    if request.method == 'POST':
        form = AuthenticationForm(data=request.POST)
        if form.is_valid():
            user = form.get_user()
            login(request, user)
            return redirect('home')
        rform = RegistrationForm(request.POST)
        if rform.is_valid():
            rform.save()
            return redirect('login')
    else:
        form = AuthenticationForm()
        rform = RegistrationForm()
    return render(request, 'login.html', {'form': form,
                                          'rform': rform})

@csrf_exempt
def logout_view(request):
    logout(request)
    return redirect('home')

@csrf_exempt
```

```
def home(request):
    leader_board=Stat.objects.all().order_by('-points')[:5]
    contents={'lb':leader_board}
    if request.user.is_authenticated:
        print("Ayee")
        try:
            st=Stat.objects.get(player=request.user)
            contents['st']=st
        except:
            contents['st']=Stat(player=request.user,
                                matches=0,
                                wins=0,
                                points=0)
    return render(request,'home.html',contents)

@csrf_exempt
def profile(request):
    return render(request,'profile.html')

@csrf_exempt
def new_movie(request,qno,score):
    contents={}
    user=User.objects.get(username=request.user.username)
    pk=randint(1,len(Movie.objects.all()))
    mvs=list(Movie.objects.all().values_list('title',flat=True))
    mv=Movie.objects.get(pk=pk)
    opts=[mv.title]+sample(mvs,3)
```

```
    shuffle(opts)
    opt1,opt2,opt3,opt4=opts
    gid=len(Game.objects.all())
    game=Game(gameid=gid,player=user,movie=mv,points=0)
    game.save()
    contents={"q":qno,
              "gid":gid,
              "p1":mv.p1,
              "p2":mv.p2,
              "p3":mv.p3,
              "points":100,
              "score":score,
              "genre":"########",
              "year":"########",
              "cast":"########",
              "opt1":opt1,
              "opt2":opt2,
              "opt3":opt3,
              "opt4":opt4}
    return render(request,"game.html",contents)

@csrf_exempt
def play(request):
    return new_movie(request,1,0)

@csrf_exempt
def quick_play(request):
```

```
    contents={}
    if request.method == 'GET':
        pk=randint(1,len(Movie.objects.all()))
        mvs=list(Movie.objects.all().values_list(
                'title',flat=True))
        mv=Movie.objects.get(pk=pk)
        opts=[mv.title]+sample(mvs,3)
        shuffle(opts)
        opt1,opt2,opt3,opt4=opts
        gid=len(Game.objects.all())
        game=Game(gameid=gid,player=None,movie=mv,points=0)
        game.save()
        contents={"gid":gid,
                  "points":100,
                  "p1":mv.p1,
                  "p2":mv.p2,
                  "p3":mv.p3,
                  "genre":"#######",
                  "year":"#######",
                  "cast":"#######",
                  "opt1":opt1,
                  "opt2":opt2,
                  "opt3":opt3,
                  "opt4":opt4}
    return render(request,"quick-game.html",contents)

@csrf_exempt
```

```
def req_hint(request):
    hint_type = request.POST.get('hint')
    gid=request.POST.get('gid')
    game=Game.objects.get(gameid=gid)
    contents={"value":""}
    if hint_type=="genre":
        contents["value"]=game.movie.genre[:-13]
    elif hint_type=="year":
        contents["value"]=game.movie.year
    elif hint_type=="cast":
        contents["value"]=game.movie.cast
    return  JsonResponse(contents)

@csrf_exempt
def check_answer(request):
    submitted_answer=request.POST.get('value')
    user=User.objects.get(username=request.user.username)
    gid=request.POST.get('gid')
    game=Game.objects.get(gameid=gid)
    score=int(request.POST.get('points'))
    correct_answer=game.movie.title
    win=submitted_answer==correct_answer
    print(submitted_answer,correct_answer,win)
    if win:
        game.points=score
        game.save()
        if not Stat.objects.filter(player=user).exists():
```

```
                stat=Stat(
                    player=user,
                    matches=1,
                    points=score,
                    wins=1)
                stat.save()
            else:
                stat=Stat.objects.get(player=user)
                stat.matches+=1
                stat.points+=score
                stat.wins+=1
                stat.save()
        else:
            if not Stat.objects.filter(player=user).exists():
                stat=Stat(player=user,matches=1,points=0,wins=0)
                stat.save()
            else:
                stat=Stat.objects.get(player=user)
                stat.matches+=1
                stat.save()
        return JsonResponse({
                'win':int(win),
                'answer':correct_answer})

@csrf_exempt
def quick_req_hint(request):
    hint_type = request.POST.get('hint')
```

