Nguyen Nhat Minh Online Movie Streaming Platform with Integrated AI 2025

THE UNIVERSITY OF DANANG
DANANG UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

GRADUATION PROJECT THESIS

MAJOR: INFORMATION TECHNOLOGY
SPECIALTY: SOFTWARE TECHNOLOGY

PROJECT TITLE:

ONLINE MOVIE STREAMING PLATFORM WITH INTEGRATED AI

Instructor**: PhD. PHAM CONG THANG**

Student:	Student ID:	Class:
NGUYEN NHAT MINH	**102210217**	**21TCLC_DT3**

Da Nang, 06/2025

THE UNIVERSITY OF DANANG
DANANG UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

GRADUATION PROJECT THESIS

MAJOR: INFORMATION TECHNOLOGY
SPECIALTY: SOFTWARE TECHNOLOGY

PROJECT TITLE:

ONLINE MOVIE STREAMING PLATFORM WITH INTEGRATED AI

Instructor: **PhD. PHAM CONG THANG**

Student:	Student ID:	Class:
NGUYEN NHAT MINH	**102210217**	**21TCLC_DT3**

Da Nang, 06/2025

DA NANG UNIVERSITY
UNIVERSITY OF SICIENCE AND TECHNOLOGY
FALCUTY OF INFORMATION TECHNOLOGY

THE SOCIALIST REPUBLIC OF VIETNAM
Independence - Freedom - Happiness

GRADUATION PROJECT REVIEW

1. General information:

1. Student Name: Nguyen Nhat Minh
2. Class: 21TCLC_DT3 Student ID: 102210217
3. Topic Title: **Online Movie Streaming Platform with Integrated AI**.
4. Instructor: PhD. Pham Cong Thang Academic title/Degree: Doctor

II. Comments and Evaluation of the Graduation Project:

1. On the necessity, originality, and applicability of the topic: (Maximum score: 2 points)

   ……..

   ……..

2. On the outcomes and how the project addresses the assigned tasks and requirements: (Maximum score: 4 points)

   ……..

   ……..

3. On the presentation, structure, and layout of the graduation thesis/project: (Maximum score: 2 points)

   ……..

   ……..

4. Scientific value / published papers / solving problems proposed by enterprises or the university: (Maximum score: 1 point)

   ……..

   ……..

5. Limitations, shortcomings that need to be supplemented or corrected:

   ……..

   ……..

III. Student's working attitude and spirit: (Maximum score: 1 point)

……..

IV. Evaluation:

1. Score: ……../10 (rounded to 1 decimal place)

2. Recommendation: ☐ Recommended for thesis defense ☐ Revise before defense
☐ Not recommended for defense

Da Nang, date month year 2025

Instructor

SUMMARY

Topic Title: **Online Movie Streaming Platform with Integrated AI**
Student Name: Nguyen Nhat Minh
Student ID: 102210217
Class: 21TCLC_DT3

In the context of global integration and technological advancement, the entertainment industry increasingly affirms its crucial role in improving quality of life and meeting the ever-diversifying needs of people. With the remarkable development of digital technology, particularly artificial intelligence (AI), online entertainment platforms are gradually becoming the dominant trend, providing users with convenient, personalized, and high-quality experiences.

Currently, the demand for online movie streaming is growing rapidly. Users not only desire a platform offering rich and diverse content across various genres but also expect intelligent features, such as quick movie content search capabilities. This presents both challenges and significant opportunities for developing online movie streaming platforms with seamless AI integration.

Based on this reality, the project "Online Movie Streaming Platform with Integrated AI" was undertaken to create a platform providing modern, intelligent online movie streaming services. The system not only offers a diverse, high-quality movie library but also integrates AI chatbots to provide a helpful channel for answering questions related to movie information. Additionally, the platform supports features such as account management, subscription plans, and content rating, delivering maximum convenience to users.

Through this project, I aim to apply the knowledge I have acquired to real-world applications while contributing to the development of Vietnam's digital entertainment industry. I hope that the product will not only meet current user demands but also have the potential for expansion, incorporating more advanced features in the future. The report is divided into the following main sections:

- **Chapter 1**: Overview of the project.
- **Chapter 2**: Implementation and integration of AI chatbot in the web platform.
- **Chapter 3**: System design analysis.
- **Chapter 4**: Building of the movie streaming website with integrated AI chatbot.

DA NANG UNIVERSITY
UNIVERSITY OF SICIENCE AND TECHNOLOGY
FALCUTY OF INFORMATION TECHNOLOGY

THE SOCIALIST REPUBLIC OF VIETNAM
Independence - Freedom - Happiness

GRADUATION PROJECT REQUIREMENTS

Student Name: Nguyen Nhat Minh Student ID: 102210217

Class: 21TCLC_DT3 Faculity: Information Technology

Major: Software Technology

1. *Topic title:* **Online Movie Streaming Platform with Integrated AI**
2. *Project topic:* ☐ *Has signed an intellectual property agreement for final result*
3. *Initial figure and data: None*

Content of the explanations and calculations:

- Introduction: Introduce the topic, purpose of the topic, objectives, scope, and research methods.
- Chapter 1. Overview of project.
- Chapter 2. Implementation and integration of ai chatbot in the web platform.
- Chapter 3. System design analysis.
- Chapter 4. Development Of The Movie Streaming Website With Integrated Ai Chatbot.
- Conclusion and development direction.

4. *Drawings, charts (specify the types and sizes of drawings):*
   - Functional Decomposition Diagram
   - Use Case Diagram
   - Sequential Diagram
   - Database Design Table
5. *Name of instructor:* PhD. Pham Cong Thang
6. *Date of assignment: ……../……./2025*
7. *Date of completion: ……../……./2025*

Da Nang, date month 06 year 2025

Head of Division …….. **Instructor**

PREFACE

Throughout my studies and the completion of this bachelor thesis at the University Of Technology And Science, University of Da Nang, I have received enthusiastic guidance and support from my professors. I would like to express my sincere gratitude to all the lecturers who have guided and mentored me, helping me develop the skills, knowledge, and confidence necessary for my future career.

In particular, I am deeply grateful to PhD. Phạm Công Thắng for his dedicated guidance, providing step-by-step support throughout the thesis process, helping me clarify objectives, and offering valuable advice on topic selection, especially regarding artificial intelligence (AI).

Additionally, I would like to thank DZFULLSTACK Company for their support and for providing me with an opportunity to intern and learn in a professional environment. The company not only offered a conducive workplace but also allowed me to engage with real-world projects and learn from IT experts, which greatly contributed to the completion of this thesis.

However, as this is a small-scale project completed within a limited timeframe, it inevitably has shortcomings and limitations in its features. I sincerely welcome valuable feedback from professors to help me refine the results and gain more experience for my future career.

ASSURRANCE

I hereby declare that:

The graduation project report titled “Online Movie Streaming Platform with Integrated AI” is my own research work, carried out under the academic supervision of Dr. Pham Cong Thang.

All content, research results, and products presented in this report have been conducted based on my learning, research, and development processes. All references, data sources, and tools used in this project are fully and clearly cited in accordance with regulations.

I also commit that this project has not been copied from any unauthorized sources, does not use any fraudulent methods, and does not violate the copyrights of any individuals or organizations.

If there is any violation, I will take full responsibility.

Student Performed

{Signature, full name of student}

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

Acronym	Abbreviation of
API	Application Programming Interface
PK	Primary key
FK	Foreign Key
AI	Artificial Intelligence
LMM	Large Language Model
RAG	Retrieval-Augmented Generation
NLP	Natural Language Processing

INTRODUCTION

1. Reason for choosing the topic

- Nowadays, the demand for online movie streaming is increasing due to the development of the internet and smart devices. Users tend to switch from traditional movie-watching methods to online platforms because of their convenience, content diversity, and accessibility anytime, anywhere.
- To meet the growing entertainment needs of users in the digital age, this topic aims to bring a smarter, more personalized, and more efficient movie-watching experience.
- Integrating Artificial Intelligence (AI) into a movie streaming website can enhance user experience by optimizing content recommendations, moderating comments, and suggesting relevant content. This helps create a safe and healthy discussion environment, strengthening safety and civility on movie streaming platforms.

2. Project objectives

- Develop a visually intuitive movie streaming website that integrates AI features such as a chatbot for Seaching Information of Movie.
- Implement a management system for content, comments, reviews, and user feedback to ensure effective content moderation, improve service quality, and optimize system performance.
- Propose mechanisms for user and content management to ensure data security and improve user experience during platform usage.

3. Scope and Research Subjects

- Research Subjects:
  - Enterprises operating in the field of film distribution and publishing.
  - Cinema owners who want to expand their distribution channels and reach online audiences.
  - Organizations aiming to promote their films on online streaming platforms.
  - Movie enthusiasts seeking diverse, convenient, and personalized entertainment, and wishing for a safe and civilized movie-watching environment.

- Scope of the Study:
  - Develop a web-based system with core features such as movie streaming, content recommendations, comment management, and movie rating.
  - The project does not focus on producing film content but solely on building an online movie streaming platform.
  - The system is designed at a small to medium scale, with potential for future expansion.

4. Research Methodology

- Study and analyze user needs regarding online movie streaming platforms.
- Build a system model, design software architecture, database, and user interface.
- Apply artificial intelligence (AI) via chatbot to optimize the movie search feature.
- Implement and test the system, evaluate performance and scalability.

5. Features

- For User: Users can perform actions such as registering, logging in, upgrading to VIP membership, viewing watch history, saving favorite movies, commenting, reporting, watching free/paid movies, and using the chatbot to search for movies.
- For Guest: Guests can only register and watch free movies. The system will not count their views in statistical data.
- For Administrator: Administrators have access to system statistics and can manage important information on the movie website, including: movie management, episode management, account management, comment management, report management, and genre management.

6. Structure

- Chapter 1: Overview of the Topic
  - The topic "Online Movie Streaming Platform with Integrated AI" aims to develop a modern technological solution that allows users to experience online movie streaming conveniently, anytime, anywhere. The system is integrated with Artificial Intelligence (AI), specifically a chatbot that supports personalized movie search based on user preferences, thereby enhancing user experience and satisfaction.
- Chapter 2: Deployment and Integration of AI Chatbot into the Web Platform
  - This chapter presents the design and implementation of an AI chatbot that supports movie information search on the web platform. The chatbot uses advanced natural language processing technology from the Gemini API, integrated into a RAG (Retrieval-Augmented Generation) system to provide accurate, personalized answers and appropriate movie recommendations. The system is built on a smart architecture with preprocessed data, delivering a smooth and efficient user experience.
- Chapter 3: System Analysis and Design
  - This chapter analyzes requirements and designs the system using illustrative diagrams such as activity diagrams, use case diagrams, and sequence diagrams. It also involves designing the database relational model and explaining the roles of each table in detail.
- Chapter 4: Building the Movie Streaming Website Integrated with AI Chatbot
  - This chapter introduces the development environment and programming tools, describes the achieved functionalities and user interface of the website components, evaluates the strengths and limitations of the project, and proposes future improvements to complete the platform in the best possible way.

CHAPTER 1: OVERVIEW OF THE TOPIC

1.1. Topic introduction

In the context of rapidly evolving information technology, the online entertainment industry, particularly movie streaming, is undergoing a significant transformation, driven by the support of the internet and advanced technologies. With the widespread adoption of smart devices and the demand for content consumption anytime, anywhere, online movie streaming platforms have become the preferred choice for users. The emergence of artificial intelligence (AI), machine learning, and chatbots has opened opportunities to build intelligent systems that not only meet entertainment needs but also enhance the viewing experience.

The thesis “Online Movie Streaming Platform with Integrated AI” addresses the challenges faced by users and businesses in movie distribution while aligning with digital technology trends. Users seek platforms that offer not only a diverse movie library but also fast interaction, relevant content recommendations, and a civilized comment environment. Meanwhile, movie distribution businesses aim to optimize content management, improve operational efficiency, and attract a broader audience. The system is developed on a web platform with a user-friendly interface, integrated with an AI chatbot for movie recommendations, delivering superior value to both users and service providers.

Beyond practical benefits, the thesis also carries social significance, contributing to the development of the online entertainment industry amid growing digital content consumption. The platform facilitates easy access to high-quality movies, enables cinema owners and businesses to expand online distribution channels, and promotes content effectively. With personalized features based on user preferences, the system encourages rich entertainment experiences and fosters long-term service usage.

The integration of AI into the streaming platform is a key highlight, optimizing content recommendations and comment management. In an era where users demand convenience and uniqueness, these features enhance satisfaction and provide a competitive edge for the platform. The thesis not only has high practical applicability but also opens opportunities for research and development in AI applications for entertainment. By combining modern technology with real-world needs, the project aims to create an efficient, intelligent, and scalable online movie streaming system.

1.2. Overview of the Support Chatbot

1.2.1. The Development History of Chatbots

A chatbot is a computer program that simulates and processes human conversations (text or voice), allowing users to interact with digital devices as if communicating with a real person. Chatbots can range from simple programs that respond to questions with short answers to complex virtual assistants capable of learning and providing increasingly personalized interactions as they collect and process information [1].

History of Chatbot Development:

- **1950s: Theoretical Foundations**
  - **1950**: Alan Turing published Computing Machinery and Intelligence in Mind journal, introducing the Turing Test, questioning whether computers could mimic human intelligence indistinguishable in conversation. This laid the theoretical foundation for dialogue systems, including chatbots [2].
- **1960s: First Chatbot - ELIZA**
  - **1966**: Joseph Weizenbaum at MIT developed ELIZA, considered the first chatbot. ELIZA simulated a Rogerian psychotherapist, using pattern-matching techniques to recognize keywords in user inputs and respond based on predefined scripts [3].
- **1970s - 1980s: Improved Dialogue Systems**
  - **1972**: PARRY, developed by Kenneth Colby at Stanford, was an advancement over ELIZA, simulating a paranoid schizophrenic patient with more complex rules for human-like conversations. PARRY was tested with psychologists and occasionally mistaken for a human [4].
  - **1980s**: Systems like Jabberwacky (developed by Rollo Carpenter) emerged, focusing on entertainment-oriented conversations without specific purposes like ELIZA or PARRY. Jabberwacky used simple machine learning to improve responses over time.
  - **Significance**: This period saw chatbots expand from psychological research to entertainment and AI experimentation.

- **1990s: Commercial Chatbots and IRC**
  - **1995**: A.L.I.C.E. (Artificial Linguistic Internet Computer Entity), developed by Richard Wallace, used AIML (Artificial Intelligence Markup Language) to enhance dialogue capabilities, storing large conversation patterns and winning multiple AI competition awards [5].
  - **Late 1990s**: Chatbots appeared on IRC platforms and online forums, such as SmarterChild on AOL Instant Messenger, providing weather updates, news, and simple chats, marking the rise of commercial chatbots.
  - **Significance**: Chatbots began integrating into online services, serving mainstream users.
- **2000s: Chatbots in Business**
  - **2000s**: Chatbots were widely adopted in customer service, using rule-based systems for FAQs, ticketing, or technical support. Example: IVR systems and e-commerce website chatbots.
  - **2006**: IBM developed Watson, initially not a chatbot but later applied to intelligent dialogue systems for healthcare and customer service [6].
  - **Significance**: Chatbots shifted from experimental to practical applications, focusing on automation and business efficiency.
- **2010s: The Rise of Modern AI and Chatbots**
  - **2011**: Apple's Siri debuted on the iPhone 4S, advancing voice recognition and natural language processing (NLP). Siri integrated deeply into device ecosystems, paving the way for other virtual assistants [7].
  - **2014 - 2016**: Other assistants like Google Now (later Google Assistant), Amazon Alexa, and Microsoft Cortana emerged, leveraging deep learning and NLP for improved conversations.
  - **2016**: Facebook introduced the Messenger Platform, enabling developers to create chatbots for e-commerce, customer service, and marketing, leading to a chatbot boom [8].
  - **Significance**: Chatbots became essential digital tools with increasingly sophisticated NLP capabilities.

- **2020s: Intelligent and Versatile Chatbots**
  - **2020**: OpenAI's ChatGPT (based on GPT-3) launched, using large language models (LLMs) for natural, human-like conversations, capable of answering diverse questions and performing various tasks [9].
  - **2021 - 2023**: Competing models like Grok (xAI), Bard (Google), and Llama (Meta AI) enhanced dialogue capabilities, integrating into applications from education to entertainment.
  - **2025 (Present):** Chatbots are embedded in most platforms, from mobile apps to websites and IoT devices, using AGI-like AI, supporting multiple languages, voice recognition, and personalized user experiences.
  - **Significance**: Chatbots have evolved from dialogue tools to intelligent assistants, playing a critical role in automation and user experience enhancement.

1.2.2. Current Types of Chatbots

- Rule-Based Chatbots: Operate on predefined rules or logic, responding to specific keywords or questions programmed in advance.
- AI-Powered Chatbots: Utilize AI and NLP to understand context, emotions, and provide more flexible responses.

1.3. Overview of C-Sharp and PostgreSQL

1.3.1. The C-Sharp Programming Language

- C-Sharp [10] is a modern, object-oriented programming language developed by Microsoft in 2000, led by Anders Hejlsberg. Initially designed for the .NET framework, C-Sharp aims to provide a powerful, user-friendly language for developing desktop, web, game, and mobile applications. Its flexibility and robust .NET ecosystem make it widely used across various domains.
- **Feature of C-Sharp:**
  - A statically typed, object-oriented language supporting cross-platform development via .NET (formerly .NET Core, now .NET 9).
  - Multithreading: Supports asynchronous programming with async and await keywords for concurrent task execution.
  - Compiled Language: C-Sharp code is compiled into Intermediate Language (IL) and executed in the Common Language Runtime (CLR).

- **Advantages:**
  - Cross-platform compatibility (Windows, Linux, macOS) via .NET.
  - Rich ecosystem with extensive libraries and tools from Microsoft.
  - Strong integration with technologies like ASP.NET, Unity (for games), and Xamarin (for mobile apps).
- **Disadvantages:**
  - May have lower performance compared to languages like C++ for high-speed applications.
  - Dependency on the .NET environment can pose challenges for unsupported systems.

1.3.2. C-Sharp Complilation Process

- C-Sharp is often referred to as a compiled language. During the build process (Figure 1.1), the compiler converts C-Sharp code into Microsoft Intermediate Language (MSIL), stored as an executable file [11].

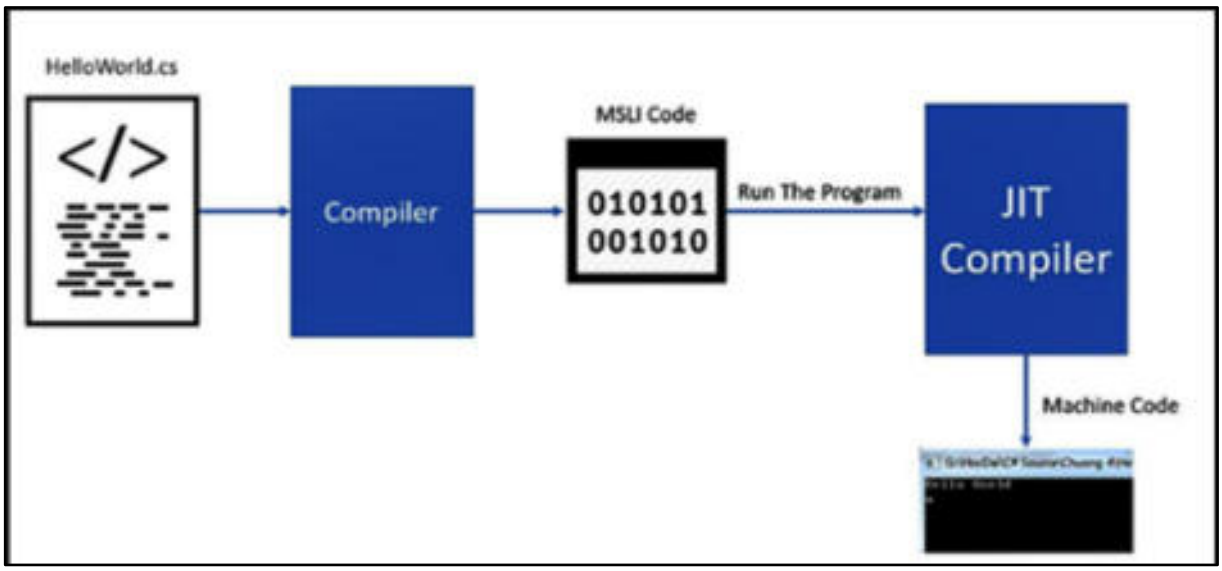

Figure 1.1. Compilation Process of C-Sharp

1.3.3. ASP.NET

- ASP.NET [12] is a powerful web development framework within the .NET ecosystem, introduced by Microsoft in 2002 and significantly enhanced with ASP.NET Core (now ASP.NET in .NET 8). It supports building high-performance web applications, from dynamic websites to APIs and real-time applications, with rapid application development (RAD) capabilities.
- ASP.NET is designed for standalone, scalable web applications, supporting cross-platform deployment with minimal manual configuration, allowing developers to focus on business logic.
- **Advantages:**
  - Builds standalone web applications runnable on Windows, Linux, or macOS.
  - Integrates with web servers like Kestrel, eliminating the need for WAR file deployments.
  - Automatic configuration of components (dependency injection, middleware) reduces coding time and boosts productivity.
  - Strong support for API development, easily integrating with frontends or mobile apps.
- **Features:**
  - Program.cs and Startup: ASP.NET uses a simple entry point (Main in Program.cs) to configure and run applications with easily set middleware.
  - Configuration Profiles: Supports environment-specific configurations (Development, Staging, Production) via appsettings.json or environment variables for flexible deployment.
  - External Configuration: Allows configuration from external sources (JSON files, databases, or cloud services) for stable operation across environments.
  - Logging: Integrates default logging with providers like Serilog or Microsoft.Extensions.Logging for consistent and customizable log management.

1.3.4. The RESTful API Model in ASP.NET Core Web API

- **Overview of RESTful API**

RESTful API (Representational State Transfer) [14] is a software architectural style widely used in ASP.NET Core Web API on .NET 8 to build modern application programming interfaces (APIs). Unlike traditional MVC with views, RESTful APIs focus on processing requests and returning data (typically JSON) via URI-defined resources, using HTTP methods like GET, POST, PUT, and DELETE (Figure 1.2).

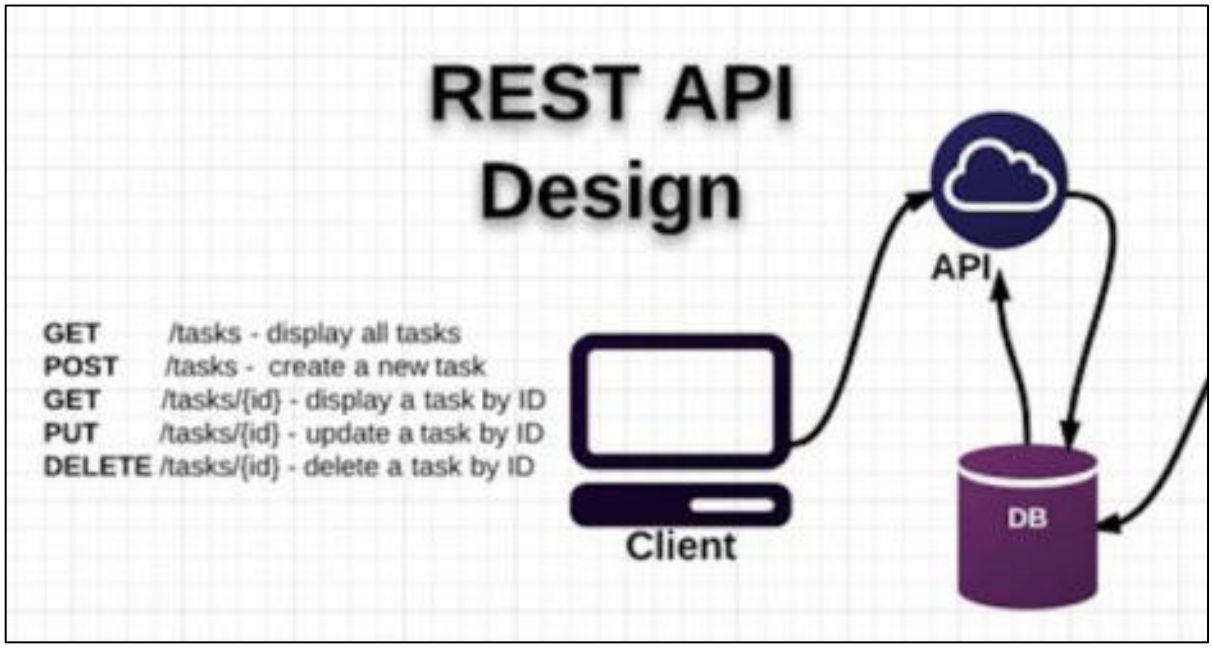

Figure 1.2. RESTful API Model

- **Layered Architecture of RESTful API**

The RESTful API [13] architecture consists of three main layers (Figure 1.3):

Application:

- Contains business logic and internal data structures of the server.
- Represents entities in the database (e.g., Movie, Category).

API:

- Acts as an intermediary between the Application and Client.
- All resources are exposed via RESTful endpoint URLs.
- Contains the project's Controllers.
- Handles HTTP operations:
  - GET: Retrieve information
  - POST: Create new resources
  - PUT: Update resources

- DELETE: Remove resources

Client:

- Can be web, mobile, desktop applications, or other services.
- Sends HTTP requests via REST libraries (e.g., Axios).
- Calls API endpoints to retrieve/create/update/delete data, receiving HTTP status codes (200, 400, 500, etc.) and JSON/XML response bodies.

