ANYANI: ANIME RECOMMENDATION SYSTEM

GROUP 11
Deng Hewen A0285707U
Luo Haiming A0285695H
Zhao Xiang A0285703B
Huang Yiqi A0285711A
Chen zhiwei A0285952N

A THESIS SUBMITTED FOR THE DEGREE OF MTECH OF INTELLIGENT SYSTEMS

INSTITUTE OF SYSTEM SCIENCE
NATIONAL UNIVERSITY OF SINGAPORE

2023

Abstract

The dynamic evolution of the animation industry has fostered a vast array of genres, styles, and techniques, catering to a diverse audience. In this expansive landscape, discovering content that aligns with individual preferences has become a chanllenging task. This report delves into the design and implementation of "AnyAni", an animation recommendation system commissioned by the Institute of Systems Science. Grounded in advanced algorithms and user-centric design philosophy, AnyAni aims to bridge the gap between the vast world of animations and the unique tastes of viewers. By analyzing user preferences, AnyAni offers tailored animation suggestions, enabling users to navigate this vast realm with ease. This system not only enhances user experience but also promotes a more inclusive representation of animations, accounting for various genres, styles, and cultural narratives. The following pages detail the objectives, methodologies, and outcomes of this project, underscoring the transformative potential of intelligent recommendation systems in the entertainment industry.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In the digital age, the fusion of technology and the internet has transformed the way we interact with media entertainment. Among this evolution, the realm of anime has gained a significant rise in global popularity. Characterized by its distinctive art style, in-depth storytelling, and meticulous production quality, anime has transitioned from a specialized hobby to becoming a cultural phenomenon across the world. By Precedence Research, the compound annual growth rate (CAGR) IS 7.9% from 2021 to 2030 [Mar22], shows a robust and sustained increase in the anime industry's global market value, highlighting its growing influence and appeal to audiences worldwide.

However, the growth of the anime industry brings its own challenges. Every season releases a set of anime spinning various genres, leading to an overwhelming array of choices for viewers. As a result, anime fans often find themselves spending more time navigating through critiques and recommendations from online communities rather than immersing themselves in the series. Moreover, once a series concludes, the unavoidable question arises: "What do I watch next?"

Given the amount of available content, specialized and evolving preferences of audiences, there is a pressing need for a system that can streamline this decision-making process and enhance the viewing experience.

1.2 Major Streamline Platforms and Problem description

In this digital era, the rise of streamline platforms serves the growing demand of anime enthusiasts. Giants such as Netflix, YouTube and Bilibili have recognized the widespread appeal of anime and have subsequently expanded their libraries to offer a diverse range of content, spanning various genres and styles.

While these platforms provide recommendations, they often lean towards showcasing popular series or titles that have garnered significant attention. This typically results in suggestions derived from generic top lists or broad recommendations that don't necessarily specialize

for the unique preferences of individual viewers. Many fans still find themselves relying on external forums, blogs, or anime communities to seek recommendations or reviews.

A notable gap in the current streaming ecosystem is the absence of a tailored recommendation system. The one-size-fits-all approach often overlooks the nuanced tastes and preferences of the vast and varied anime audience. There's a demanding need for platforms to harness advanced algorithms and user data to provide personalized anime recommendations, enhancing user satisfaction and engagement.

1.3 Market Research

To further investigate the potential commercial value of this customized anime recommendation system, a comprehensive survey and corresponding analysis for each question has been conducted as below.

- Demographic Analysis:

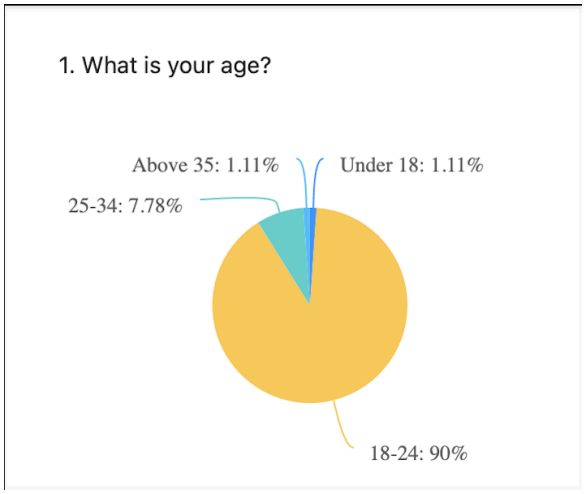

Figure 1.1: "What is your age?"

  - Age Group: Most respondents (90%) are in the age range of 18-24, a demographic known for its high online presence. They represent a key audience for streaming platforms and are typically more receptive to personalization features.

- Streaming Habits:

  - Frequency of Streaming: An overwhelming 87.78% of respondents use streaming platforms daily. This high frequency underscores the potential impact of any feature enhancements, including personalized recommendations, on user experience.

  - Anime Watching Patterns: 41.11% of respondents watch anime sometimes, while 31.11% regularly watch anime. This highlights that a significant portion of this

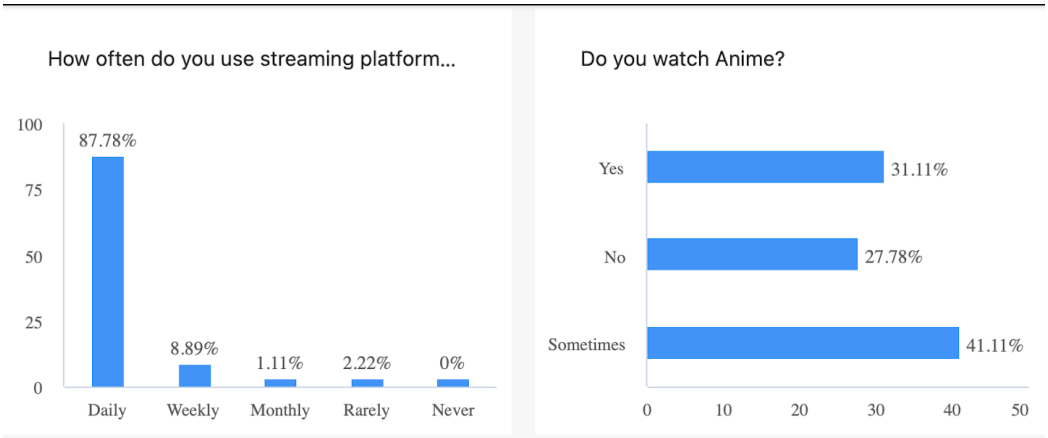

Figure 1.2: "How Often do you use streaming platforms?" "Do you watch anime?"

audience is actively engaging with anime content.

- Content Discovery:

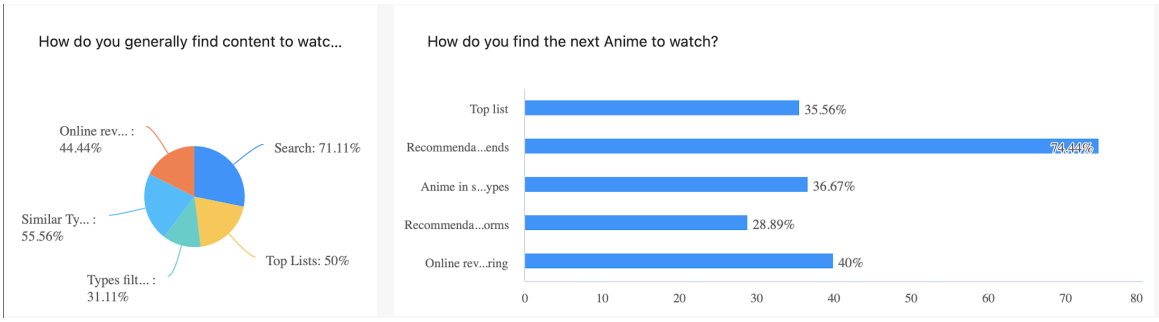

Figure 1.3: "Whar are the ways to find the content to wathc next?"

  - General Content Discovery: The majority (71.11%) rely on search functions to discover content, followed by online reviews (44.44%) and similar types of content (56.56%). These methods, while effective, can be time-consuming.
  - Recommendations from friends (74.44%) seem to be the primary method of finding new anime to watch, followed by online reviewing (40%). This indicates a reliance on broad-based suggestions rather than individualized content recommendations.

- Streaming Platforms Preference:
  - Tencent, YouTube, BiliBili and Aiqiyi are among the most preferred platforms, suggesting a diversified streaming market with potential opportunities for platforms that can offer differentiated features, like personalized recommendations.

- Satisfaction Levels:

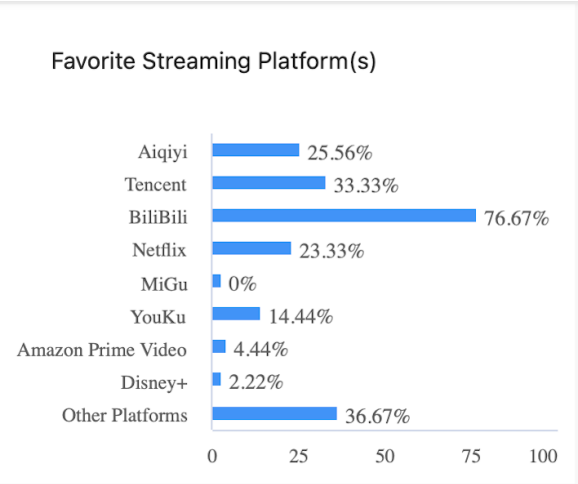

Figure 1.4: "What are some streamline platforms you usually use?"

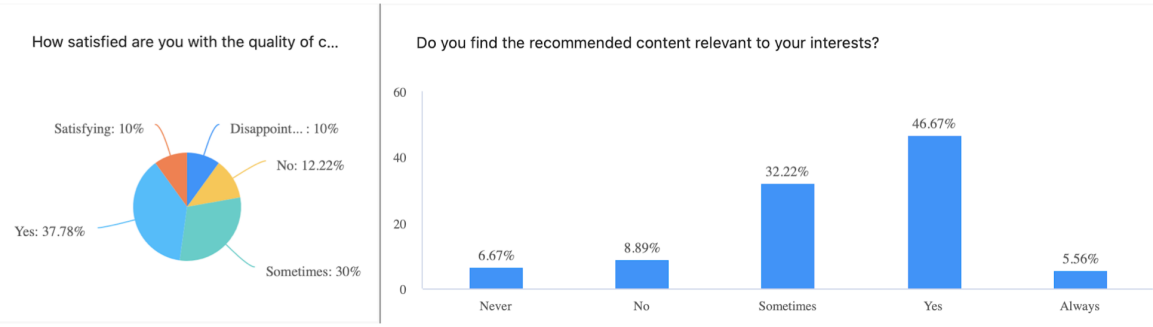

Figure 1.5: The satisfaction level of recommendation content

- Quality of Recommended Content: Only 10% of the respondents are totally satisfied with teh recommendation content, while there are a portion of respondents partially satisfied (37.78% somehow yes and 30% sometimes) with the recommendations. However, a notable 22.22% are found not that satisfied overall, pointing to potential gaps in content offerings or mismatches between viewer preferences and recommendations.
- Relevance of Recommended Content: Only 5.56% always find the recommended content relevant to their interests. A concerning 32.22% only find it relevant "sometimes", and 6.67% do not find it relevant at all. This clearly underlines the inadequacy of current recommendation systems and the demand for more personalized recommendations.

- Feedback:
  - The world cloud derives from comments suggesting that respondents are primarily

concerned with "Recommendation", "Top lists", and "Genre". This reiterates the sentiment that while broad-based top lists are popular, there's a clear desire for more nuanced recommendations tailored to individual preferences. There are some other mentions including leakage of personal information, and deliberately eye-catching headlines.