```
    gid=request.POST.get('gid')
    game=Game.objects.get(gameid=gid)
    contents={"value":""}
    if hint_type=="genre":
        contents["value"]=game.movie.genre[:-13]
    elif hint_type=="year":
        contents["value"]=game.movie.year
    elif hint_type=="cast":
        contents["value"]=game.movie.cast
    return JsonResponse(contents)

@csrf_exempt
def quick_check_answer(request):
    submitted_answer=request.POST.get('value')
    gid=request.POST.get('gid')
    game=Game.objects.get(gameid=gid)
    correct_answer=game.movie.title
    win=submitted_answer==correct_answer
    print(submitted_answer,correct_answer,win)
    return JsonResponse({
            'win':int(win),
            'answer':correct_answer})
```

urls.py

```
from django.urls import path
from .views import *
```

```
urlpatterns = [
    path('login/', login_view, name='login'),
    path('logout/', logout_view, name='logout'),
    path('profile/',profile,name='profile'),
    path('play/',play,name='play'),
    path('', home, name='home'),
    path('hint/',req_hint,name='req_hint'),
    path('check/',check_answer,name='check_answer'),
    path('quick-play/',quick_play,name='quick-play'),
    path('quick-hint/',quick_req_hint,name='quick-hint'),
    path('quick-check/',quick_check_answer,name='quick-check'),
    path('new_movie/<int:qno>/<int:score>',
        new_movie,name='new-movie'),
]
```

modals.py

```
from django.db import models
from django.contrib.auth.models import AbstractUser

class User(AbstractUser):
    # add any additional fields or methods here
    pass

class Movie(models.Model):
    title=models.CharField(max_length=255)
    year=models.IntegerField()
    genre=models.CharField(max_length=511)
```

```
    cast=models.CharField(max_length=511)
    description=models.CharField(max_length=10000)
    p1=models.URLField(max_length=255)
    p2=models.URLField(max_length=255)
    p3=models.URLField(max_length=255)
    class Meta:
        db_table="Movie"

class Game(models.Model):
    gameid=models.IntegerField()
    player=models.ForeignKey(
            User,
            on_delete=models.CASCADE,
            null=True)
    movie=models.ForeignKey(Movie, on_delete=models.CASCADE)
    points=models.IntegerField()
    class Meta:
        db_table="Game"

class Stat(models.Model):
    player=models.ForeignKey(User, on_delete=models.CASCADE)
    matches=models.IntegerField()
    points=models.IntegerField()
    wins=models.IntegerField()
    class Meta:
        db_table="Stat"
```

forms.py

```
from django.contrib.auth.forms import UserCreationForm,
AuthenticationForm, UsernameField
from .models import User
from django import forms

class RegistrationForm(UserCreationForm):
    username = UsernameField(widget=forms.TextInput(
        attrs={'id': 'r-username','label':'Username'}
        ))
    email = forms.EmailField(widget=forms.EmailInput(
        attrs={'id': 'r-email','label':'Email'}
        ))
    password1 = forms.CharField(widget=forms.PasswordInput(
        attrs={'id': 'r-password','label':'Password'}
        ))
    password2 = forms.CharField(widget=forms.PasswordInput(
        attrs={'id': 'r-cpassword','label':'Confirm Password'}
        ))
    def __init__(self, *args, **kwargs):
        super(RegistrationForm, self).__init__(*args, **kwargs)
        self.fields['username'].label = "Username"
        self.fields['email'].label = "Email"
        self.fields['password1'].label = "Password"
        self.fields['password2'].label = "Confirm Password"

    class Meta:
```

```
        model = User
        fields = ['username', 'email', 'password1', 'password2']

class LoginForm(AuthenticationForm):
    def __init__(self, *args, **kwargs):
        super(LoginForm, self).__init__(*args, **kwargs)

    username = UsernameField(widget=forms.TextInput(
        attrs={'id': 'username'}
        ))
    password = forms.CharField(widget=forms.PasswordInput(
        attrs={'id': 'password'}
        ))
```

settings.py

```
"""
Django settings for guessit project.

Generated by 'django-admin startproject' using Django 4.1.7.