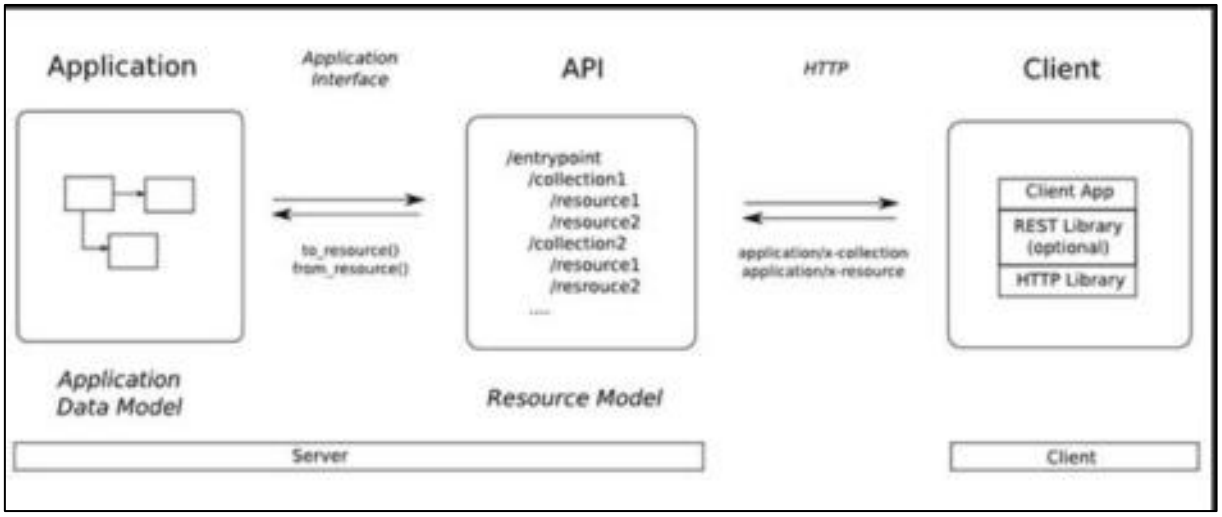

Figure 1.3. Layered Architecture of RESTful API

- **RESTful API Processing Flow in ASP.NET Core Web API:**

When a client request is sent to the ASP.NET Core server, the processing flow is as follows:

- The Controller receives the RESTful request via routing and identifies the HTTP method.
- The Controller calls the Service to execute business logic based on client input.
- The Service interacts with the Repository to query or update data from the DatabaseContext.
- Data is returned via the Controller as a JSON response, adhering to RESTful standards, and sent to the client.

1.3.5. PostgreSQL

- **Overview:**

PostgreSQL [15] often called Postgres, is a powerful, open-source object-relational database management system (ORDBMS) known for its robustness and reliability. It not only stores data but also provides a solid foundation for developing complex applications requiring high data integrity and scalability.

- **Data Types:**

  - **Numeric Types**: Includes integers (2, 4, 8 bytes), floating-point numbers with varying precision, and customizable decimals.
  - **Character Types**: Supports CHAR(n) for fixed-length strings and VARCHAR(n) for variable-length strings up to n characters, with efficient storage.
  - **Binary Data Types:** BYTEA for storing binary data like images or files.
  - **Date/Time Types**: Includes TIMESTAMP (date and time), DATE (date only), and INTERVAL (time intervals).
  - **Boolean**: Stores TRUE, FALSE, or NULL for logical states.
  - **Enumerated Types**: ENUM for defining fixed value sets (e.g., order statuses or days of the week).
  - **JSON/JSONB**: Supports JSON for data storage and JSONB (binary format) for high-performance querying and indexing.

- **Advantages:**
  - Easily extensible with advanced features, suitable for complex applications.
  - Compatible with multiple operating systems (Linux, Windows, macOS) and modern platforms.
  - Superior performance for complex queries and large datasets due to optimization.
  - Ensures data integrity with full ACID compliance, ideal for critical transactions.

- **Disadvantages:**
  - Initial setup can be complex, requiring specialized knowledge for optimization.
  - Consumes more resources compared to some systems, especially for small-scale data.

1.4. Overview of ReactJS and Bootstrap

1.4.1. ReactJS

React (also known as React.js or ReactJS) is an open-source JavaScript library for building user interfaces, developed by Meta (formerly Facebook) and the developer community. It enables fast and efficient web application development [16].

Key Features of React:

- **Component-Based:** React builds interfaces using components, each managing its own state, combined to create complex UIs, enhancing maintainability and reusability.
- **JSX (JavaScript XML):** A syntax extension for JavaScript, resembling HTML, making code more readable and writable.
- **Virtual DOM:** React uses a virtual DOM to optimize performance, updating only changed parts instead of the entire real DOM, improving application speed.

ReactJS Operation Mechanism:

- React builds interfaces using a component-based structure.
- Each component has its own state and lifecycle.
- When a component's state changes, React automatically re-renders its UI.
- The Virtual DOM optimizes UI updates, minimizing real DOM operations.

Component Lifecycle in ReactJS (Figure 1.4):

- Each component has a lifecycle with phases like Mounting, Updating, and Unmounting.
- Components can perform actions like data initialization, state updates, or resource cleanup during these phases.
- Lifecycle methods like componentDidMount(), componentDidUpdate(), and componentWillUnmount() allow intervention at specific stages.

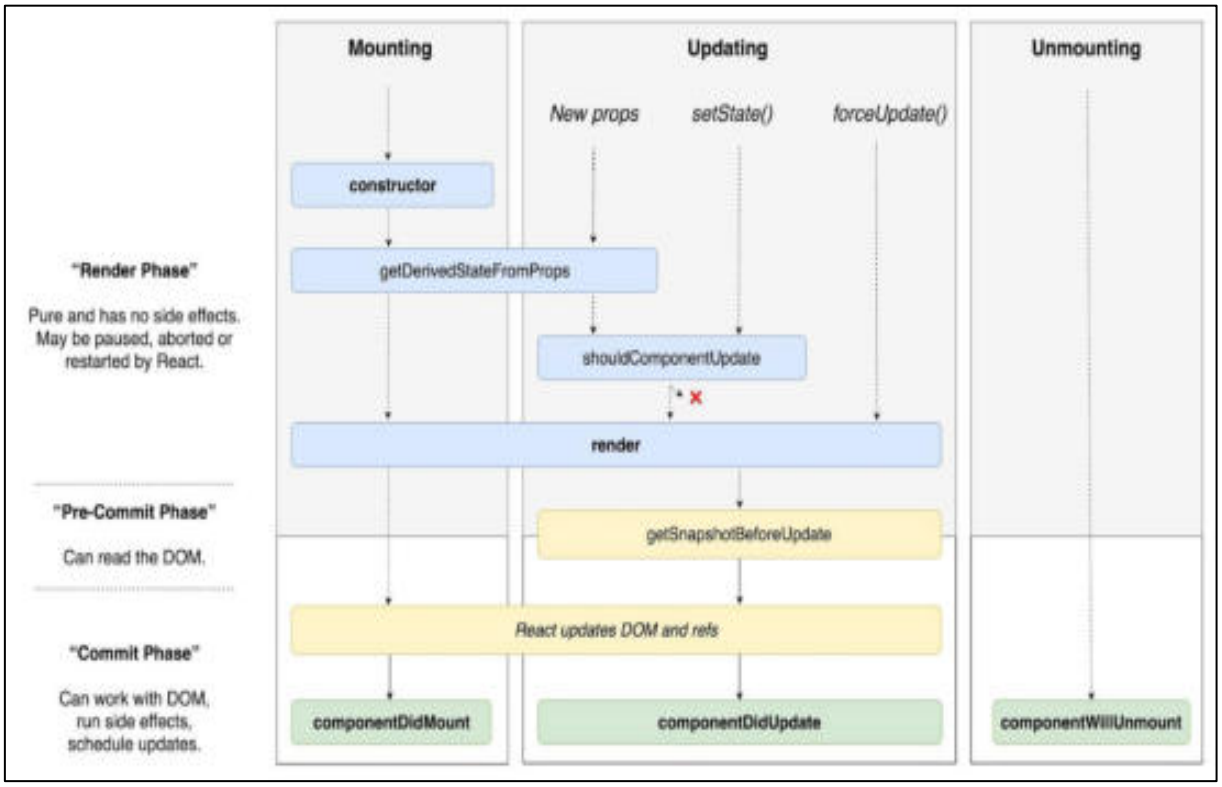

Figure 1.4. Description of the Lifecycle of a ReactJS Component [17]

1.4.2. Bootstrap

Bootstrap is a popular CSS framework developed by Twitter, enabling rapid and consistent web interface development. It provides pre-built UI components like buttons, tables, forms, navigation bars and utility classes, reducing the need for custom CSS.

Key Advantages of Bootstrap:

- **Ease of Use and Integration**: Offers ready-to-use components like buttons and navigation bars, speeding up interface development.
- **Responsive Design**: Uses a Grid System for designs compatible with various screen sizes.
- **Consistency**: Ensures uniform interfaces across browsers and devices with standardized designs.
- **Comprehensive Documentation**: Detailed documentation and a large community support users.
- **Extensibility**: Allows CSS variable customization or component extension for specific project needs.

1.5. Electronic Payment with PayOS

1.5.1. Overview of payOS

PayOS is an online payment solution designed for fast and secure transactions, particularly in e-commerce. It supports generating bank QR codes (e.g., VietQR) for each order, enabling easy payments at online points of sale without complex integration. Marketed as an open-source or free service, payOS integrates well with platforms like WooCommerce via plugins and supports multiple Vietnamese banks compatible with VietQR. It simplifies payment processes, offering one-time or reusable QR codes, bank management, and fraud prevention features. payOS is suitable for small to medium businesses and AI-driven payment automation applications.

1.5.2. Advantages of PayOS in the System

- **Easy and Fast Integration**: Provides APIs and VietQR QR codes for seamless integration with ASP.NET Core backend, enabling dynamic payment code generation for movie ticket purchases or service upgrades.
- **Support for Vietnam Market**: VietQR compatibility with local banks (e.g., Vietcombank, Techcombank, BIDV) meets Vietnamese user payment needs, enhancing project accessibility.

- **Automation and Synchronization**: Webhook support for automatic transaction status updates, enabling AI-driven personalized content recommendations based on payment history, improving user experience.
- **Low Initial Cost**: As an open-source or partially free solution, payOS reduces deployment costs, suitable for academic or startup projects.

1.5.3. Disadvantages of PayOS in the System

- **Limited Documentation and Support**: payOS documentation may lack details for handling rare errors or complex integrations, posing challenges for ASP.NET Core backend development.
- **Webhook Dependency**: Webhook speed and reliability may be inconsistent, causing delays in transaction status updates, affecting seamless payment-AI integration.
- **Limited International Scope**: Focused on Vietnam's VietQR, payOS struggles with global payment integrations (e.g., PayPal) for future expansion.
- **Stable Internet Requirement**: Users need reliable internet for QR code scanning and payment confirmation, potentially inconvenient in areas with weak connectivity, impacting streaming experience.

1.6. Chapter Conclusion

Chapter 1 provides a comprehensive overview of the thesis "Online Movie Streaming Platform with Integrated AI," from the context of the online entertainment industry's growth to the theoretical foundations of AI chatbots. The history of chatbot development, from ELIZA (1966) to modern large language models, highlights AI's potential in enhancing user experience. The selected technologies – C-Sharp with ASP.NET Core, PostgreSQL, ReactJS, and Bootstrap - form a robust, scalable tech stack. The RESTful API model ensures compatibility and efficiency in building modern web services. The integration of an AI chatbot into the streaming platform delivers personalized content recommendations and fosters an intelligent interaction environment, meeting users' growing demands. This chapter establishes a solid foundation for the system's design and implementation in subsequent chapters.

CHAPTER 2: DEPLOYMENT AND INTEGRATION OF AI CHATBOT INTO THE WEB PLATFORM

2.1. Requirement Introduction

In today's digital era, where information can be accessed with just a few clicks, developing intelligent systems that enhance user experience has become more crucial than ever. One of the most prominent trends in the technology sector is the use of chatbots to assist in searching and interacting with large databases. A chatbot is not merely a tool for answering questions - it acts as an intelligent virtual assistant that enables users to quickly access necessary information with high accuracy.

With the rapid growth of the entertainment industry, the demand for searching movie-related information has become more widespread than ever. Users often want to discover new movies, explore details of their favorite films, or simply receive suggestions based on their personal preferences. However, searching through a massive movie database can be a major challenge, especially when the data is complexly structured or the user interface is not user-friendly.

To address these issues, I have developed an intelligent chatbot designed to support movie information search, with the goal of optimizing user experience. This chatbot not only helps users easily retrieve necessary information but also offers valuable suggestions, enabling them to discover more engaging content. It is especially designed to be user-friendly for all types of users—from beginners who are new to online search to tech-savvy users.

The chatbot's features focus on meeting user needs in a fast and comprehensive manner, including: searching for detailed movie information, suggesting films based on preferences, checking showtimes or release status, and more. With powerful natural language processing (NLP) capabilities, the chatbot can not only understand direct queries but also analyze context to provide the most accurate responses.

More than just a tool, this chatbot represents a combination of advanced technology and well-structured data, delivering a smooth and convenient user experience. It not only supplies information but also enhances user-system interaction and promotes efficient, enjoyable content discovery.

2.2. Requirement Analysis

2.2.1. Chatbot User Groups

- Casual movie watchers: Interested in exploring new films or obtaining basic and detailed information.
- Fans of directors/actors: Looking for movies related to specific individuals.
- Genre/movie lovers: Searching for films by genre or by country of origin.

2.2.2. Key Features of the Chatbot

- Provide detailed information about movies based on the existing database.
- Suggest movies by genre, country, or release status.
- Answer inquiries related to showtimes and movie quality.

2.2.3. Chatbot Workflow

- Initial greeting by the chatbot:
  - "Hello! What movie are you looking for today?"
- Receiving and responding to queries:
  - Provide movie information based on user questions.
  - Display a list of relevant movies upon request.

2.3. System Design

2.3.1. System Architecture

- The chatbot architecture consists of the following components:
  - User Interface (UI): Users interact through platforms such as websites.
  - Natural Language Processing (NLP): Ensures accurate understanding of user intent by analyzing the semantics and context of each query.
  - Embedding Data: Preprocessed movie data is converted into embeddings and stored in the Chroma database for efficient retrieval.
  - GPT model: Generates natural and accurate responses based on the context and the user's question.

2.3.2. Chatflow Process

- Figure 2.1 illustrates the interaction between the user and the chatbot in the movie information query process
  - The user asks a question about movie information.
  - The chatbot analyzes the query and searches for relevant data from the vector store.
  - The chatbot responds based on the retrieved and re-ranked information.

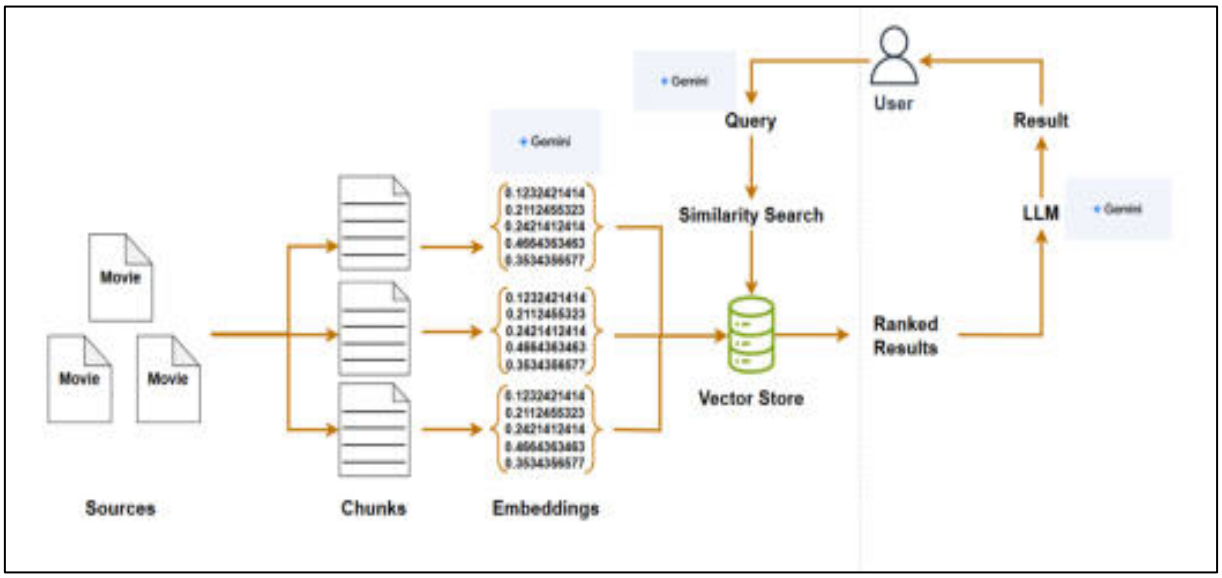

Figure 2.1. Chatbot Model

2.3.3. Chatbot Model

2.3.3.1. Data Sources

- Movie database information: Includes descriptions, genres, countries, actors, directors, status, number of episodes, and ratings.
- Movie schedules and latest episodes.

2.3.3.2. Data Preprocessing

a. Data Cleaning and Preparation

- Remove redundant data, clean up information, and normalize values.
- Split the data into smaller segments (≤ 1000 characters) to optimize the embedding process.

b. Data Embedding

- After being segmented into smaller passages of up to 1000 characters, the text is embedded into vector representations. This length is suitable for generating embeddings and optimizing document retrieval. The embedding process converts the text data into numerical vectors that can be understood by machines. It not only encodes the data but also preserves its semantic meaning. The resulting vectors are then stored in the Chroma database, allowing the system to efficiently store and retrieve semantic information in vector form.

2.3.4. Model

2.3.4.1. Introduction

- At the core of the chatbot for movie information retrieval is advanced natural language technology, built upon the Gemini API — one of the most powerful and modern AI systems available today. Gemini is not merely a natural language processing model; it is a smart ecosystem capable of deep contextual understanding, delivering accurate responses, and personalizing the user experience.

- The Gemini API stands out with a specialized architecture designed to optimize language-related tasks, offering the following outstanding features:

  - Comprehensive Context Understanding: Gemini goes beyond identifying individual keywords - it captures the overall semantics of user queries, enabling the chatbot to provide accurate answers even when the questions are complex or vague.

  - Fast Response Speed: Optimized for parallel processing, Gemini is capable of responding to user queries in real time, offering a smooth and seamless user experience.

  - Personalization Capabilities: Based on user-provided information, Gemini can generate tailored responses for each user, enhancing satisfaction and the effectiveness of interactions.

2.3.4.2. Gemini API Architecture and Generative Capabilities

- The Gemini API is built upon a cutting-edge architecture that emphasizes generative capabilities and self-learning from data. It represents a seamless integration of deep learning algorithms with the flexibility to adapt to various real-world scenarios. Key highlights of Gemini's architecture include:
  - Attention Mechanism: This allows Gemini to "focus" on the most important words or phrases in a user's question, significantly enhancing its ability to understand context and generate appropriate responses.
  - Multi-turn Dialogue Handling: Gemini supports extended conversations with multiple back-and-forth exchanges, enabling the chatbot to track and remember information from earlier interactions. This ensures more accurate and contextually relevant responses in subsequent turns.
- The generative capabilities of Gemini API empower the chatbot not only to answer questions, but also to creatively generate responses based on context. This is especially useful when users require detailed replies or highly personalized suggestions - for example, movie recommendations based on previous search history or individual preferences.

2.3.4.3. Application in RAG-based Chatbot

- When integrated into a Retrieval-Augmented Generation (RAG) system, the Gemini API functions as the "brain" of the chatbot. In this workflow:
  - Context Retrieval: RAG retrieves relevant information from the movie database, which has been converted into vectors through embeddings.
  - Smart Answer Generation: Based on the retrieved context, the Gemini API generates coherent, relevant, and natural responses.
- The synergy between RAG and Gemini API not only enhances the accuracy of responses but also enables the chatbot to provide intelligent suggestions, delivering real value to users.

2.3.4.4. Conclusion on the Model

- The use of the Gemini API as the foundation not only ensures that the chatbot operates efficiently but also delivers a superior user experience. With cutting-edge technology and powerful processing capabilities, the Gemini API enables the chatbot to become more than just a movie information retrieval tool — it transforms into a trusted companion, helping users explore the world of cinema in a more engaging and effortless way.

2.4. Chapter Conclusion

This chapter has described the design and implementation steps of a chatbot that supports movie information retrieval. With a strong data foundation, intelligent system architecture, and natural language processing capabilities provided by the Gemini API, the chatbot promises to deliver an outstanding user experience, improve user engagement, and enhance the overall value of the service.

CHAPTER 3: SYSTEM ANALYSIS AND DESIGN

3.1. Requirement Overview

System Requirement Analysis: Clarifies the main business processes for each user type (guests, users, and administrators). Identifies necessary functions for managing information, movie streaming, and related activities.

Overall System Design: Develops operating principle diagrams to illustrate component interactions. Designs use case diagrams to describe user-system relationships.

Detailed Function Design: Provides sequence diagrams for key functions like login, movie streaming, information management, and payment. Describes function interactions to ensure efficiency and stability.

Database Analysis: Models data for storing user, movie, payment, and management information. Converts entity models into relational models, ensuring consistency and scalability.

3.2. Requirements Specification

3.2.1. Guest Business Processes

Guest Users: Visitors who have not logged in or registered.

- **View Public Movies**: Display a list of movies with basic information (title, genre, release year, poster).
- **Filter and Search**: Search movies by title or filter by genre or release year.
- **View Movie Details**: Includes descriptions, actors, directors, trailers, and user ratings.
- **Interact with AI Chatbot**: Get movie recommendations, information, or website usage support.
- **Register Account**: Guests can sign up for a member account using an email.

3.2.2. User Business Processes

Users: Registered users with full access to platform features.

- **Account Management**: Login, logout, update personal information, change password.
- **Movie Streaming**: Watch movies in HD, control video playback, use subtitles, and resume from the last position.
- **List Management**: Add movies to favorites.
- **Content Interaction**: Rate movies, comment, provide feedback, and rank movies. Participate in community activities.
- **VIP Features**: Subscribe to VIP to access exclusive movies.
- **AI Chatbot Interaction**: Receive personalized movie recommendations, search by description, or ask movie-related questions.
- **Sharing and Connectivity**: Share movie links with friends or family.

3.2.3. Admin Business Processes

Administrators: Have the highest system privileges, responsible for managing the platform and ensuring stable operations.

- **Visual Statistics**: Provide an overview via statistical dashboards (movie counts, views, users).
- **Movie Management**: Add, edit, or delete movies; categorize genres; set access levels for user groups.
- **User Account Management**: View registered account lists.
- **Comment Management**: Moderate user comments and handle violation reports.
- **Feedback Management**: Receive and process user feedback to improve content quality.
- **AI Chatbot Management**: Train and refine the chatbot model to optimize interaction and user experience.

