

THE UNIVERSITY OF DA NANG
DA NANG UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

GRADUATION PROJECT THESIS

MAJOR: INFORMATION TECHNOLOGY

SPECIALTY: SOFTWARE ENGINEERING

PROJECT TITLE:

**BUILD A BOOKMARK MANAGEMENT SYSTEM WITH
INTEGRATED AI**

Instructor	: MSC. MAI VAN HA
Student	: TRUONG BICH QUYNH
Student ID	: 102200150
Class	: 20TCLC_DT3

Da Nang, 6/2024

**THE UNIVERSITY OF DA NANG
DA NANG UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY**

GRADUATION PROJECT THESIS

MAJOR: INFORMATION TECHNOLOGY

SPECIALTY: SOFTWARE ENGINEERING

PROJECT TITLE:

**BUILD A BOOKMARK MANAGEMENT SYSTEM WITH
INTEGRATED AI**

Stucter	: MSC. MAI VAN HA
Student	: TRUONG BICH QUYNH
Student ID	: 102200150
Class	: 20TCLC_DT3

Da Nang, 6/2024

SUMMARY OF THE GRADUATION PROJECT RESULTS

I. General Information

1. *Student name: Truong Bich Quynh*
2. *Student ID: 102200150*
3. *Class: 20TCLC_DT3*
4. *Topic title: Build a Bookmark management system with integrated AI*

II. Project summary

1. Description of the problem

In the 4.0 technology era, searching for and researching information on the internet has become an indispensable part of daily life. However, this process often requires users to consider multiple different websites to gather information and achieve the best results for each issue. From a personal perspective, managing and organizing the websites one is researching becomes more important than ever.

For this reason, I propose a project with the goal of building a utility bookmark management system combined with an integrated chatbox featuring GPT chat. The purpose of the project is to help users easily search, store, and share information from various websites efficiently.

In reality, although this need is quite essential, there still hasn't been a comprehensive solution that meets these requirements. Therefore, through this project, I hope to provide a convenient solution for users in their studying and researching endeavors.

2. Author contributions

System design analysis from the inception of website development, programming, and deployment, to system maintenance and upkeep.

3. Main functions and results of program implementation

The system provides solutions for bookmark management along with utilities for learning and research, enabling users to do so quickly and efficiently.

DA NANG UNIVERSITY
UNIVERSITY OF SCIENCE AND
TECHNOLOGY
FACULTY: INFORMATION
TECHNOLOGY

THE SOCIALIST REPUBLIC OF
VIETNAM
Independence - Freedom - Happiness

GRADUATION PROJECT REQUIREMENTS

Student Name: Truong Bich Quynh

Student ID : 102200150

Class: 20TCLC_DT3 Faculty: Information Technology

Major: Information Technology

1. *Topic title: Build a Bookmark management system with integrated AI*
2. *Project topic : ☒has signed intellectual property agreement for final result*
3. *Initial figure and data:*

.....
.....

Content of the explanations and calculations:

.....
.....

4. *Drawings, charts (specify the types and sizes of drawings):*

.....
.....
.....
.....

5. <i>Name of instructor:</i>	<i>Content parts:</i>

6. *Date of assignment :*/...../201

7. *Date of completion :*/...../201

Da Nang, date month year 2024

Head of Division.....

Instructor

THANK YOU

First and foremost, I would like to express my gratitude and appreciation to Mr. Mai Van Ha. He has been attentive, pinpointing shortcomings and providing the best conditions for me to complete my graduation project.

I sincerely thank all the professors of the Faculty of Information Technology at the University of Science and Technology - The University of Danang for guiding and teaching me, both in terms of specialized knowledge and study spirit, enabling me to acquire the knowledge necessary for my graduation project.

Lastly, due to my lack of experience and being relatively new to this, I understand that there may be imperfections in my graduation project. I eagerly welcome the valuable feedback from all the professors and my peers to improve the quality of my work.

Once again, I express my sincere gratitude!

Truong Bich Quynh

ASSURANCE

I hereby declare:

- 1. The content of this project was carried out by me under the direct supervision of MSc. Mai Van Ha.*
- 2. All references used in this project are clearly cited with the author's name, title, time, and place of publication.*
- 3. If there are any unauthorized copies or violations, I take full responsibility.*

Da Nang, date month year 2024

Students perform

Truong Bich Quynh

TABLE OF CONTENTS

CHAPTER 1: THEORETICAL BASIS	2
1.1. THEORETICAL RESEARCH	3
1.1.1. PROGRAMMING LANGUAGE C#	3
1.1.2. INTRODUCING ENTITY FRAMEWORK CORE	3
1.1.3. INTRODUCTION TO .NET CORE	5
1.1.4. INTRODUCTION TO SIGNALR IN ASP.NET	7
1.1.5. OBJECT-RELATIONAL-MAPPING (ORM)	9
1.1.6. INTRODUCTION TO VUE.JS	11
1.1.7. INTRODUCTION TO NUXT.JS	12
1.1.8. ABOUT AZURE	13
1.1.9. ABOUT OPENAI'S TEXT GENERATION MODELS	13
CHAPTER 2: SYSTEM ANALYSIS AND DESIGN	14
2.1. BUILD A BOOKMARK MANAGEMENT SYSTEM WITH INTEGRATED AI	15
2.2. SYSTEM ANALYSIS AND DESIGN	15
2.2.1. FUNCTIONAL DIAGRAM	15
2.2.2. USE CASE CHART	17
2.2.3. ACTIVITY DIAGRAM	20
2.2.4. SEQUENCE DIAGRAM	22
2.2.5. DESIGN CLASS CHART AND DATABASE	23
2.2.6. INTRODUCTION OF FUNCTION	26
2.2.7. CONNECTION MODEL WITH GPT API	27
2.3. CONCLUDE	28
CHAPTER 3: EVALUATION	28
3.1. SOME IMAGES USED BY WEBSITE	29
3.2. CONCLUDE	40
REFERENCES	42

LIST OF IMAGES

Image 1.1: EF Core processes interact with the database.	4
Image 1.2: Overall architecture of Entity Framework Core.	4
Image 1.3: The process of converting source code into machine code for execution.	7
Image 1.4: SignalR in .NET.	8
Image 1.5: Using Server-Sent Event (SSE) technology.	8
Image 1.6: Signal Hub.	9
Image 2.1: Functional decomposition diagram of the bookmark management system with integrated GPT chat.	16
Image 2.2: Overall use case diagram of the bookmark management system with integrated GPT chat.	17
Image 2.3: Activity diagram describing the login functionality.	20
Image 2.4: Activity diagram describing the bookmark management functionality.	21
Image 2.5: Activity diagram describes the functionality of using the chatbox.	22
Image 2.6: Sequence diagram depicting the bookmark management system with integrated GPT chat.	23
Image 2.7: Class diagram illustrating the functionality of bookmark management with integrated GPT chat.	24
Image 2.8: Overview of the connectivity model.	27
Image 3.1: Login screen.	29
Image 3.2: Register screen.	30
Image 3.3: Directory list and list of bookmarks within each respective directory.	30
Image 3.4: UI create new folder.	31
Image 3.5: UI create new bookmark.	31
Image 3.6: Function bar on each bookmark.	32
Image 3.7: Chatbox UI.	33
Image 3.8: Search function in chatbox.	34
Image 3.9: Download content chatbox function.	35
Image 3.10: Content of chatbox in file downloaded.	35
Image 3.11: Chatbox screen displaying responses to questions unrelated to bookmark content.	36
Image 3.12: The chatbox screen displays the answer to the user.	36
Image 3.13: Tools that users can interact with the responses.	37
Image 3.14: Page containing stored messages.	38
Image 3.15: Adding note.	39
Image 3.16: Import bookmark list.	39