For more information on this file, see
https://docs.djangoproject.com/en/4.1/topics/settings/

For the full list of settings and their values, see
https://docs.djangoproject.com/en/4.1/ref/settings/
"""
```

```
from pathlib import Path
import os

# Build paths inside the project like this: BASE_DIR / 'subdir'.
BASE_DIR = Path(__file__).resolve().parent.parent


# Quick-start development settings - unsuitable for production
# See
https://docs.djangoproject.com/en/4.1/howto/deployment/checklist/

# SECURITY WARNING: don't run with debug turned on in production!
DEBUG = True

ALLOWED_HOSTS = ['127.0.0.1','guessit-fyp.herokuapp.com']


# Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'accounts',
```

```
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'whitenoise.middleware.WhiteNoiseMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'guessit.urls'

TEMPLATES = [
    {
    'BACKEND': 'django.template.backends.django.DjangoTemplates',
    'DIRS': [],
    'APP_DIRS': True,
    'OPTIONS': {
        'context_processors': [
            'django.template.context_processors.debug',
            'django.template.context_processors.request',
            'django.contrib.auth.context_processors.auth',
            'django.contrib.messages.context_processors.messages',
            ],
```

```
        },
    },
]

WSGI_APPLICATION = 'guessit.wsgi.application'


# Database
# https://docs.djangoproject.com/en/4.1/ref/settings/#databases

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}


# Password validation
# https://docs.djangoproject.com/en/4.1/
ref/settings/#auth-password-validators

AUTH_PASSWORD_VALIDATORS = [
    {
    'NAME': 'django.contrib.auth.password_validation.
    UserAttributeSimilarityValidator',
    },
```

```
    {
    'NAME': 'django.contrib.auth.password_validation.
␣␣␣␣MinimumLengthValidator',
    },
    {
    'NAME': 'django.contrib.auth.password_validation.
␣␣␣␣CommonPasswordValidator',
    },
    {
    'NAME': 'django.contrib.auth.password_validation.
␣␣␣␣NumericPasswordValidator',
    },
]


# Internationalization
# https://docs.djangoproject.com/en/4.1/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_TZ = True
```

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/4.1/howto/static-files/

STATIC_URL = 'static/'
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')



# Default primary key field type
# https://docs.djangoproject.com/en/4.1/
ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'

AUTH_USER_MODEL = "accounts.User"
```