3.3. System Analysis

3.3.1. System Workflow Diagram

- Figure 3.1 illustrates the activity flow of guest visitors accessing the website:

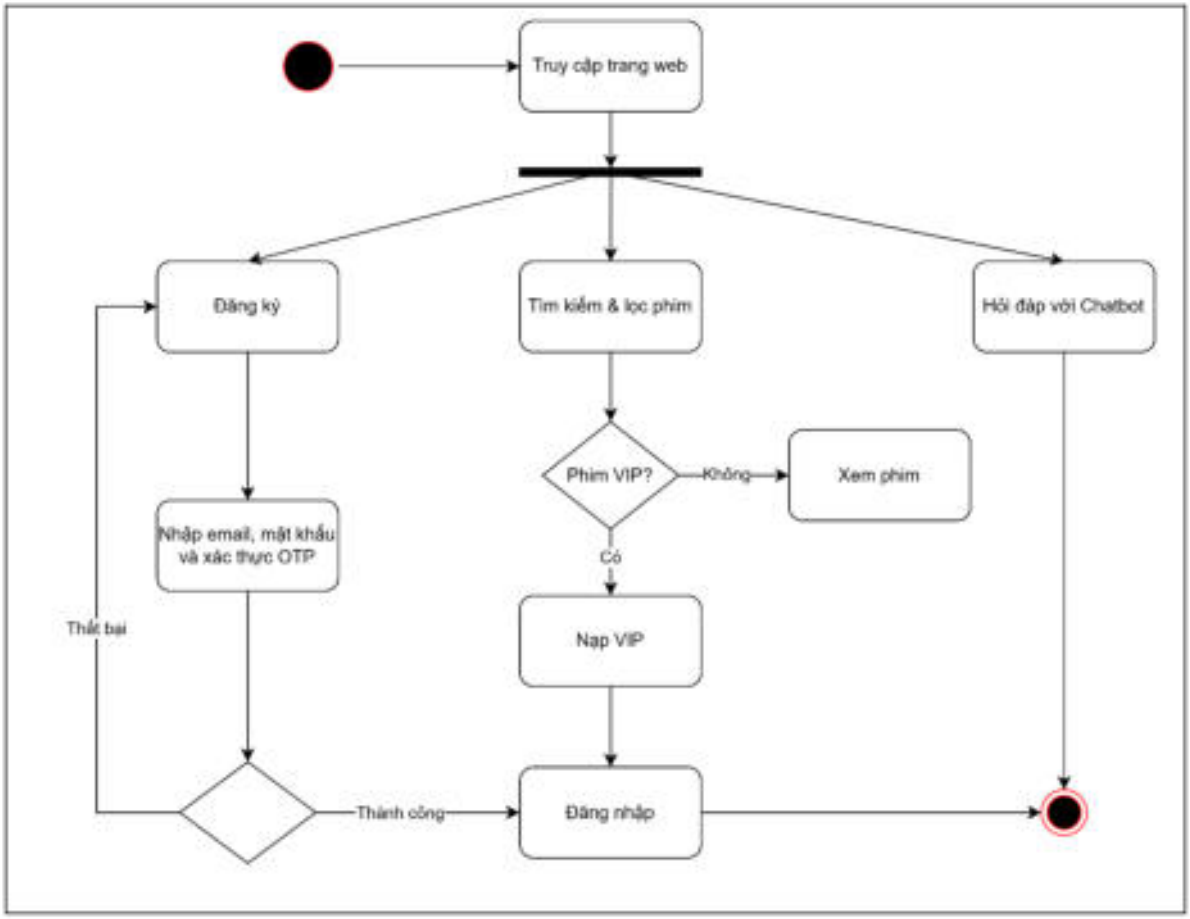

Figure 3.1. Guest User Workflow

- Figure 3.2 illustrates the activity flow of users accessing the website:

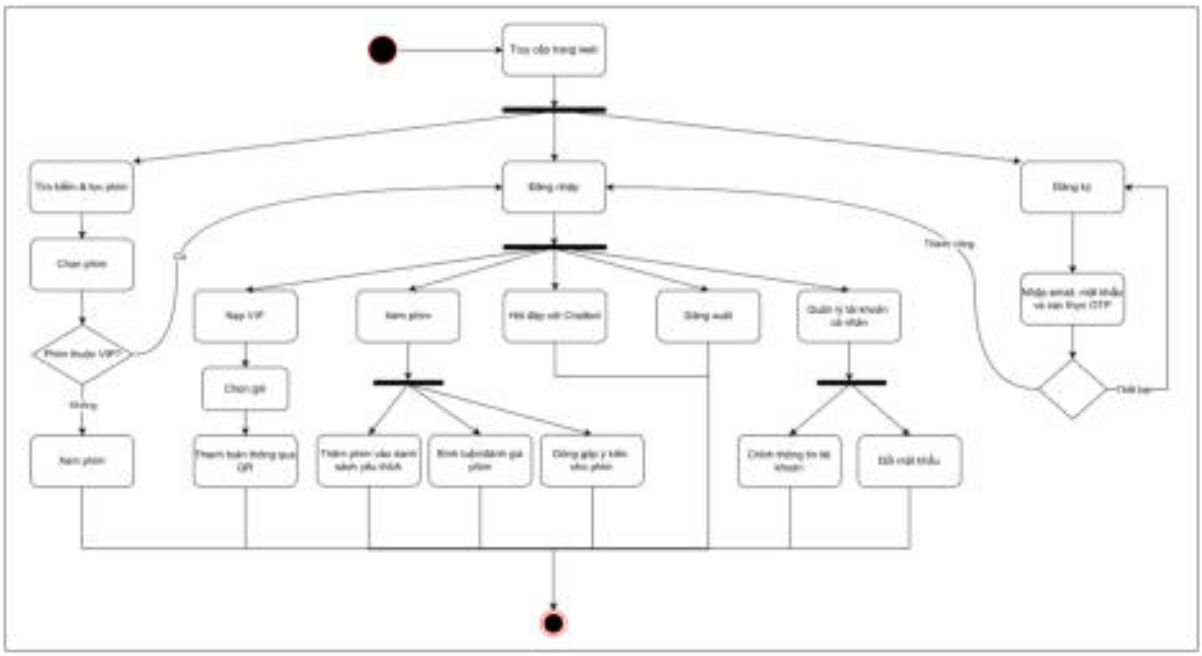

Figure 3.2. User Workflow

- Figure 3.3 illustrates the activity flow of administrators accessing the website:

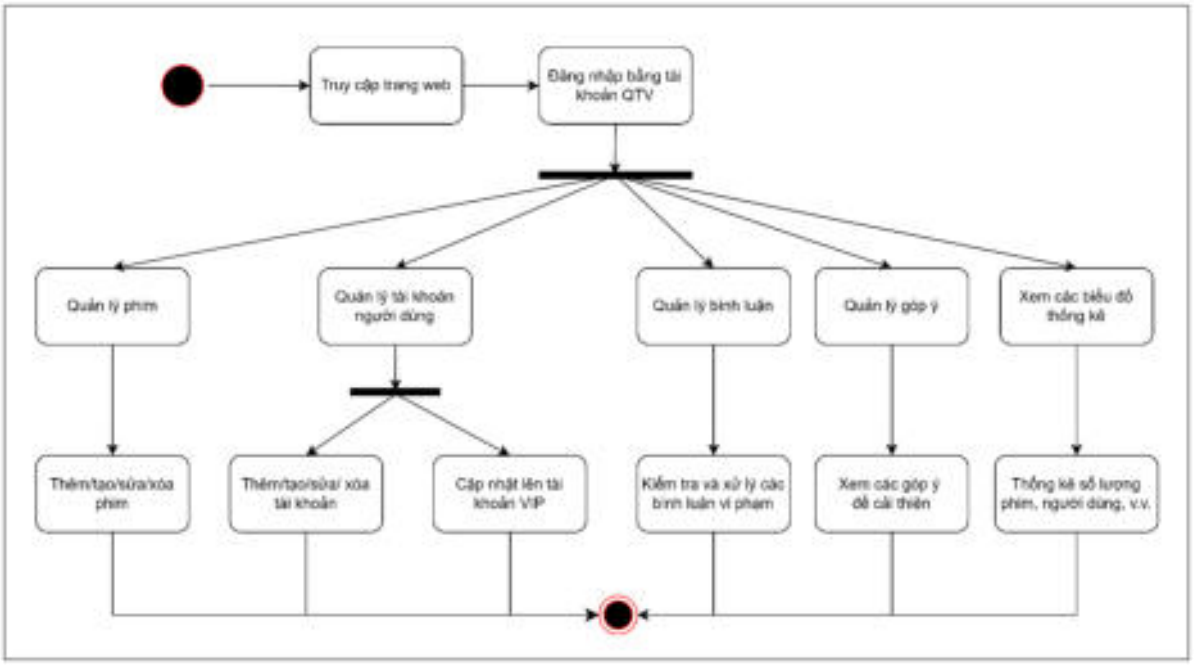

Figure 3.3. Administrator Workflow

3.3.2. Use Case Diagrams

Use case diagrams are a software engineering technique to capture system functional requirements. They describe typical interactions between external actors and the system, reflecting system behavior from the user's perspective.

3.3.2.1. Overview of System Functionalities

- The use case diagram of the system is shown in Figure 3.4:

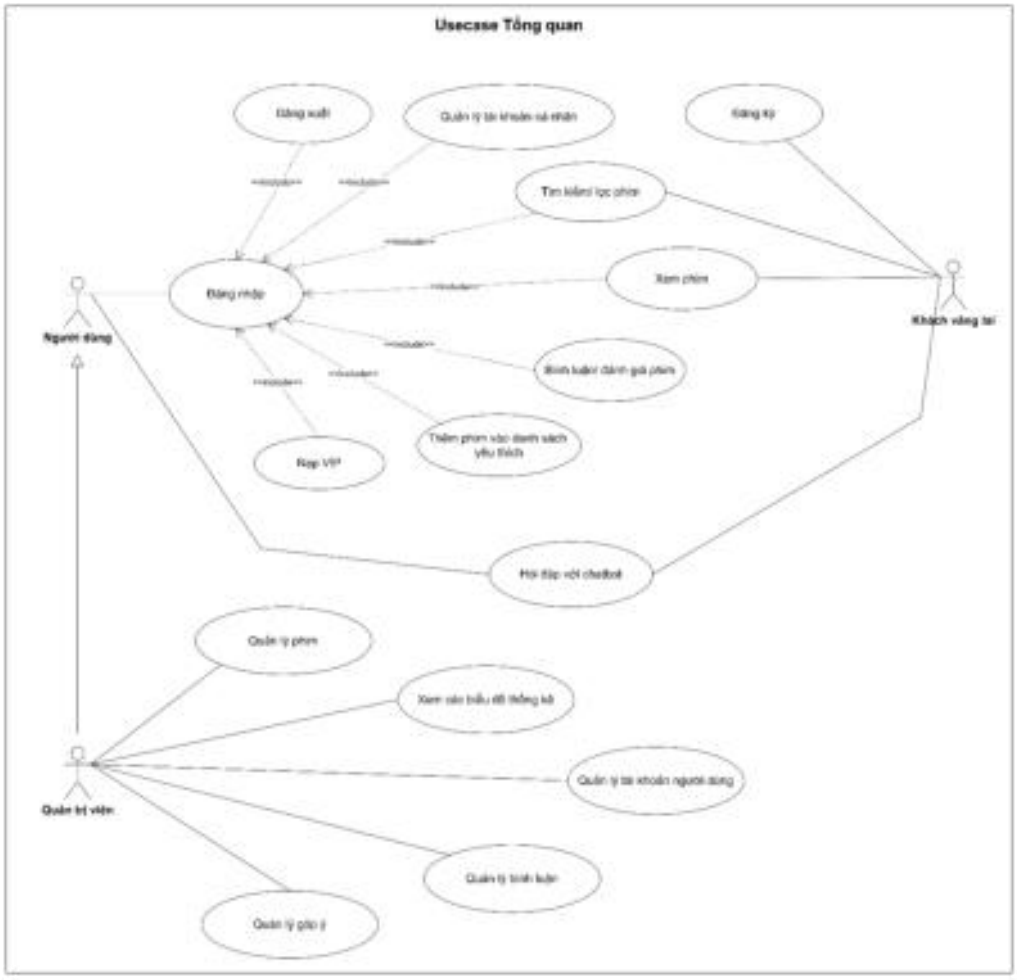

Figure 3.4. Use Case Diagram of the System

3.3.2.2. Overview of User Functions

- Figure 3.5 illustrates the use case diagram for managing personal accounts:

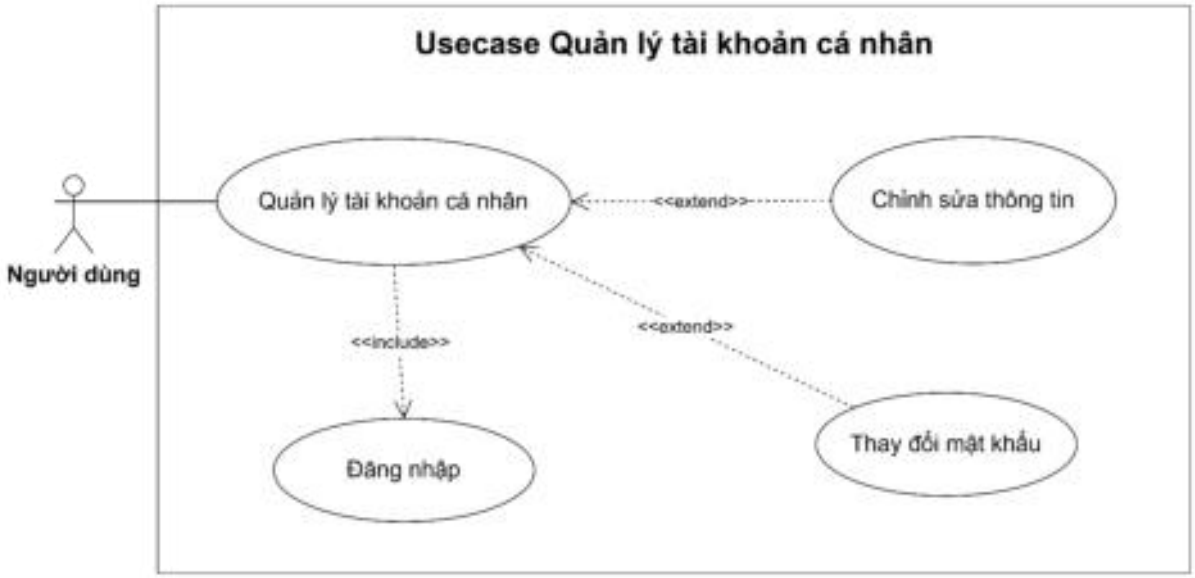

Figure 3.5. Use Case Diagram of Movie Management

- Figure 3.6 illustrates the use case diagram for the movie search function:

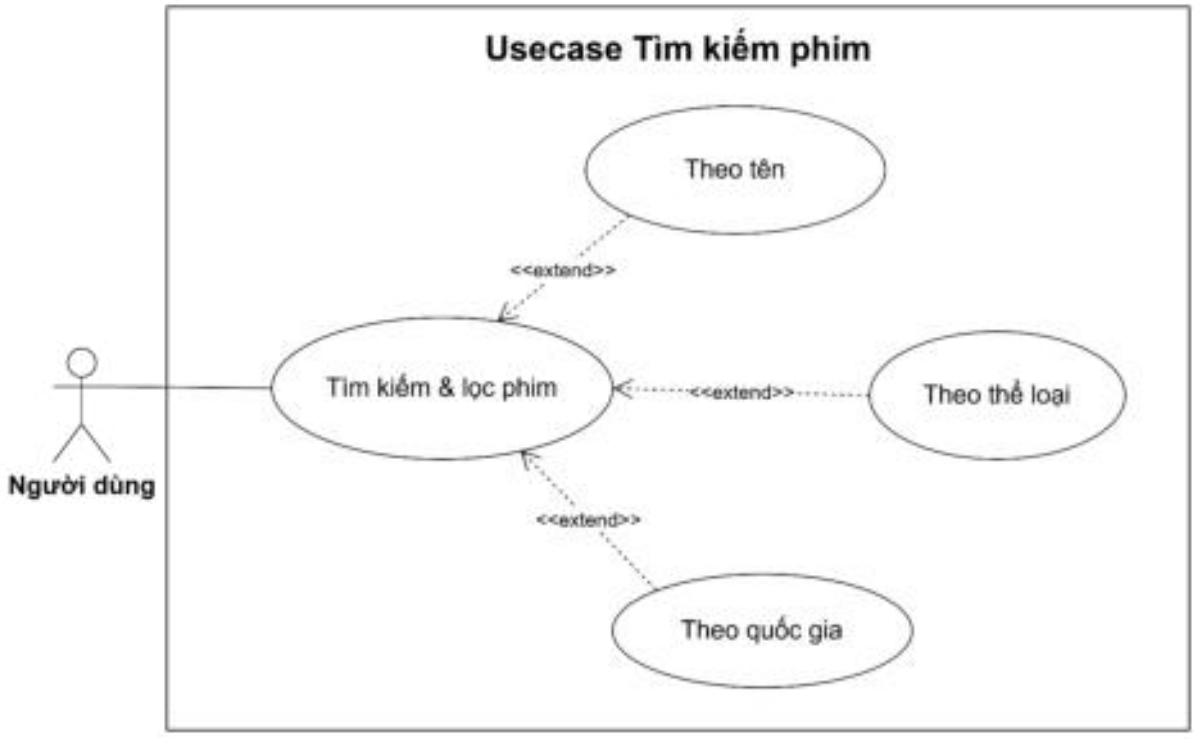

Figure 3.6. Use Case Diagram of Movie Searching

- Figure 3.7 illustrates the usecase diagram for the account upgrade function:

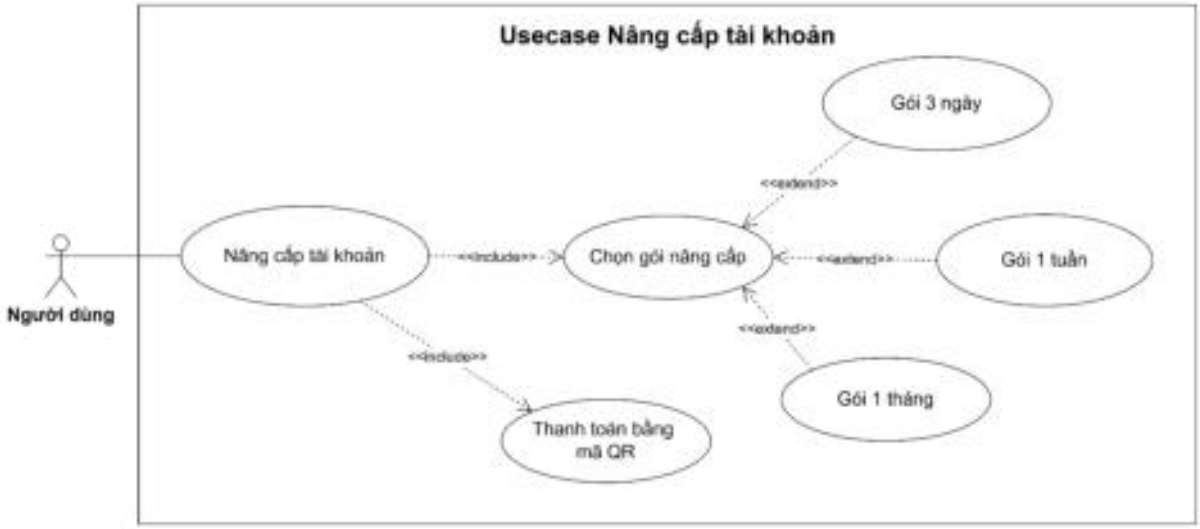

Figure 3.7. Use Case Diagram of Account Upgrade Functionality

- Figure 3.8 illustrates the use case diagram for the chatbot Q&A function:

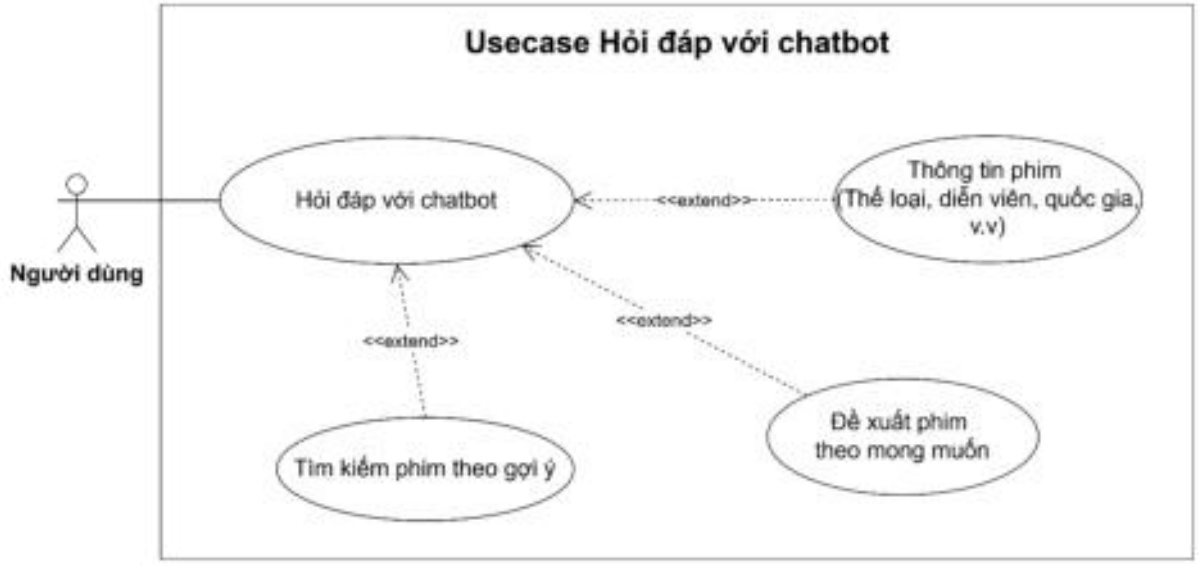

Figure 3.8. Use Case Diagram of Chatbot Q&A Functionality

- Figure 3.9 illustrates the use case diagram for the movie viewing function:

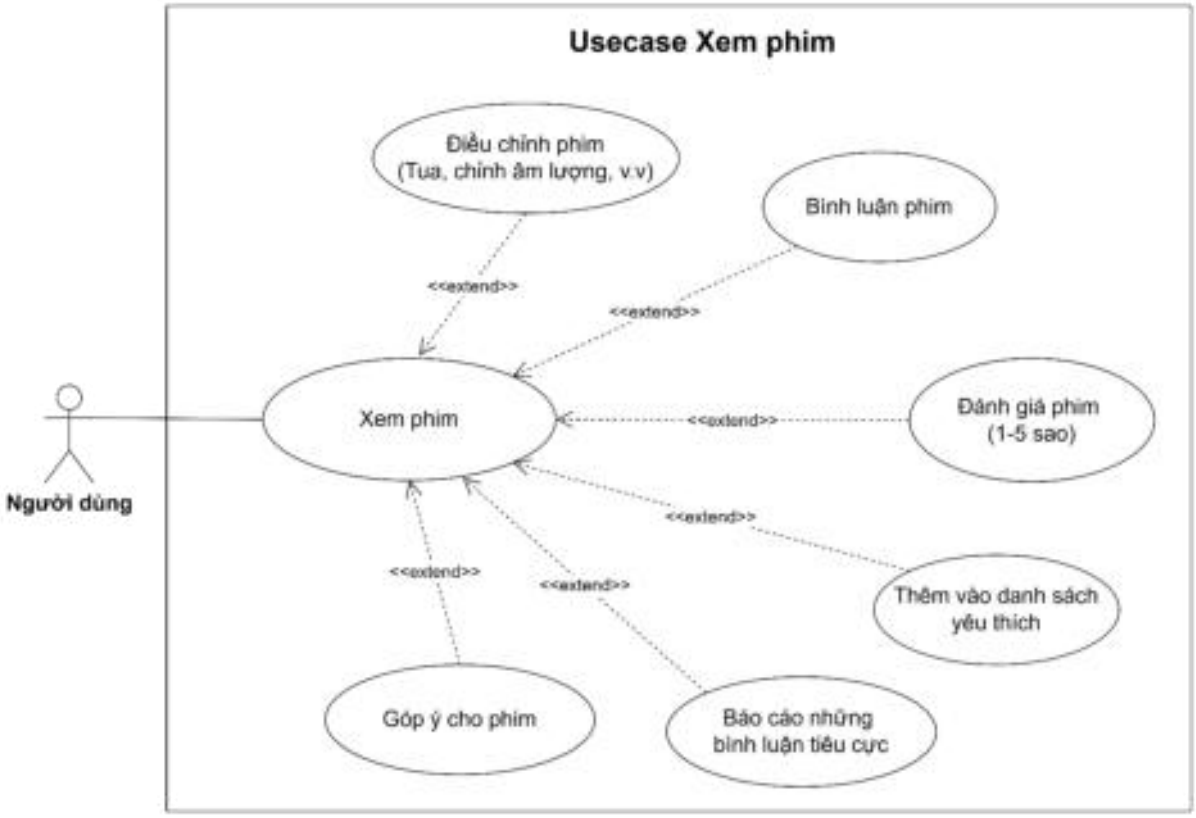

Figure 3.9. Use Case Diagram of Watching Movies

3.3.2.3. Overview of Admin Functions

- Figure 3.10 illustrates the use case diagram for the statistics chart viewing function:

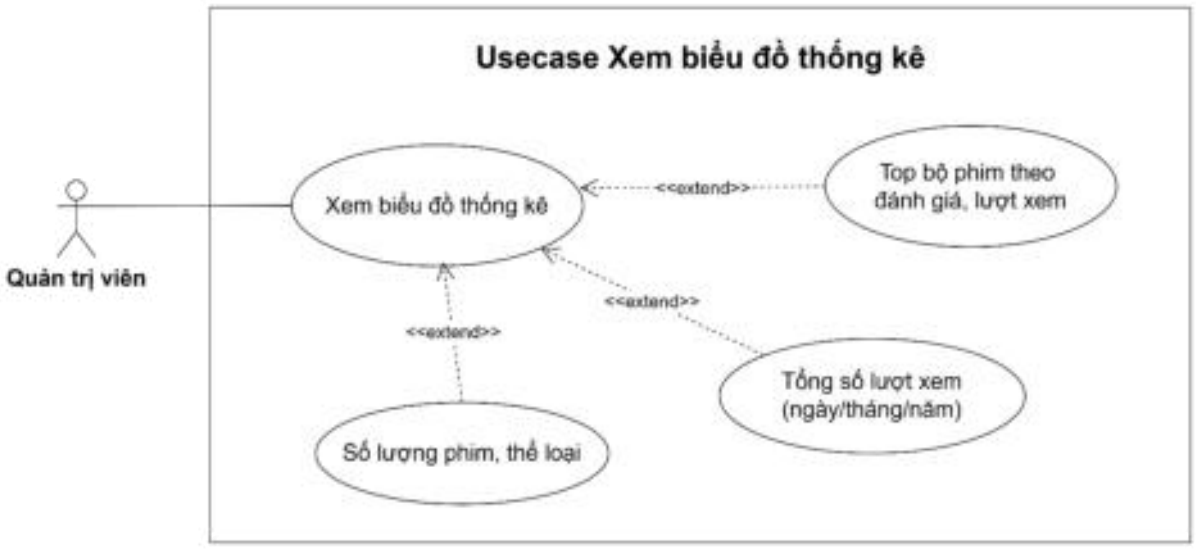

Figure 3.10. Use Case Diagram of Viewing Statistical Charts

- Figure 3.11 illustrates the use case diagram for the movie management function:

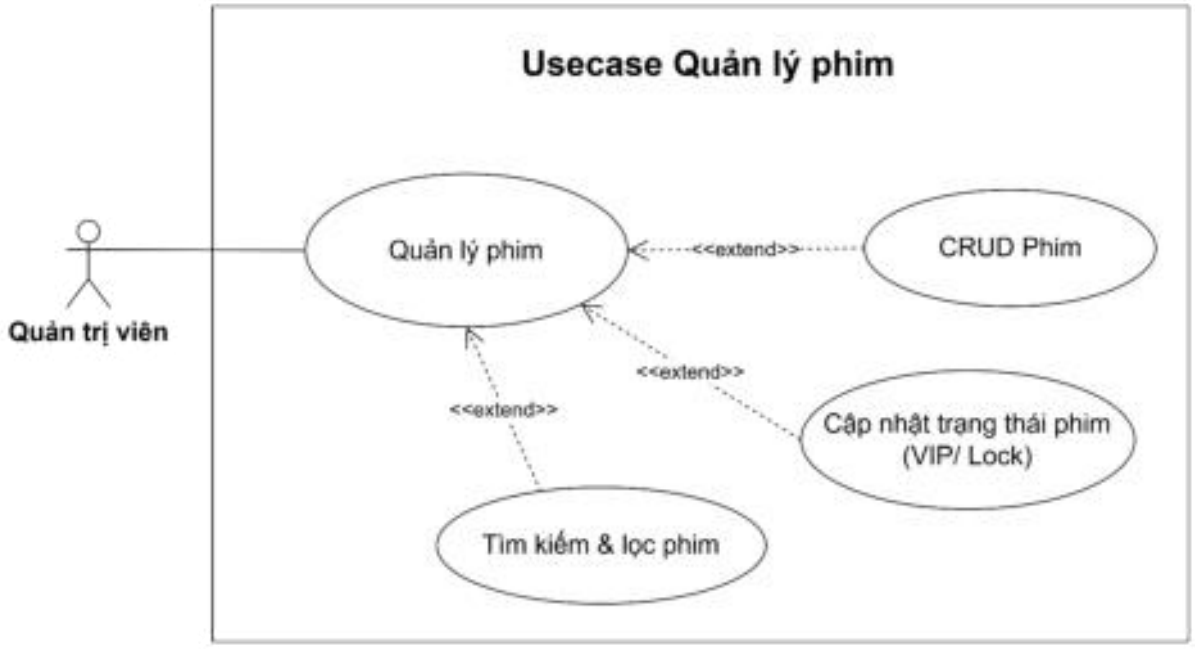

Figure 3.11. Use Case Diagram of Movie Management

- Figure 3.12 illustrates the use case diagram for the user management function:

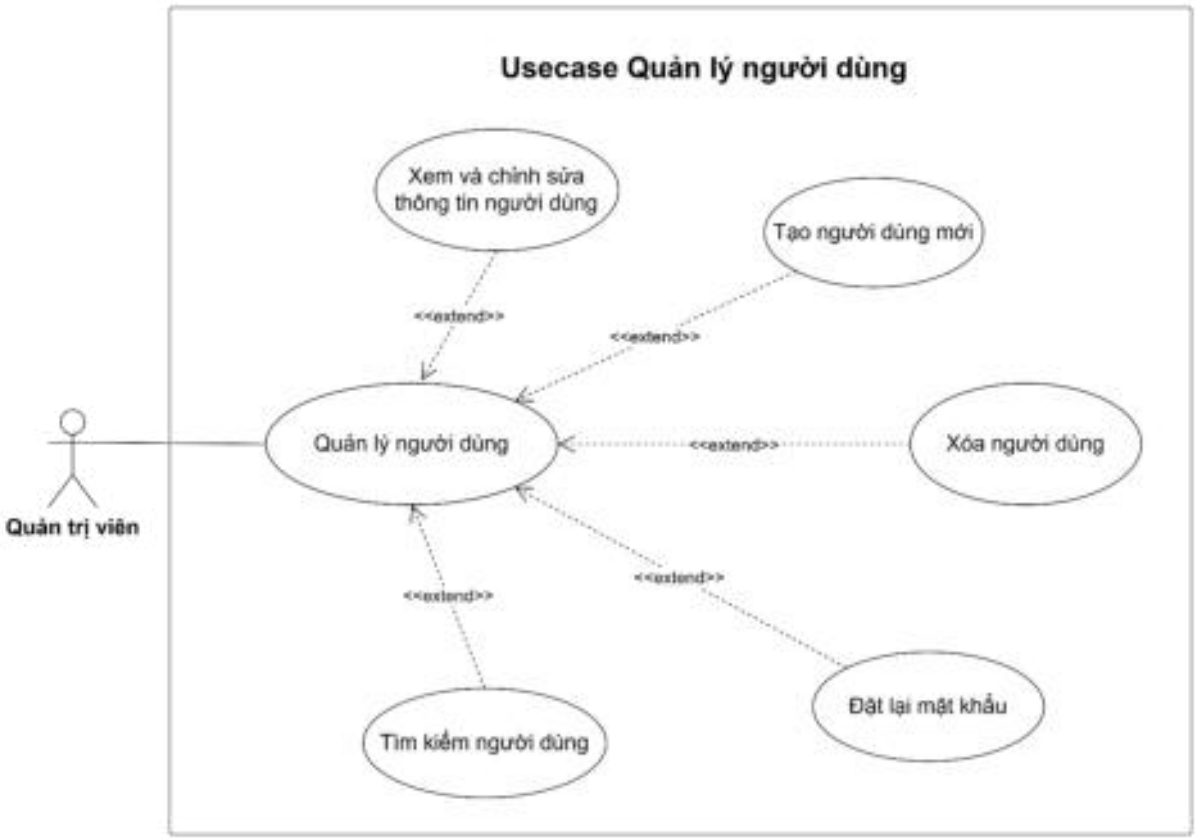

Figure 3.12. Use Case Diagram of User Management

- Figure 3.13 illustrates the use case diagram for the comment management function:

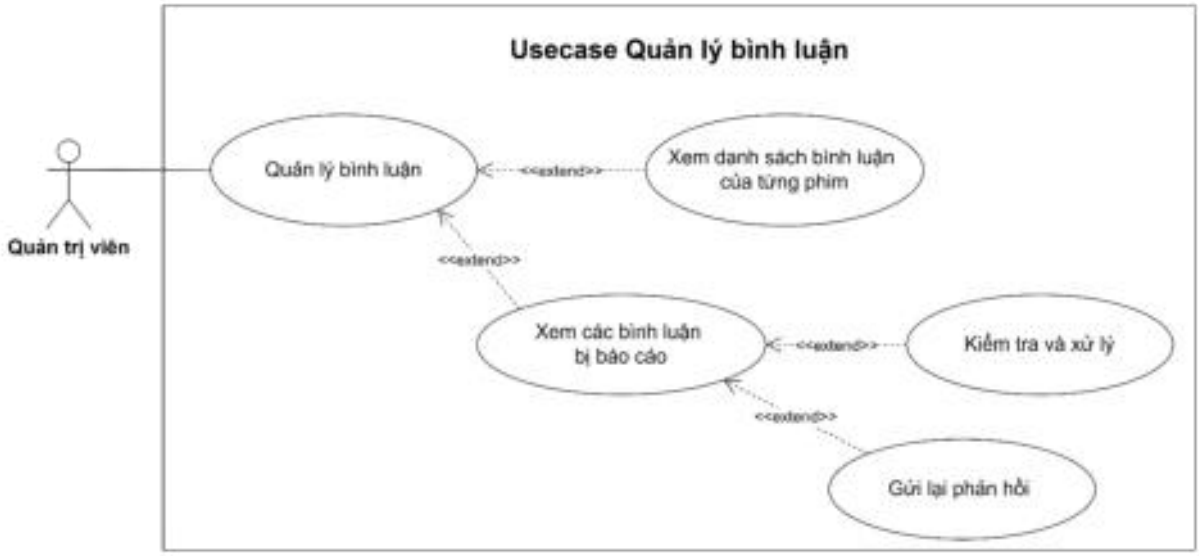

Figure 3.13. Use Case Diagram of Comment Management

- Figure 3.14 illustrates the use case diagram for the feedback management function:

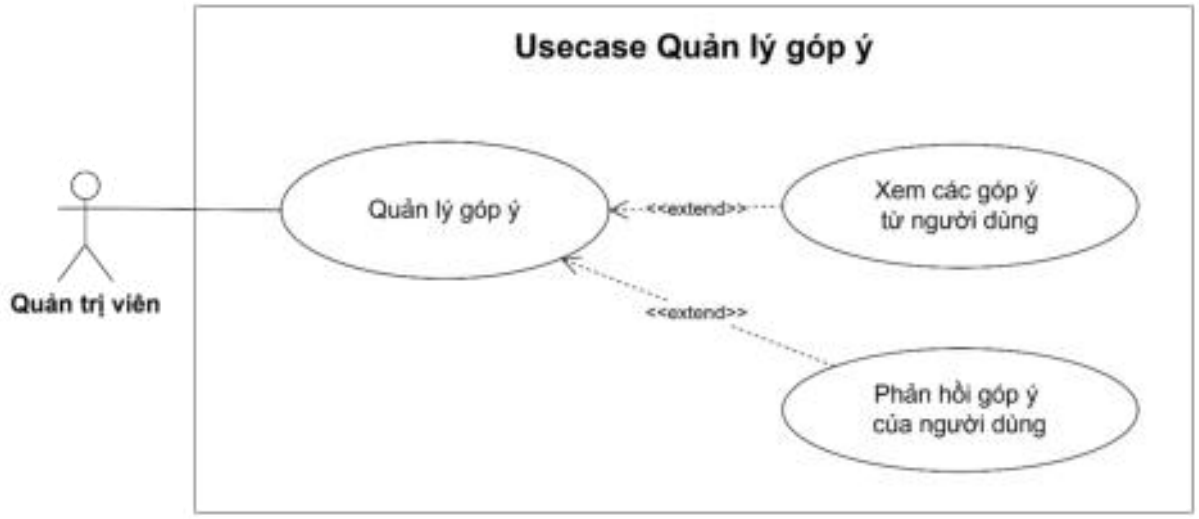

Figure 3.14. Use Case Diagram of Feedback Management

3.3.3. Sequence Diagrams

- Figure 3.15 presents the sequence diagram for the login function:

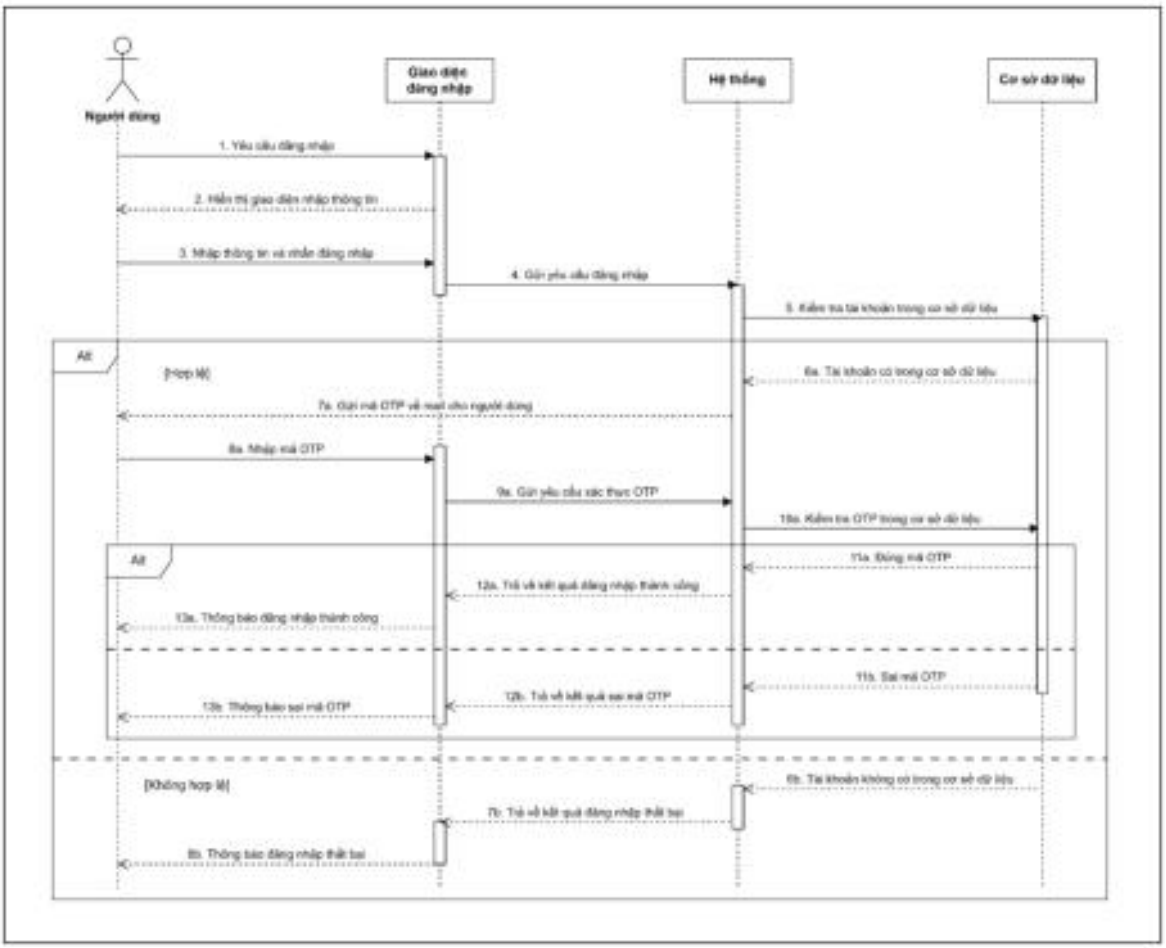

Figure 3.15. Sequence Diagram of User Login Functionality

- Figure 3.16 presents the sequence diagram for the registration function:

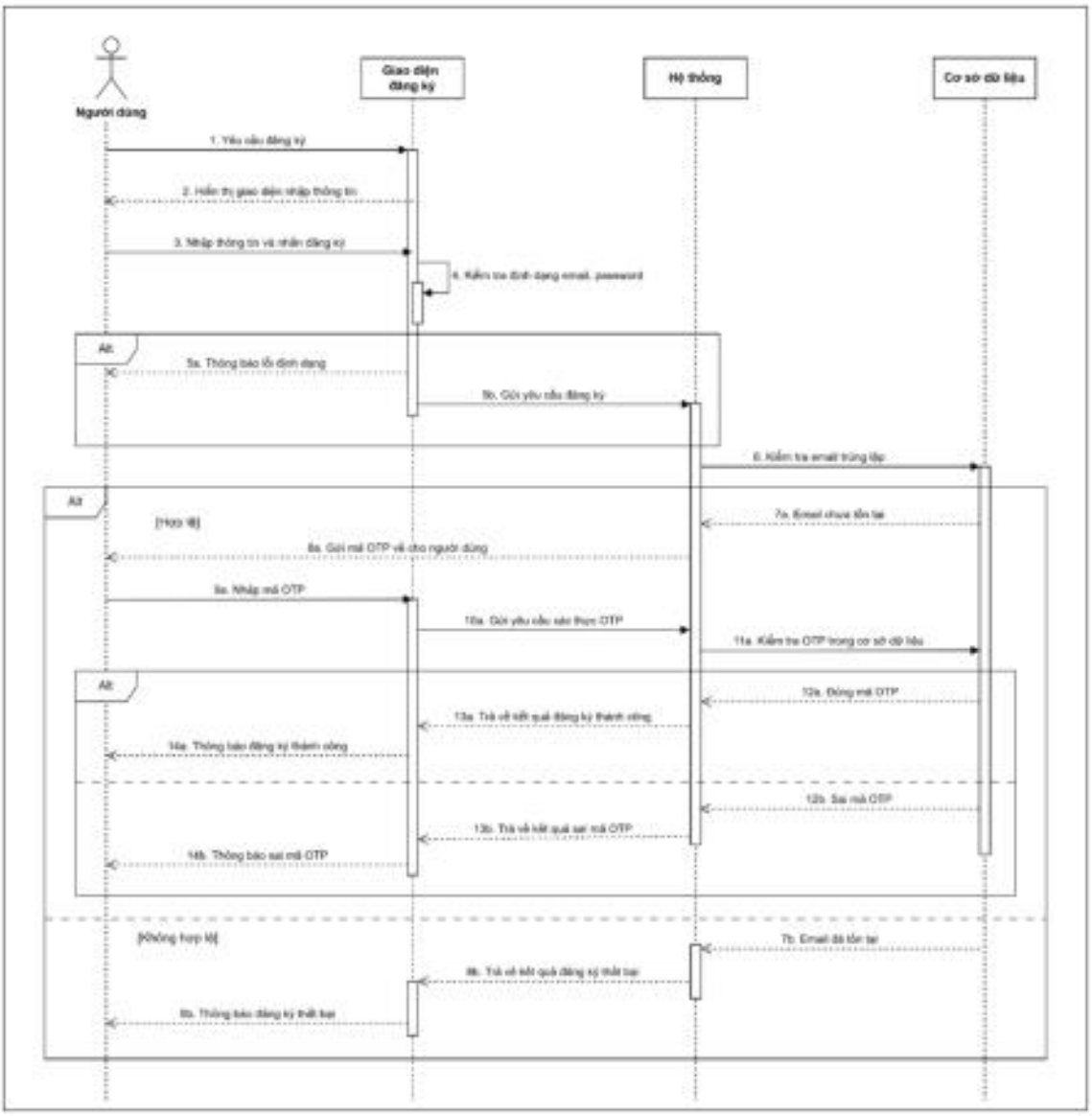

Figure 3.16. Sequence Diagram of Registration Functionality

- Figure 3.17 presents the sequence diagram for the logout function:

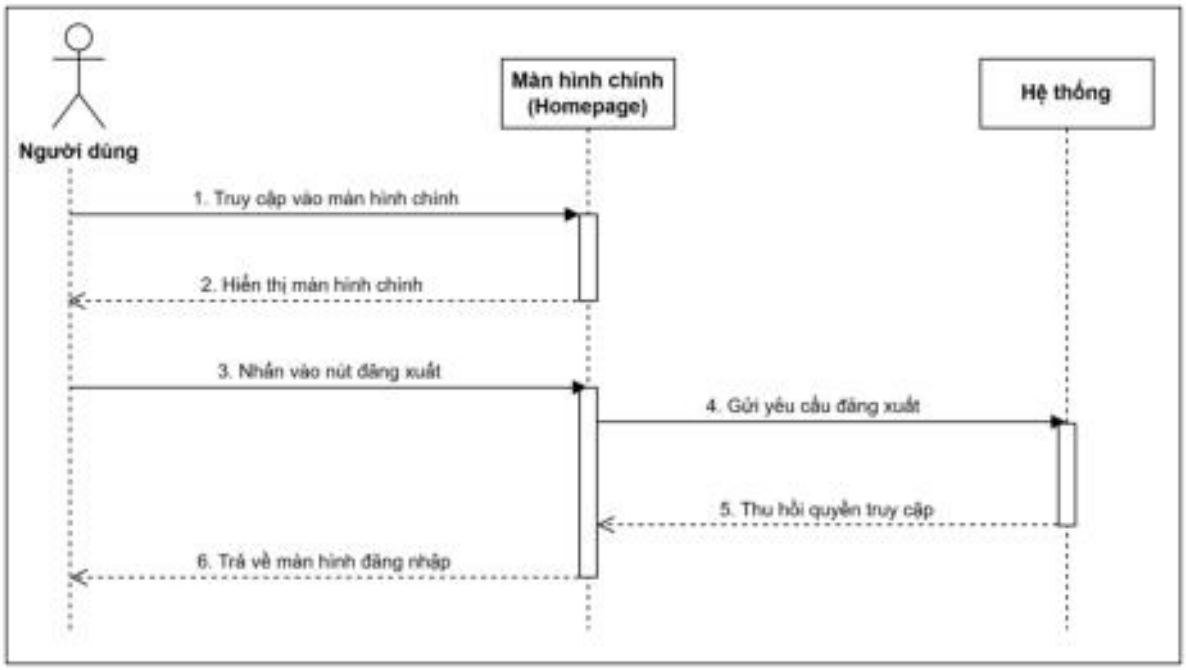

Figure 3.17. Sequence Diagram of the Logout Function

- Figure 3.18 presents the sequence diagram for the personal account management function:

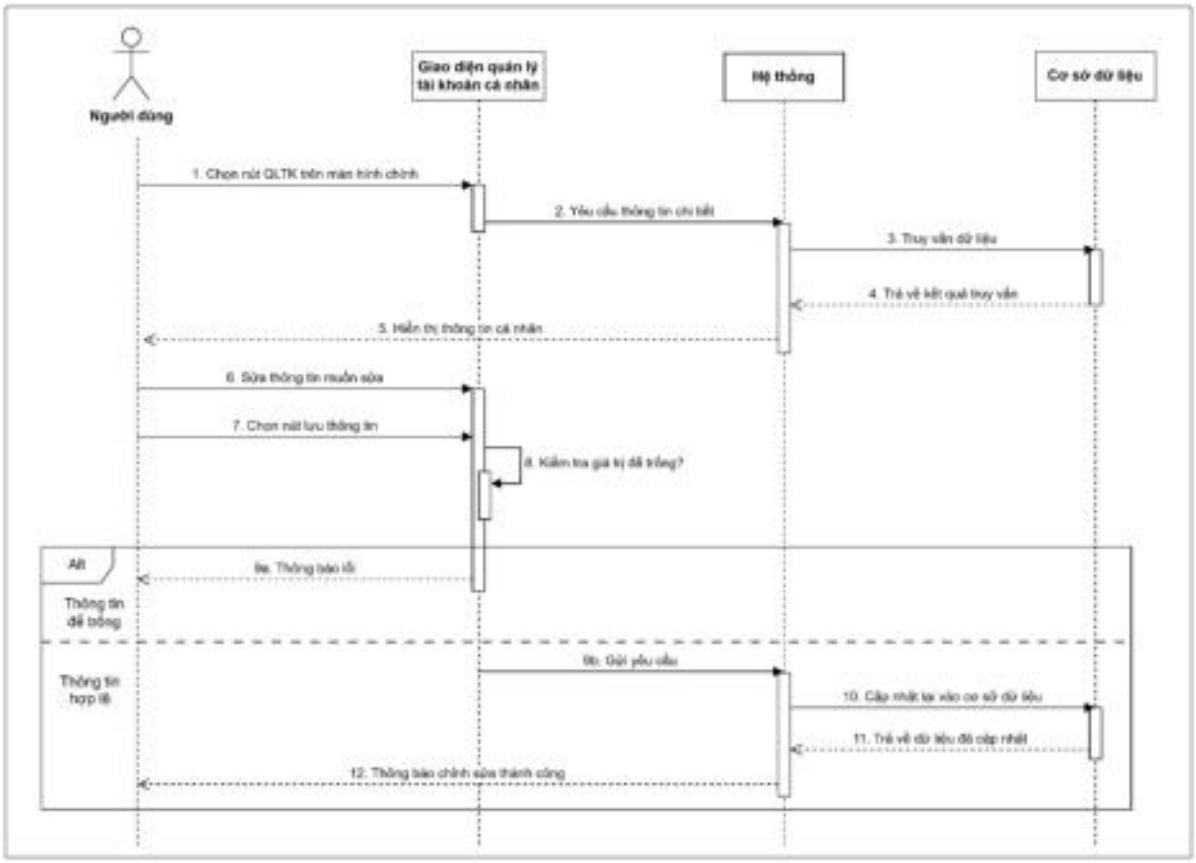

Figure 3.18. Personal Account Management Sequence Diagram

- Figure 3.19 presents the sequence diagram for the password change function:

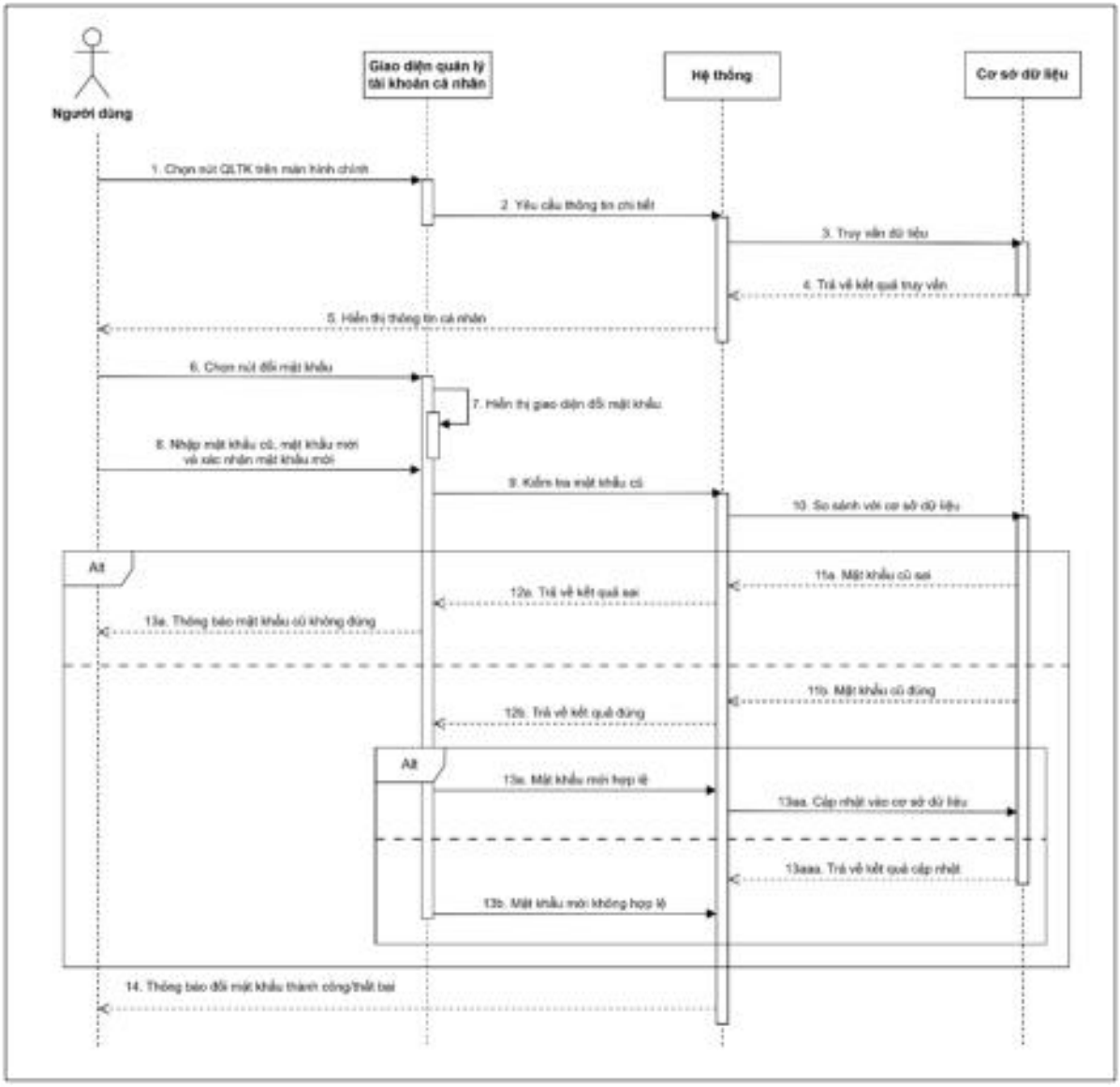

Figure 3.19. Sequence Diagram of Password Change Functionality

- Figure 3.20 presents the sequence diagram for the user account management function:

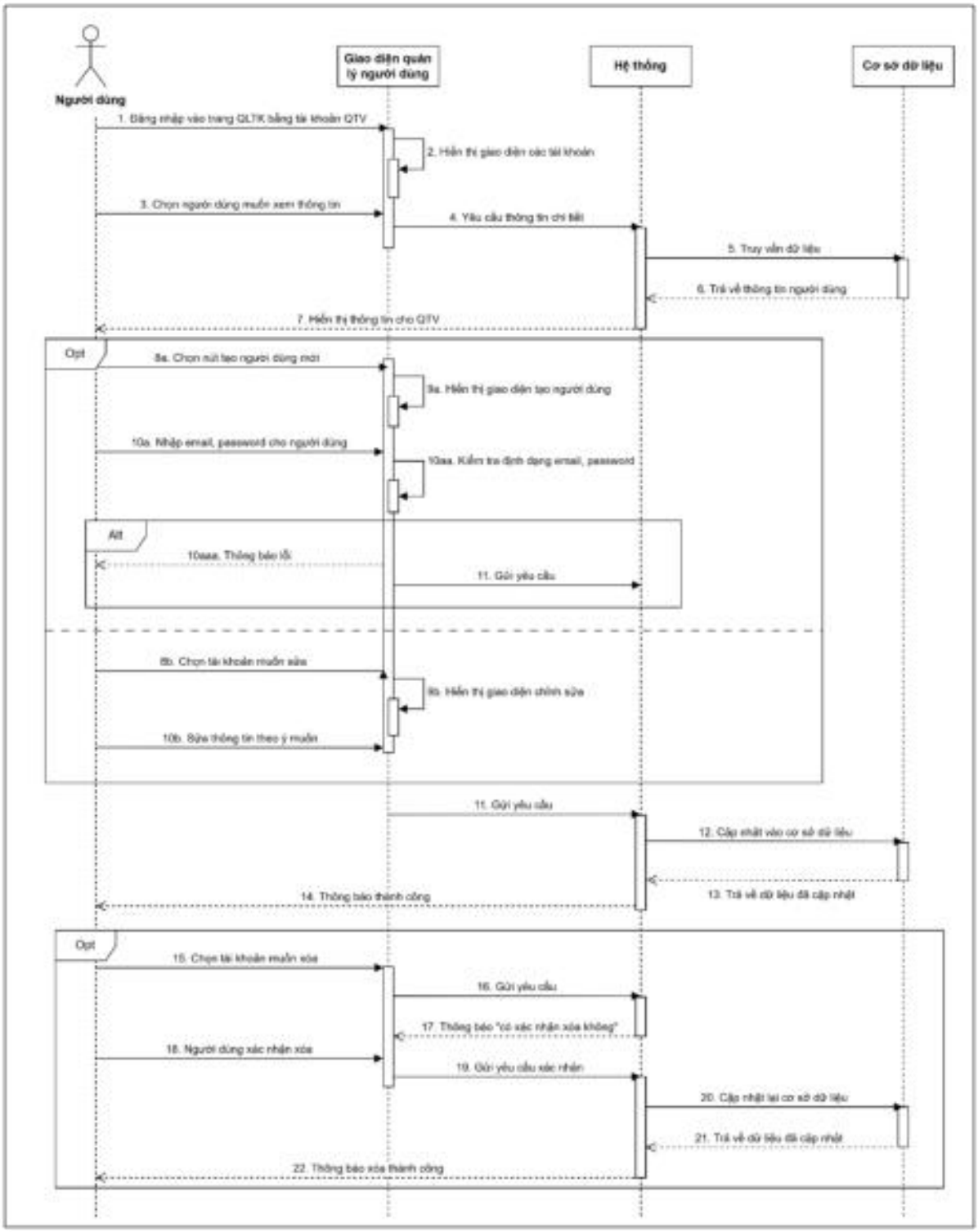

Figure 3.20. Sequence Diagram of User Account Management

- Figure 3.21 presents the sequence diagram for the movie viewing function:

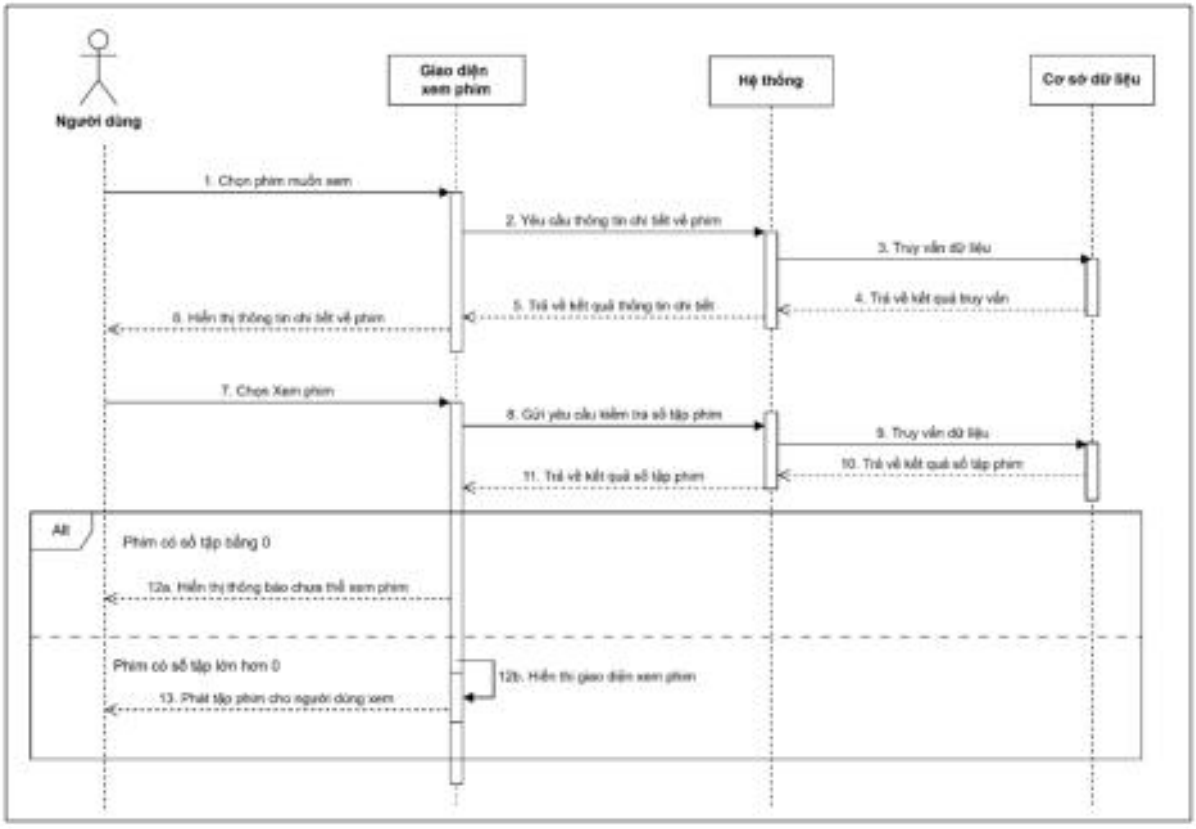

Figure 3.21. Sequence Diagram of Movie Watching Functionality

- Figure 3.22 presents the sequence diagram for the movie search function:

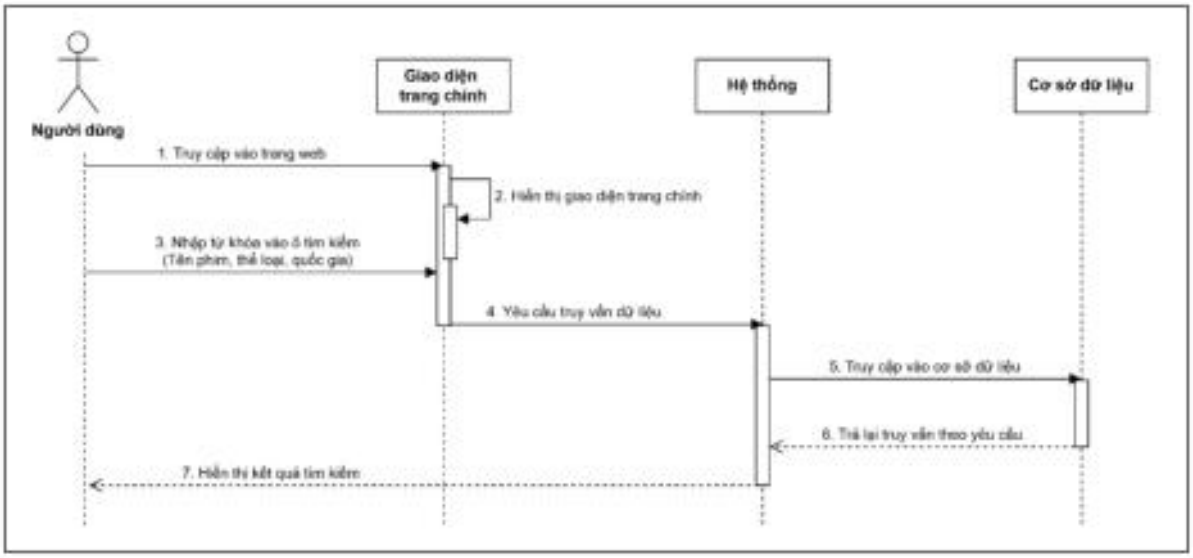

Figure 3.22. Sequence Diagram of Movie Search Functionality

- Figure 3.23 presents the sequence diagram for the movie commenting/rating function:

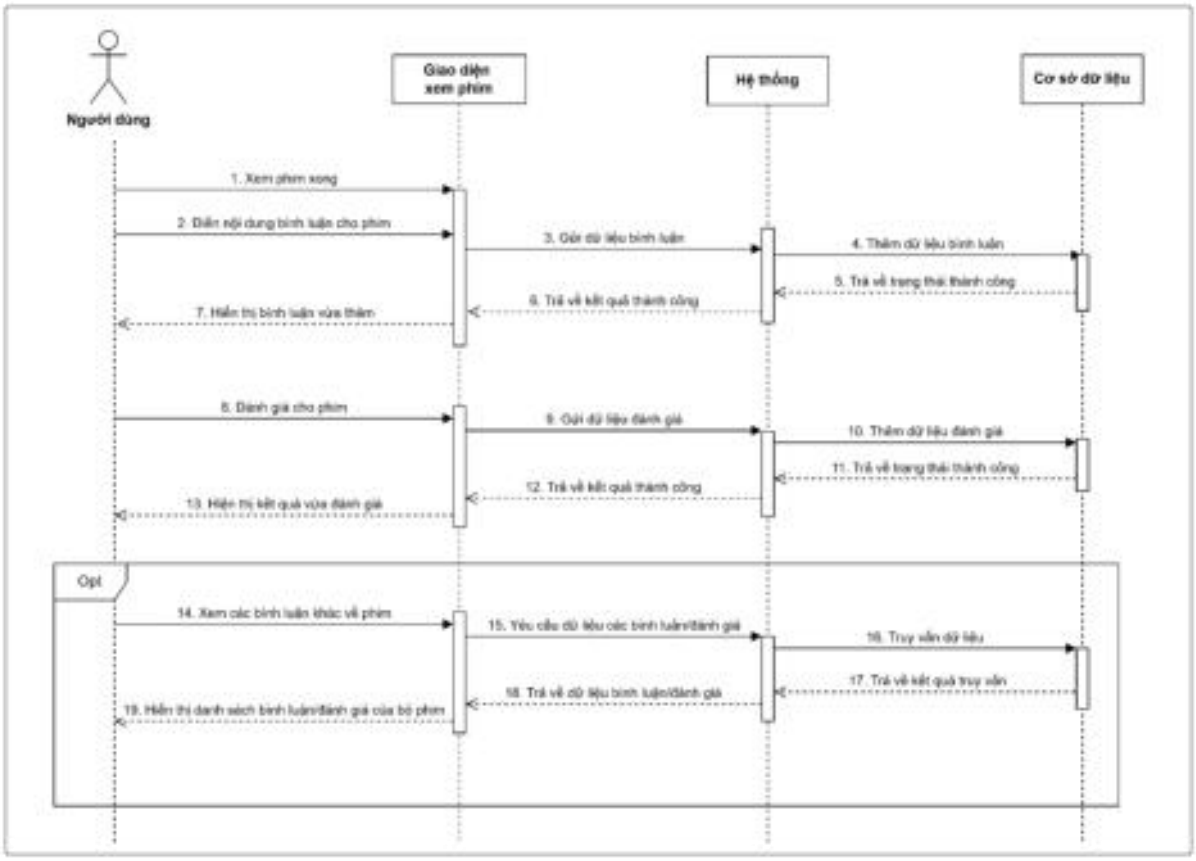

Figure 3.23. Sequence Diagram of Movie Commenting and Rating Functionality

- Figure 3.24 presents the sequence diagram for the VIP subscription function:

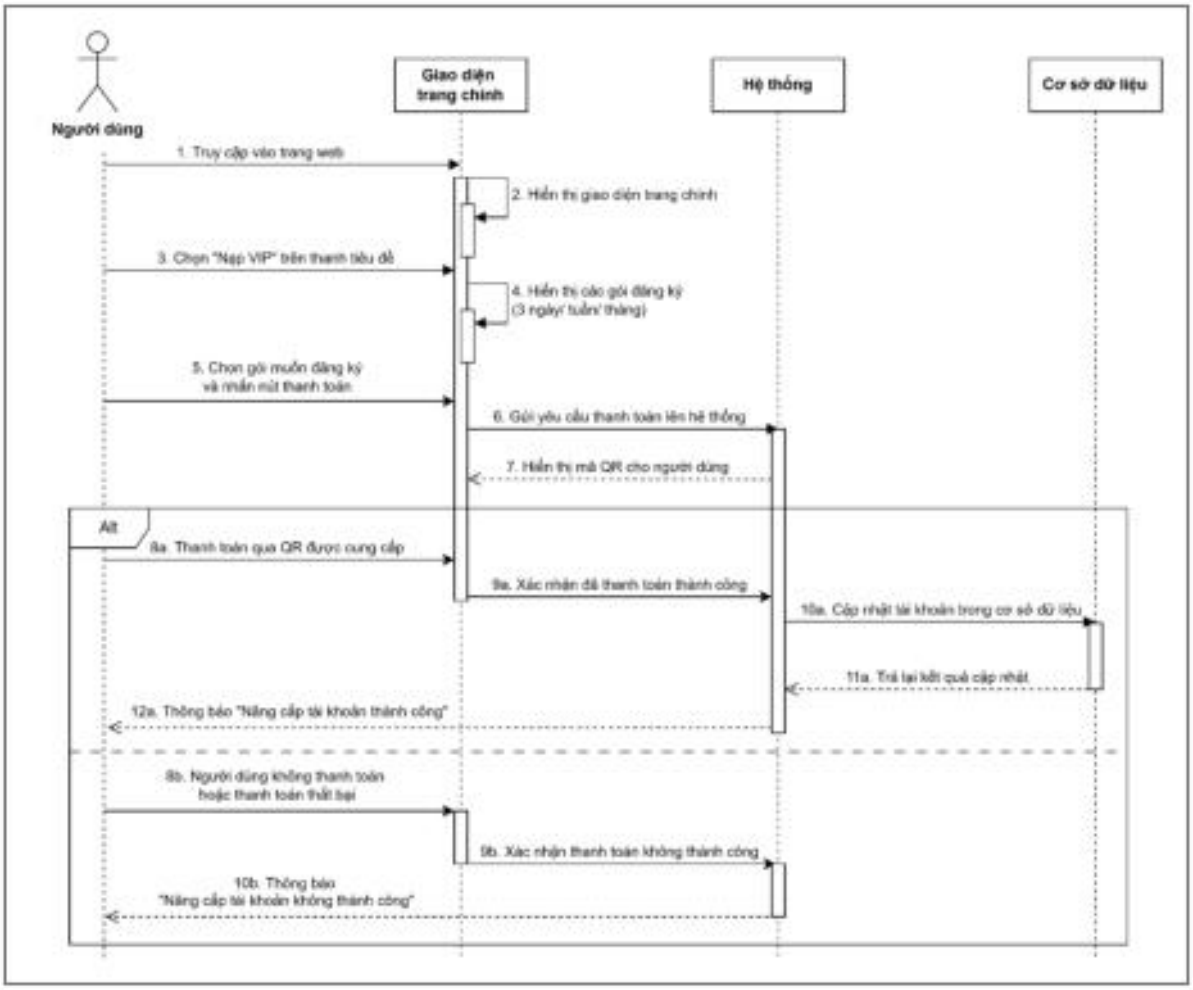

Figure 3.24. Sequence Diagram of VIP Top-up Functionality

- Figure 3.25 presents the sequence diagram for the chatbot Q&A function:

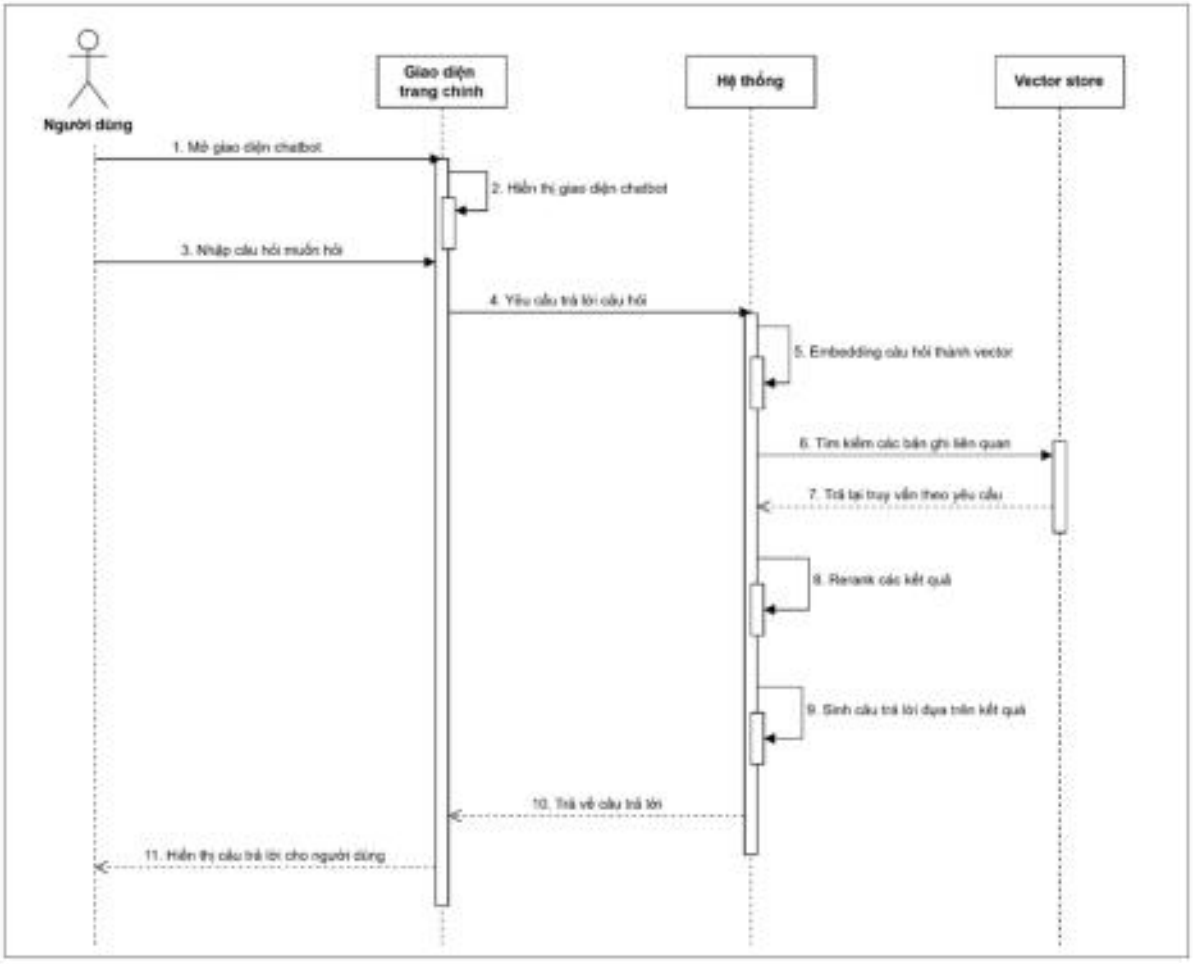

Figure 3.25. Sequence Diagram of Chatbot Q&A Functionality

3.4. Database Design

3.4.1. Relational Diagram

- Figure 3.26 presents the relational diagram of the syste:

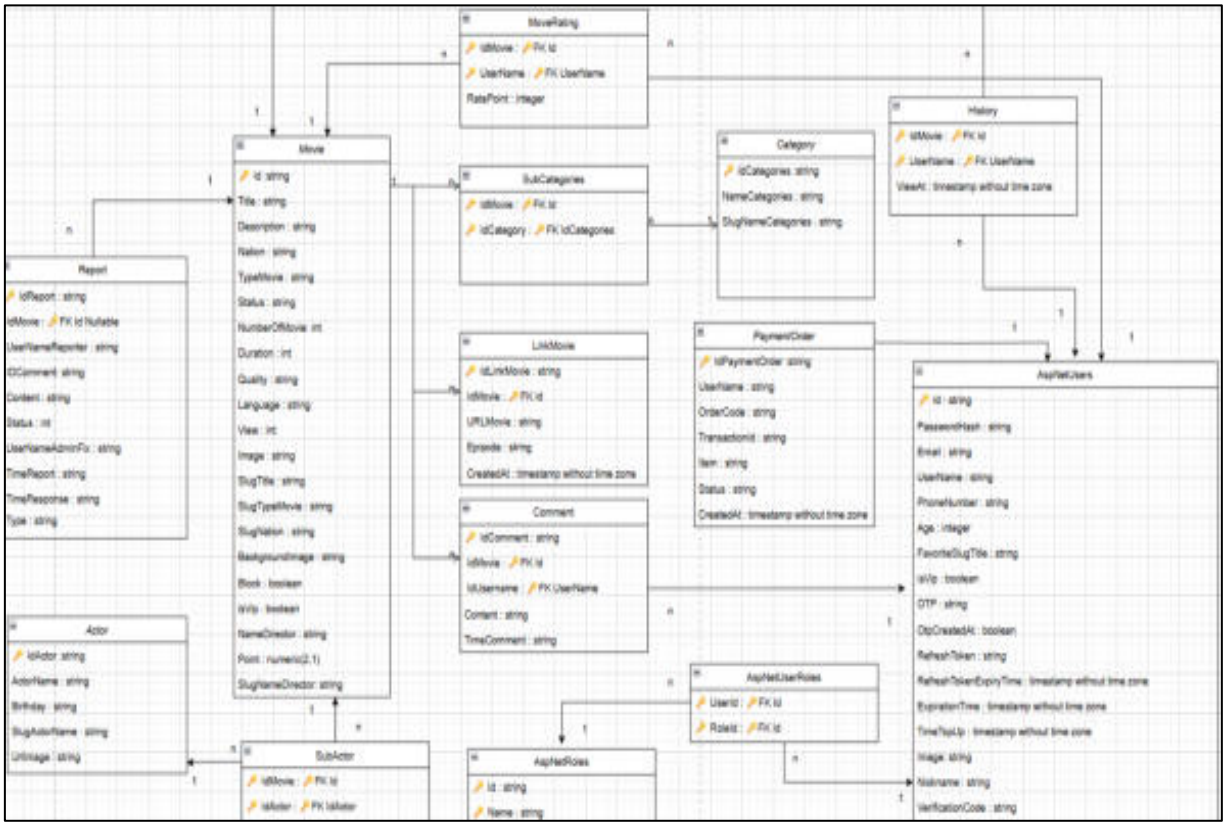

Figure 3.26. Relational Diagram

3.4.2. Converting the Entity Model into a Relational Diagram

Converting Entity Model to Relational Diagram:

1. Movie (Id, Title, Description, Nation, TypeMovie, Status, NumberOfMovie, Duration, Quality, Language, View, Image, SlugTitle, SlugTypeMovie, SlugNation, BackgroundImage, Block, isVip, NameDirector, Point, SlugNameDirector): Stores detailed movie information.
2. SubCategories (IdMovie, IdCategory): Intermediate table for many-to-many relationships between Movie and Category.
3. Category (IdCategories, NameCategories, SlugNameCategories): Stores movie genre details.
4. SubActor (IdMovie, IdActor): Intermediate table for many-to-many relationships between Movie and Actor.
5. Actor (IdActor, ActorName, Birthday, SlugActorName, UrlImage): Stores actor details.
6. Report (IdReport, IdMovie, UserNameReporter, IDComment, Content, Status, UserNameAdminFix, TimeReport, TimeResponse, Type): Stores report/feedback details.
7. LinkMovie (IdLinkMovie, IdMovie, URLMovie, Episode, CreatedAt): Stores episode details for a movie.
8. MovieRating (IdMovie, UserName, RatePoint): Stores user ratings for movies.
9. Comment (IdComment, IdMovie, IdUserName, Content, TimeComment): Stores user comments for movies.
10. History (IdMovie, UserName, ViewAt): Stores user viewing history.
11. PaymentOrder (IdPaymentOrder, UserName, OrderCode, TransactionId, Item, Status, CreatedAt): Stores user payment history.
12. AspNetUsers (Id, PasswordHash, Email, UserName, PhoneNumber, Age, FavoriteSlugTitle, isVip, OTP, OtpCreatedAt, RefreshToken, RefreshTokenExpiryTime, ExpirationTime, TimeTopUp, Image, Nickname, VerificationCode): Stores detailed user information.
13. AspNetUserRoles (UserId, RoleId): Intermediate table for many-to-many relationships between AspNetUsers and AspNetRoles.
14. AspNetRoles (Id, Name): Stores user role details.