LIST OF TABLES

Table 2.1 Login use case specification.....	17
Table 2.2: Specification of the IAM system account group permission editing use case.....	17
Table 2.3: Features of using chatbox.....	18
Table 2.4: Table Description Table Bookmark.....	25
Table 2.5: Table Description table Conversation.....	25
Table 2.6: Table Description table Message.....	25
Table 2.7: Table Description table Folder.....	26
Table 2.8: Table Description table User.....	26
Table 2.9: Table Description table UserToken.....	26

ACRONYMS LIST

Acronym	Full word
EF Core	Entity Framework Core
MS	Microsoft
COM	Component Object Model
LINQ	Language-Integrated Query
MSIL	Microsoft Intermediate Language
ASP	Active Server Page
URL	Uniform Resource Locator
IIS	Internet Information Server
DDD	Domain Driven Design
CQRS	Command and Query Responsibility Segregation
MVC	Model View Controller

PREFACE

1. Overview

The project 'Build a Bookmark management system with integrated AI' aims to create a bookmark management system combined with utilities and a chatbox integrated with GPT chat. The goal of the project is to provide users with a multifunctional tool to manage, interact with, and extract information from websites efficiently and conveniently.

2. Problem

In today's digital age, searching for and accessing information on the internet is an indispensable part of daily life. However, managing and organizing information from websites can sometimes present many challenges. Users often have to spend considerable time reviewing numerous websites and may find it difficult to store and retrieve important information.

What is bookmark?

- Definition: A bookmark is a saved link to a web page that a user wants to easily return to later. It's like a digital marker that allows quick access to specific pages on the internet.
- Purpose: Bookmarks help users keep track of valuable and frequently visited web pages without needing to remember the URLs or search for them again.

Challenges with traditional bookmark management:

- Organization: As the number of bookmarks grows, organizing them in a meaningful way becomes increasingly difficult.
- Searchability: Finding a specific bookmark among many can be time-consuming, especially if they are not well-organized.
- Context Loss: Traditional bookmark systems do not provide contextual information, making it hard to remember why a particular page was saved.

3. Purpose

The purpose of this project is to build a bookmark management system integrated with a GPT chatbox to address these issues. Additionally, the project aims to create a useful tool that helps users save time and enhance their efficiency when interacting with information on the internet. By combining the automatic

response capabilities of the GPT chatbot with flexible bookmark management features, the project aims to provide the best experience for users in their research, study, and work online.

4. Theoretical basic

4.1 Technical

- C# - ASP.NET MVC Web API;
- .NET CORE 7;
- Entity Framework Core;
- Typescript - Vue.js;
- Python.

4.2 Development tool and environment

- Microsoft Visual Studio Code.

5. Layout of the project

The project include following content:

Preface

Chapter 1: THEORETICAL BASIC

Chapter 2: SYSTEM ANALYSIS AND DESIGN

Chapter 3: EVALUATION

Conclude.

CHAPTER 1: THEORETICAL BASIS

1.1. THEORETICAL RESEARCH

The system uses modern popular technologies and platforms to minimize risks during the development process while still ensuring flexibility and meeting all set requirements.

To better understand the structure and technologies used, in this section I would like to present the content of the main technologies along with how to deploy them in the system.

1.1.1. PROGRAMMING LANGUAGE C#

C# (C-Sharp) [3] is a modern and general-purpose programming language developed by Microsoft in the 2000s. C# is part of Microsoft's .NET Framework and is widely used for development. Develop Windows applications, web applications, mobile applications and more on Microsoft platforms. C# is built on two strong platforms: C++ and Java.

C# supports full object-oriented programming with concepts such as classes, objects, inheritance, polymorphism, and encapsulation. Besides, C# provides many condensed syntax for writing code, helping to reduce length and increase readability, typically lambda expressions, attributes, iterator (yield), ...

1.1.2. INTRODUCING ENTITY FRAMEWORK CORE

Entity Framework Core (EF Core) [4] is a .NET framework that allows interaction with relational databases through ORM (Object-Relational Mapping) mapping mechanism using Language Integrated Query (LINQ). [5] to query data, minimizing the writing of pure query statements.

The following figure depicts the EF Core process interacting with the database via LINQ:

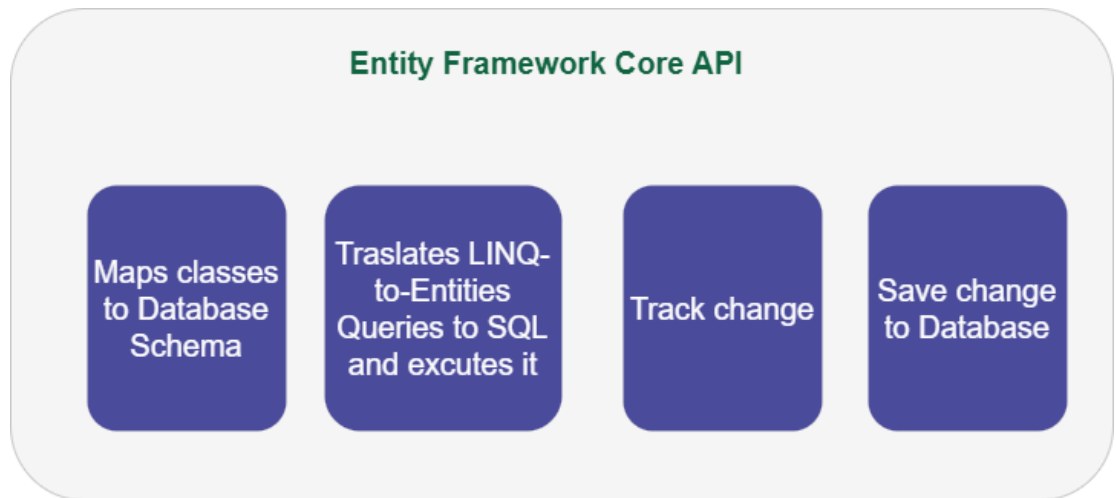


Image 1.1: EF Core processes interact with the database.