A.2.2 Results

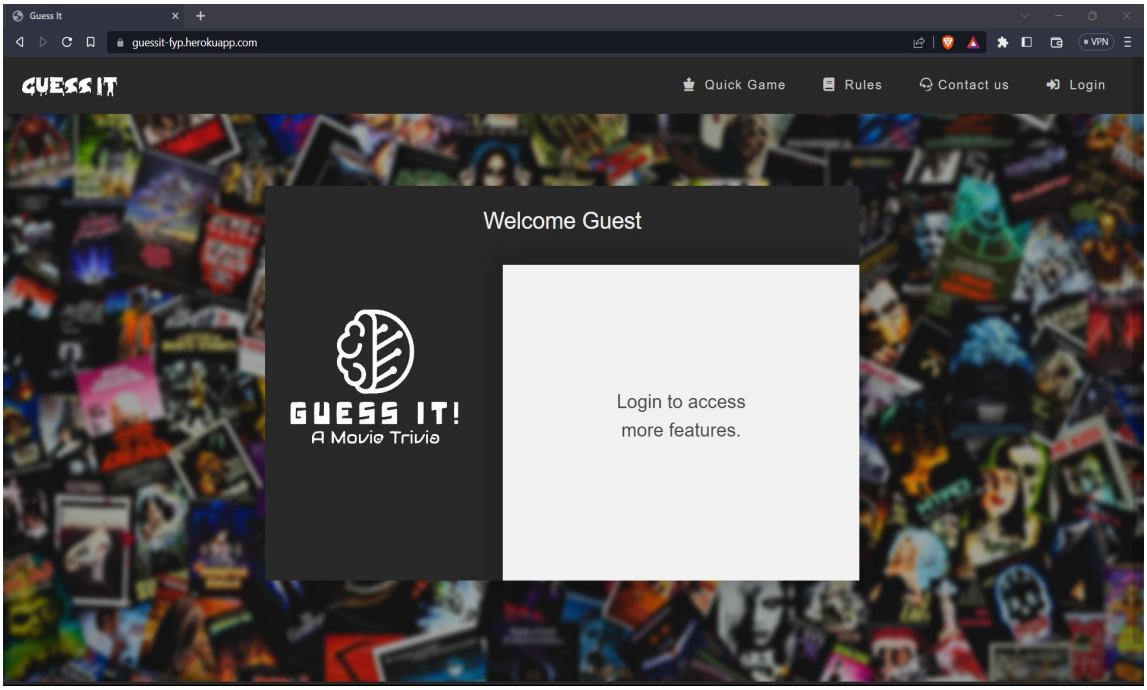

Figure A.4: Guest Home Page

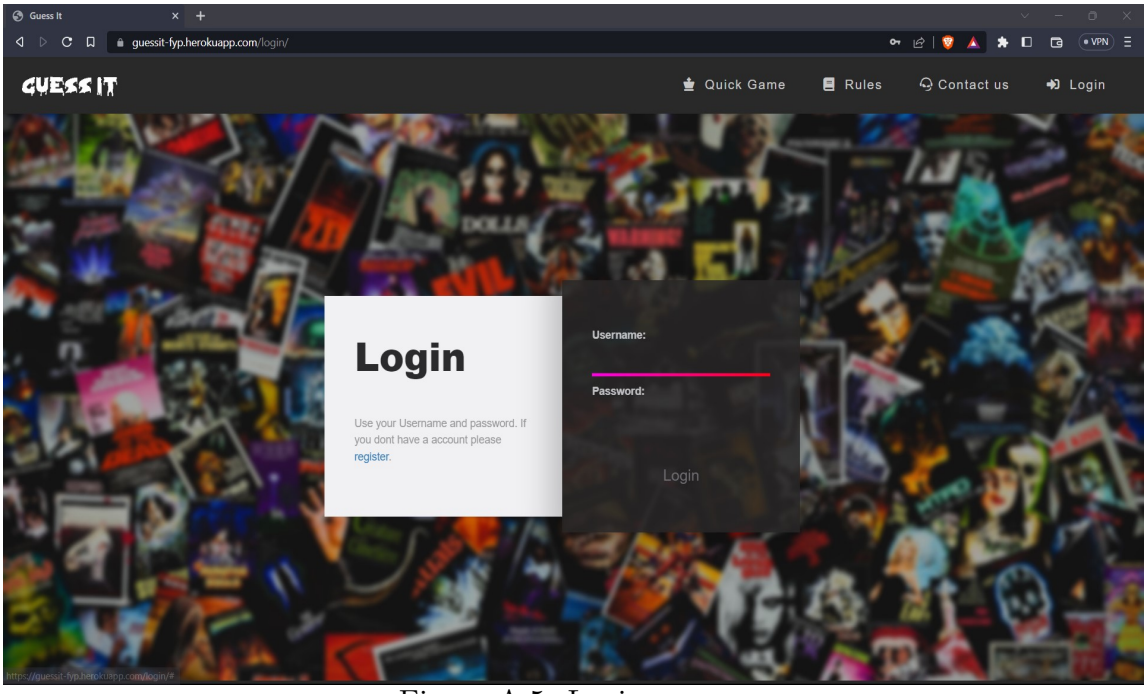

Figure A.5: Login