1.4 Commercial Value

It is vital to explore and define the commercial opportunities of this project from a more practical perspective. This section will describe three potential aspects of the business value of AnyAni.

- Sponsorships & Partnerships: Aligning with production studios offers the potential for exclusive anime releases, enhancing the platform's appeal. Additionally, collaborations with merchandise brands can bring exclusive offers to our users, thus boosting engagement and retention rates.
- Subscription Models: Introducing a premium tier provides multiple benefits. Not only can users enjoy an ad-free browsing experience, but they also gain access to exclusive contents. For example, the content our team designed for ISY5002 project, Aniverse, where the user can generate their own images to different anime styles. Moreover, premium subscribers benefit from our advanced recommendation algorithms, ensuring a truly tailored anime discovery experience.
- Community Integration: Future enhancements to our platform include the creation of a thriving community where users can discuss, share, and review content. This community hub will become an attractive spot for targeted advertising, especially for brands looking to tap into the anime culture and fandom. The interactive nature of a community also increases platform stickiness, ensuring users spend more time and engage more frequently.

The commercial avenues outlined above not only ensure sustainability but also promise growth, setting the stage for a richer, more personalized anime viewing journey.

1.5 Aim and Objectives

Aim:

Our primary aim is to revolutionize the process of anime content discovery, filling the prevalent market gaps. By introducing "AnyAni", our state-of-the-art Anime Recommendation System, we aspire to provide a thrilling and personalized viewer experience for anime enthusiasts around the world.

Objectives:

To achieve our ultimate goal, the following specific objectives have been determined.

1. User Interface Development: Design and implement a user-friendly interface that provides easy navigation, smooth browsing, and a modern aesthetic appeal to engage users effectively.
2. Database Construction: Build a comprehensive database, ensuring it encompasses a vast range of anime titles, genres, and metadata to support diverse user preferences.
3. Recommendation System Enhancement: Continually refine the recommendation algorithms, leveraging machine learning and user feedback, to deliver accurate and tailored suggestions to the users.
4. User Profile Customization: Allow users the flexibility to customize their profiles, adjusting preferences, viewing history, and other parameters, to further enhance the recommendation accuracy.
5. Backend Integration: Ensure seamless integration between frontend and backend systems, optimizing performance, scalability, and ensuring data security and integrity.

Chapter 2

Data

2.1 Data Source

The primary dataset employed in this project is sourced from the "Anime Database for Recommendation System" available on Kaggle [Man20]. This dataset encompasses 16,737 distinct anime entries, each comprehensively detailed as outlined in the subsequent table:

Features	**Description**
Title	Refers to the name of the anime
Genre	Highlights the primary genre or themes of the anime
Synopsis	Provides a brief description or summary of the anime's storyline
Type	Categorizes the anime as either a 'Movie' or 'TV' series
Producer	The individual(s) or company responsible for funding the anime's production
Studio	The animation studio that produced the anime
Rating	Gives an aggregate rating of the anime, reflecting its quality and reception
ScoredBy	Represents the total number of users who have rated the given anime
Popularity	Assigns a rank to the anime based on its popularity
Members	Indicates the number of users who have added the given anime to their watchlist or favorites
Episodes	Specifies the number of episodes that the anime has, if it's a series

Table 2.1: Description of Anime Features

Features like 'Genre' and 'Synopsis', are vital for content-based recommendations, providing a deeper insight into each anime, thereby refining the precision of our recommendations.

However, it's worth noting that the dataset isn't without its challenges. Some inconsistencies arise due to the presence of missing values and duplicate records. Such irregularities, though addressed to the best of our capabilities, underline the importance of ongoing refinement and quality checks to maintain the reliability of our recommendation system.

2.2 Data preprocessing

After acquiring our dataset, several preprocessing has been carried through. Here is a step-by-step breakdown of the measures taken to ensure the data was primed for optimal functioning:

- Text Cleaning Function (text_cleaning): This function is designed to purify the textual data. Upon receiving a text string, it identifies and eliminates unwanted characters and patterns. The result is a sanitized version of the original text, free from unnecessary distractions and inconsistencies that might hinder the recommendation process.
- Weighted Rating Function (weighted_rating): The essence of a recommendation system lies in how it weighs and rates content. With this function, we aim to provide a balanced rating score for each anime. By considering the number of votes an anime has received (ScoredBy) alongside its actual rating (Rating), a more accurate viewer sentiment is derived. This balanced rating ensures that popular animes with a substantial number of votes don't overshadow underrated gems with fewer votes but high quality.
- Data Cleaning Function (clean_data): Standardizing our dataset is imperative to ensure seamless processing. The clean_data function plays a pivotal role in this endeavor. It takes in data—whether a list or a string—and transforms it: converting characters to lowercase and eliminating spaces. These steps ensure that every piece of data adheres to a consistent format, laying the groundwork for the subsequent stages of the recommendation process.
- Create Soup Function (create_soup): An integral part of our recommendation strategy is to consider multiple attributes of an anime when making suggestions. The create_soup function is a testament to this approach. It amalgamates key features of an anime—like its Genre, Type, Producer, Studio, and Synopsis—into a singular string called 'soup'. This consolidated data structure enables a comprehensive evaluation of each anime, facilitating richer and more relevant recommendations. free from unnecessary distractions and inconsistencies that might hinder the recommendation process.

Prepared with these functionalities, the preprocessing stage is meticulously designed to enhance the recommendation system's accuracy and efficiency, ensuring users get suggestions closely aligned with their preferences.

2.3 Database Structure

The foundation of our recommendation system is built upon a robust database architecture, which is structured to store and retrieve vital data efficiently. The database is split into three primary tables, each serving a distinct purpose:

1. ‘cleaned_anime_data‘ Table: This table is home to the preprocessed anime dataset, which has undergone a series of conversion and enhancements as part of the data preprocessing phase. An integral feature added to this table is the ‘Soup’, a comprehensive representation of an anime’s attributes from create_soup function mentioned in section 2.2.

2. ‘user_data‘ Table: User information is the vital part of our recommendation system. The ‘user_data‘ table ensures that this experience remains seamless and personalized. It is tasked with storing user-centric information, including:

   - ‘account_id‘: A unique identifier for every user, ensuring that each user’s data remains distinct and retrievable.
   - ‘user_name‘: The chosen name or alias of the user, facilitating a more personal interaction with the system.
   - ‘password‘: Encrypted for security, this ensures that the user’s account remains protected from unauthorized access.

3. ‘ratings‘ Table:

   The ‘ratings‘ table is designed to capture and store the scoring feedback of the user from the frontend. It stores user ratings in a structured format, providing insights into user preferences and viewing patterns. The table includes:

   - ‘rating_id‘: A unique identifier for each rating entry, ensuring the traceability of every piece of feedback.
   - ‘account_id‘: This links back to the ‘user_data‘ table, identifying which user provided a particular rating.
   - ‘scores‘: Capturing the essence of user feedback, this denotes the rating a user has accorded to an anime.
   - ‘anime_id‘: Tying the feedback to a specific anime, this ensures that the scores can be utilized to refine recommendations for that particular anime.

The commercial avenues outlined above not only ensure sustainability but also promise growth, setting the stage for a richer, more personalized anime viewing journey.

The provided Entity-Relationship (ER) diagram showcases the intricate connections and structure of the database designed the three tables. The ‘user_data‘ table, with primary key (PK) of account_id, retains user particulars unique. The ‘cleaned_anime_data‘ table, encapsulating a number of anime specifics, adopts Anime_id as its PK. The ‘ratings‘ table link the relation between user information and anime data together. While it’s distinctly identified by its PK, rate_id, it also incorporates two foreign keys (FK) – account_id

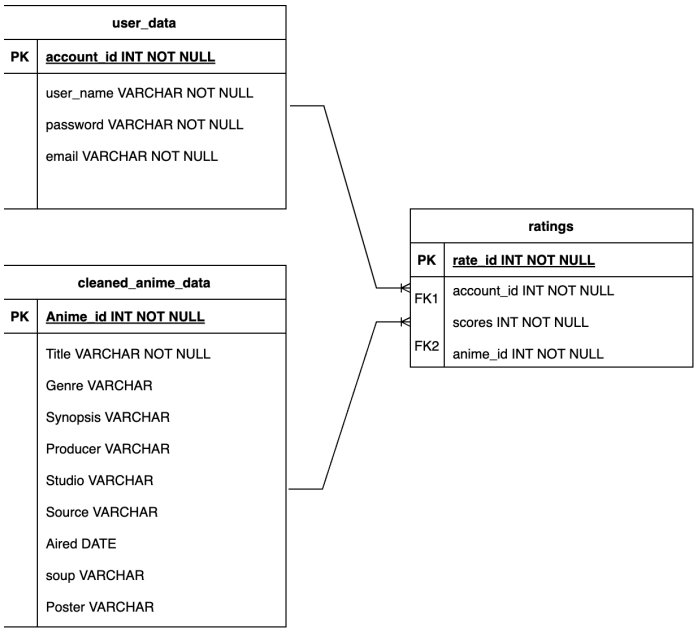

Figure 2.1: ER diagram of database construction of AnyAni system

and anime_id, establishing connections to the user_data and cleaned_anime_data tables respectively.

Together, the tables and their relationships form a comprehensive database structure, ensuring efficient storage, retrieval, and processing of both user and anime data.

Chapter 3

Methodology and System structure

3.1 Common recommending methods

Different recommendation systems implement specialized algorithms. Driven by different commercial objectives and data availability, filters to apply to a particular users's specific situation can be selected. It facilitates marketers to maximize conversions and average order value. There are two common recommending methods, content-based and collaborative recommendations [LJM+19].

3.1.1 Content-based Recommendation System:

Content-based recommendation is a popular technique in the domain of recommendation systems. It's used to suggest items by comparing the content of the items and a user profile, with content being described in terms of several descriptors that are inherent to the item. cosine similarity [HKP12]:

$$\text{sim}(u, i) = \frac{u \cdot i}{\|u\| \times \|i\|} \tag{3.1}$$

How It Works:

- Item Profile: First, to define the characteristics or features of each item. For instance, in a movie recommendation system, features might include genre, director, main actor, etc.
- User Profile: For each user, a profile is created based on the features of items the user has interacted with (e.g., watched, liked, or purchased movies).
- Recommendation: To recommend new items to the user, the system compares the user's profile with the profiles of the items. The more the user's profile aligns with an item's profile, the higher the chance that the item will be recommended.

3.1.2 Collaborative filtering Recommendation System:

Collaborative filtering is another widely used technique in recommendation systems. Instead of focusing on the content (features) of items, collaborative filtering focuses on the interactions or relationships between users and items.

How It Works:

- User-Item Interactions: The primary input is the matrix of interactions between users and items. This matrix might be explicit or implicit.
- Finding Similarities: Collaborative filtering looks for users who have similar preferences or items that have similar interaction patterns.
- Recommendation: For a given user, items liked or preferred by similar users are recommended.

There are mainly two types of collaborative filtering:

- User-based Collaborative Filtering: Finds similar users based on their item interactions and suggests items that similar users have liked but the target user hasn't interacted with yet. User similarity (based on Pearson correlation coefficient) [SK09]:

$$\text{sim}(u,v) = \frac{\sum_{i \in I}(r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I}(r_{ui} - \bar{r}_u)^2} \times \sqrt{\sum_{i \in I}(r_{vi} - \bar{r}_v)^2}} \tag{3.2}$$

  , where r_{ui} and r_{vi} are the ratings given by users u and v to item i respectively. $\bar{r}_u$ and $\bar{r}_v$ are the average ratings of users u and v respectively.

  Prediction using User-Based Collaborative Filtering:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v \in U} \text{sim}(u,v)(r_{vi} - \bar{r}_v)}{\sum_{v \in U} |\text{sim}(u,v)|} \tag{3.3}$$

  where, $\hat{r}_{ui}$ is the predicted rating of user u for item i.