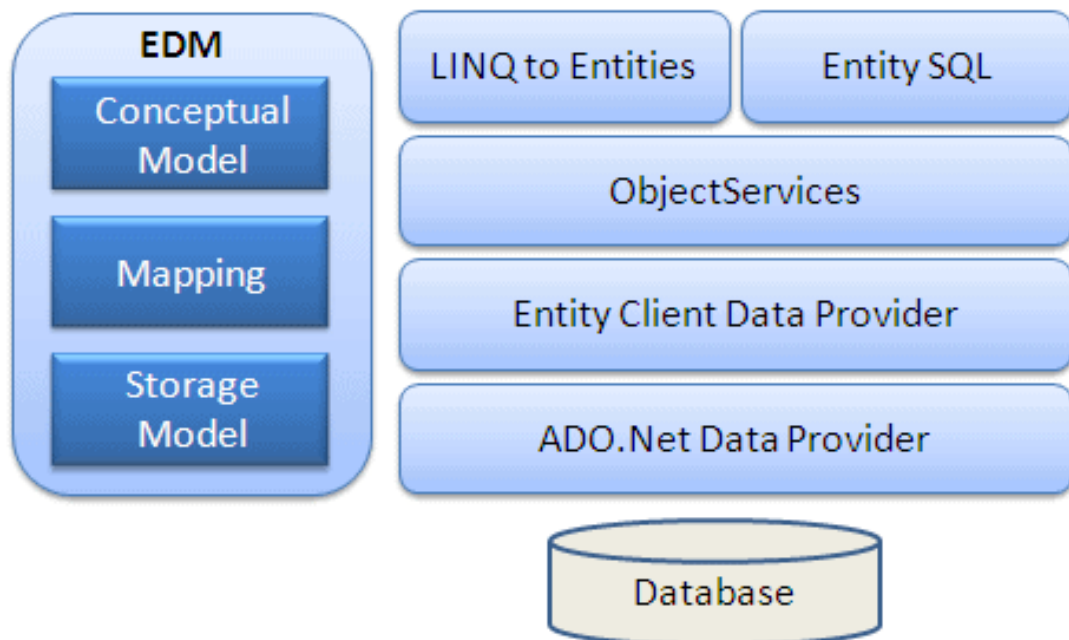


Image 1.2: Overall architecture of Entity Framework Core.

The overall architecture of Entity Framework is shown in the image above, in which:

- EDM (Entity Data Model): Data entity model that represents all components into a visual designer that allows programmers to have an overview of the interacting database without need to go through another tool. Includes 3 main parts:

- + Conceptual Model: contains models of classes, relationships between them in the database and is independent of the database;
- + Storage Model: This is a database design model containing all database things such as tables, views, stored procedures, relationships, keys;
- + Mapping: Includes information describing how the Conceptual Model is connected to the Storage Model.
- Linq To Entities: is a type of LINQ (Language Integrated Language) - a data query language based on ORM (Object-Relational Mapping) so that programmers can manipulate data (here, Entities) instead of having to write pure queries in SQL format;
- Entities SQL: is a query language similar to SQL, used when performing complex queries - when it is difficult for LINQ to Entities to fully meet the business needs, providing the ability to query and manipulate flexible data. dynamic, efficient for .NET applications that use the Entity Framework to access data;
- Object Service: Provides a set of tools for programmers to work with Entities and their relationships, including query support, data editing, data validation, and structure conversion. from the code to the real table in the database. The main benefit of Object Service is that it provides an intuitive way to work with relational data versus having to use pure SQL queries or use lower data access layers like ADO.NET, which facilitates transformations. convert the data returned from the query into entities. Programmers will work with entities and use familiar object-oriented relationships rather than primitive database structures;
- Entity Client Data Provider: responsible for converting LINQ or other ORM language into corresponding pure sql statements for execution in the database and providing a set of features for working with data returned from the database.

1.1.3. INTRODUCTION TO .NET CORE

Component object model (COM)

COM is a Microsoft technology developed to build windows applications and websites. However, COM has two disadvantages: it does not fully support OOP features and depends on the Windows platform.

.NET

.NET was born to overcome the disadvantages of COM, is a platform for developers made up of tools and libraries to build many different types of applications including websites.

There are many different implementations of .NET, each allowing execution of .NET code in various platforms such as Windows, Linux, macOS, android.

- .NET Framework is the original implementation of .NET, a powerful tool that fully supports OOP features and develops websites and application services on the windows platform;
- .NET Core is implemented after .NET Framework to meet the needs of cross-platform development on Windows, macOS and Linux.

.NET Core architecture

.NET Core mainly consists of 4 components:

- CLI Tools: is a cross-platform tool that supports developing, compiling, launching and publishing .NET applications under command lines and can be integrated into the terminal;
- Roslyn: C# language compiler;
- CoreFX: collection of libraries developed by Microsoft to support the deployment of applications and websites using .NET;
- CoreCLR: used to convert Microsoft Intermediate Language (MSIL) code into machine code for execution.

The following figure describes the process of converting C# source code into executable machine code:

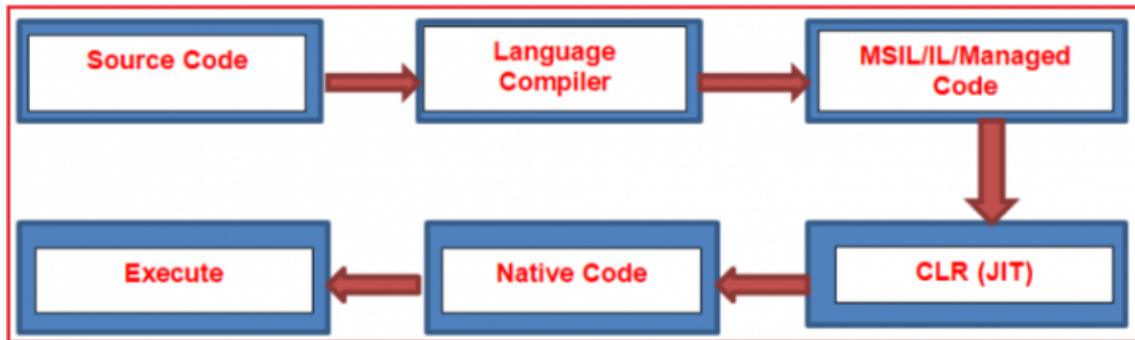


Image 1.3: The process of converting source code into machine code for execution.

In .NET, the source code is compiled twice:

- The source code will first be compiled into autonomous code as an intermediate language called Microsoft Intermediate Language (MSIL) or Intermediate Language (IL) by the respective language compilers;
- Then when executing the application, depending on the user's instructions, the CLR's Just In Time (JIT) compiler will convert the autonomous code into native machine code.

1.1.4. INTRODUCTION TO SIGNALR IN ASP.NET

SignalR[1] is a library in ASP.NET that allows building real-time web applications. With SignalR, you can create features like chat, live notifications, and real time data updates without having to refresh the page. This creates a better user experience and increases web application interactivity. SignalR automatically handles connection and messaging between client and server, letting you focus on developing your app's realtime features without worrying about communication implementation.