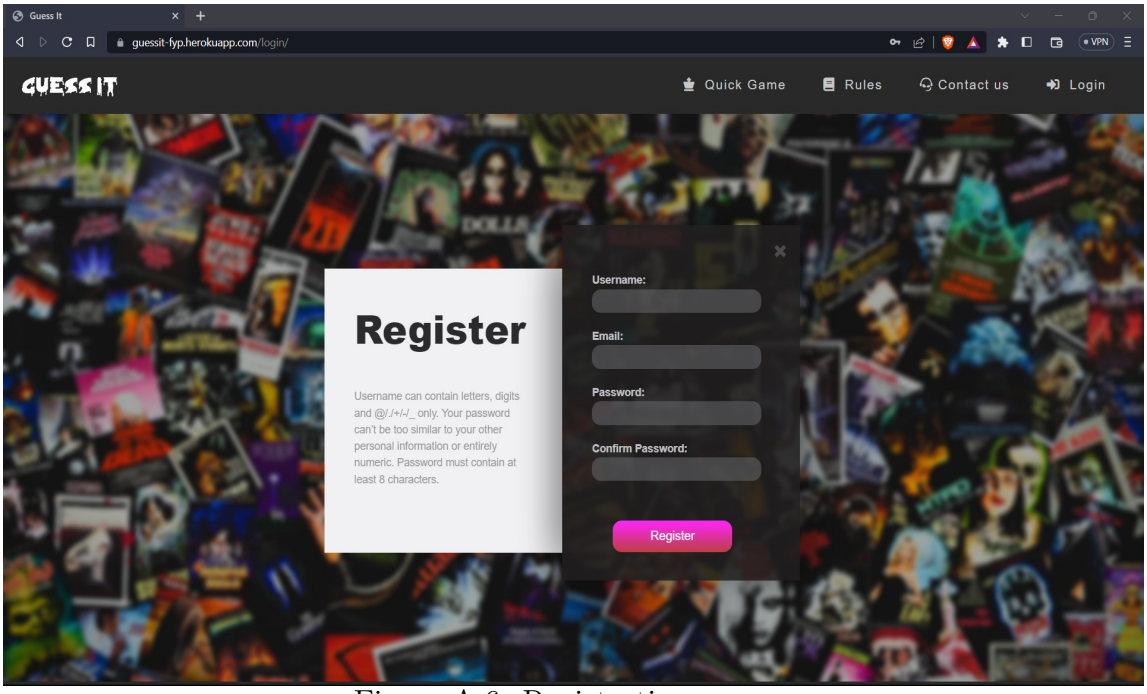

Figure A.6: Registration

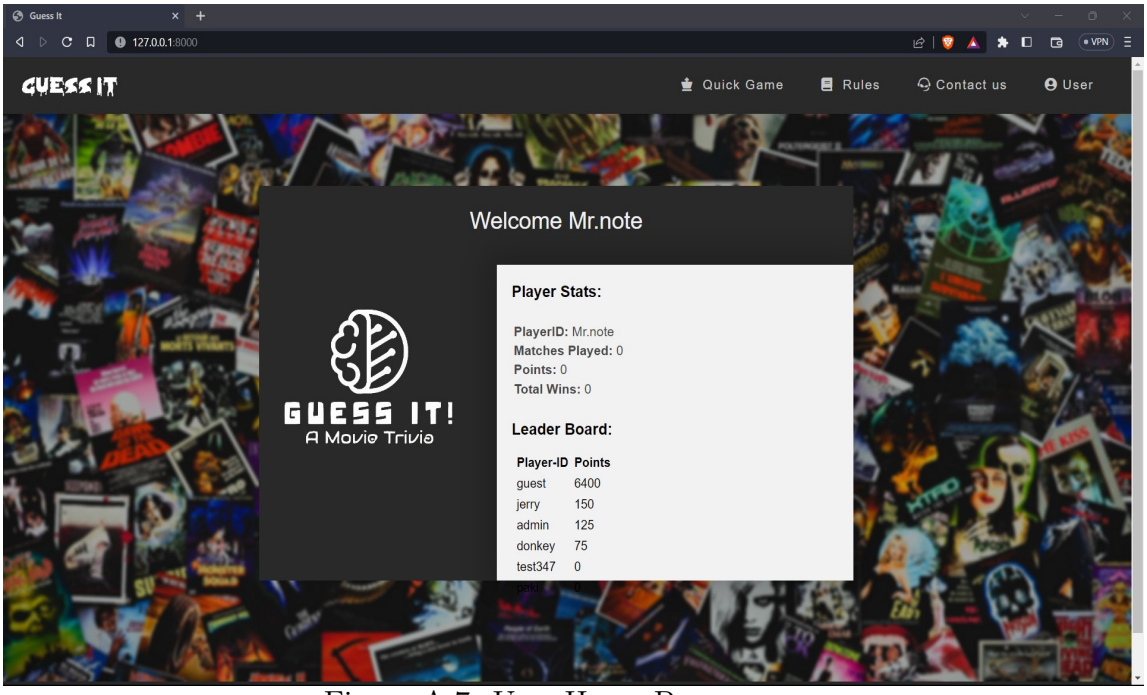

Figure A.7: User Home Page

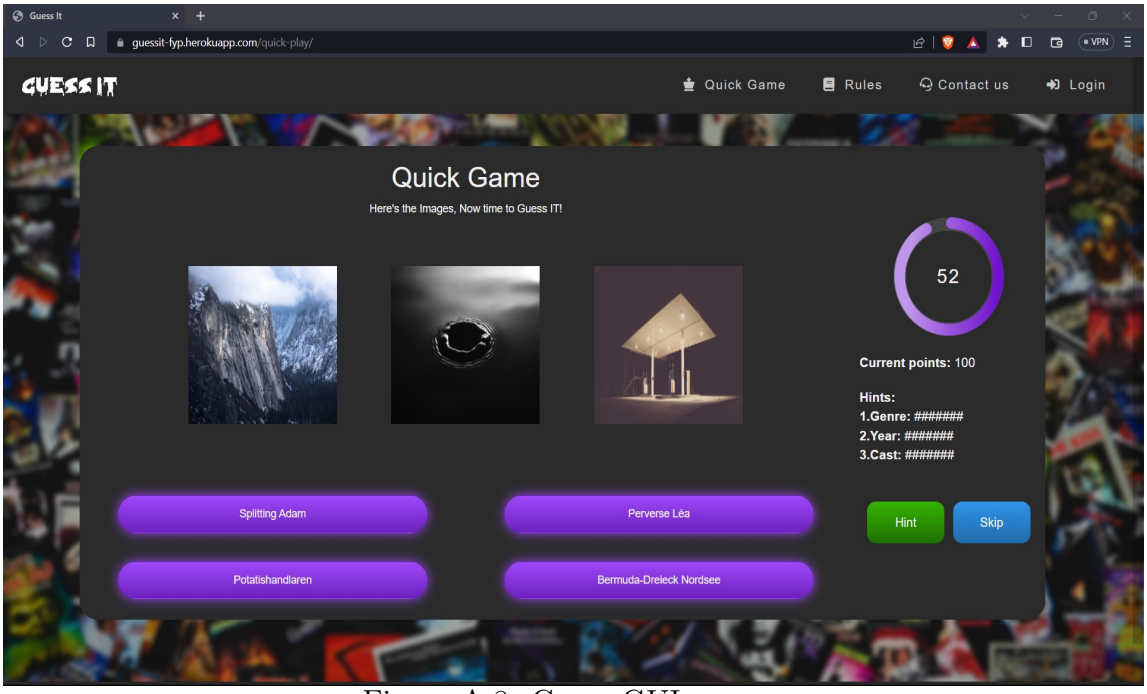