- Item-based Collaborative Filtering: Finds similar items based on how users interacted with them and recommends the most similar items to those that the user liked. Item Similarity (based on Cosine Similarity):

$$\text{sim}(i,j) = \frac{\sum_{u \in U} r_{ui} r_{uj}}{\sqrt{\sum_{u \in U} r_{ui}^2} \sqrt{\sum_{u \in U} r_{uj}^2}} \tag{3.4}$$

  where, r_{ui} and r_{uj} are ratings for items i and j by user u respectively.

- Prediction using Item-Based Collaborative Filtering:

$$\hat{r}_{ui} = \frac{\sum_{j \in I} \text{sim}(i,j) r_{uj}}{\sum_{j \in I} |\text{sim}(i,j)|} \tag{3.5}$$

  where, I is the set of items rated by user u.

Implementation: Popular algorithms and methods for collaborative filtering include:

- Matrix Factorization: Techniques like Singular Value Decomposition (SVD) where the user-item interactions matrix is factorized into multiple matrices representing latent factors.
- K-Nearest Neighbors (K-NN): For user-based or item-based similarity computations.
- Deep Learning: Methods like autoencoders or recurrent neural networks to capture complex patterns in user-item interactions.
- Hybrid Methods: Combining collaborative filtering with other techniques like content-based filtering.

3.2 System architecture

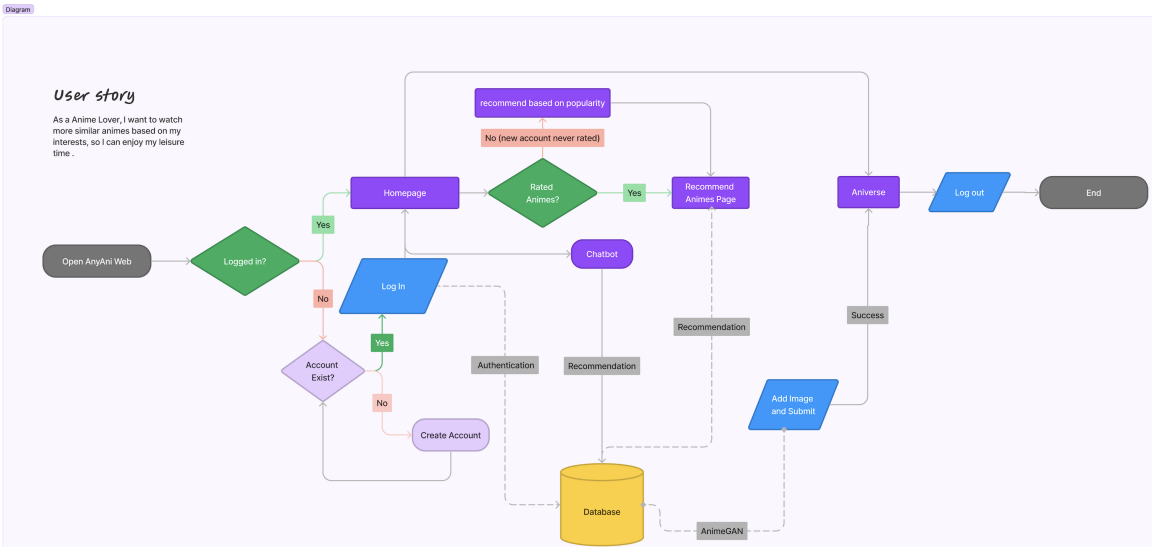

Figure 3.1: System Architecture flow chart

At the heart of the AnyAni Web Platform lies a thoughtful designed recommendation system anchored on robust data collection and advanced modeling techniques.

Upon the user's engagement with the AnyAni web interface, the system intelligently detects their authentication status. New users are smoothly transitioned into the authentication flow where they can either log in or sign up, depending on their existing association with the

platform.

As users explore the platform, their actions, from rating animes to selecting favorite genres, are meticulously recorded. This interaction data becomes a pivotal input for the recommendation engine. For newcomers or those yet to rate any anime, the system offers suggestions based on trending popularity.

The recommendation engine then processes this user input, combining it with its extensive database and leveraging state-of-the-art algorithms to generate tailored anime suggestions. These recommendations are stored in a MySQL database.

Additionally, the platform features an interactive chatbot, designed to further enhance user engagement by assisting them throughout their journey. In the "Aniverse" section, users can transform their real images into anime-style avatars based on three distinct styles. This unique feature allows users to see themselves in an anime format, adding a touch of personalization and fun to their journey on the platform.

Finally, the web front end fetches these personalized recommendations from the MySQL database, presenting users with a list of anime suggestions tailored to their tastes and preferences.

This architecture, as shown in Fig 3.1, ensures a smooth and immersive experience for every user, from the moment they access the platform to their eventual exploration of personalized anime recommendations.

3.3 System Design and features

3.3.1 UI

The UI designed in our AnyAni System provide a intuitive and user-friendly experience, embedded with various functionalities.

- Login:

  Upon launching our AnyAni system, users are prompted to either log in (if they're an existing user) using their username and password, or to register as a new user with a unique username and email address.

- Home Page:

  The home page showcases a series of anime from our dataset. Beneath each anime poster, there's a heart icon, allowing users to rate the anime from the range 0-10, based on their preferences.

- Scored Anime:

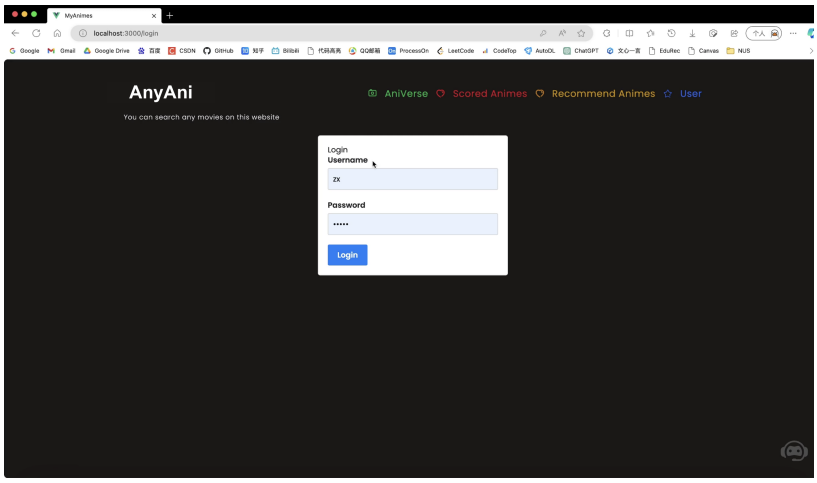

Figure 3.2: Login Page of the system

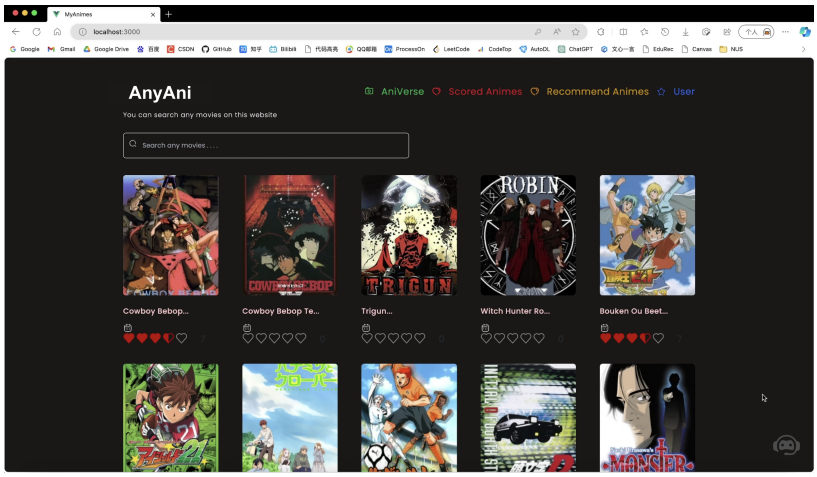

Figure 3.3: Main Page of the system

This section displays the anime rated by the current user. If a user chooses to remove their rating from an anime, that particular anime will no longer appear on this page.

- Recommend Anime:

  Here, users are presented with the top 10 anime recommendations, curated by our recommendation algorithms based on their prior ratings. Additionally, users have the option to delve deeper in to detailed page for each anime as shown in Fig.xx. In this page, users can view the aired period, anime genre and synopsis of the anime, allowing a more detailed description for users.

- Aniverse:

  As an additional feature, Aniverse allows users to transform their images to mimic the

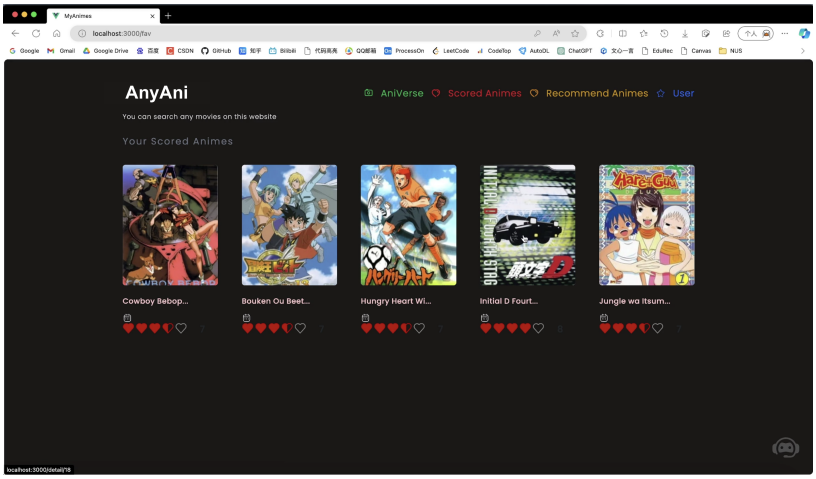

Figure 3.4: Page of scored Anime

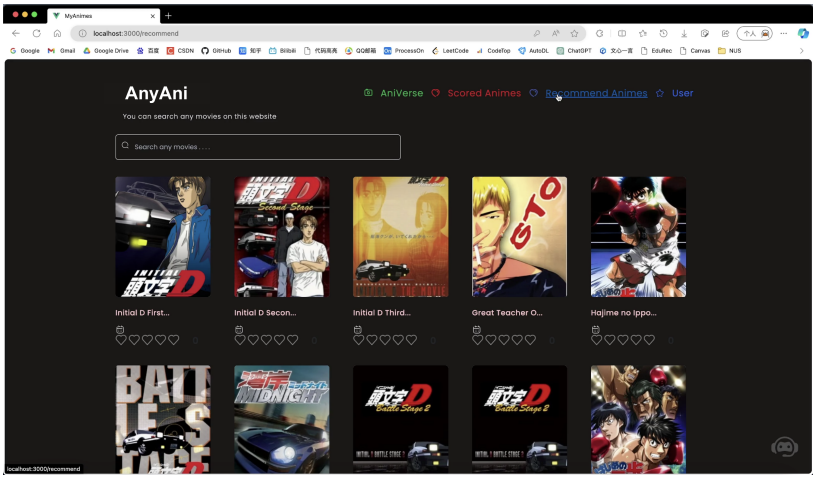

Figure 3.5: Page of top 10 recommend Anime

styles of three renowned anime artists, Paprika, Hayao and Shinkai styles.

More detailed and technical explanation of those embedded functions will be included in the following sections.

3.3.2 Chatbot

Overview

The chatbot is designed as a web-based chatbot using Flask, which recommends animes based on the provided inputs and performs natural language processing (NLP) tasks using PaddleNLP. The recommendation is driven by both KMeans and content-based methods.