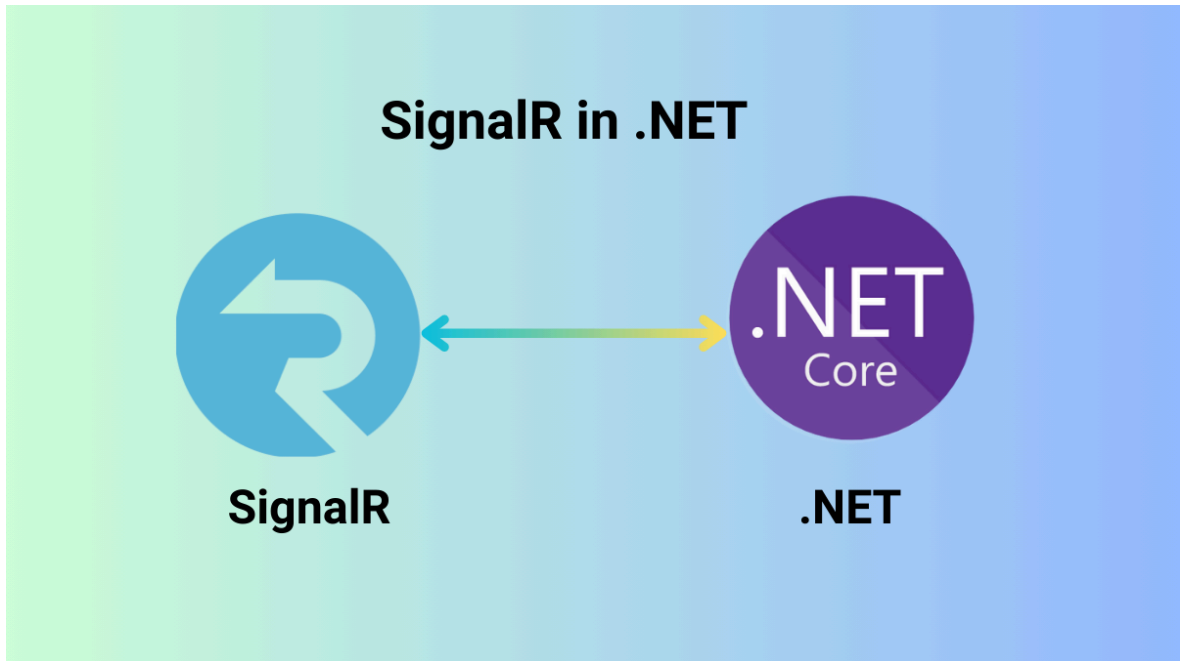


Image 1.4: SignalR in .NET.

Using Server-Sent Events (SSE) is a web technology that allows the server to send data directly to the client over HTTP. SSE provides a simple mechanism to establish a connection from a client to a server and receive updates from the server without generating new requests from the client.

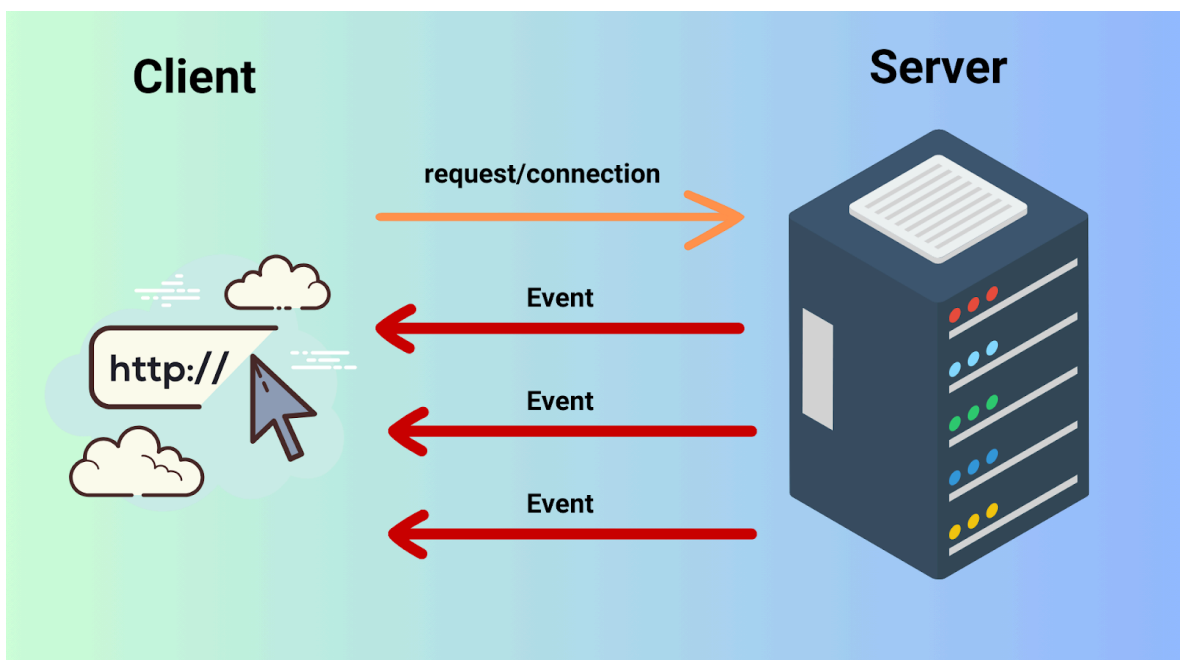


Image 1.5: Using Server-Sent Event (SSE) technology.

The Hub in SignalR is an important component for real time communication between client and server in ASP.NET web applications. It provides a central

mechanism for establishing connections, sending data both from the client and the server, and managing connections automatically, simplifying real time application development.

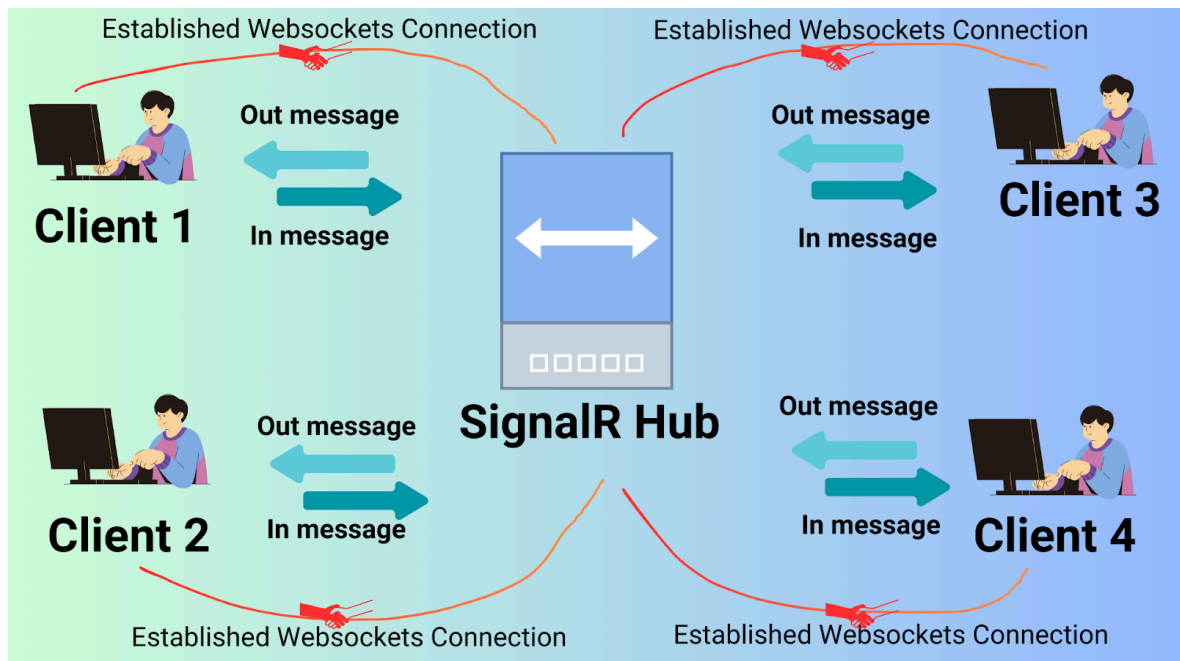


Image 1.6: Signal Hub.