Figure A.8: Game GUI

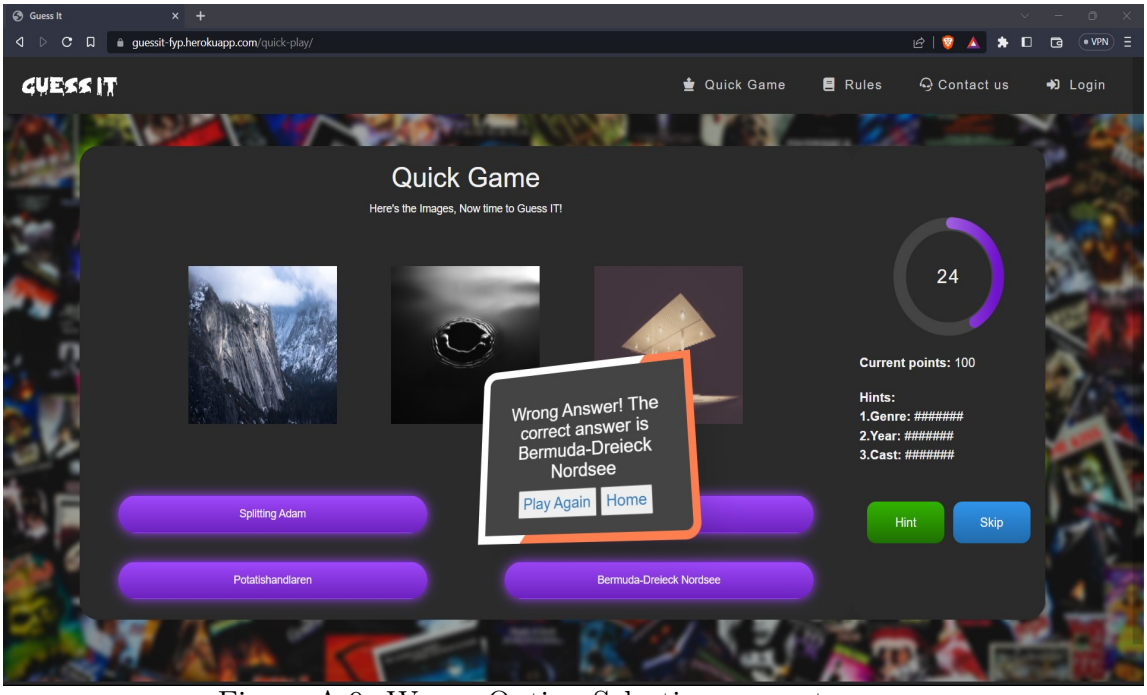

Figure A.9: Wrong Option Selection prompt

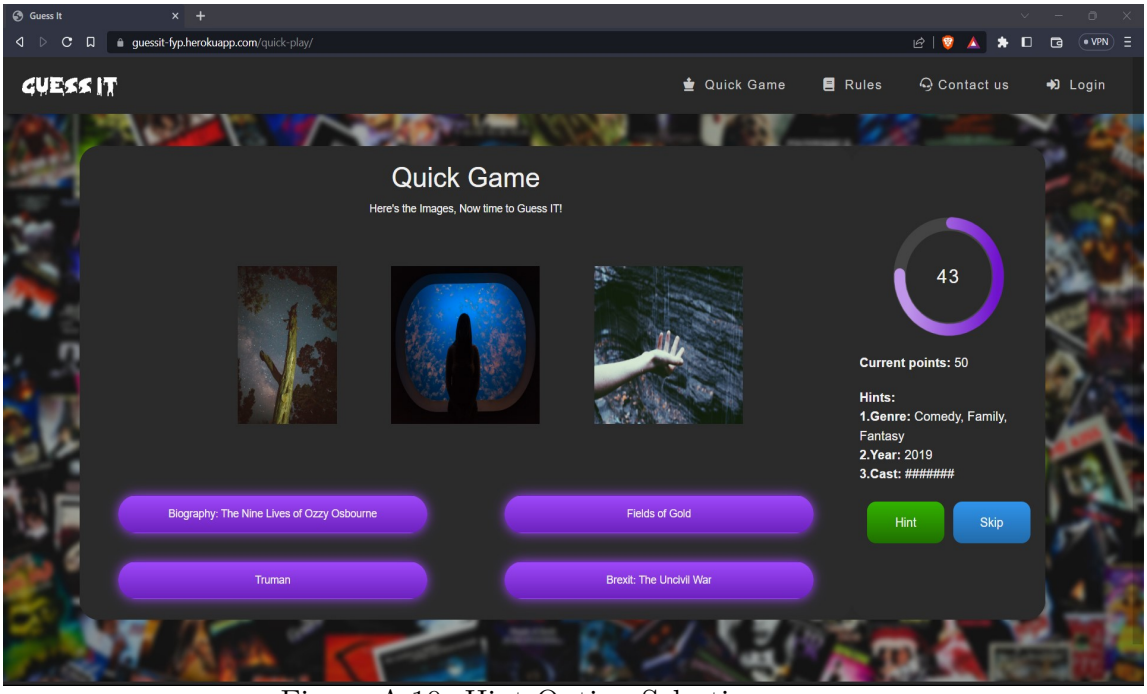