3.4.3. Logical-Level Database Design

To facilitate the programming process, the table names and field names have been translated into English.

- Movie Table: Table 3.1 presents the structure of the Movie table, which stores information related to movies in the system.

Table 3.1. Movie Table

Field	Data Type	Allow Null?	Key	Description
Id	String	No	PK	Unique movie identifier
Title	String	No		Movie title
Description	String	No		Movie description
Nation	String	No		Country of production
TypeMovie	String	No		Movie genre
Status	String	No		Releas status
NumberOfMovie	Integer	No		Number of episodes
Duration	Integer	No		Duration per episode (in minutes)
Quality	String	No		Video quality
Language	String	No		Language (e.g. Vietsub, Dubbed)
View	Integer	No		Number of views
Image	String	No		Thumbnail image URL
SlugTitle	String	No		Slug of movie title
SlugTypeMovie	String	No		Slug of genre
SlugNation	String	No		Slug of country
BackgroundImage	String	No		Background image URL
Block	Boolean	No		Lock status of the movie

IsVip	String	No		VIP status of the movie
NameDirector	String	No		Director's name
Point	Numeric (2,1)	No		Movie rating
SlugNameDirector	String	No		Slug of director's name

- SubCategories Table: Table 3.2 presents the structure of the SubCategories table, which establishes a many-to-many relationship between movies and their associated categories or genres.

Table 3.2. SubCategories Table

Field	Data Type	Allow Null?	Key	Description
IdMovie	String	No	PK, FK Movie(Id)	Unique movie identifier
IdCategory	String	No	PK, FK Category(IdCategories)	Unique category/genre identifier

- Category Table: Table 3.3 presents the structure of the Category table, which stores information about movie genres used in the system.

Table 3.3. Category Table

Field	Data Type	Allow Null ?	Key	Description
IdCategories	String	No	PK	Unique identifier of the movie genre
NameCategories	String	No		Name of the movie genre
SlugNameCategories	String	No		Slug of the movie genre name

- SubActor Table: Table 3.4 presents the structure of the SubActor table, which defines the many-to-many relationship between movies and actors.

Table 3.4. SubActor Table

Field	Data Type	Allow Null?	Key	Description
IdMovie	String	No	PK, FK Movie(Id)	Unique movie identifier
IdActor	String	No	PK, FK Actor(IdActor)	Unique identifier of the actor

- Actor Table: Table 3.5 presents the structure of the Actor table, which stores detailed information about actors featured in the system.

Table 3.5. Actor Table

Field	Data Type	Allow Null ?	Key	Description
IdActor	String	No	PK	Unique identifier of the actor
ActorName	String	No		Actor's name
Birthday	String			Actor's date of birth
SlugActorName	String	No		Slug of the actor's name
UrlImage	String			Actor's image URL

- Report Table: Table 3.6 presents the structure of the Report table, which stores user-submitted reports or feedback on movies or comments, along with the corresponding administrative responses.

Table 3.6. Report Table

Field	Data Type	Allow Null ?	Key	Description
IdReport	String	No	PK	Unique identifier of the report/feedback
IdMovie	String			Unique identifier of the movie
UserNameReporter	String	No		Reporter’s account
Content	String	No		Report content

UserNameAdminFix	String			Admin account handling the report
Response	String			Admin's response to the reporter
TimeReport	String	No		Report submission time
Status	integer	No		Report status
IdComment	string			Unique identifier of the comment
Type	string	No		Type of report

- LinkMovie Table: Table 3.7 presents the structure of the LinkMovie table, which stores information about the video links of movie episodes.

Table 3.7. LinkMovie Table

Field	Data Type	Allow Null ?	Key	Description
IdLinkMovie	String	No	PK	Unique identifier of the episode
IdMovie	String			Unique identifier of the movie
URLMovie	String	No		Episode video URL
Episode	integer	No		Episode number
CreatedAt	Timestamp without time zone	No		Time the episode link was added

- MovieRating Table: Table 3.8 presents the structure of the MovieRating table, which stores user-submitted ratings for movies.

Table 3.8. MovieRating Table

Field	Data Type	Allow Null?	Key	Description
IdMovie	String	No	PK, FK Movie(Id)	Unique identifier of the movie genre
UserName	String	No	PK	User account rating the movie
RatePoint	String	No		User rating score

- PaymentOrder Table: Table 3.9 presents the structure of the PaymentOrder table, which stores information about user payment transactions within the system.

Table 3.9. PaymentOrder Table

Field	Data Type	Allow Null?	Key	Description
IdPaymentOrder	String	No	Primary Key	Unique identifier of the payment order
UserName	String	No	Foreign Key AspNetUsers (UserName)	Name of the user making the payment
OrderCode	String	No		Order code
TransactionId	String	No		Transaction ID
Item	String	No		Item or service being paid for
Status	String	No		Payment status
CreatedAt	DateTime	No		Payment order creation time

- AspNetUsers Table: Table 3.10 presents the structure of the AspNetUsers table, which stores essential authentication and profile information of all users in the system.

Table 3.10. AspNetUsers Table

Field	Data Type	Allow Null?	Key	Description
Id	String	No	Primary Key	Unique identifier of the user
PasswordHash	String	No		Hashed password
Email	String	No		User's email address
UserName	String	No		Username
PhoneNumber	String	Yes		User's phone number
Age	Integer	Yes		User's age
FavoriteSlugTitle	String	Yes		Slug of favorite movie title
isVip	Boolean	No		User's VIP status
OTP	String	Yes		OTP verification code
OtpCreatedAt	DateTime	Yes		OTP creation time
RefreshToken	String	Yes		Refresh token
RefreshTokenExpiryTime	DateTime	Yes		Refresh token expiration
ExpirationTime	DateTime	Yes		VIP account expiration
TimeTopUp	DateTime	Yes		Time of balance recharge
Image	String	Yes		User's avatar URL
Nickname	String	Yes		User's nickname
VerificationCode	String	Yes		Account verification code

- AspNetUserRoles Table: Table 3.11 presents the structure of the AspNetUserRoles table, which defines the many-to-many relationship between users and roles in the system.

Table 3.11. AspNetUserRoles Table

Field	Data Type	Allow Null?	Key	Description
UserId	String	No	Primary Key, Foreign Key AspNetUsers (Id)	Unique identifier of the user
RoleId	String	No	Primary Key, Foreign Key AspNetRoles (Id)	Unique identifier of the role

- AspNetRoles Table: Table 3.12 presents the structure of the AspNetRoles table, which defines the different roles that can be assigned to users in the system.

Table 3.12. AspNetRoles Table

Field	Data Type	Allow Null?	Key	Description
Id	String	No	Primary Key	Unique identifier of the role
Name	String	No		Name of the user role

3.5. Chapter Conclusion

Chapter 3 has provided a comprehensive approach to business analysis and system design for the project. The activity diagrams, use case diagrams, and sequence diagrams not only clarified the system's functionalities but also ensured that business processes from the perspectives of customers, tour guides, and administrators were effectively modeled and handled.

In addition, the database design - comprising relational tables and transformed entity models - has established a logical data structure that meets storage and information processing requirements. These design foundations serve as a crucial premise for deploying the actual system, ensuring accuracy, stability, and scalability in the future.

CHAPTER 4: DEVELOPING A MOVIE STREAMING WEBSITE INTEGRATED WITH AI CHATBOT

4.1. Development Environment and Tools

- Environment:
  - Backend: C-Sharp, ASP.NET Core Web API (.Net 8), Python, FastAPI.
  - Frontend: ReactJS version 19.0.0, Bootstrap version 5.3.5.
  - Database: PostgresDB version 16.5.
- Tools:
  - Integrated Development Environments (IDEs) such as Visual Studio 2022 and Visual Studio Code.
  - Source code management and storage using GitHub.
  - Used for API testing, including sending HTTP requests, inspecting responses, and automating test scripts using Postman.

4.2. Description of Implemented Features

4.2.1. General Functional Interfaces

- Figure 4.1 shows the Registration Page interface where users can sign up for a new account:

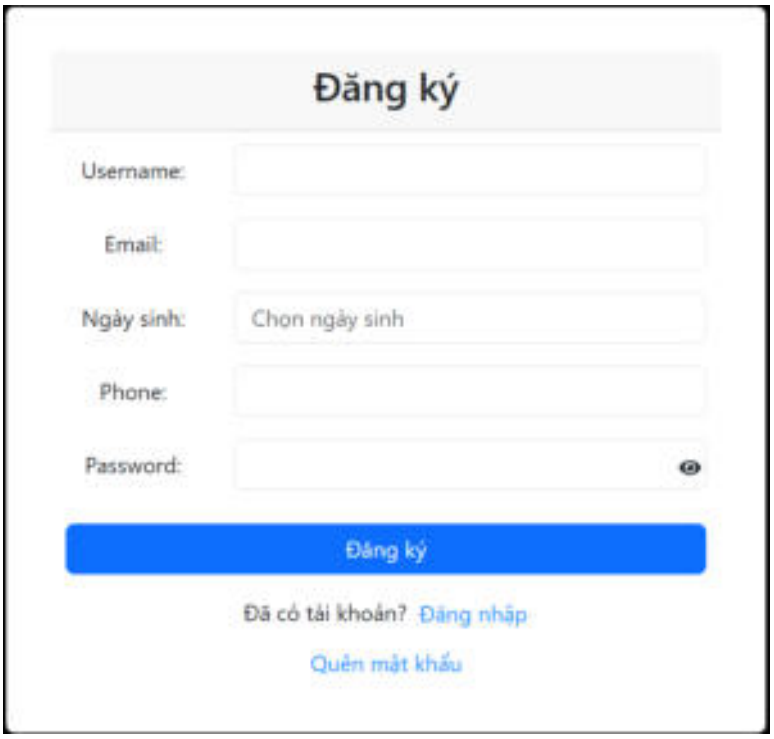

Figure 4.1. Registration Interface

- Figure 4.2 shows the Login Page interface for user authentication:

Figure 4.2. Login Page

- Figure 4.3 shows the Forgot Password popup, allowing users to reset their password by providing their registered email address:

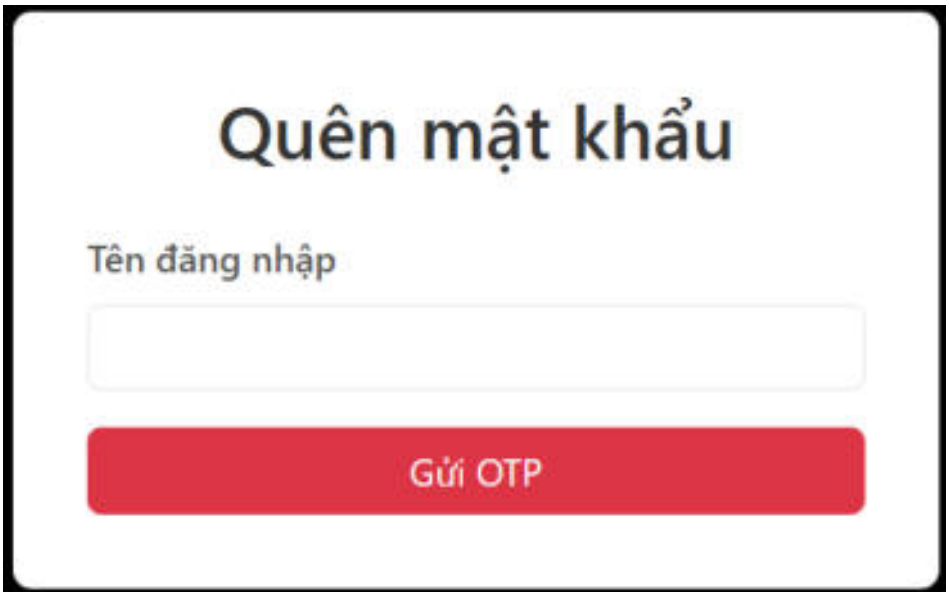

Figure 4.3. Forgot Password Interface

- Figure 4.4 displays the main screen interface shown to users when they first access the website:

Figure 4.4. Main Screen Interface When Accessing the Website

4.2.2. User Interface for Movie Viewers

- This is an interface that allows users to search for movies based on criteria such as title, genre and country of production.

+ Figure 4.5 displays the result interface for searching a specific movie title:

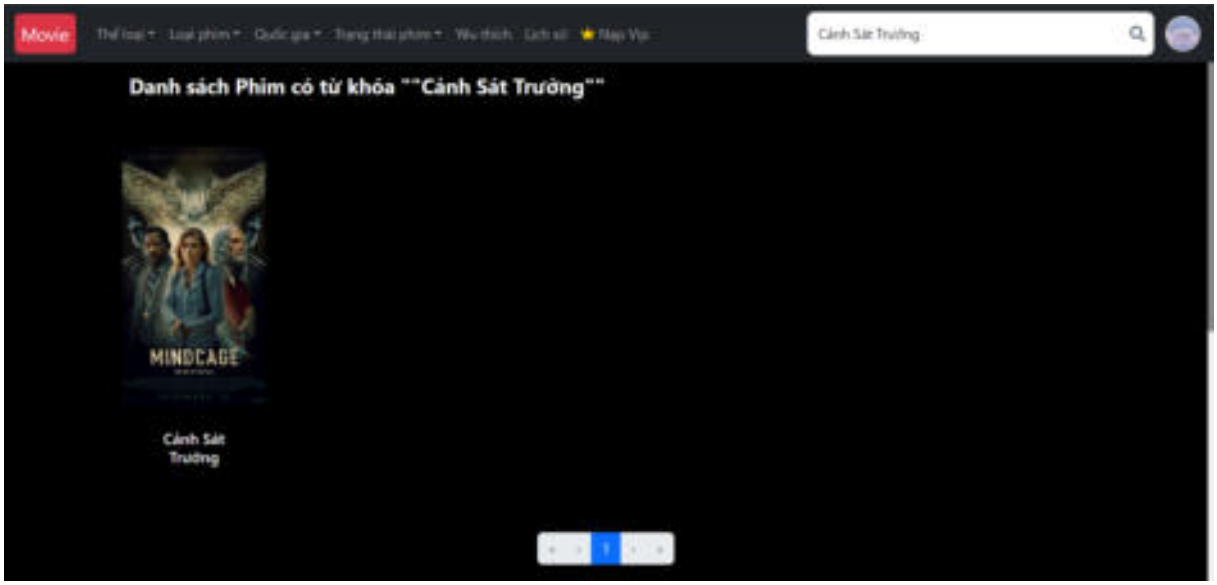

Figure 4.5. Movie Title Search Screen Interface

+ Figure 4.6 displays the search results for movies in the "Science Fiction" genre:

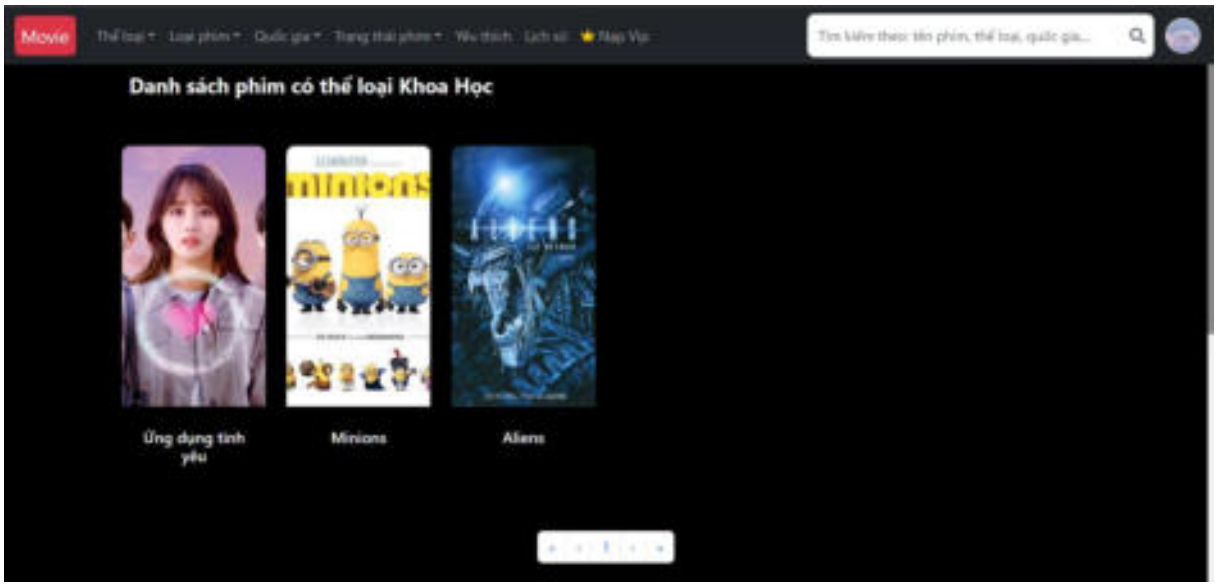

Figure 4.6. Genre-Based Movie Search Screen Interface

- Figure 4.7 shows the interface for managing user personal information:

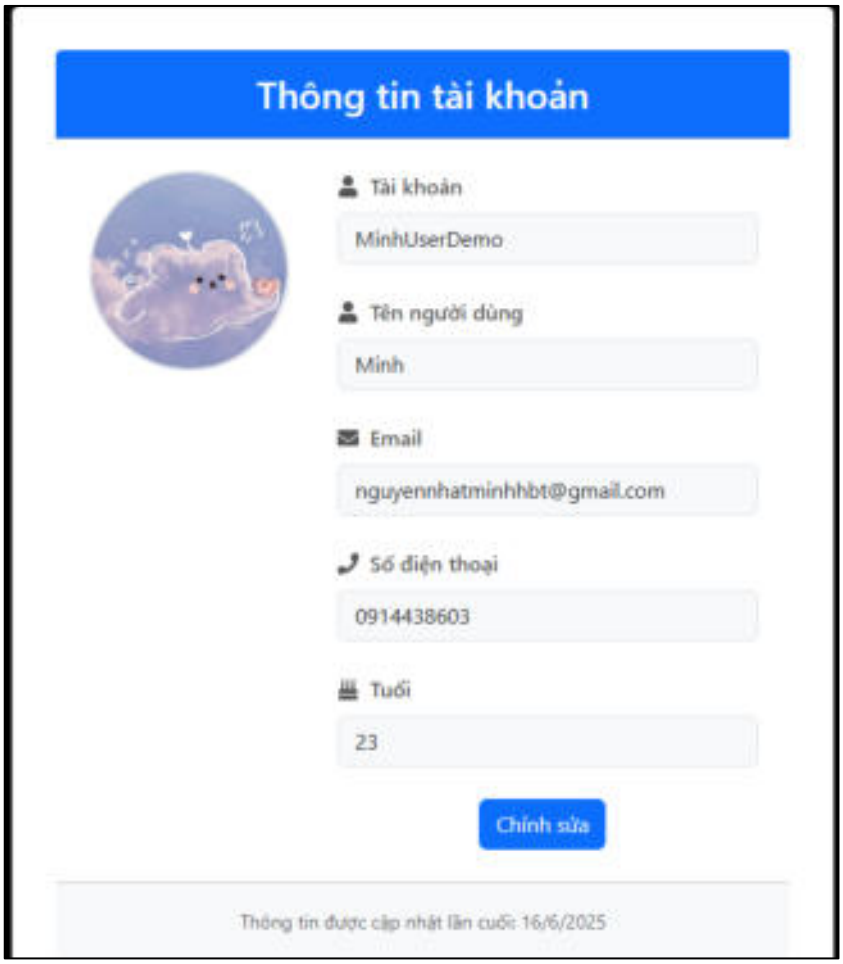

Figure 4.7. Personal Information Management Screen Interface

- Figure 4.8 shows the interface for viewing detailed information about a selected movie, including title, description, genre, duration, and rating:

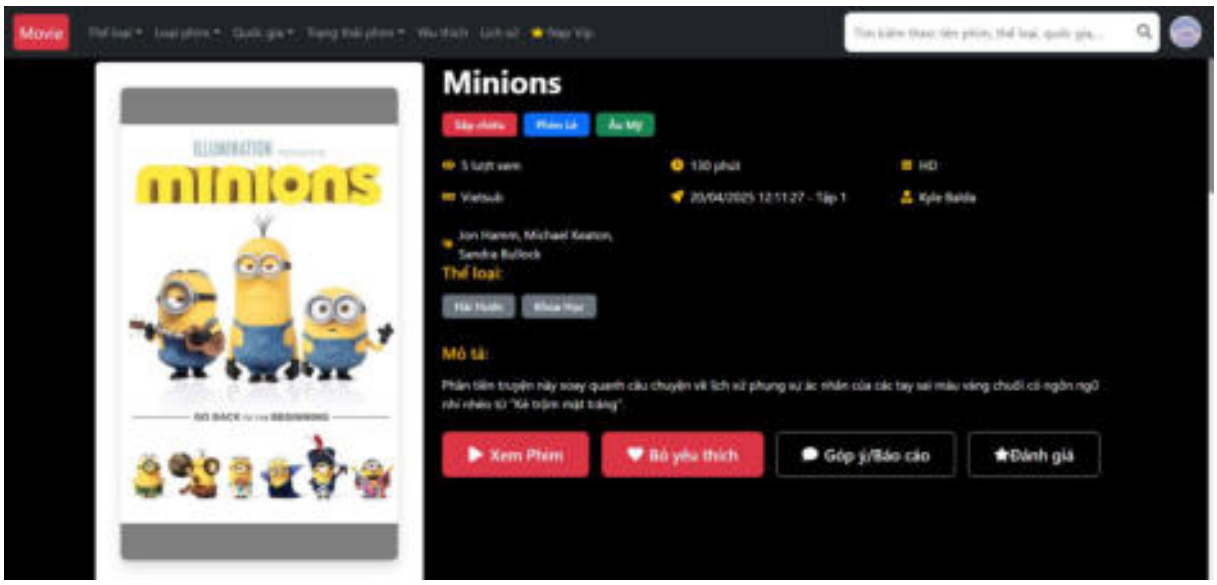

Figure 4.8. Movie Information Viewing Screen Interface

- Figure 4.9 shows the interface for viewing the user's list of favorite movies, allowing quick access to previously marked content:

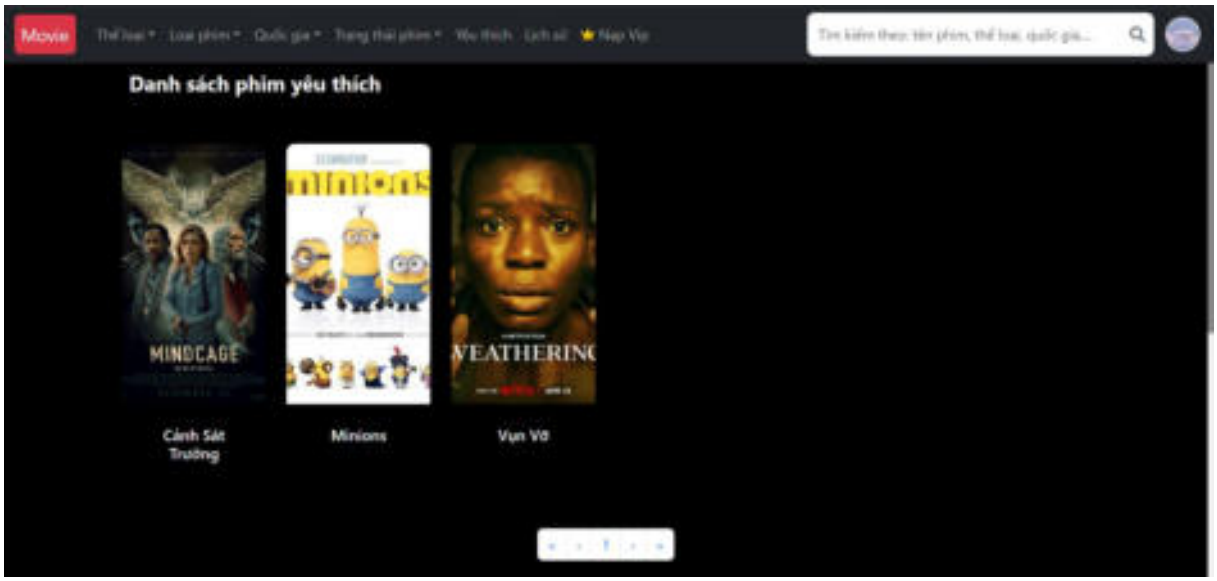

Figure 4.9. Favorite Movies List Viewing Screen Interface

- Figure 4.10 shows the interface for viewing the user's watch history, displaying a list of previously watched movies:

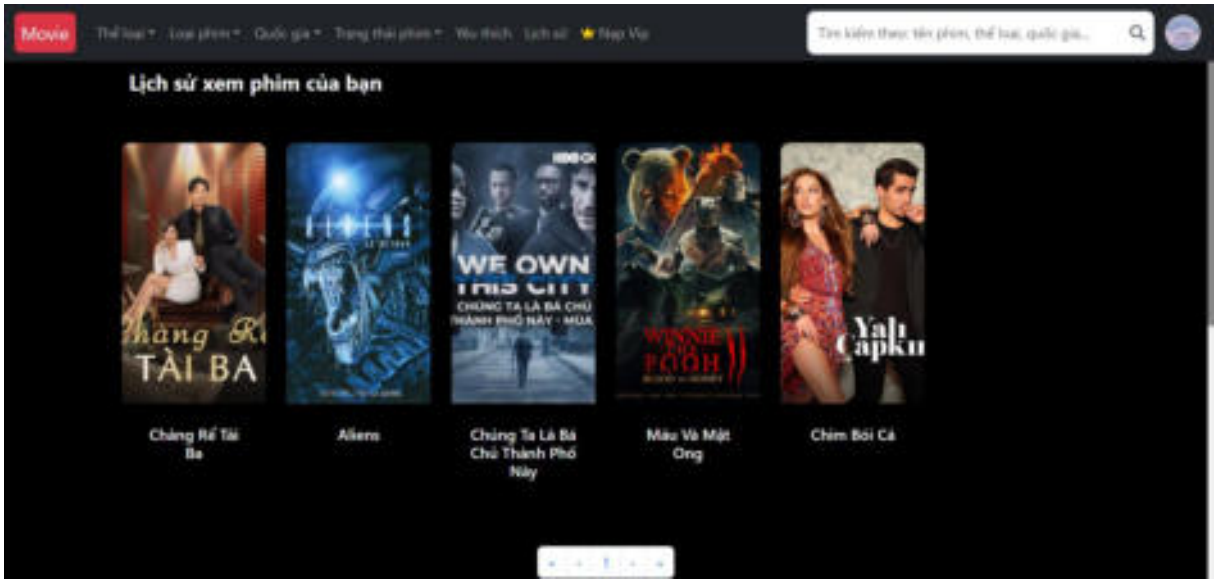

Figure 4.10. Watch History Viewing Screen Interface

- Figure 4.11 and Figure 4.12 show the VIP subscription and payment interfaces, where users can select a VIP package and proceed with the payment process to upgrade their account:

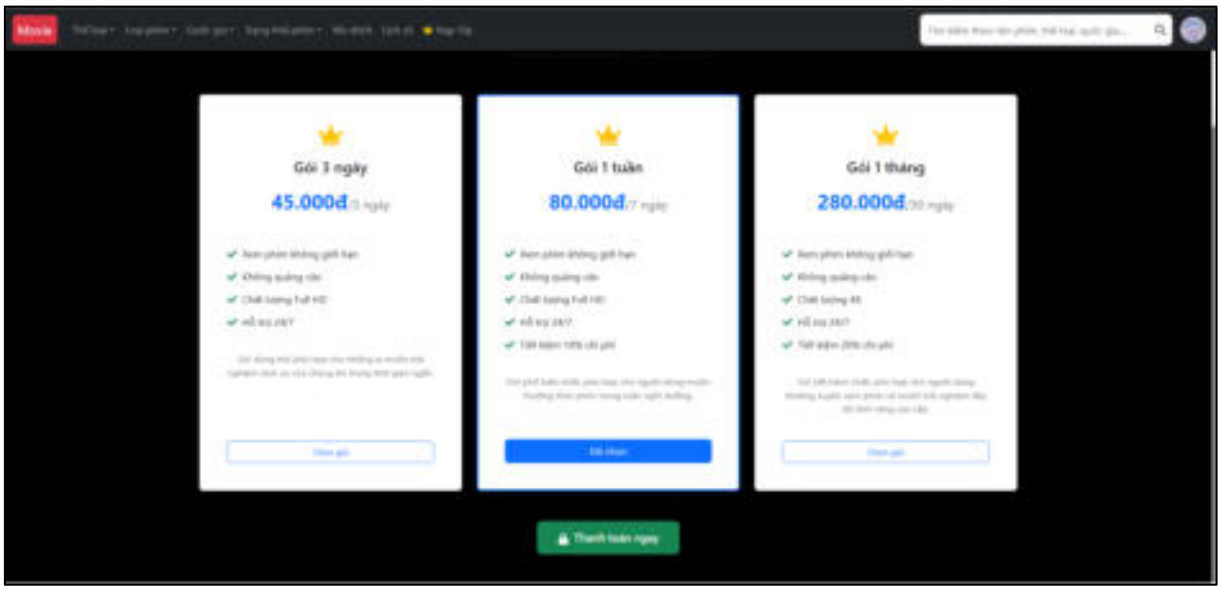

Figure 4.11. VIP Subscription Registration Screen Interface

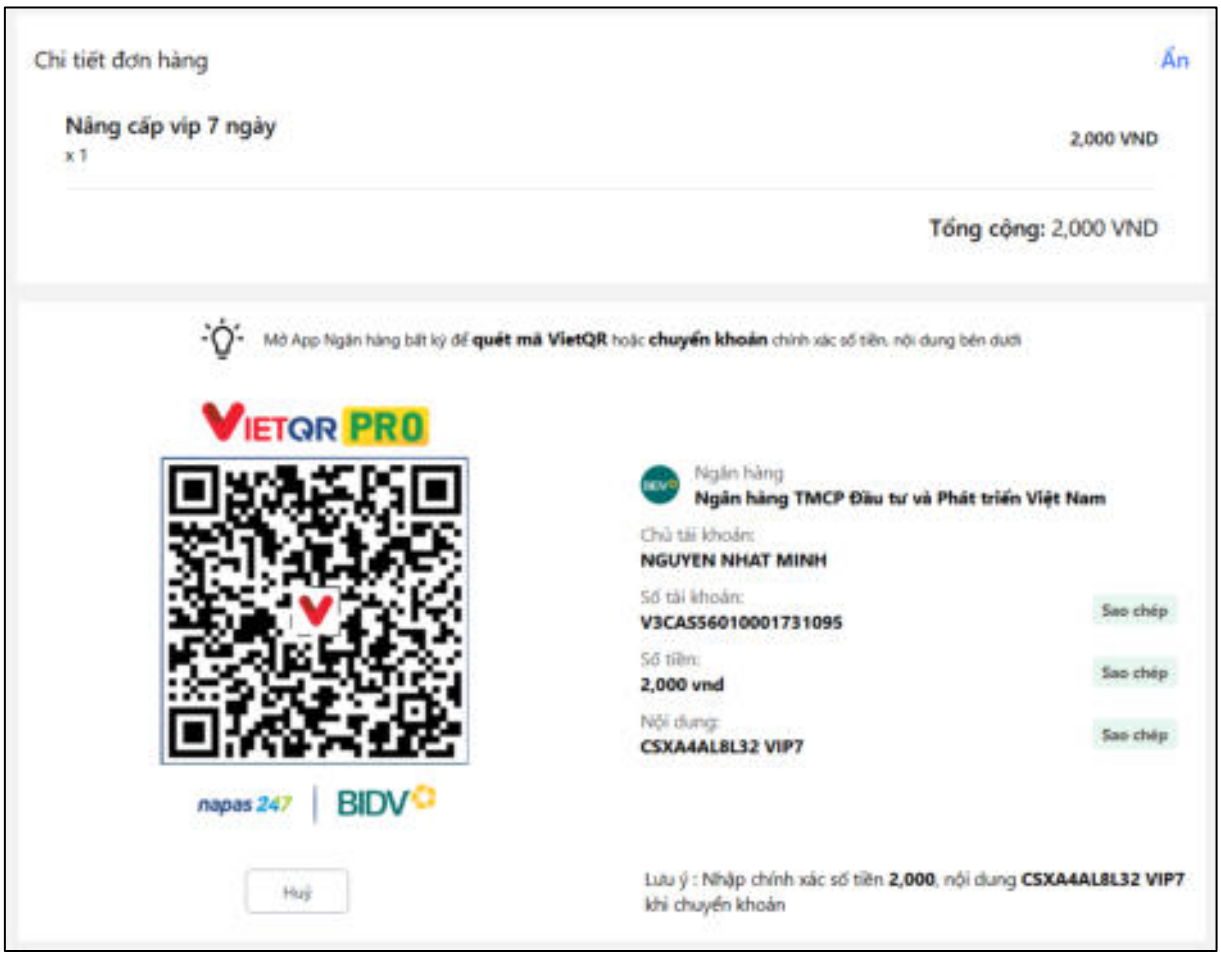

Figure 4.12. Payment Screen Interface

- Figure 4.13 shows the interface for viewing the user's payment history, including details such as transaction ID, status, and date:

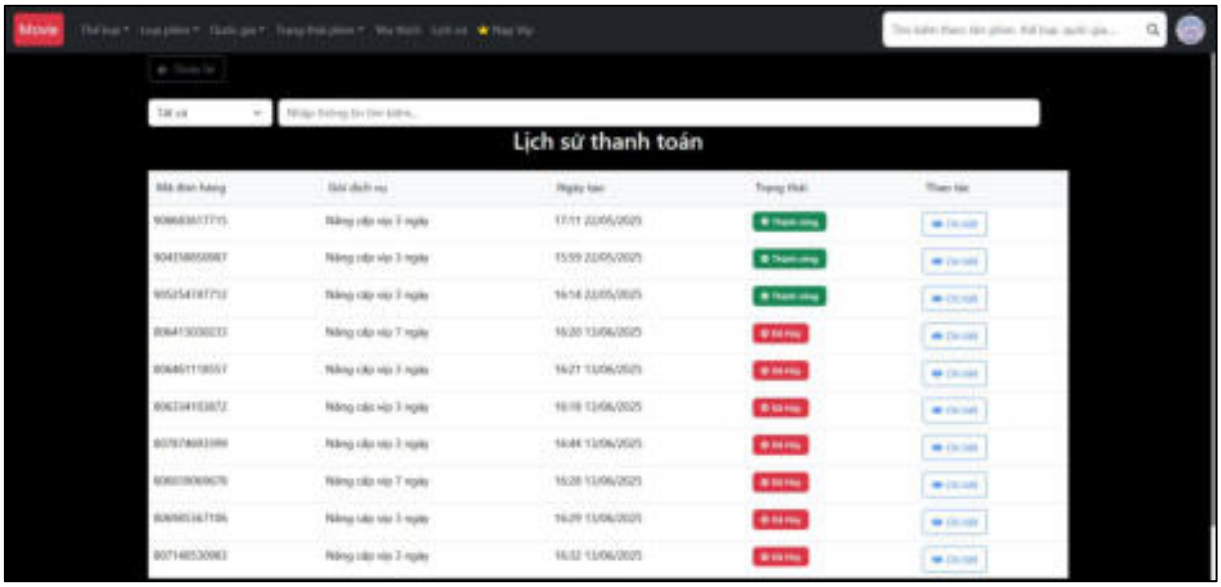

Figure 4.13. Payment History Screen Interface

4.2.3. Admin Functional Interface

- The administrator has full access to all user functionalities, in addition to exclusive management features.
- Dashboard Page includes Figure 4.14, Figure 4.15 and Figure 4.16.
- Figure 4.14 shows the total number of movies, the total number of genres, the total number of comments and suggestions. There are also viewing statistics by date, month, and year:

Figure 4.14. Statistical Dashboard Screen Interface (1)

- Figure 4.14 shows the number of movies by genre and movie status:

Figure 4.15. Statistical Dashboard Screen Interface (2)

- Figure 4.14 sorts movies by views or user ratings:

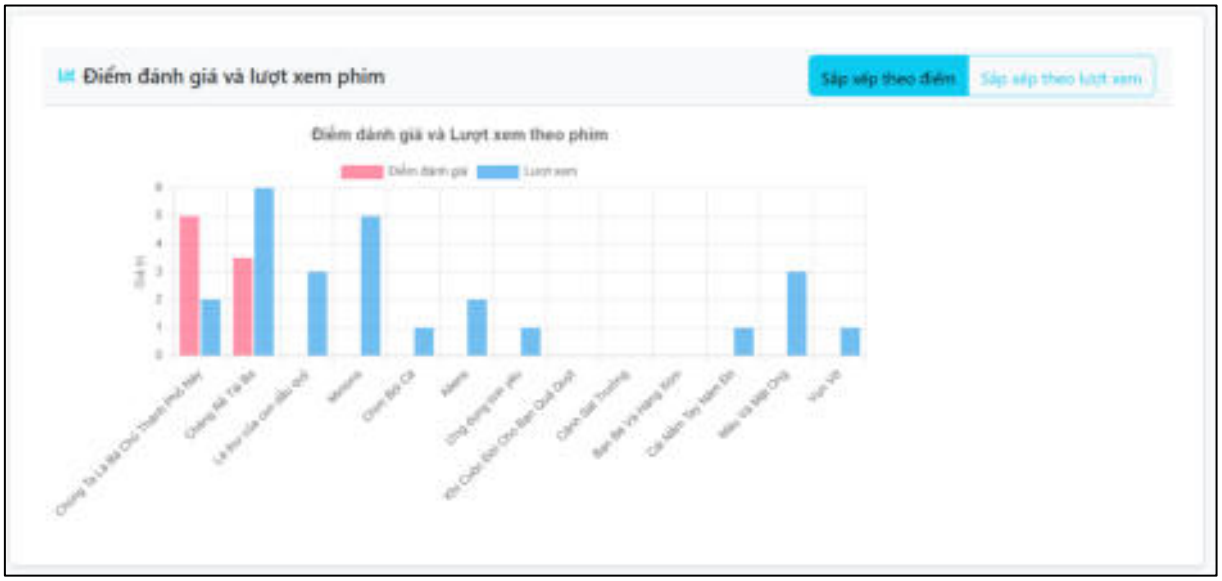

Figure 4.16. Statistical Dashboard Screen Interface (3)

- Figure 4.17 shows the interface for managing movie information, including title, description, genre, and more:

Figure 4.17. Movie Management Screen Interface

- Figure 4.18 shows the Episode Management interface, allowing administrators to add, update, or delete episodes, including details such as episode number, video URL, and upload time:

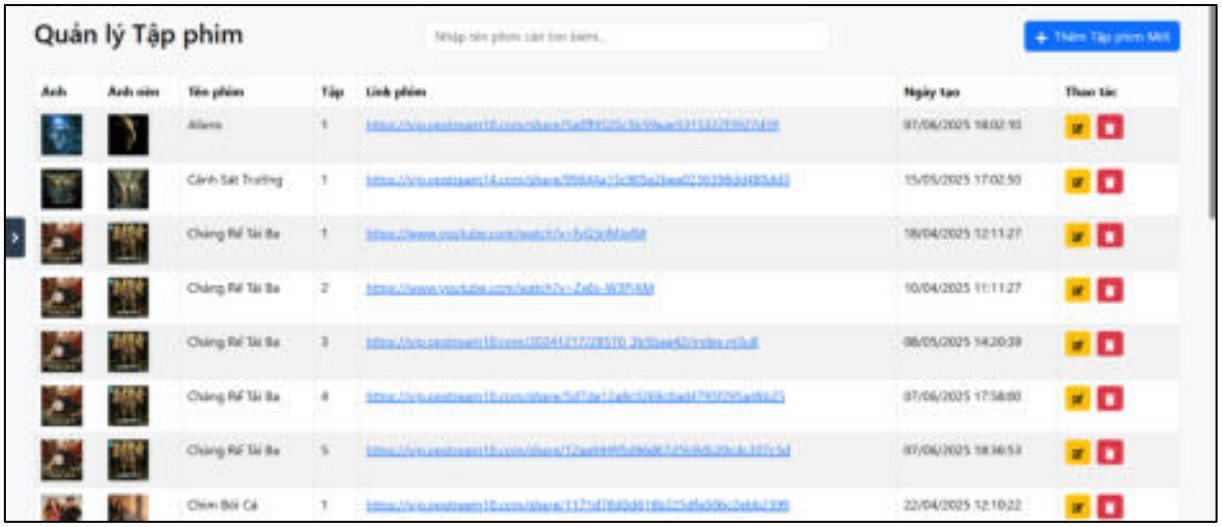

Figure 4.18. Episode Management Screen Interface

- Figure 4.19 shows the Genre Management interface, where administrators can perform CRUD operations (Create, Read, Update, Delete) on movie genres:

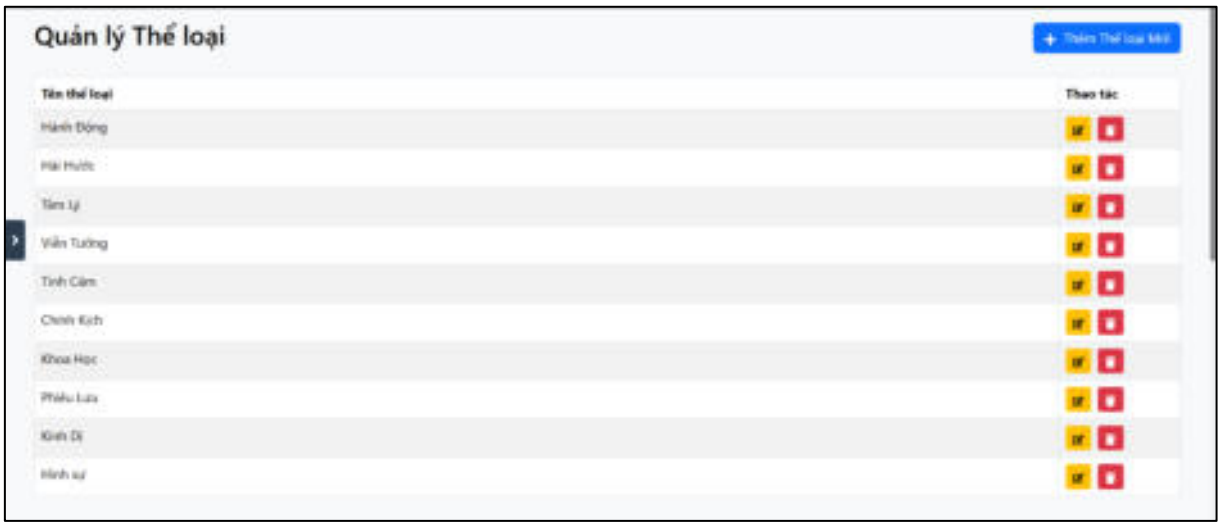

Figure 4.19. Genre Management Screen Interface

- Figure 4.20 shows the User Account Management interface, where administrators can view, edit, and manage user information:

Figure 4.20. User Account Management Screen Interface

- Figure 4.21 shows the Comment Management interface, where administrators can view, filter, and manage user comments:

Figure 4.21. Comment Management Screen Interface

- Figure 4.22, Figure 4.23, and Figure 4.24 show the user report management interface, where admins review, respond to, and track report statuses:

Figure 4.22. System Report Management Screen Interface

Figure 4.23. Comment Report Management Screen Interface

Figure 4.24. Movie Report Management Screen Interface

4.3. Software Testing

4.3.1. Definition and Purpose Of Software Testing

- Software testing is a systematic process of evaluating a system or its components with the objective of determining whether it meets the specified requirements and identifying any deviations between the actual implementation and the intended specifications. In essence, testing aims to uncover defects, bugs, errors, or omissions in requirements, as well as inconsistencies between the documented requirements and the real-world expectations [18].

4.3.2. Type Of Software Testing

- Software testing can be categorized into several types based on purpose, scope, and the aspect of the system being tested. Below are four fundamental classifications commonly used in software engineering:
  - Testing of function (Functional testing).
  - Testing of software product characteristics (Non-Functional testing).
  - Testing of software structure/architecture (Structural testing).
  - Testing related to changes (Confirmation and regression testing).

4.3.3. Postman

- Postman is a popular collaboration platform for API development, widely used by developers and testers to design, test, and document APIs efficiently. It provides an intuitive user interface that allows users to send HTTP requests (GET, POST, PUT, DELETE, etc.), inspect responses, manage environments, and create automated test scripts without writing complex code.
- Postman is a tool commonly used for API testing, which is considered a part of integration testing within the broader category of functional testing. It helps verify that different modules or services interact correctly through APIs, ensuring that the system behaves as expected based on the functional requirements.
- Moreover, Postman supports test automation through its scripting capabilities using JavaScript, making it suitable for regression testing as well. It also allows users to organize requests into collections, share workspaces across teams, and integrate with CI/CD pipelines for continuous testing.

4.3.4. Postman Testing Workflow

- Set up Environment:
  - Base URL: http://localhost:5285 (local environment).
  - Header: Content-Type: application/json.

- **Endpoint**: GET/api/movie/GetMovieById/{id}: Table 4.1 presents the test cases for the GetMovieById endpoint, verifying the system's responses to valid, missing, and invalid movie ID inputs.

Table 4.1. Test Cases for GetMovieById Endpoint

Test Case ID	Description	Input	Expected Output	Actual Output
TC01	Retrieve the movie with a valid id	id = 01dcd72c-03b7-4b24-b23b-b48b5f2ac1dc	HTTP Status Code 200, a movie object with properties such as Title, NameCategories, ...	In [Figure 4.25]
TC02	Retrieve the movie with none id		HTTP Status Code 404 Not Found	In [Figure 4.26]
TC03	Retrieve the movie with invalid id	id = 123	HTTP Status Code 200 with no movie found	In [Figure 4.27]

- The figure 4.25 shows the response returned by the GetMovieById endpoint when a valid movie ID is provided. It includes detailed information such as the movie's title, description, genre, nation, director, cast, video URL, and related images. The HTTP status code 200 OK confirms a successful retrieval.

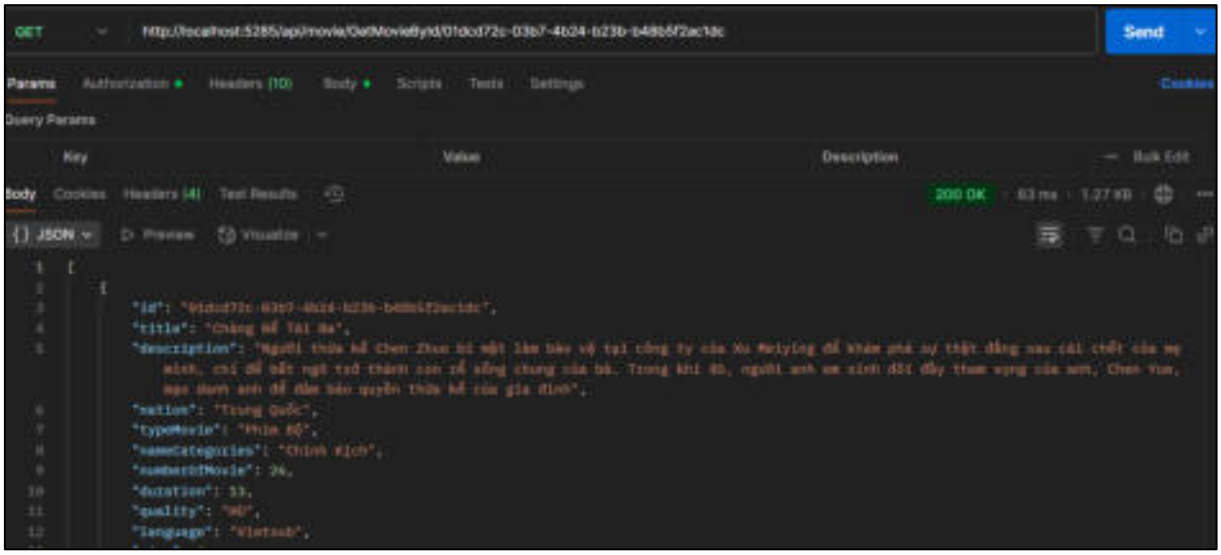

Figure 4.25. JSON response for a valid movie ID request

- The figure 4.26 displays the response from the GetMovieById endpoint when no ID parameter is included in the request. The server returns an HTTP status code 404 Not Found, indicating that the requested resource could not be located due to the missing movie identifier.