1.1.5. OBJECT-RELATIONAL-MAPPING (ORM)

Object-Relational Mapping (ORM) is a programming technique used to bridge the gap between object-oriented programming languages such as Java, C# or PHP and typical relational databases. Looks like MSSQL Server, MySQL, Postgrester or Oracle.

Databases store data in tables, while object-oriented programming deals with objects and relationships between them. ORM is to serve as an abstraction layer that allows working directly with objects defined in code, automatically mapping to the corresponding table and retrieving the data stored in the table, eliminating the way of basic queries. traditional database and process returned results.

Typically ORM operations will include the following steps:

- Object Model: the programmer defines classes (objects) that represent tables in the database, for example users, categories... These classes have properties and methods of defining their behavior;
- Mapping (Mapping to database): ORM frameworks provide a mapping mechanism to establish correspondence between the object model and the

database structure. Typically, mapping is done through configuration files or annotations in the code. Determine how each class, attribute, and relationship maps to the database's tables, columns, and foreign keys;

- CRUD Operations: Once the mapping is established, allows the use of an ORM framework to perform Create, Read, Update, and Delete (CRUD) operations on objects. The ORM framework converts these operations into SQL statements and executes them on the corresponding database.

Transparency between object and database: The ORM framework tries to provide transparency between object-oriented and relational programming models. This allows interaction with objects in code without concern for back-end database operations. The ORM framework handles the conversion of object operations into database operations and vice versa.

Query language: ORM frameworks often provide a native query language, often a variant of SQL. This query language allows database queries to be written in a more object-oriented way, using object properties and relationships instead of pure SQL statements.

Benefits of using ORM include:

- Reduce application development time: ORM eliminates the need to write SQL statements and map results, making working with databases easier and faster;
- Portability: ORM abstracts the details of database-specific dependencies, allowing applications to work with different DBMSs without changing the main source code;
- Ease of maintenance: Database structure changes can be easily handled by updating the mapping configuration instead of editing multiple SQL statements in the source code;
- Performance optimization: ORM frameworks often provide features such as caching, lazy loading, and batch operations, which help optimize database access and improve performance.

However, it must be noted that ORM is not always the best choice for every situation. In some cases, direct SQL queries or other data access patterns may be more appropriate, depending on the specific requirements and performance standpoint of the application.

1.1.6. INTRODUCTION TO VUE.JS

Vue.js [7] is an open source JavaScript framework used to build dynamic web applications and user interfaces. It focuses on creating efficient and maintainable Single Page Applications (SPAs) using a flexible and easy-to-understand approach.

Created by Evan You, Vue.js is designed to optimize web application development by providing a flexible and easy-to-use structure. Vue.js allows developers to create reusable components, interact with data, and manipulate the DOM easily.

Some outstanding advantages include:

- Flexible structure: Vue.js offers a highly adaptable architecture, enabling developers to integrate it into projects incrementally. This flexibility means you can use Vue.js for a small part of your application or as the central framework for a large, complex project. Its design also allows easy integration with other libraries or existing projects, making it a versatile choice for web development.
- Bidirectional data binding: One of the standout features of Vue.js is its bidirectional data binding. This feature ensures that any change in the application state reflects instantly in the user interface and vice versa. By synchronizing data between the model and the view, Vue.js simplifies the handling of dynamic content and reduces the amount of boilerplate code.
- Directives system: Vue.js includes a powerful directives system, which allows developers to extend HTML with custom tags and attributes. Directives such as v-bind, v-model, and v-for enable reactive behavior and DOM manipulation with minimal effort. This system promotes cleaner and more readable code, enhancing maintainability.
- Easy-to-use tool: Vue.js is known for its simplicity and ease of use. The framework's core library focuses on the view layer only, making it easy to pick up and integrate with other libraries or existing projects. The learning curve is relatively shallow, especially for those with experience in HTML, CSS, and JavaScript.
- High performance: Performance is a critical aspect of Vue.js. Its reactivity system and virtual DOM implementation ensure efficient updates and

rendering. Vue.js minimizes the number of DOM manipulations, leading to faster and more responsive applications.

1.1.7. INTRODUCTION TO NUXT.JS

Nuxt.js[6] is an open source JavaScript framework built on top of Vue.js used to develop simple web applications (SPAs) and static web applications. What's special about Nuxt.js is that it simplifies the Vue.js application development process by providing a number of powerful features like pre-rendering, server-side rendering (SSR), and dynamic generation website.

Advantages of Nuxt.js:

- Server-side rendering: Server-side rendering is a core feature of Nuxt.js, which allows the initial page load to be rendered on the server rather than the client. This improves the performance of the application by reducing the time to first meaningful paint, making the application more SEO-friendly and providing a better user experience. SSR is particularly beneficial for content-heavy applications where quick rendering is crucial.
- Pre-rendering: Nuxt.js supports pre-rendering, which generates static HTML pages at build time. This feature combines the benefits of SSR with the speed of static sites, providing an excellent balance between performance and functionality. Pre-rendering is ideal for static content that doesn't change frequently, ensuring that users receive fast-loading pages.
- Automatic routing: Nuxt.js automates the routing process based on the directory structure of the project. This means that developers don't need to manually configure routes for each page, simplifying the development process and reducing the potential for errors. Each .vue file in the pages directory automatically becomes a route.
- Standard directory structure: Nuxt.js enforces a standard directory structure, promoting best practices and consistency across the codebase. This structure includes directories for components, layouts, middleware, pages, plugins, and static files. Adhering to this organized structure helps in maintaining a clean and scalable project architecture.
- Middleware support: Middleware in Nuxt.js provides a way to execute custom logic before rendering a page or performing a route transition. This is useful for tasks such as authentication, logging, or setting global state.

Middleware functions can be applied globally, to specific routes, or to individual components, offering flexible control over the application flow.

1.1.8. ABOUT AZURE

Azure is a comprehensive cloud computing platform developed by Microsoft, providing a range of services and solutions to deploy, manage, and operate applications. With Azure, users can easily build, deploy, and manage applications across a global network of Microsoft-managed data centers. Azure services include compute, storage, database, analytics, artificial intelligence and machine learning, networking, and more. The platform supports multiple programming languages, tools and frameworks, enabling businesses and developers to deploy applications quickly and flexibly, while taking advantage of security features, Microsoft Azure reliability and scalability.

1.1.9. ABOUT OPENAI'S TEXT GENERATION MODELS

OpenAI's text generation models (often called generative pre-trained transformers or large language models) have been trained to understand natural language, code, and images. The models provide text outputs in response to their inputs. The text inputs to these models are also referred to as "prompts". Designing a prompt is essentially how you “program” a large language model, usually by providing instructions or some examples of how to successfully complete a task. I

About OpenAI's text generation models: GPT-3.5 turbo

- GPT-3.5 Turbo is a state-of-the-art language model developed by OpenAI. It is an enhanced version of the GPT-3 model, designed to generate human-like text based on the input it receives;
- Natural Language Understanding: GPT-3.5 Turbo can understand and respond to a wide range of prompts in natural language, making it highly versatile.
- Contextual Awareness: It can maintain context over long conversations, providing coherent and contextually relevant responses.
- Text Generation: Capable of generating text for various applications such as chatbots, content creation, and more.