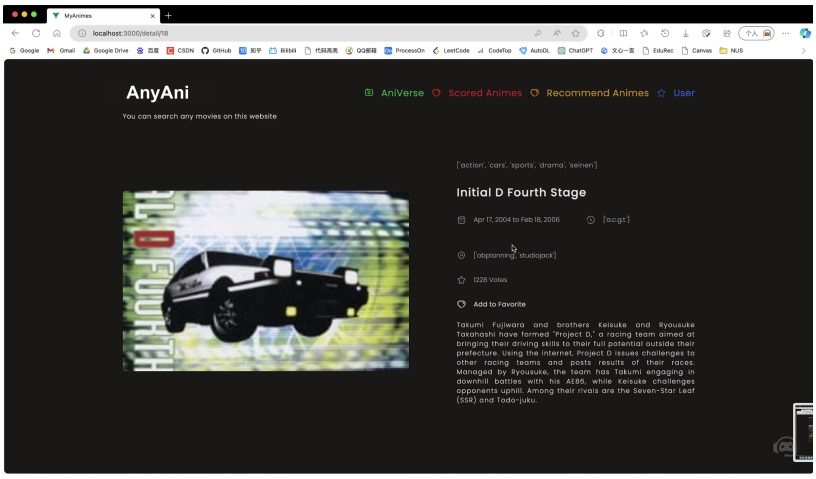

Figure 3.6: Detail page of an anime with all the information

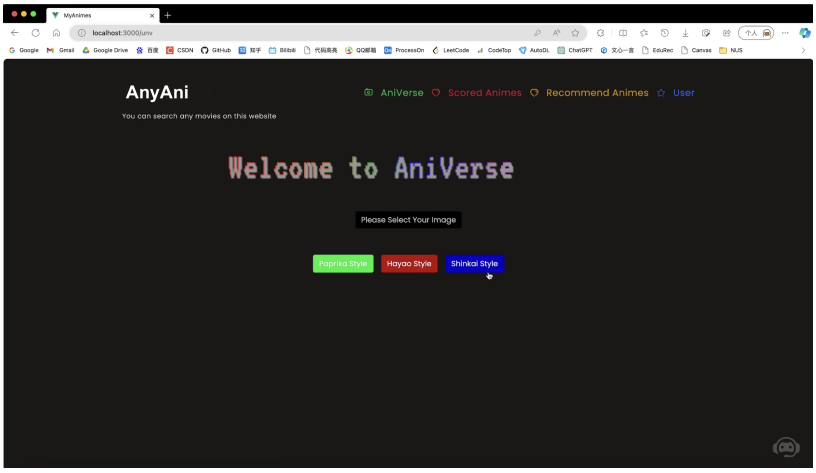

Figure 3.7: Aniverse Page

Features

- Named Entity Recognition (NER): This bot employs two NER functions. While the former captures the movie title, the latter recognizes opinions and sentiments like "comedy" and "action". Both use the information_extraction from the PaddleNLP to get the correct word and bring it back to the backend.
- Anime Recommendation: The system provides anime recommendations in two ways: By Soup and Genre: The bot can offer recommendations based on genres such as comedy, using the soup_classfication function, using the KMeans clustering.
- Scored Anime: This section displays the anime rated by the current user. If a user chooses to remove their rating from an anime, that particular anime will no longer

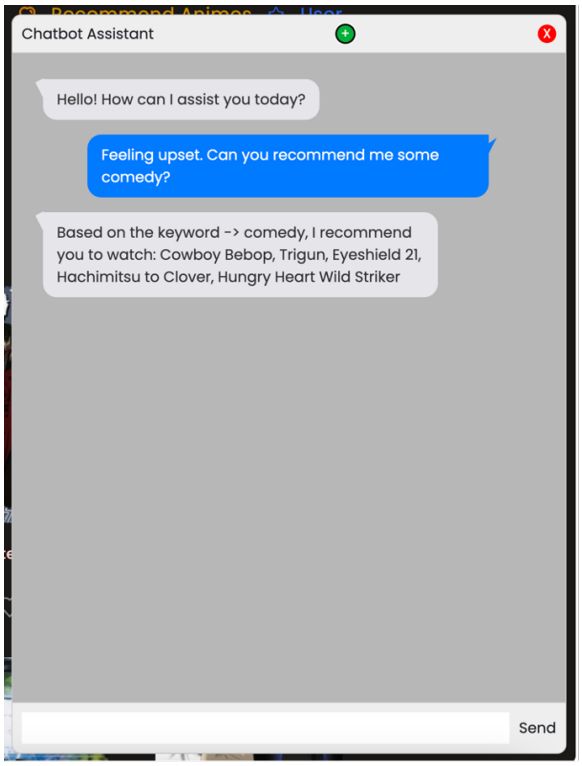

Figure 3.8: An example conversation with the chatbot assistant. When the user is upset and wantst to watch some comedy type anime, the chatbot will suggest a list of comedy anime.

appear on this page.

- CORS Handling: To handle cross-origin requests, the system responds to CORS preflight requests, allowing it to be accessed from various origins, specifically by web-based interfaces. Error Handling: The system can identify invalid HTTP request methods and respond with an appropriate error message.

Backend Technologies

- Flask: A lightweight web framework for Python.
- PaddleNLP: Used for NLP tasks, particularly the Taskflow for information extraction.
- Pandas: Employs the data manipulation and analysis capabilities of pandas, especially while handling anime recommendations.
- scikit-learn: Utilizes its CountVectorizer and cosine_similarity functions, presumably in the recommendation logic within the AnimesRecommendation module.

```
b'{"userMessage":"Please recommend me some comedy"}'
Cluster 0:
 ova source tv comedy school
Cluster 1:
 music video song nhk uta
Cluster 2:
 movie dementia film comedy kids
Cluster 3:
 special dvd comedy episode included
Cluster 4:
 tv ona action fantasy comedy
```

Figure 3.9: Keyword Clustering from User Messages Highlighting Comedy Preferences

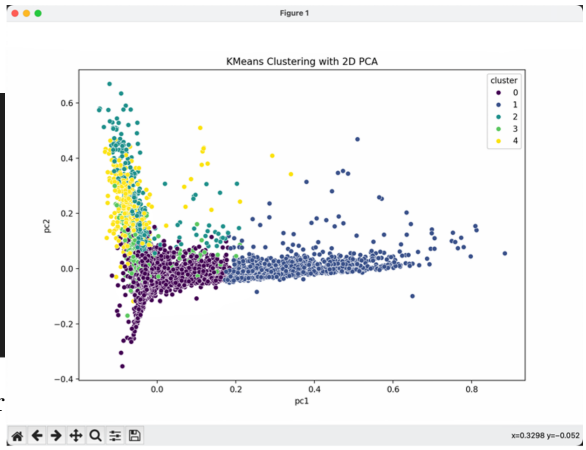

Figure 3.10: Visualization of Data Points in Multi-dimensional Clustering Analysis

System Flow

1. The user interacts with the chatbot through a web interface.
2. The frontend sends the user's message to the backend via a POST request.

```
await axios.post(`${"http://127.0.0.1:8282"}/chatbot`)
```

Listing 3.1: Chatbot POST endpoint

3. If the user's message contains a recognizable entity, such as a anime title, the backend will process it.

```
ie_en = Taskflow('information_extraction', schema=schema, model='uie-
    base-en')
```

Listing 3.2: anime title recognizability

4. If the user's message contains a recognizable entity, such as a genre, the backend will process it.

```
list = soup_classfication.find_movies_by_keyword(genre)
```

Listing 3.3: anime genre recognizability

5. The backend, based on the input and the underlying logic, fetches appropriate anime recommendations.
6. The bot then sends back a message, recommending anime based on the recognized entity or genre.
7. The frontend displays the bot's reply to the user.

3.3.3 Scoring

Overview

The scoring feature acts as an interactive bridge between the system and its users, ensuring that users not only passively consume content but actively provide feedback. It offers a simple, visual, and intuitive method for users to rate anime, thus enhancing the user experience. This feedback mechanism, apart from its direct benefit, serves as a critical input for the recommendation algorithm, allowing it to refine suggestions based on user preferences.

Features

- **Heart Icon**:
  - Beneath each anime title, there's a line-up of five heart icons. This visualization gives users the flexibility to rate an anime on a scale from 0 to 10.
  - The heart icons are split between full, half, and empty, offering a visually pleasant way for users to understand and decide on their ratings.
  - An immediate representation of the score is visible next to the heart icons, making the process transparent.
- **Database Interaction**:
  - As users assign or modify scores, the system ensures that these ratings are immediately updated in the `ratings` table of the database.
  - This ensures that the data remains current and that any subsequent operations (like recommendations) are made based on the most recent data.
- **Scored Anime Page**:
  - A dedicated page called "Scored Anime" is available for users to revisit. This page lists out all the anime they have rated.
  - By offering this recapitulation, the system not only helps users remember their past interactions but also provides them with a snapshot of their anime preferences.

Backend Techniques

- **Flask with flask_mysqldb**: This lightweight web framework integrates seamlessly with MySQL through the flask_mysqldb extension. The extension provides a direct connection to the MySQL database, allowing the backend to fetch, update, or delete scores as users rate anime.
- **MySQLdb.cursors**: Serving as the cursor class for MySQL database connections, MySQLdb.cursors enables the backend to execute raw SQL queries directly. This

direct execution capability ensures precision and control over database operations, particularly when handling rating-related actions in the ratings table.

System Flow

1. **User Interaction**: The user interacts with the heart icons beneath an anime title to indicate their rating.

```
const toggleHeart = (index) => {
if (ratings.value[index] === 0) {
  // Make all previous hearts full
  for (let i = 0; i < index; i++) {
    ratings.value[i] = 1;
  }
  ratings.value[index] = 0.5; // Make the current heart half-filled
} else if (ratings.value[index] === 0.5) {
  // Make all previous hearts and the current one full
  for (let i = 0; i <= index; i++) {
    ratings.value[i] = 1;
  }
} else if (ratings.value[index] === 1) {
  // Make the current heart and all subsequent hearts empty
  for (let i = index; i < ratings.value.length; i++) {
    ratings.value[i] = 0;
  }
}


```

Listing 3.4: Toggle Heart Interaction

2. **Frontend Processing**:

   - The frontend assesses the total score based on user interactions, translating it into a range from 0 to 10. The scoring process is executed by the `currentTotalScore` function:

```
   const currentTotalScore = ratings.value.reduce(
  (sum, rating) => sum + (rating) * 2, 0);
   if (currentTotalScore === 0) {
     uploadScore(0);
     // If the score is zero, remove the anime from the
  scoredAnime store
     scoredAnimeStore.removeScoredAnime(props.Anime_id);
   }
 };


```

Listing 3.5: Score Calculation

This computed score instantly appears next to the heart icons and is readied for back-end transmission.

3. **Backend Communication**: Through an Axios API call, the frontend conveys the computed score to the backend, typically adopting a POST request directed to the `/rating/upload_ratings` endpoint:

```
1    axios.post(`${"http://127.0.0.1:8282"}/rating/upload_ratings`,
     upload_score)
2
```

Listing 3.6: upload ratings endpoint

4. **Database Update**: The backend processes this data, validates it, and then updates the `ratings` table with the new score, using the same endpoint as above.

5. **Scored Anime Page Update**: To keep the user experience coherent and up-to-date, the Scored Anime page undergoes an update, showcasing the latest or modified anime ratings. This synchronization is achieved by the following call:

```
1    await axios.get(`${"http://127.0.0.1:8282"}/rating
2    /nonzero_rating/${sessionStorage.getItem("accountID")}`);
3
```

Listing 3.7: stored score fetching

This flow ensures that the user's interactions are captured in real-time, processed efficiently, and stored securely, making the scoring feature robust and user-friendly.