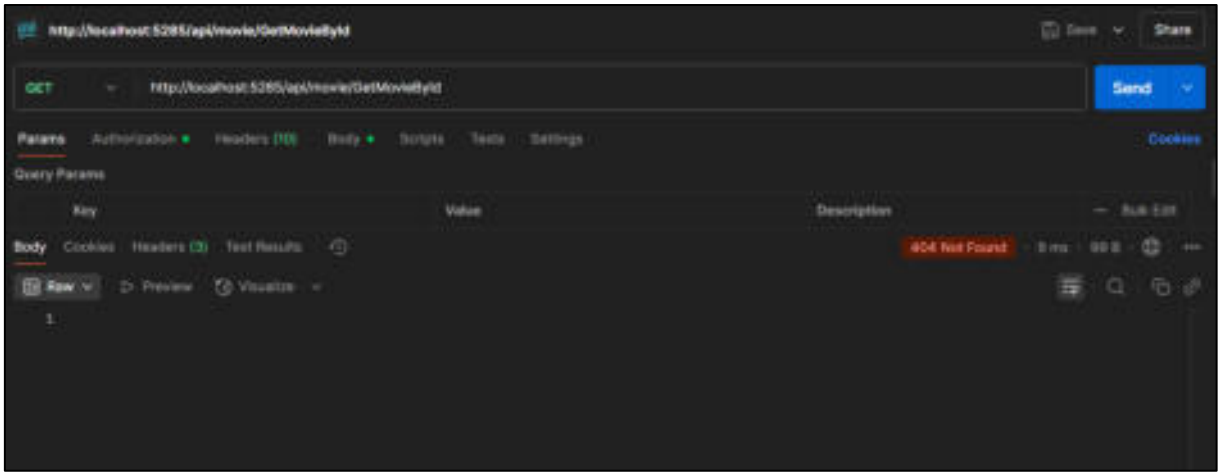

Figure 4.26. JSON response when no movie ID is provided

- The figure 4.27 shows the response from the GetMovieById endpoint when an invalid movie ID (123) is provided. The server returns an HTTP status code 200 OK with an empty JSON array, indicating that no matching movie was found for the given ID.

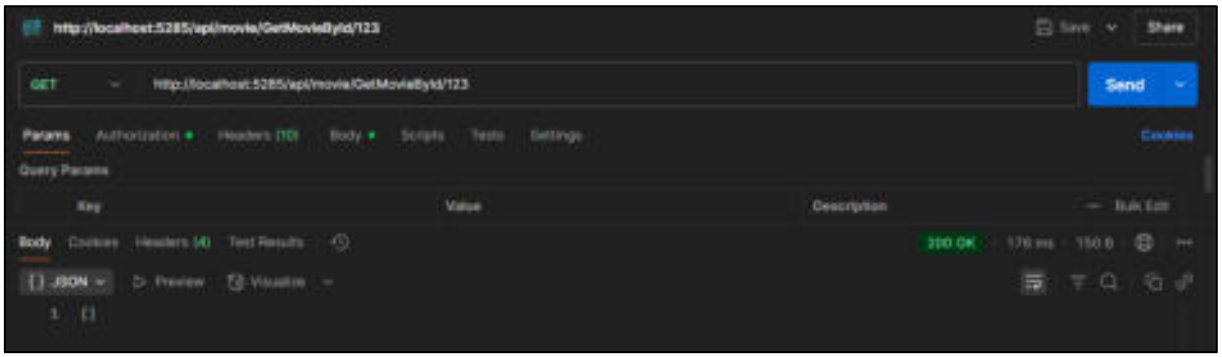

Figure 4.27. JSON response for an invalid movie ID

- **Endpoint**: GET/api/movie/FavoriteMovies: Table 4.2 presents the test cases for the FavoriteMovies endpoint, which validate the system's behavior with different authentication states, pagination parameters, and user-specific conditions.

Table 4.2. Test Cases for FavoriteMovies Endpoint

Test Case ID	Description	URL	Expected Output	Actual Output
TC01	Missing authentication token	http://localhost:5285/api/movie/FavoriteMovies?pageNumber=1&pageSize=36	HTTP Status Code 401, Unauthorized	In [Figure 4.28]
TC02	Retrieve the list of favorite movies with valid pagination parameters and valid authentication token.	http://localhost:5285/api/movie/FavoriteMovies?pageNumber=1&pageSize=36	HTTP Status Code 200, an array of movies with properties such as id, title, nameCategories, urlMovie, etc., along with pagination information including totalRecords, totalPages, pageNumber, and pageSize	In [Figure 4.29]
TC03	User with no favorite movies.	http://localhost:5285/api/movie/FavoriteMovies?pageNumber=1&pageSize=36	HTTP Status Code 200 with message : Danh sách yêu thích trống	In [Figure 4.30]
TC04	Missing pageNumber and pageSize parameters.	http://localhost:5285/api/movie/FavoriteMovies	HTTP Status Code 200, an array of movies with properties such as id, title, nameCategories, urlMovie, etc., along with pagination	In [Figure 4.31]

			information including totalRecords, totalPages, pageNumber, and pageSize	
TC05	Invalid pageNumber (pageNumber = 0)	http://localhost:5285/api/movie/FavoriteMovies?pageNumber=0	HTTP Status Code 400 with message : Số trang không hợp lệ	In [Figure 4.32]
TC06	Invalid pageSize(pageSize =0)	http://localhost:5285/api/movie/FavoriteMovies?pageSize=0	HTTP Status Code 400 with message : pageSize không hợp lệ	In [Figure 4.33]

- The figure 4.28 displays the JSON response from the FavoriteMovies endpoint when a request is made without providing an authentication token. The server returns an HTTP status code 401 Unauthorized, indicating that access is denied due to missing authentication credentials.

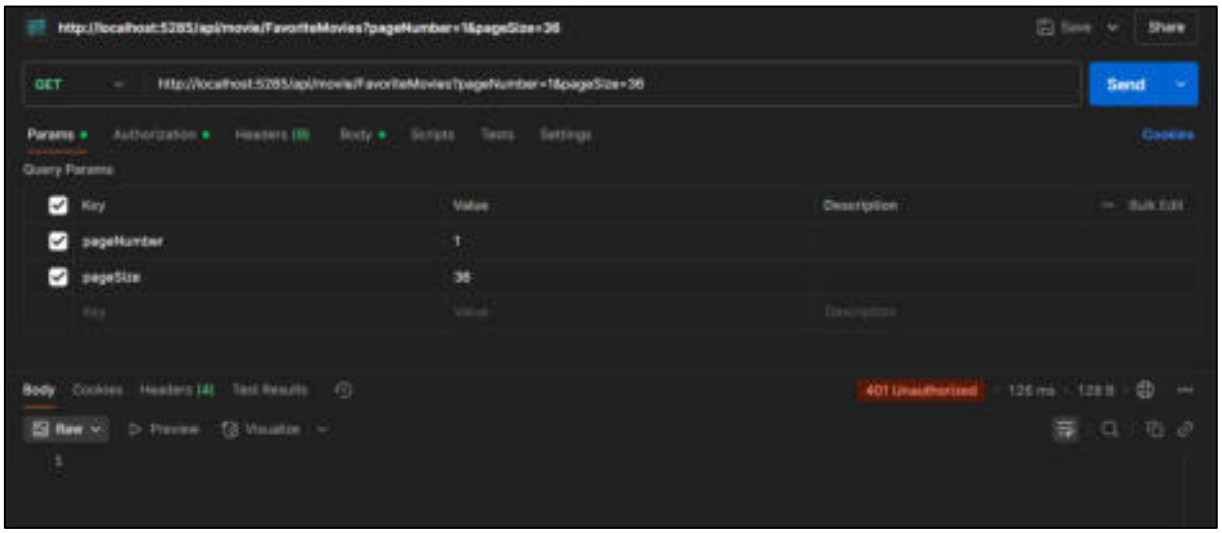

Figure 4.28. JSON response when authentication token is missing

- The figure 4.29 shows a 200 OK response with a list of favorite movies and pagination info when valid authentication and parameters are provided.

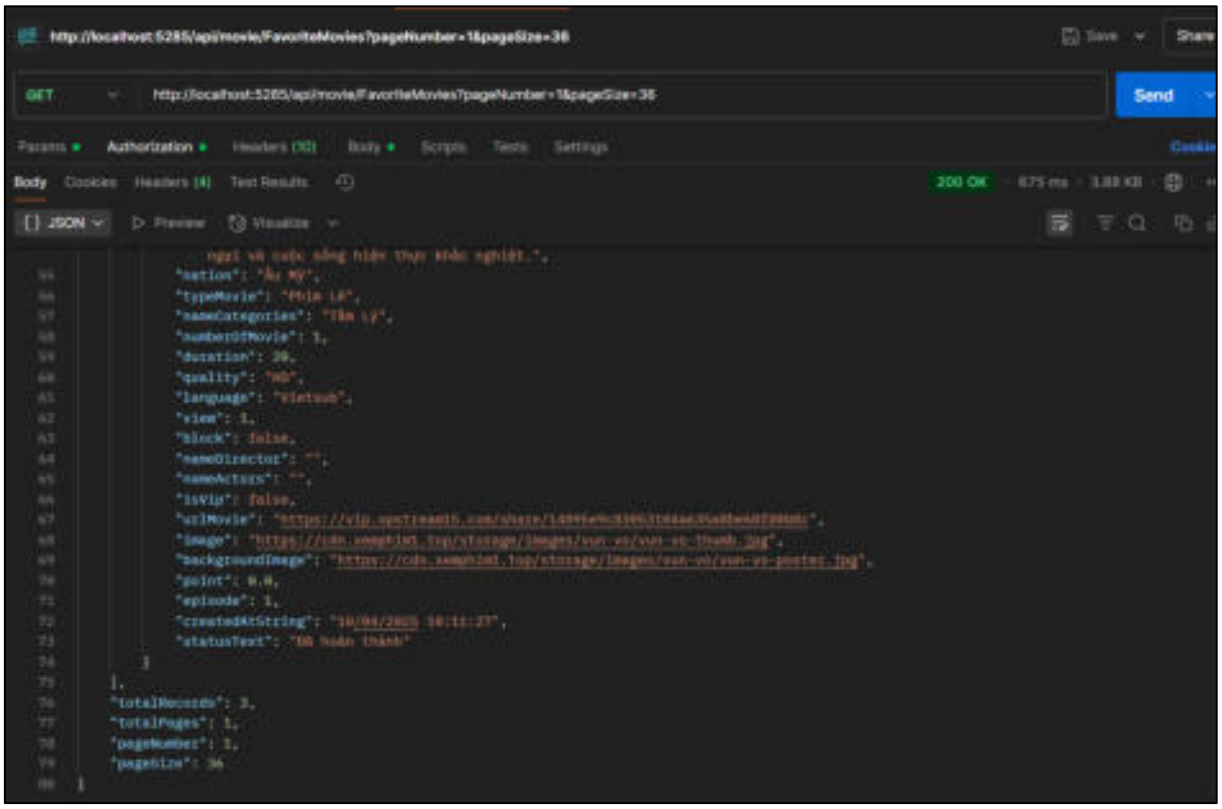

Figure 4.29. JSON response with valid authentication and pagination

- The figure 4.30 shows a 200 OK response indicating that the user has no favorite movies.

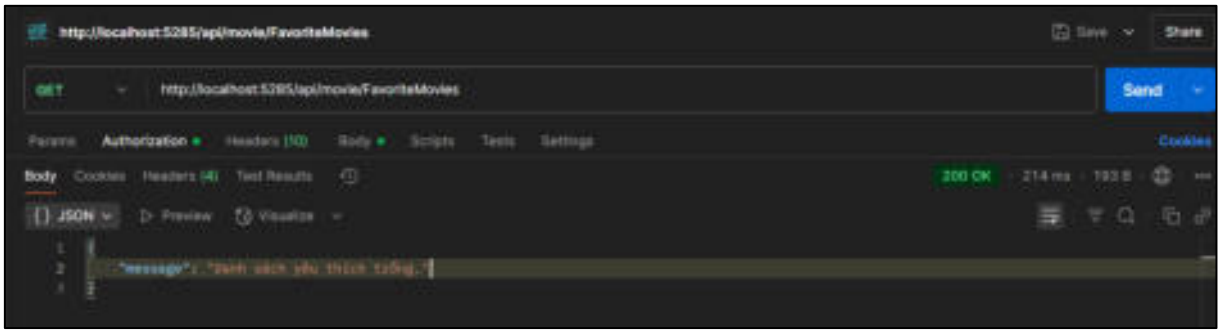

Figure 4.30. JSON response when user has no favorite movies

- The figure 4.31 shows the JSON response returned when a GET request is sent to the FavoriteMovies API. It responds with status code 200 OK and includes movie details such as name, type, nation, image, and pagination information.

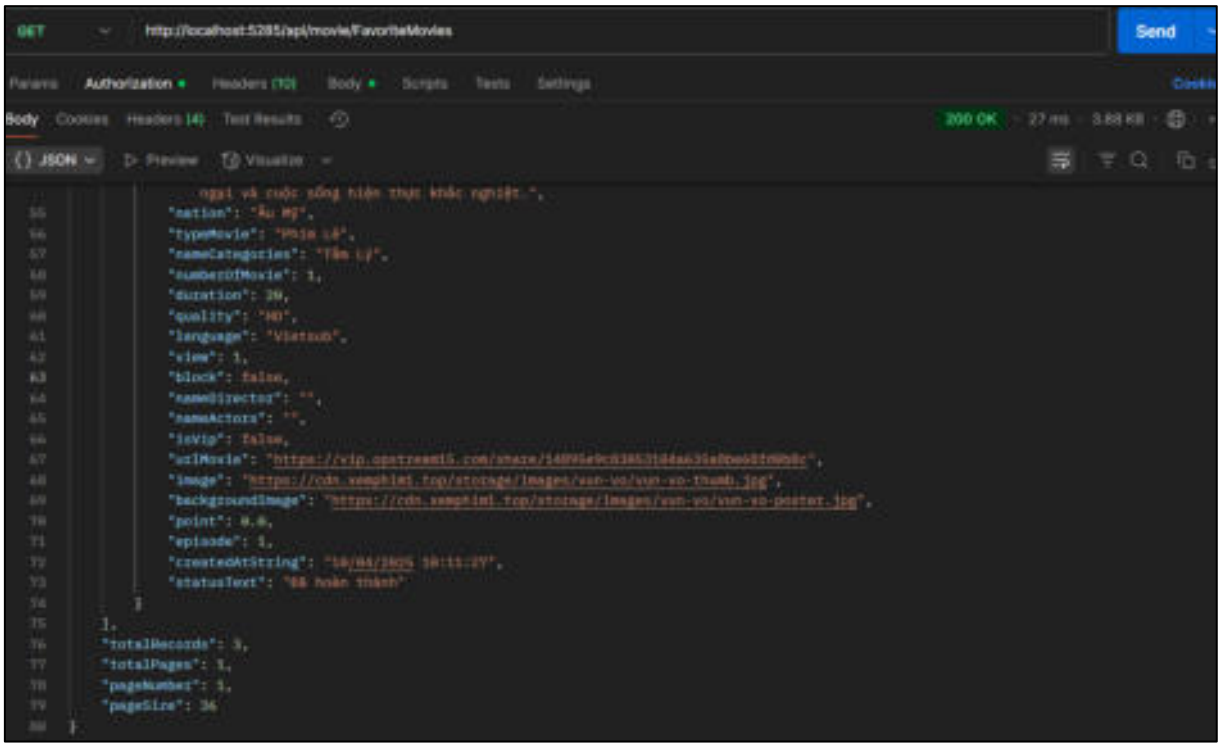

Figure 4.31. JSON response when successfully requesting the FavoriteMovies API

- The figure 4.32 shows the JSON response returned when a GET request is sent to the FavoriteMovies API with an invalid pageNumber = 0. The server responds with HTTP status code 400 Bad Request, indicating that the provided page number is invalid.

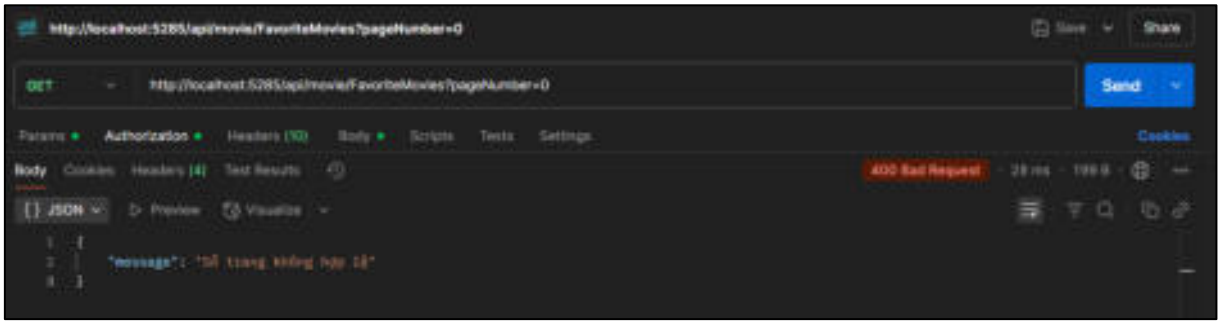

Figure 4.32. JSON response when providing an invalid page number

- The figure 4.33 shows the JSON response returned when a GET request is sent to the FavoriteMovies API with an invalid pageSize = 0. The server responds with HTTP status code 400 Bad Request, indicating that the provided page size is invalid.

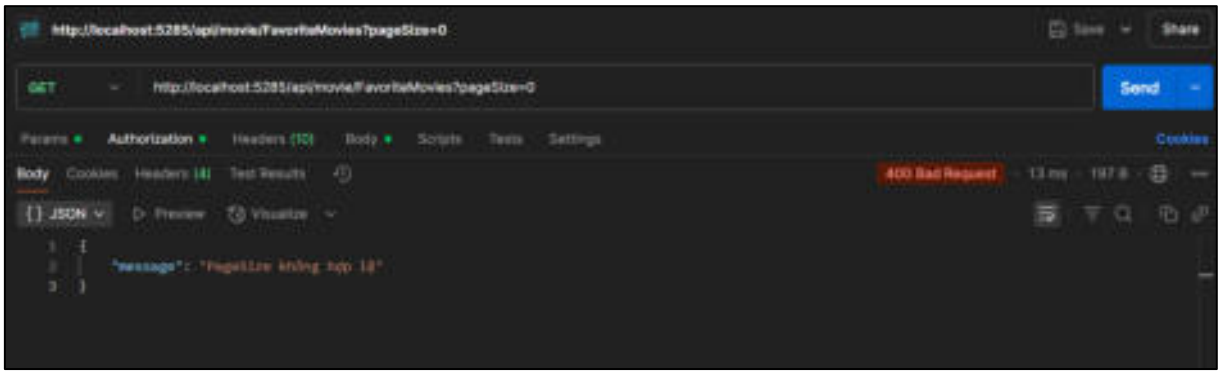

Figure 4.33. JSON response when providing an invalid page size

4.4. Evaluation of Results

The online movie streaming web application has generally met the business requirements and design objectives. The user interface was developed with a user-friendly approach, making it easier for new users to operate and get familiar with the system. However, potential interface or functionality issues may still arise during real-world operation.

4.5. Chapter Conclusion

In Chapter 4, the process of building and implementing the system was presented in detail, covering everything from choosing the development environment and programming tools to interface design and core functionalities. The website not only meets basic requirements such as movie search, online viewing, and categorization by genre, country, or director, but also integrates advanced features such as electronic payment via PayOS and account role management. Notably, the integration of an AI chatbot into the system significantly enhances user interaction and support by providing quick and flexible assistance.

In addition to the achieved outcomes, the project also identified some existing limitations and proposed future development directions, aiming to continuously improve user experience and increase the value the system brings to the digital entertainment domain.

CONCLUSION AND FUTURE DEVELOPMENT

1. Achieved Results

During the research, learning, and implementation process of this project, the following results have been accomplished:

- **In terms of theoretical knowledge:**
  - Through exploring and executing the project, I have gained a solid understanding and successfully applied core knowledge required to build a web application using technologies such as C-Sharp, frameworks and libraries like ASP.NET Core Web API, ReactJS, and the database management system PostgresDB.
  - Effectively designed and applied the RESTful API model for deploying the application, ensuring a well-structured, maintainable, and scalable software architecture.
- **In terms of practical application:**
  - Successfully built and deployed an online movie streaming website, implementing complete functionalities such as user account management, movie management, and more, meeting the basic needs of users.
  - Integrated an AI chatbot that assists users in quickly and easily searching for movies, supporting the Vietnamese language to optimize the user experience for Vietnamese users.
  - Integrated an electronic payment feature, allowing users to conveniently and efficiently make transactions. This not only enhances personalized experiences but also supports businesses in optimizing content development strategies and improving user satisfaction.

2. Limitations

Despite the results achieved during the project implementation, the system still has several limitations. Specifically, the current interface has not been fully optimized for mobile platforms, which negatively affects the user experience on handheld devices. Additionally, pricing policy issues between users and platform owners have not yet been addressed in a comprehensive and detailed manner.

Due to limited time and practical experience, the problem analysis was carried out relatively thoroughly, but it still lacks a complete description of all aspects of the issue.

3. Future Development Directions

Feature Enhancement: Integrate advanced data analysis tools to predict viewing trends and recommend content tailored to each user's preferences. Optimize the user experience across both mobile and desktop platforms to ensure consistent performance, a unified interface, and user-friendly design. Upgrade the AI-powered chatbot with modern artificial intelligence capabilities, enabling it to recognize user emotions and provide natural responses, thereby enhancing interaction and user satisfaction.

Scalabitily: Implement multilingual support to cater to international users and integrate global payment solutions such as PayPal for broader transaction capabilities. Collaborate with entertainment industry professionals, especially in the film sector, to expand content and services.

User Community Development: Establish forums, comment systems, and review sharing features to encourage users to interact, exchange opinions, and build a vibrant film-loving community.

REFERENCES

[1] Oracle, "What is a chatbot?." [Online]. Available: https://www.oracle.com/chatbots/what-is-a-chatbot/. [Accessed 05 May 2025].

[2] A. M. Turing, "Computing Machinery and Intelligence," *Mind*, vol. 59, no. 236, pp. 433–460, 1950, doi: 10.1093/mind/LIX.236.433.

[3] J. Weizenbaum, "ELIZA-A Computer Program For the Study of Natural Language Communication Between Man and Machine," *Communications of the ACM*, vol. 9, no. 1, pp. 36–45, 1966, doi: 10.1145/365153.365168.

[4] K. M. Colby, *Artificial Paranoia: A Computer Simulation of Paranoid Processes*, Oxford: Pergamon Press, 1975.

[5] R. S. Wallace, "The Anatomy of A.L.I.C.E.," in *Parsing the Turing Test*, Dordrecht: Springer, 2009, pp. 181–210.

[6] IBM Research, "Watson: A System Designed for Answers," 2006. [Online]. Available: https://www.ibm.com/watson. [Accessed 10 April 2025].

[7] Apple Inc., "Introducing Siri," 2011. [Online]. Available: https://www.apple.com/siri. [Accessed 20 March 2025].

[8] Apple Inc., "Introducing Siri," 2011. [Online]. Available: https://www.apple.com/siri. [Accessed 25 March 2025].

[9] OpenAI, "Introducing ChatGPT," 2020. [Online]. Available: https://openai.com/blog/chatgpt. [Accessed 06 March 2025].

[10] Microsoft, "Tour of C#: Overview." [Online]. Available: https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/overview. [Accessed 10 April 2025].

[11] ITNavi, "C# là gì?." [Online]. Available: https://itnavi.com.vn/blog/c-la-gi/. [Accessed 05 February 2025].

[12] Microsoft, "Introduction to ASP.NET Core." [Online]. Available: https://learn.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-9.0. [Accessed 19 February 2025].

[13] Xây Dựng Số, "Giải thích REST là gì và các loại vi-cb." [Online]. Available: https://xaydungso.vn/blog/giai-thich-rest-la-gi-va-cac-loai-vi-cb.html. [Accessed 16 March 2025].

[14] TopDev, "RESTful API là gì?," [Online]. Available: https://topdev.vn/blog/restful-api-la-gi/. [Accessed 20 April 2025].

[15] Wiweb, "PostgreSQL là gì?," [Online]. Available: https://wiweb.vn/postgresql-la-gi/. [Accessed 07 March 2025].

[16] Xây Dựng Số, "Hướng dẫn cơ bản React là gì trên nền tảng web vi-cb," [Online]. Available: https://xaydungso.vn/blog/huong-dan-co-ban-react-to-la-gi-tren-nen-tang-web-vi-cb.html. [Accessed 19 March 2025].

[17] Q. P. Nguyen, "React - Vòng đời của một Component," [Online]. Available: https://viblo.asia/p/react-vong-doi-cua-mot-component-co-gi-hay-eW65G1NYZDO. [Accessed 18 May 2025].

[18] GP Coder, "Tìm hiểu về kiểm thử (Tesing) trong phát triển phần mềm," [Online]. Available: https://gpcoder.com/5194-tim-hieu-ve-kiem-thu-tesing-trong-phat-trien-phan-mem/#Phan_loai_Tesing. [Accessed 03 June 2025].