1.2. REALITY

Huge workload: Internet users increasingly have access to huge amounts of information from many different sources. Finding, storing and organizing information becomes a major challenge.

Information dispersion: Necessary information is often scattered across many different websites and documents, making tracking and management complicated and time-consuming.

Limited management tools: Currently, bookmark management tools only provide basic features such as storing and organizing web pages, lacking interactive features and smart support.

From the above, it can be seen that a bookmark management system that integrates the necessary features is necessary for many users today:

- Optimize the research and learning process: A bookmark management system with integrated chatbox will help users store and organize web pages systematically. They can quickly access and organize information according to specific needs, thereby improving research and learning efficiency;
- Intelligent Interaction: Integrated GPT technology allows users to ask questions and receive intelligent answers about the content of archived web pages. This not only saves time but also provides information quickly and accurately;
- Effective information management: The system allows users to note, copy, download and share information to increase convenience and flexibility;
- Import bookmark list from Chrome.

1.3. CONCLUDE

In the context of increasingly large and scattered information volumes, a bookmark management system integrating the GPT chatbot not only helps users organize and store information effectively but also provides intelligent interactions, personalization and support for information sharing. This is a comprehensive solution, meeting the increasing needs of users in managing and interacting with information on the internet.

CHAPTER 2: SYSTEM ANALYSIS AND DESIGN

2.1. BUILD A BOOKMARK MANAGEMENT SYSTEM WITH INTEGRATED AI

The system includes a number of main functions as follows:

- **Manage bookmarks:** Build a bookmark list management interface, allowing users to add, delete, edit and organize bookmarks easily. Integrates features to import bookmark lists exported from different browsers (Chrome).
- **Interact with chatbox:** Ability to create a text summarizing the website's content along with questions related to the website's content for users to refer to. Besides, users can ask questions and receive intelligent answers from the GPT chatbox about the content of the bookmark.
- **Utilities on chatbox:** Allows users to copy and store the content of messages in the chatbox. Note information, highlight messages, store responses from chatbox. Search for messages, download the entire chat content into a file.

2.2. SYSTEM ANALYSIS AND DESIGN

2.2.1. FUNCTIONAL DIAGRAM

A functional decomposition diagram is a diagram that analyzes and helps viewers clearly understand the structure and relationships between functional components in a system or project. Focus on decomposing main functions into sub-functions until the required level of detail is reached.

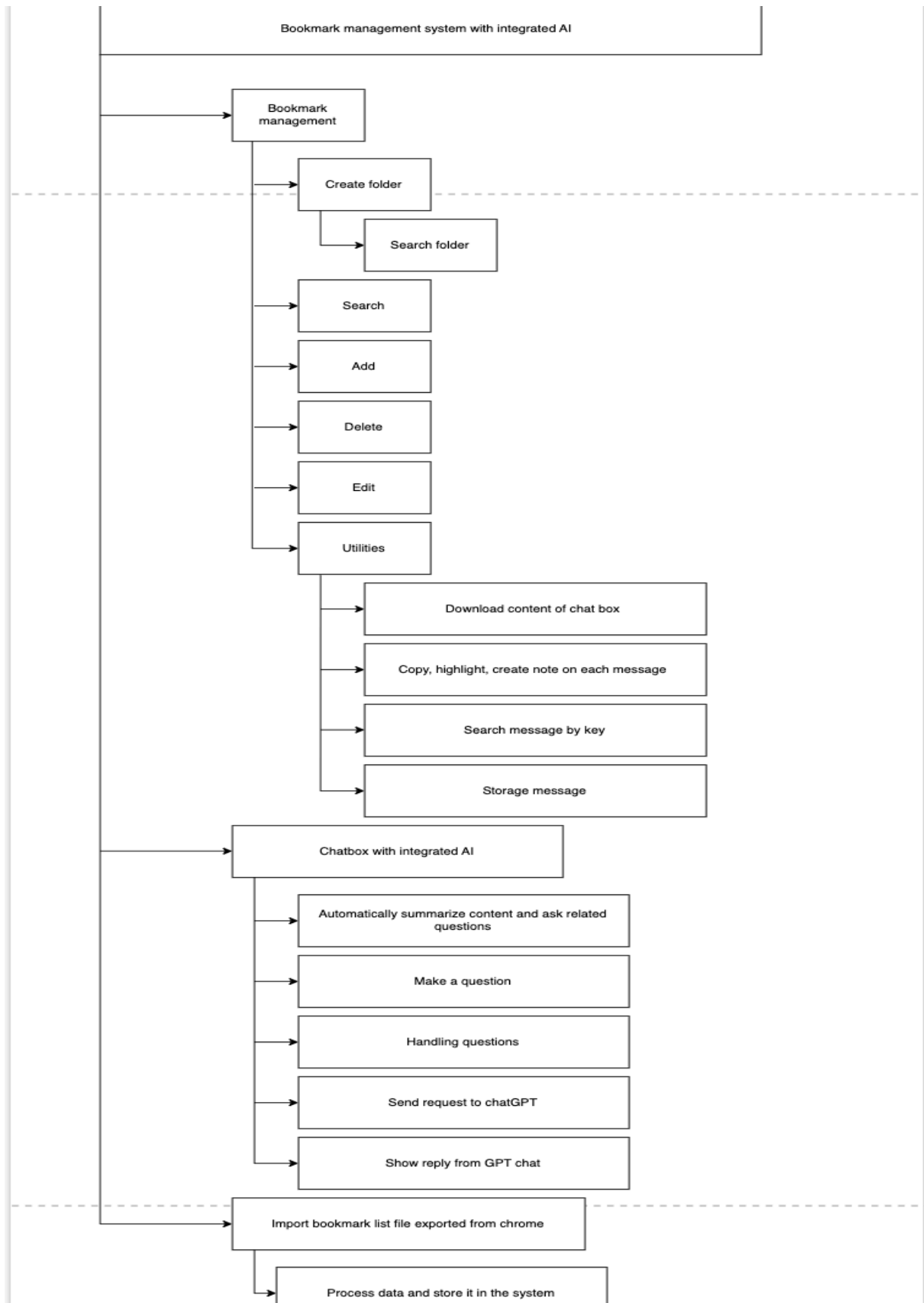


Image 2.1: Functional decomposition diagram of the bookmark management system with integrated GPT chat.

2.2.2. USE CASE CHART

Use case diagrams help viewers understand the interactions between users and the system in a specific environment with a defined purpose.