3.3.4 Aniverse

Overview

Aniverse is a leading platform in the field of turning images into anime. Driven by quality, Aniverse has developed a GAN-based model that not only matches models like CycleGAN [ZPIE17] and CartoonGAN [CLL18]but aims to outdo them in speed and result quality.

Features and System Flow

- **Artist-specific Anime Transformations:**

  Aniverse excels in turning real images into specific anime styles. It's designed for both fans of certain anime styles and those wanting to capture a unique artistic look.

- **Selfie-to-Anime Avatars:**

  Aniverse offers a unique feature that lets users turn their selfies into anime-style pictures, providing a fun way to see oneself in the anime world.

Model design

The generator structure that has been crafted exhibits a balanced encoder-decoder configuration. To enhance efficiency while maintaining performance, certain standard convolution layers have been replaced with depthwise separable convolutions. Furthermore, to make the model more streamlined and to accelerate its processing speed, standard residual blocks have been substituted with inverted residual blocks. Recognizing the potential degradation

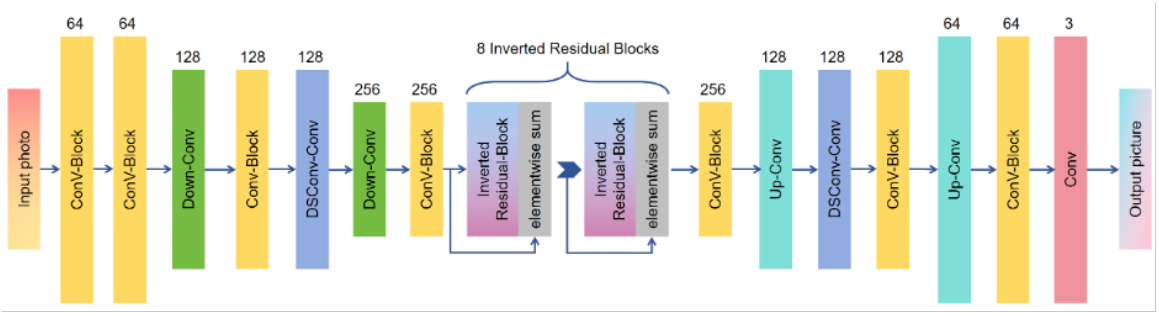

Figure 3.11: Generator structure of CartoonGan

of image quality through pooling in image generation tasks, the design employs Down Convolution, taking cues from established GAN models, rather than resorting to maximum pooling. The primary focus has been on optimizing the generator's performance, and as such,

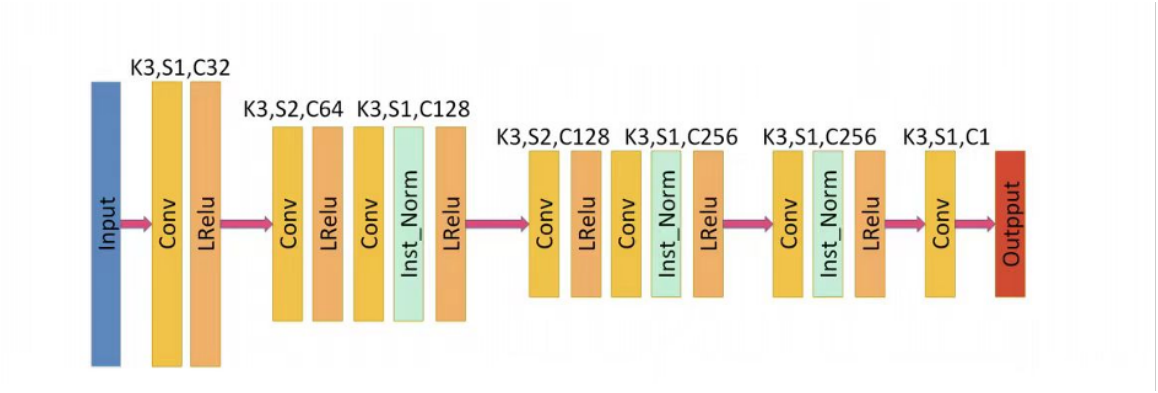

Figure 3.12: Discriminator structure of CartoonGan

the discriminator from CartoonGAN remains unchanged at this stage. However, revisions to its design will be considered as needed in subsequent phases.

Loss function design Upon reviewing relevant literature, the following observations have been made: The defining aspect of the anime style hinges on a particular texture, primarily informed by grayscale data. Consequently, When the loss function is designed, the Gram Matrix is incorporated to glean the texture details of an image. Additionally, two specific loss functions, namely grayscale adversarial loss and grayscale style loss, are introduced. Together, they are designed to ensure that the resultant image aptly mirrors the nuances of the desired anime style. Furthermore, a color loss mechanism is employed to guarantee the preservation of the image's inherent details, with a particular emphasis on maintaining color

and brightness consistency.

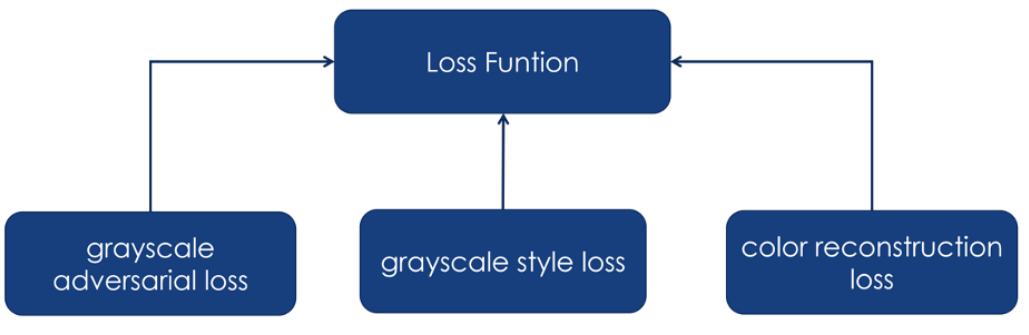

Figure 3.13: Component of loss function including gray-scale adversarial loss, gray-scale style loss and color reconstruction loss.

Training&Testing The suggested Aniverse can be trained from start to finish using unmatched training data. Given the GAN model's complexity, starting with random values can lead to less-than-ideal results. Based on research, starting the training of the generator can speed up the GAN's learning process. So, the generator G begins its training using just the content loss Lcon(G, D). This initial training lasts for one cycle with a learning speed of 0.0001. When fully training Aniverse, the learning speeds for the generator and the part that distinguishes between real and fake images are 0.0001 and 0.0002. Aniverse is trained for 10 cycles, using batch-size=4. The Adam method is used to reduce the total loss. The training is done on A100 server with Tensorflow and CUDA.

When the trained model is deployed on a laptop (RTX3060 graphics card, i7-12700H) in the form of ONNX, the following results are obtained):

Aniverse, with 12,253,152 network parameters, boasts a model size of 64.38M. It operates at 108.98B FLOPs and has an efficient inference time of 61 ms/image.

3.3.5 Recommendation

Overview

An intuitive way of checking whether a recommendation system for anime works from user end is simply click the 'Recommended Anime' button and watch what it happens. And the 'recommend' feature was totally developed following this instinct.

The task was solved by concertrating on one piece at a time: 1) poster displaying; 2) backend recommend python code; 3) frontend-backend communications; 4) detail pages and further more.

Features and System Flow

- **Poster displaying:**

The origin dataset comes without a poster image or link. A python scipt with some web crawl skills was used to fetch posters from given link ‘field‘ in the dataset.

```
def get_url_by_link(anime_item):
    if anime_item['Poster'] != 'https://github.githubassets.com/images/modules/logos_page/GitHub-Mark.png': # default poster link
        return
    try:
        # Send an HTTP GET request to the provided URL
        with requests.Session() as session:
            response = session.get(anime_item['Link'], headers=HEADERS, verify=False)
        # Check if the request was successful (status code 200)
        if response.status_code == 200:
            # Parse the HTML content
            soup = BeautifulSoup(response.text, 'html.parser')
            # Find the image URL using BeautifulSoup
            # (assuming it's in the meta tags)
            img_url = soup.find('meta', property='og:image')['content']
            anime_item['Poster'] = img_url

    except Exception as e:
        print("[E]", str(e))
        return
```

Listing 3.8: Function to fetch image URL given an anime URL

The fetched image url would be written back to database to reduce response time for next time, when the same anime data is requested.

- **Backend recommend code:**

  There are two kinds of recommendation algorithms that are generally used: collaborative filtering alogrithm and content-base recommendation algorithm. The project only adopted the first one, as the recommendation algorithm are but one of the core components of a recommendation system, most rest of which still require time to code, such as general frontend code and ‘score‘ feature and so on.

  We use a rating matrix to store the anime and its corresponding scores. We use nearest neighbors algorithms and the cosine metric to train our recommendation system. The code we used to train a recommendation system is as followed:

```
def train_recommendation_system():
    # Load anime and rating data
    rating = pd.read_csv(os.path.join(prefix_data, 'rating.csv'), encoding='latin')
    anime = pd.read_csv(os.path.join(prefix_data, 'cleaned_anime_data.csv'), encoding='latin')

    # Filter anime and users based on rating counts
```

```
    anime_rating = rating.groupby(by='anime_id').count()
    anime_rating = anime_rating['rating'].reset_index().rename(columns
    ={'rating': 'rating_count'})
    final_anime = anime_rating[anime_rating['rating_count'] > 50]

    user_rating = rating.groupby(by='user_id').count()
    user_rating = user_rating['rating'].reset_index().rename(columns={'
    rating': 'rating_count'})
    final_user = user_rating[user_rating['rating_count'] > 80]

    print(rating.columns)
    print(final_anime.columns)

    final_anime_dt = rating[rating['anime_id'].isin(final_anime['
    anime_id'])]
    final_dt = final_anime_dt[final_anime_dt['user_id'].isin(final_user
    ['user_id'])]

    # Create a rating matrix
    rating_matrix = final_dt.pivot_table(index='anime_id', columns='
    user_id', values='rating').fillna(0)
    csr_rating_matrix = csr_matrix(rating_matrix.values)

    # Train collaborative filtering recommender
    recommender = NearestNeighbors(metric='cosine')
    recommender.fit(csr_rating_matrix)

    # Save rating matrix and recommender to files
    with open(os.path.join(prefix_data, 'rating_matrix.pkl'), 'wb') as
    rating_matrix_file:
        pickle.dump(rating_matrix, rating_matrix_file)
    with open(os.path.join(prefix_data, 'recommender.pkl'), 'wb') as
    recommender_file:
        pickle.dump(recommender, recommender_file)
```

Listing 3.9: Training the recommendation system

In the predict process, we first extract the anime title from database, and then locate the anime in the rating matrix, finally use k neighbors to find the nearest neighbors, namely the "recommendation" results. Code we used as followed:

```
def get_recommendation(anime_title):
    try:
        # Load anime data
        anime = pd.read_csv(os.path.join(prefix_data, '
    cleaned_anime_data.csv'), encoding='latin')

        # Load rating matrix and recommender
        with open(os.path.join(prefix_data, 'rating_matrix.pkl'), 'rb')
     as rating_matrix_file:
```

```
            rating_matrix = pickle.load(rating_matrix_file)

        with open(os.path.join(prefix_data, 'recommender.pkl'), 'rb') as recommender_file:
            recommender = pickle.load(recommender_file)

        # Find the index of the user's chosen anime
        user_anime = anime[anime['Title'] == anime_title]
        user_anime_index = np.where(rating_matrix.index == int(user_anime['Anime_id']))[0][0]

        # Get recommendations based on the chosen anime
        user_anime_ratings = rating_matrix.iloc[user_anime_index]
        user_anime_ratings_reshaped = user_anime_ratings.values.reshape(1, -1)
        _, indices = recommender.kneighbors(user_anime_ratings_reshaped, n_neighbors=16)
        nearest_neighbors_indices = rating_matrix.iloc[indices[0]].index[1:]
        nearest_neighbors = pd.DataFrame({'Anime_id': nearest_neighbors_indices})
        result = pd.merge(nearest_neighbors, anime, on='Anime_id', how='left')

        return result
    except Exception as e:
        return str(e)
```

Listing 3.10: Getting anime recommendations based on a given title

- **frontend-backend communications:**

  The frontend backend code includes many functions to do.

```
@app.route('/recommend', methods=['GET', 'POST'])
def recommend_anime():
    username = request.args.get('username')
    animeId = request.args.get('animeid', -1,type=int)
    if username is None and animeId == -1:
        return anime.get_all_animes(mysql, page=1)
    elif username is not None and animeId == -1:
        return anime.get_recommend_animes(mysql, username)
    elif username is None and animeId != -1:
        return anime.get_recommend_animes_by_anime_id(mysql, animeId)
    else:
        return anime.get_all_animes(mysql, page=1)

```

Listing 3.11: Recommend Animes

In the above code, the 'get_recommend_animes()' is to get recommendation from a

username. We simply use the highest scored anime by the user and do recommnedations base on that. The ‘get_recommend_animes_by_anime_id()‘ simply find the corresponding anime title and revoke the ‘get_recommendation()‘ function. The logic for all recommendation is simple yet effective.