Figure A.10: Hint Option Selection

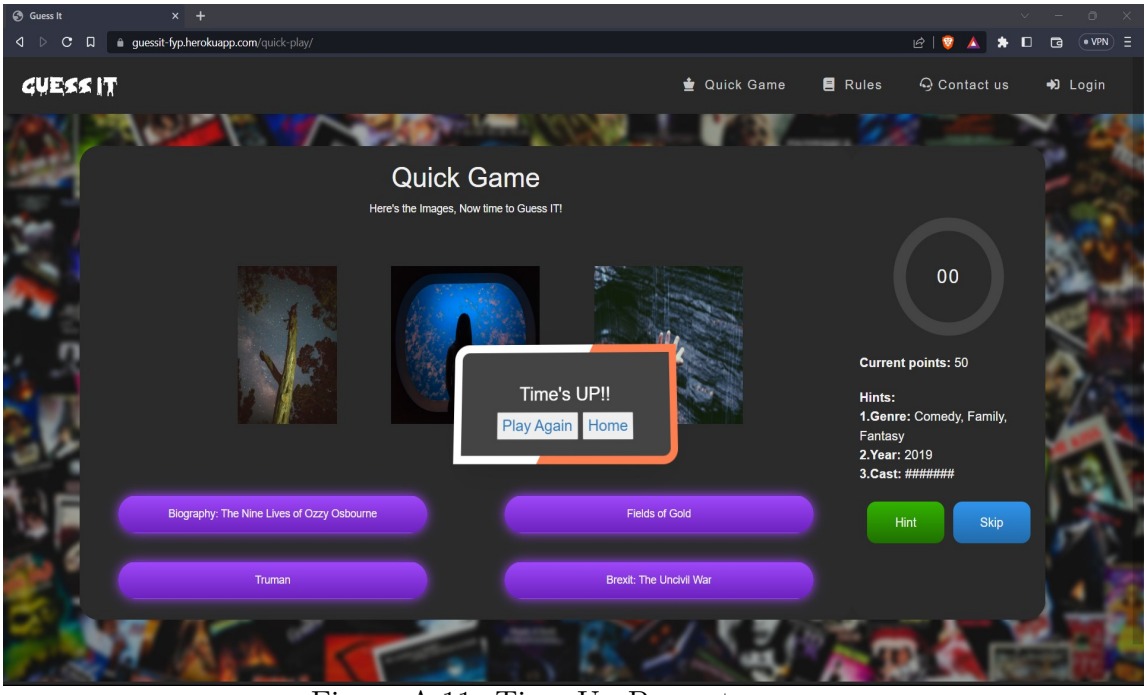

Figure A.11: Time Up Prompt

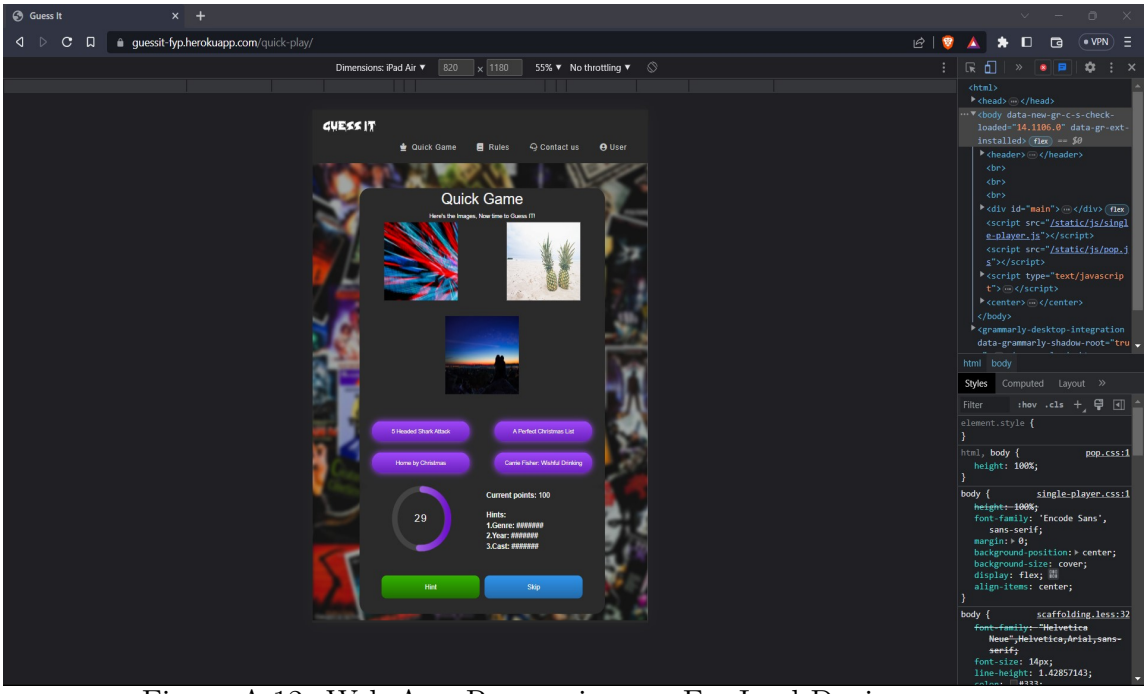