The general use case diagram of the system is described as shown below:

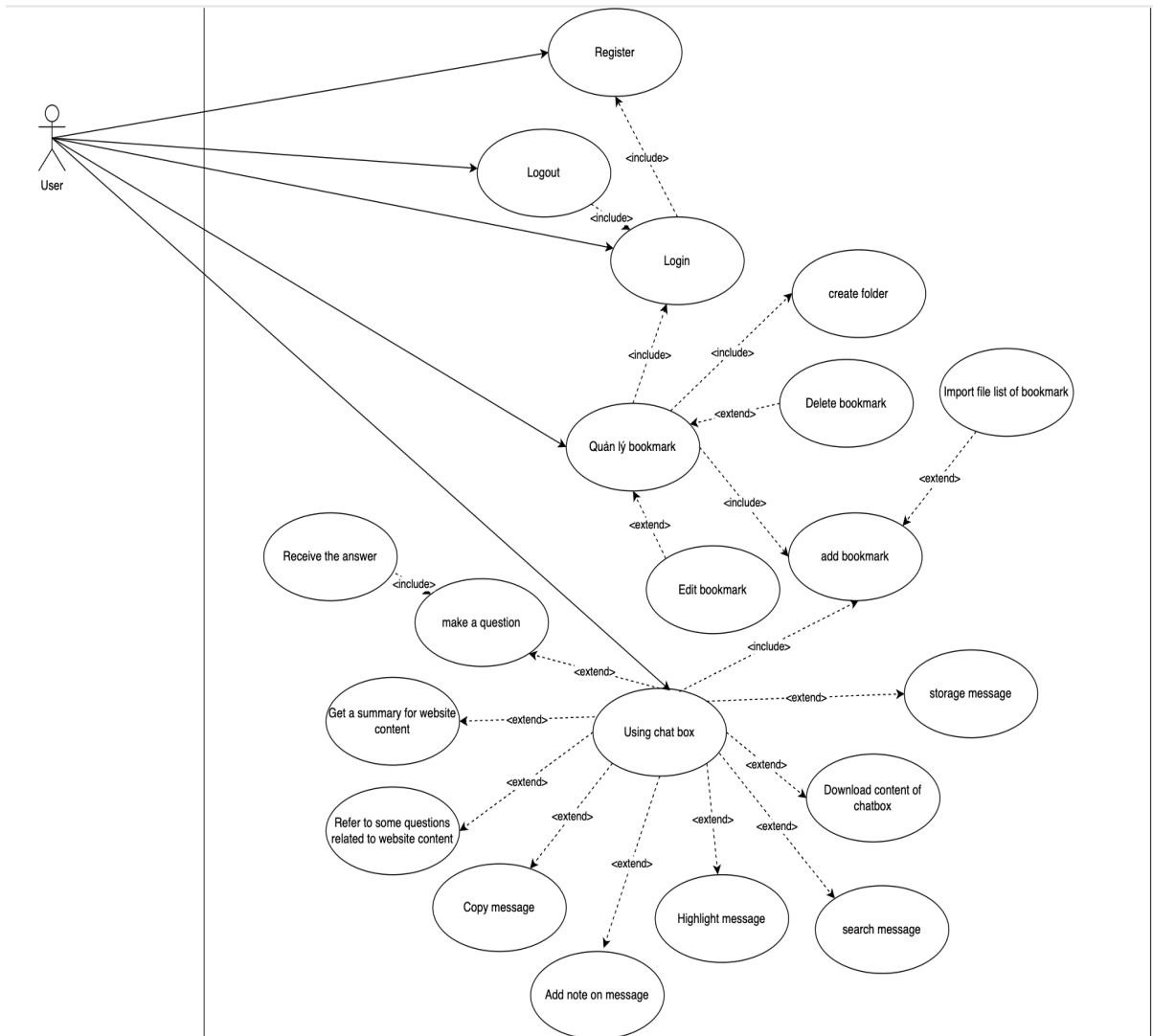


Image 2.2: Overall use case diagram of the bookmark management system with integrated GPT chat.

The login use case specification

Table 2.1 Login use case specification.

Use case name	Log in
User	User
Describe	Allows users to log in to the system.
Activation conditions	Users accessing pages need to log in.
Implementation steps	1. Enter the login name and password. 2. The system checks that the input data is correct. 3. Taken to the home page.
Result	Successfully logged in and taken to the home page
Error case	1. Enter the login name and password information. 2. The system checks is the username and password are entered incorrectly. 3. Error message appears on the screen.

Bookmark management use case specification

Table 2.2: Specification of the IAM system account group permission editing use case.

Use case name	Manage bookmark
User	The user already has an account.
Describe	Allows users to manage and perform the functions of adding, editing, deleting, importing and searching for bookmarks.
Activation conditions	User logged in successfully.
Implementation step	1. User logs in. 2. The system will retrieve the list of bookmarks and display it to the user. 3. Users use functions to manage their bookmark list. 4. The system stores information when users perform bookmark management functions. 5. Return the bookmark list screen.
Result	Data changes.
Error case	

Chatbox use case specification

Table 2.3: Features of using chatbox.

Use case name	Using chatbox
User	The user already has an account.
Describe	Chatbox provides users with a summary and questions related to the website's content when the user creates a chatbox. Ask questions and get smart answers from the chatbox. Allows users to use features integrated into the chatbox such as: copy, highlight, create notes, store chatbox responses, search for messages in the chat by keywords, download the entire content chat as .txt file.
Activation conditions	The user successfully logs in, has at least one bookmark in the bookmark list, clicks on the "eye icon" button on the toolbar of each bookmark to create a chatbox from that bookmark.
Implementation step	1. The user logged in successfully. 2. The person who clicks on the view button on the bookmark's toolbar wants to create a chatbox. 3. The system will automatically create and display to the user the first message with the content including a summary of the content and three questions related to the content of that website. 4. After that, the user can: Either enter the question that they want to ask related to the website content or click on the "sent icon" behind each question that the chatbox has previously created to receive the answer from the chatbox. 5. The system receives questions, processes and answers the user. If the answer is not found in the web page being read, the chatbox will display the message "Answer not found: 6. Users hover over each message to display the toolbar. 7. User clicks on the icon on the toolbar (each icon will present a corresponding function) to interact with the

	chatbox: such as storing, copying, highlighting answers from the chatbox, searching, downloading chat content. 8. The system stores information. 9. Notification of successful information saving.
Result	Successfully created chat and saved changed data.
Error case	

2.2.3. ACTIVITY DIAGRAM

Activity diagram is a form of diagram that generally describes the steps of activities or a series of activities to perform a system function.