- **Detail pages and further more:**

  Beyond the work above, we found that it would be better if we can add a detail page for each specific page. Using the frontend code to render a detail page as followed:

```
<template>
  <main>
    <!-- <IsLoading v-if="animeStore.isLoading" /> -->
    <IsLoading v-if="combinedRef.isLoading" />
    <article class="lg:flex lg:gap-5 lg:justify-between lg:items-center
    ">
      <div class="w-full h-64 rounded-md overflow-hidden md:h-80 lg:w
    -6/12 lg:h-96">
        <img :src="animeStore.movie.Poster" class="w-full h-full object
    -cover"
        :alt="animeStore.movie.Title" />
      </div>
      <div class="my-5 lg:w-5/12 lg:mt-0">
        <p class=" text-gray-400 font-light text-xs  mt-2
        tracking-wider  md:text-sm ">
          {{ animeStore.movie.Genre }}
        </p>
        <h3 class="my-8 font-medium text-lg text-gray-200
        tracking-wider md:text-2xl">
          {{ animeStore.movie.Title }}
        </h3>
        <div class="flex justify-between items-center flex-wrap
        gap-2 md:justify-start md:gap-14">
          <div class="flex items-center text-gray-400 font-light text-
    sm">
            <CalendarIcon />
            {{ animeStore.movie.Aired }}
          </div>

          <div class="flex items-center text-gray-400 font-light text-
    sm">
            <TimeIcon />
            {{ animeStore.movie.Studio }}
          </div>

          <div class="flex items-center text-gray-400 font-light text-
    sm">
            <LocationIcon />
            {{ animeStore.movie.Producer }}
          </div>
```

```
35          </div>
36          <div class="flex items-center text-gray-400 font-light text-sm
    my-8">
37            <StarIcon />
38            {{ animeStore.movie.Popularity }} Votes
39          </div>
40          <button @click="toggleFav(animeStore.movie.Anime_id, $event)"
41              class="flex items-center mt-8 cursor-pointer"
42              :class="getClass(animeStore.movie.Anime_id)">
43            <HeartIcon />
44
45            <span class="text-sm tracking-wide">
46              {{ handleTextFav(animeStore.movie.Anime_id) }}
47            </span>
48          </button>
49          <br>
50          <p class="text-gray-300 text-justify tracking-wider
51              text-sm font-light md:tracking-widest ">
52            {{ animeStore.movie.Synopsis }}
53          </p>
54        </div>
55      </article>
56      <hr class="mt-24 mb-16 opacity-10" />
57      <div>
58        <h3 class="text-gray-300 text-sm md:text-lg">
59          You migh also like . . . . .
60        </h3>
61        <Movies :movies="films" />
62      </div>
63    </main>
64  </template>
65
```

Listing 3.12: Detail page

Also we use the detail page anime to enable an extra "you may also like" feature, so that it looks more natural, and enable user to explore more.

Chapter 4

Results

4.1 Recommendation Algorithms

4.1.1 Content-based Recommendation Algorithm

This algorithm suggests anime based on the content and attributes of the anime. For instance, if a user has previously shown interest or rated in a particular genre or theme, the system recommends other anime with similar content.

4.1.2 Collaborative Filtering Algorithm

This technique is based on user behavior. It recommends anime by identifying patterns in user ratings and preferences. For example, if User A and User B both liked a particular anime, and User A liked another anime X, the system might recommend anime X to User B.

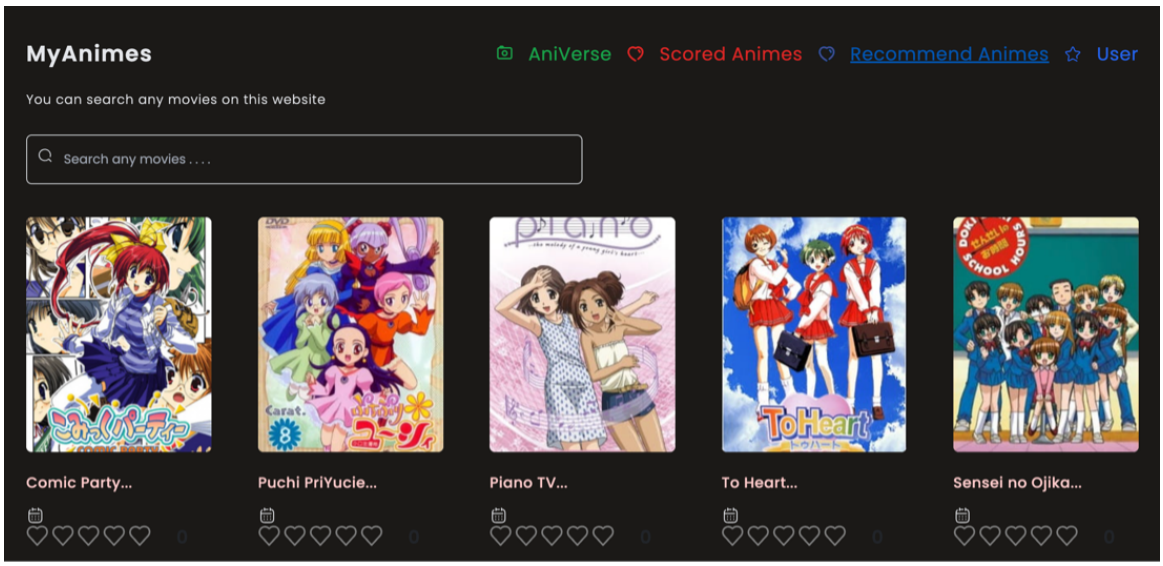

Figure 4.1: Mainpage of AnyAni

4.1.3 K-means Clustering

This unsupervised machine learning algorithm is based on similarities in soup or genre. By categorizing users into clusters based on the soup column of the animes, the system can make group-wise recommendations.

4.2 Frontend & Backend Architecture

4.2.1 Vue

The frontend of the system is built using Vue.js, a progressive JavaScript framework. It provides a responsive and interactive user interface, ensuring a seamless user experience. Within our Vue application, we use the axios library to send asynchronous HTTP requests to the Flask backend. These requests are triggered based on user interactions, such as when a user requests anime recommendations or submits a new rating.

4.2.2 Flask

The backend services are implemented using Flask, a lightweight Python web framework. It manages the recommendation algorithms, data processing, and interaction with the database. Our backend, developed using Flask, exposes various RESTful API endpoints. These endpoints are designed to respond to HTTP requests, facilitating tasks like fetching anime recommendations, submitting user ratings, and more.

4.2.3 Chatbot Integration

The system also includes a chatbot based on the PaddleNLP, which can interact with users in real-time. It can answer queries, recommend animes and provide a more interactive experience.

4.2.4 Aniverse

The inclusion of AnimeGAN allows the system to generate anime-style images. This can be utilized for various purposes, such as creating unique anime artwork, enhancing existing images, or offering a unique visual experience for the users.

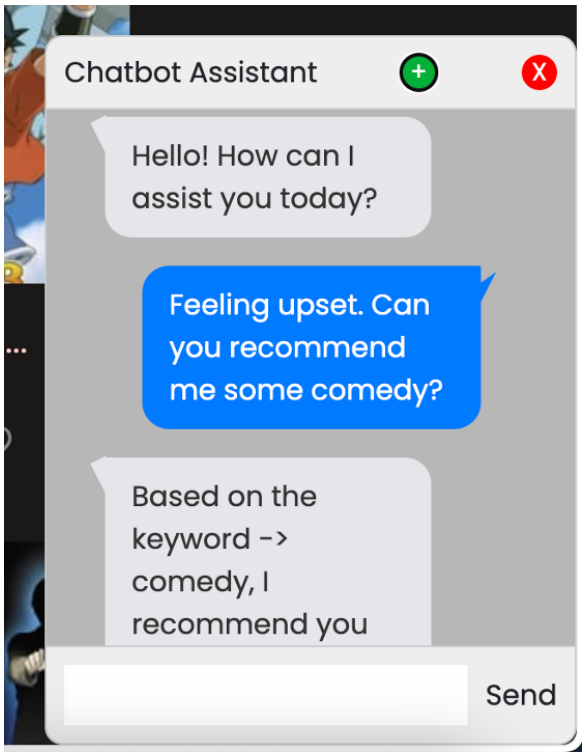

Figure 4.2: Chatbot conversation example

Figure 4.3: Generated anime style of Dwayne Douglas Johnson from Aniverse page

Chapter 5

Conclusion and Future Work

In the dawn of the digital age, the intersection of technology and the internet has revolutionized our engagement with media entertainment, with anime standing out as a testament to this transformation. From its niche origins to its current status as a global cultural sensation, anime's trajectory is indicative of its profound impact and resonance with audiences worldwide. Yet, as the industry continues to flourish, it grapples with the challenge of catering to the diverse and evolving tastes of its viewers. Major streaming platforms, despite their vast repositories, often fall short in offering personalized recommendations, leading enthusiasts to seek guidance from external sources. As we reflect on the journey of anime from a specialized hobby to a worldwide phenomenon, it becomes evident that the future lies in harnessing the power of advanced algorithms to curate a viewing experience that is as unique as the viewers themselves. Only by addressing this pressing need can we truly elevate the anime viewing experience, ensuring that fans spend less time searching and more time enjoying the rich tapestry of stories that the world of anime has to offer. Our anime recommendation system AnyAni integrates a combination of advanced algorithms, a user-friendly interface, and innovative features like AnimeGAN. The synergy of content-based, collaborative filtering, and K-means clustering ensures precise and tailored recommendations for each user. With the added interactivity of a chatbot and the visual allure of generated anime art, it offers a comprehensive and immersive experience for anime enthusiasts. As we look ahead, several enhancements and expansions are envisioned for the anime recommendation system to further refine its capabilities and broaden its reach: Community Integration: Recognizing the value of shared experiences and collective insights, we plan to integrate a community feature. This will allow users to engage in discussions, share reviews, and exchange recommendations, fostering a vibrant and interactive anime community.

- Hyperlink Additions: To provide users with a more immersive experience, hyperlinks will be incorporated, directing users to anime trailers and the series itself. This will offer a quick preview and immediate access, streamlining the viewing process.
- Region and Age-based Recommendations: Understanding that cultural nuances and age-specific preferences play a pivotal role in anime appreciation, the system will be enhanced to offer recommendations based on different regions or age groups. This

ensures a more tailored viewing experience, resonating with the specific tastes and sensibilities of diverse audiences.