Figure A.12: Web App Responsiveness For Ipad Device

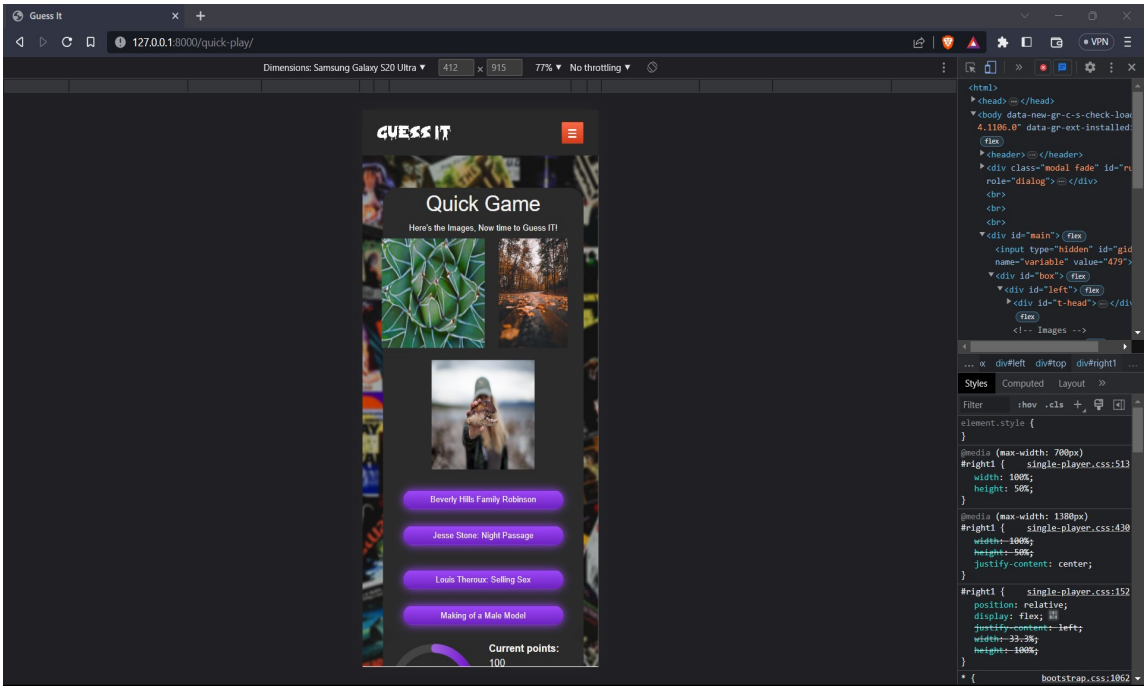

Figure A.13: Web App Responsiveness For Mobile Device