- Advanced Chatbot: To address queries and assist users in real-time, efforts will be directed towards developing a more powerful chatbot. Equipped with advanced AI capabilities, the chatbot will offer instant recommendations, answer questions, and provide a more interactive user experience.
- Mobile Application: Recognizing the growing trend of mobile consumption, we aim to launch a dedicated mobile application for the recommendation system. This will offer users the flexibility to receive anime recommendations on-the-go, ensuring a seamless experience across devices.

References

[CLL18] Yang Chen, Yu-Kun Lai, and Yong-Jin Liu. Cartoongan: Generative adversarial networks for photo cartoonization. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

[HKP12] Jiawei Han, Micheline Kamber, and Jian Pei. Getting to know your data. *Data Mining*, page 39–82, 2012.

[LJM+19] Pasquale Lops, Dietmar Jannach, Cataldo Musto, Toine Bogers, and Marijn Koolen. Trends in content-based recommendation. *User Modeling and User-Adapted Interaction*, 29(2):239–249, 2019.

[Man20] Vishal Mane. Anime database for recommendation system, Jun 2020.

[Mar22] Anime Market, 2022.

[SK09] Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:1–19, 2009.

[ZPIE17] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

Appendix

A.1 Project Proposal

Date of proposal: 30th September 21, 2023

Project Title: ISS Project – Animation Recommendation System – AnyAni

Sponsor/Client: Institute of Systems Science (ISS) at 25 Heng Mui Keng Terrace, Singapore NATIONAL UNIVERSITY OF SINGAPORE (NUS)

Contact: Mr. GU ZHAN / Lecturer & Consultant

Telephone No.: 65-6516 8021

Email: zhan.gu@nus.edu.sg

Background/Aims/Objectives:

Background The vast realm of animation has been proliferating rapidly, transcending cultural and demographic boundaries. This form of artistic expression has not only evolved in terms of style, genre, and technique but also in the diversity of its audience. Users often find it challenging to navigate through the myriad of animation available and to find content that aligns with their preferences and interests. This conundrum has highlighted the need for a sophisticated and user-friendly animation recommendation system.

Aims

The primary aim of our animation recommendation system is to personalize and enhance user experience by providing them with animation suggestions that are closely aligned with their preferences, viewing history, and search patterns. The system strives to offer relevant and diversified content, allowing users to explore and discover animations across genres, styles, and cultures, thereby promoting a more comprehensive and inclusive representation of animations.

Objectives

To realize the above-mentioned aim, the following objectives have been defined:

Develop a Robust Recommendation Algorithm:

1. To design algorithms that leverages user data, such as rating, preferences, and so on, to generate accurate recommendations.
2. To incorporate diverse filtering techniques like collaborative filtering and content-based filtering to refine the recommendations.

User-Centric Interface Design:

1. Develop an intuitive and user-friendly interface that facilitates seamless interaction, easy navigation, and enriched user experience.
2. Integrate features enabling users to refine their preferences, rate animations, and provide feedback on recommendations.
3. Provide a search box to allow users to look for their favorite anomies.

Incorporate Diversity and Inclusivity:

1. Ensure representation of animations from different cultures, languages, and traditions to foster a diversified content pool.
2. Implement features allowing users to explore and discover content beyond their usual preferences and get exposed to a varied range of animation styles and genres.

Optimization and Scalability:

1. Continuously optimize the recommendation algorithm to adapt to changing user preferences and emerging animation trends.
2. Scale the system to accommodate a growing number of users and an expanding animation database without compromising performance.

User Engagement and Retention:

1. Employ strategies such as push notifications, user engagement metrics analysis, and feedback loops to enhance user retention and encourage active participation.
2. Regularly update content and integrate trending animations to maintain user interest and engagement.

Legal and Ethical Compliance:

1. Uphold the highest standards of data privacy and security to protect user information.
2. Ensure that the content provided complies with intellectual property laws and the system operates within the legal and ethical framework.

Performance Evaluation and Continuous Improvement:

1. Implement analytical tools and performance metrics to monitor the efficiency and accuracy of the recommendation system.
2. Utilize user feedback and system analytics to continually refine and enhance the recommendation algorithm and user interface.

Conclusion: By achieving these objectives, the animation recommendation system aspires to revolutionize the way users interact with animation content, providing them with personalized experiences and exposing them to the richness and diversity of the animation world. The system is envisioned to be a catalyst in promoting animation as a significant form of art and entertainment, appealing to a broad spectrum of audiences, and fostering a global appreciation of animation culture.

Requirements Overview:

Research Ability
Programming ability
System integration
Algorithm understanding
Deep understanding in computer network operation system

Resource Requirements

Hardware proposed for consideration:

- GPU, CPU, Router
- Application container, e.g., Docker
- Cloud Server

Software:

- PyCharm
- VSCode
- MySQL
- Flask
- Vue
- GitHub

Programming Language:

- Python

Dataset:

- Anime Database for Recommendation system

Individual Role:
Deng Hewen: Back-end and Front-end and Algorithm
Chen Zhiwei: Back-end and Front-end and Algorithm
Luo Haiming: Back-end and Front-end and Algorithm
Zhao Xiang:Back-end and Front-end and Algorithm
Huang Yiqi: Back-end and Front-end and Algorithm

Team Formation & Registration

Team Name:	AnyAni - Group 11
Project Title (repeated):	AnyAni - animation recommendation system
System Name (if decided):	AnyAni
Team Member 1 Name:	Deng Hewen
Team Member 1 Matriculation Number:	A0285707U
Team Member 1 Contact (Mobile/Email):	1348478403@qq.com
Team Member 2 Name:	Chen Zhiwei
Team Member 2 Matriculation Number:	A0285952N
Team Member 2 Contact (Mobile/Email):	zweicen@icloud.com
Team Member 3 Name:	Zhao Xiang
Team Member 3 Matriculation Number:	A0285703B
Team Member 3 Contact (Mobile/Email):	e1221515@u.nus.edu
Team Member 4 Name:	Luo Haiming
Team Member 4 Matriculation Number:	A0285695H
Team Member 4 Contact (Mobile/Email):	e1221507@u.nus.edu
Team Member 5 Name:	Huang Yiqi
Team Member 5 Matriculation Number:	A0285711A
Team Member 5 Contact (Mobile/Email):	e1221523@u.nus.edu

A.2 User Guide

A.2.1 Requirement

Python server configuration

1. First, check the system's pre-installed python version, Ubuntu typically comes with Python 3 pre-installed. You can check if it's already installed by running the following command in your terminal.

```
<python3 --version>
```

   If you system already has Python3, you can skip 2 and processed to step 3.

2. Then install python with the following command:

```
<sudo apt update>
<sudo apt install python3>
```

3. After that, let's install and use venv module to create a virtual environment for python.

```
<sudo apt install python3-venv>
<python3 -m venv myenv>
```

   This will create a virtual environment named myvenv in the current directory. And we can use this script to activate the virtual environment:

```
<source myenv/bin/activate>
```

   Now, you can use pip to install Python package within the virtual environment without affecting the system-wide Python installation. For example, to install a package , you can run:

```
<pip install package_name>
```

MySQL configuration

First, open a terminal, and update the package list by running the following command in the terminal to ensure your system's package list is up to date.

```
<sudo apt update>
```

And then install MySQL server by running the following command:

```
<sudo apt install mysql-server>
```

During the installation, you will be prompted to set a password for the MySQL root user. Remember this password, as you will need it to log in to the MySQL server. Once the MySQL server is installed, it should start automatically, you can use the following command to make sure MySQL is running:

```
        <sudo systemctl status mysql>

```

Now, you have successfully installed MySQL on Ubuntu. You can use the MySQL client to connect to the server, manage databases, and data.

Vue.js configuration

Ubuntu repositories:

```
        <sudo apt update>
        <sudo apt install nodejs>
        <sudo apt install npm>

```

Once Node.js and npm are installed, you can use npm to install Vue CLI globally, which allow you to create and manage Vue.js projects.

```
        <sudo npm install -g @vue/cli>

```

After that, you should install Yarn to manager you project. Yarn is a package manager for JavaScript and Node.js projects, it was designed to be a more efficient and reliable alternative to the npm. To install Yarn on ubuntu, you can use the official method provided by Yarn, which involves adding the Yarn repository and then installing it.

```
        <curl -sS https://dl.yarnpkg.com/debian.pubkey.gpg | sudo apt-key
    add ->
        <echo "deb https://dl.yarnpkg.com/debian/ stable main" | sudo tee /
    etc/apt/sources.list.d/yarn.list>

```

After that, update your package list and install Yarn:

```
        <sudo apt update>
        <sudo apt install yarn>

```

Browser requirement

IE10, IE11, Edge, Firefox, Chrome, safari, opera

A.2.2 Best Practice

User Information

Login/Logout

Register

Rating & Anime Information

Anime details

Rating anime

Scored anime

Recommendation function

Get recommendation by rating

Get recommendation by chatbot

Functions of system

Search video

A.2.3 Interface document

A.3 Individual Report

A.3.1 Deng Hewen A0285707U

Personal Contribution

In this project, I mainly acted as the project manager, designed the user UI as well as the program interface of the server system, and defined the functional design of the overall system and the task assignment of program development. In the program function implementation part, I mainly designed the user-based collaborative filtering algorithm, and designed the user management function, including registration and login and logout. In the management part of the project, I was mainly responsible for the version control of the project, branch merging in github, and task coordination and conflict handling.

What learnt is most useful for you?

In this group project, I learned a lot of practical knowledge and generated many good ideas for application in other scenarios.

1. Github version control skills: In the process of project development, I will assign tasks to different people, and each person will develop on his or her own GitHub

Method	URL	parameters	remarks
POST	/login	{"username": "145", "password": "145"}	Login
POST	/register	{"username": "145", "password": "145", "email": 123@163.com}	Register
GET	/detail	None	detail
GET	/anime	None	anime
GET	/recommend	None	recommend
GET	/get_userid	None	Get_userid
GET	/rating/fetch_ratings/<account_id>/<anime_id>	None	Fetch_ratings
POST	/rating/upload_ratings	{"account_id": "145", "anime_id": "145", "score": 10}	Upload_ratings
GET	/rating/nonzero_rating/<account_id>	None	Nonzero_rating
POST	/Anyani/upload_image	{"image": file}	Upload_image
POST	/chatbot	{"userMessage": text}	chatbot

Table A.1: Table detailing API endpoints

branch, and merge it into the master branch after the development is completed. In the process of development, I often encountered the problem of version rollback and conflict resolution, and I have a clearer understanding of version control in the process of solving this problem.

2. Recommendation Algorithm: In the process of developing recommendation algorithm, I did the cleaning of dataset, data preprocessing and data importing into the database, and finally developed the user-based collaborative filtering algorithm, and learned how to deploy the algorithm model in the server.

3. UI design and development: I learned how to use Vue.js for user interaction interface development, and understood the means of interaction between the front-end interface and the server system.

How you can apply the knowledge and skills in other situations or your workplaces?

1. Project Management: The task assignment and version control that I learned in this project will help me a lot in my future work, which will enable me to better manage a

team and carry out agile development.

2. Problem solving skills: During the development of the project, we encountered a lot of problems, but we gradually discussed and solved them in the group meetings held three times a week.

A.3.2 Zhao Xiang A0285703B

Personal Contribution

During this project, my primary responsibilities encompassed both the front-end and back-end development of the chatbot, the Aniverse's front-end design, the implementation of the KMeans recommendation algorithm, and the integration of certain Application Programming Interfaces (APIs).

Front-end Development with Vue: I spearheaded the integration of a chatbot icon into the user interface, ensuring its presence on every page by embedding it within the App.vue. This aesthetically pleasing icon, when clicked, establishes a connection to the back-end, facilitating real-time interactions.

Back-end Chatbot Development with Flask: I engineered the back-end chatbot functionality using Flask. This system is designed to receive user messages from the front-end via axios, which are then processed through PaddleNLP. Leveraging the KMeans clustering algorithm, the system curates a recommendation list based on specific criteria like soup and genre. These recommendations are subsequently relayed to the front-end, appearing within the chatbot interface for the user's perusal.

Front-end Design for Aniverse: For the Aniverse interface, my contributions primarily revolved around the integration of interactive elements, including multiple buttons and a file selection feature. This design allows users to upload images, which are then processed in the back-end. Once processed, these images are displayed on the front-end, enhancing the user experience.

What learnt is most useful for you?

In this group project, I learned a lot of practical knowledge and generated many good ideas for application in other scenarios.

1. Frontend and Backend Integration: One of the most valuable skills I acquired was understanding how to integrate frontend and backend interfaces through axios and Flask. This knowledge has empowered me to create seamless interactions between user interfaces and server-side processes, ensuring efficient and responsive applications.

2. Recommendation Algorithms: Diving into the world of recommendation algorithms has been enlightening. I've gained insights into how to analyze user preferences

through ratings to curate personalized anime suggestions, enhancing user experience and engagement.

3. Advanced Database Knowledge: My exposure to databases has expanded significantly. I've delved deeper into database structures, optimization techniques, and advanced queries, equipping me with the tools to manage and manipulate data more effectively.
4. Vue + Flask Framework Setup: Learning how to set up and utilize the Vue and Flask framework combination has been instrumental. This has provided me with a robust foundation to build dynamic web applications, combining the reactive user interface of Vue with the powerful backend capabilities of Flask.

How you can apply the knowledge and skills in other situations or your workplaces?

1. Web Application Development: The proficiency I've gained in frontend and backend integration, especially with Vue and Flask, can be applied to develop scalable and responsive web applications in diverse industries, from e-commerce platforms to enterprise solutions. The git version control is useful as well.
2. Data-Driven Decision Making: With my enhanced understanding data manipulation techniques, I can contribute to data-driven decision-making processes in organizations, helping them derive insights from vast datasets.
3. Chatbot Integration: The experience of developing a chatbot can be applied in numerous sectors, from customer support in e-commerce websites to interactive assistants in educational platforms. Chatbots are becoming an integral part of digital interactions, and my expertise can aid in creating more intuitive and user-friendly chat interfaces.
4. Collaborative Teamwork: Working on this group project has honed my collaborative skills. In any workplace, effective teamwork is crucial, and the ability to collaborate and integrate various components is invaluable.
5. Problem Solving and Innovation: The challenges faced and overcome during this project have sharpened my problem-solving skills. In any workplace, unforeseen challenges arise, and the ability to innovate and find effective solutions is a sought-after skill.

A.3.3 Chen Zhiwei A0285952N

Personal Contribution

In this project, I am responsible for the 'recommend' feature. Finished part including: implement 'recommend' related routes; write codes for fetching/loading posters of anime and corresponding detail page in real time; frontend render code for homepage poster layout.

What learnt is most useful for you?

In this group project, I learned a lot of practical knowledge that I've never learned before.

1. Git version control skills. In the process of developing the project, I learned to use 'git' version control, so that we can focus on different feature of the project but also able to merge one's work to another conveniently.
2. Fast hands-on frontend codes. I am totally new to frontend (JavaScript, Vue.js) before, but I get to read through the frontend code and able to do my own contribution to this project.
3. Backend development. I learned how to implement the interface of many functions, so that in the backend I only have to focus on abstract process. Codes including writing python codes for recommendation; codes for fetching posters which doesn't not exist in the dataset originally; codes for detailed page, etc.

How you can apply the knowledge and skills in other situations or your workplaces?

1. 'git' knowledge. The 'git' tool is very common in industry development so the gain knowledge of 'git' would be helpful.
2. Problem solving skills: There are a lot of questions that bugs me from moving forward. But I finally got to solve them either on my own or after discussion with teammates.

A.3.4 Huang Yiqi A0285711A

Personal Contribution

- Resources Acquisition
  - Dataset Acquisition: Played a crucial role in sourcing the right datasets to ensure that the system had quality data to rely upon. This foundation was essential for building robust recommendation algorithms and ensuring meaningful user interaction.
- Project Management
  - Timeline and Scope: Designed the project timeline, aligning it with team members' availability and strengths. Defined the project scope to provide a clear roadmap for the team, ensuring we remained focused and met our deliverables on time.
- Development & Design for scoring feature
  - Frontend UI Design: Spearheaded the design and development of the scoring feature, utilizing Vue.js. Ensured an intuitive and user-friendly interface, promoting

engagement and easy ratings. Additionally, designed the Scored Anime page to provide users with a snapshot of their preferences.

- Database Management: Devised and constructed the ratings table, ensuring data integrity and efficient storage.
- Backend Development: Designed endpoints for consistent communication between the frontend and backend. This involved data fetching and updating scores in the database, ensuring real-time and accurate reflections of user ratings.
- Integration: Integrated frontend and backend components, ensuring smooth data flow and user experience. Addressed and resolved any integration hiccups promptly.

What learnt is most useful for you?

- End-to-End Development Insight: Rather than concentrating only on a single aspect like frontend development, my involvement in the complete software app development process has significantly broadened my horizon. Building the scoring feature end-to-end improved my coding skills and augmented my ability to think and design more comprehensively.
- Text & Language Processing: Enhanced my skills in text preprocessing and natural language processing. Practically applying lecture materials to my favorite interest, anime, has driven a deeper passion for blending technology with anime culture and intensified my commitment to innovating in areas of personal interest.
- Recommendation Algorithms: Delved into various recommendation algorithms, particularly content-based and collaborative methods. Gained a nuanced understanding of their strengths, limitations, and use-cases.
- Collaboration & Version Control: Gained hands-on experience in collaborative coding, employing Git for version control, branch management, and code merging. This not only enhanced technical prowess but also fostered teamwork, communication, and problem-solving skills.

How you can apply the knowledge and skills in other situations or your workplaces?

- End-to-End Development: The comprehensive understanding of software development, both frontend and backend, equips me to handle a range of IT projects. Whether I'm in a startup environment designing a new product from scratch or optimizing existing systems in established companies, this knowledge ensures effective planning, implementation, and troubleshooting throughout the software lifecycle.

- Text & Language Processing: In today's data-driven world, the ability to process and extract meaningful insights from textual data is invaluable. Beyond anime recommendations, this skill is transferable to various sectors, from market analysis in business, sentiment analysis in social media, to document classification in legal and administrative domains. My hands-on experience in text preprocessing allows for efficient data wrangling in any context.
- Recommendation Algorithms: With businesses increasingly looking to offer personalized experiences, recommendation algorithms play a pivotal role. Whether it's e-commerce platforms suggesting products, streaming services recommending shows, or even job portals matching candidates to roles, the knowledge of recommendation systems is universally relevant. My grounding in both content-based and collaborative methods allows for adaptable and effective solutions tailored to diverse needs.
- Collaboration & Version Control: The teamwork skills fostered through this project, coupled with my proficiency in Git, are assets in any collaborative environment. Modern workplaces demand effective teamwork and the ability to manage projects with multiple stakeholders. Version control, in particular, is not just restricted to software development; it's essential in content creation, document management, and any scenario where multiple revisions and contributions are involved. Git ensures that collaboration remains seamless, errors are minimized, and productivity is maintained.

A.3.5 Luo Haiming A0285695H

Personal Contribution

In this project, my main responsibilities include generative algorithm research, Aniverse algorithm construction, training, testing, tuning, deployment, collection and preprocessing of training data sets, and production of demonstration videos.

Research on generative algorithms: I read many papers on generative deep learning algorithms and tried the effects of some open source projects. After fully learning GAN algorithms and running through the demos of some open source GAN projects, I determined that it is feasible to develop a series of GAN algorithms to convert pictures into different animation styles.

Aniverse algorithm construction, training, testing, tuning, deployment: I used the Tensorflow framework, with the help of basic CNN and resnet structures, to build the generator and discriminator in my GAN algorithm, and designed a novel loss function, and then I The training and testing of the Aniverse algorithm were completed on the A100 server. After observing the slow inference speed of large generators, I changed some CNN layers in the generator to Depthwise Separable Convolution layers, and introduced Inverted Residual Block to replace the classic residual.Quickly, after fully verifying the availability of the generator, I converted the Tensorflow model and checkpoint files to ONNX format, and

deployed the ONNX files on the backend of AnyAni to complete the inference (generation) task more efficiently.

Training data set collection and preprocessing: I found a suitable animation image data set and animation avatar data set on the Kaggle platform. After studying some common features of image generation algorithms, I preprocessed the data set images, including edge smoothing, brightness adjustment, etc., so that the data set can train the model more effectively and reduce the noise of the generated images. and false color.

Produce presentation videos: At the end of the project, I was mainly responsible for producing two presentation videos. After fully reviewing the main work, technical points, and project highlights of the project, I carefully designed and produced two fascinating presentation videos.

What learnt is most useful for you?

Research & Feasibility Analysis: By deeply exploring generative algorithms, I understood the need for detailed research. Reading many papers and trying out demos from various projects expanded my knowledge. This gave me the confidence to judge which techniques would work best. With this knowledge, I was able to make better decisions about which methods to use.

Algorithm Development & Optimization: Working on the Aniverse algorithm, and especially with GANs, I got a better understanding of the details involved. Creating a new way to measure model accuracy and improving the model's structure, like adding certain types of layers, showed me the need for fine-tuning. I learned that it's not just about making a model that works but one that's also fast and easy to use.

Practical Challenges in Deployment: Moving models to be more user-friendly, like going from TensorFlow to ONNX, showed me the real-world challenges I might face. Seeing issues like slow processing times made it clear that making an AI tool involves more than just building an algorithm. It's important to make sure the tool works well in real-world settings.

Data Preprocessing: Handling the training data emphasized the importance of good data in machine learning. Using methods like smoothing out rough edges and adjusting lighting, I found out that even small changes can make a big difference in the model's results.

End-to-End Project Management: Making presentation videos and summarizing the project's main points taught me the importance of clear communication in tech projects. I learned the skill of explaining technical details in a way that's easy for everyone to understand.

How you can apply the knowledge and skills in other situations or your workplaces?

Versatility in Algorithm Development: Through my experience with generative algorithms like GANs, I've developed a comprehensive skill set that I can apply across various sectors. Whether it's designing graphics for marketing campaigns, enhancing database quality, detecting fraud in financial transactions, or aiding medical research, my algorithmic expertise can be a game-changer in any data-driven workplace.

Optimization Skills: My ability to optimize models isn't just about speed; it's about efficiency and reliability. In sectors like automotive (for self-driving cars), finance (for real-time stock analysis), or technology (in app development), a faster and more reliable model can lead to improved user experience and trust. As industries race to incorporate AI, my skills can streamline their transition, giving them a competitive edge.

Deployment Expertise: Transitioning a model from development to real-world use is a challenge many companies face. My experience here is directly applicable, whether it's a startup aiming to launch a groundbreaking AI app or an established firm looking to integrate AI into their existing infrastructure. I can guide teams through this transition, ensuring smooth and effective deployment.

Data Handling: Data is the new gold in almost every industry, from retail to finance to healthcare. My expertise in collecting, refining, and enhancing data can help businesses make more informed decisions, leading to improved outcomes. Whether it's tailoring marketing strategies, making investment choices, or diagnosing health conditions, my data handling skills can be pivotal.

Effective Communication: In any role, from project management to sales to training, my ability to break down and convey complex technical concepts can bridge the gap between tech teams and non-technical stakeholders. This ensures smoother project execution, clearer client communication, and more effective team collaborations.