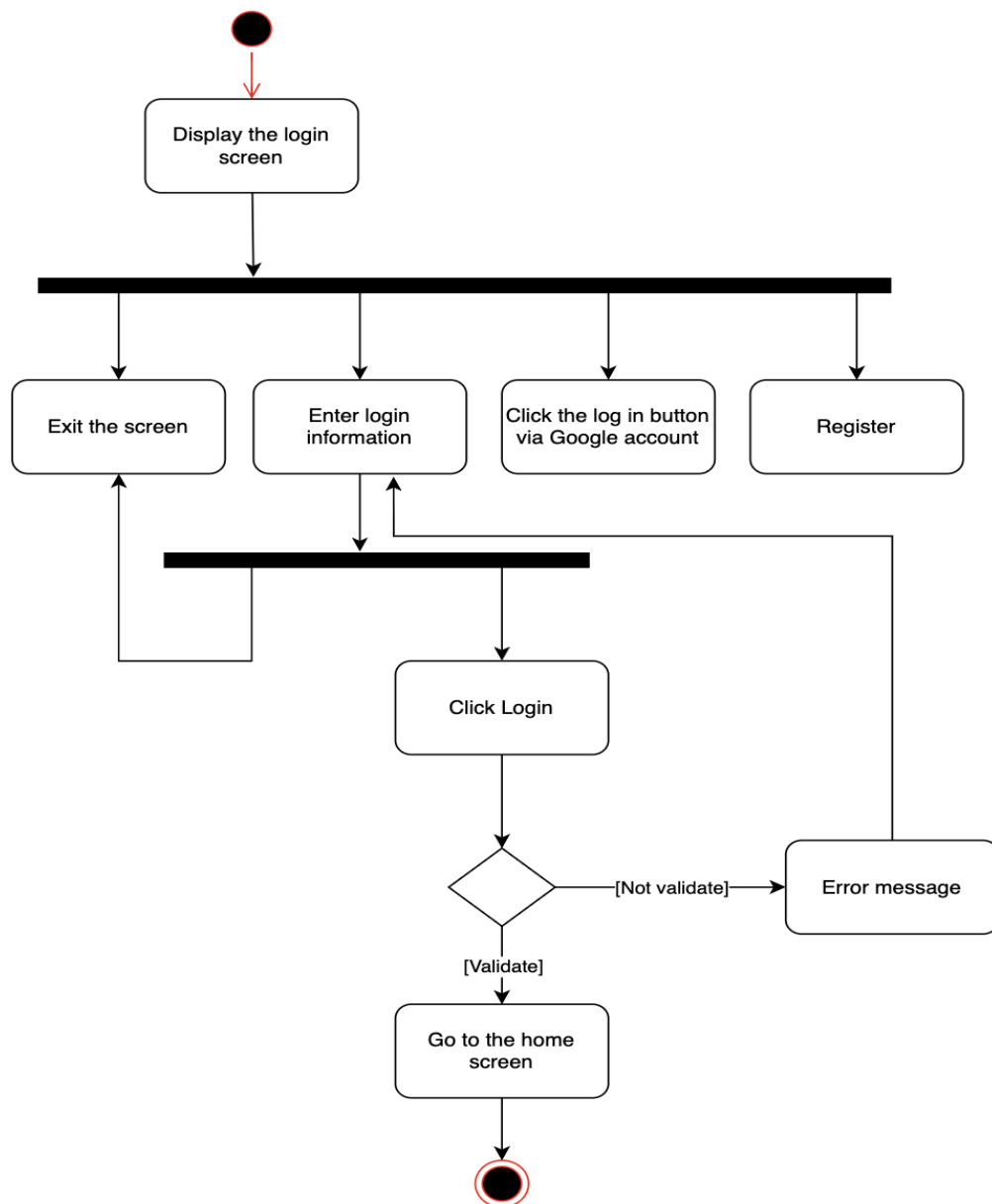


Image 2.3:Activity diagram describing the login functionality.



Image 2.4: Activity diagram describing the bookmark management functionality.

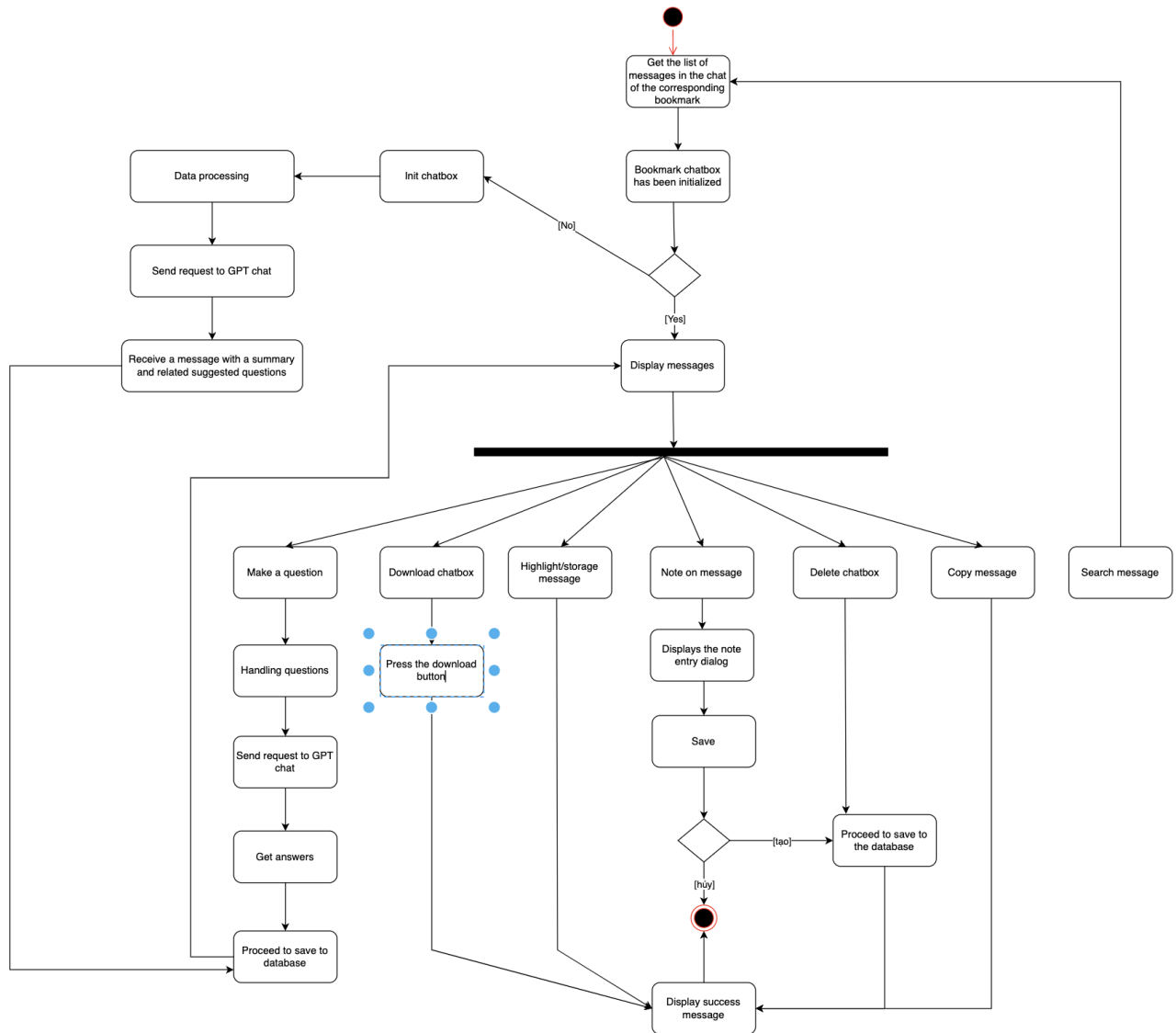


Image 2.5: Activity diagram describes the functionality of using the chatbox.

2.2.4. SEQUENCE DIAGRAM

Sequence diagrams use software and database design to ensure that the system will perform as required and assist in finding potential errors before implementation.

The sequence diagram describing the Login Via IAM function is depicted in the following figure:

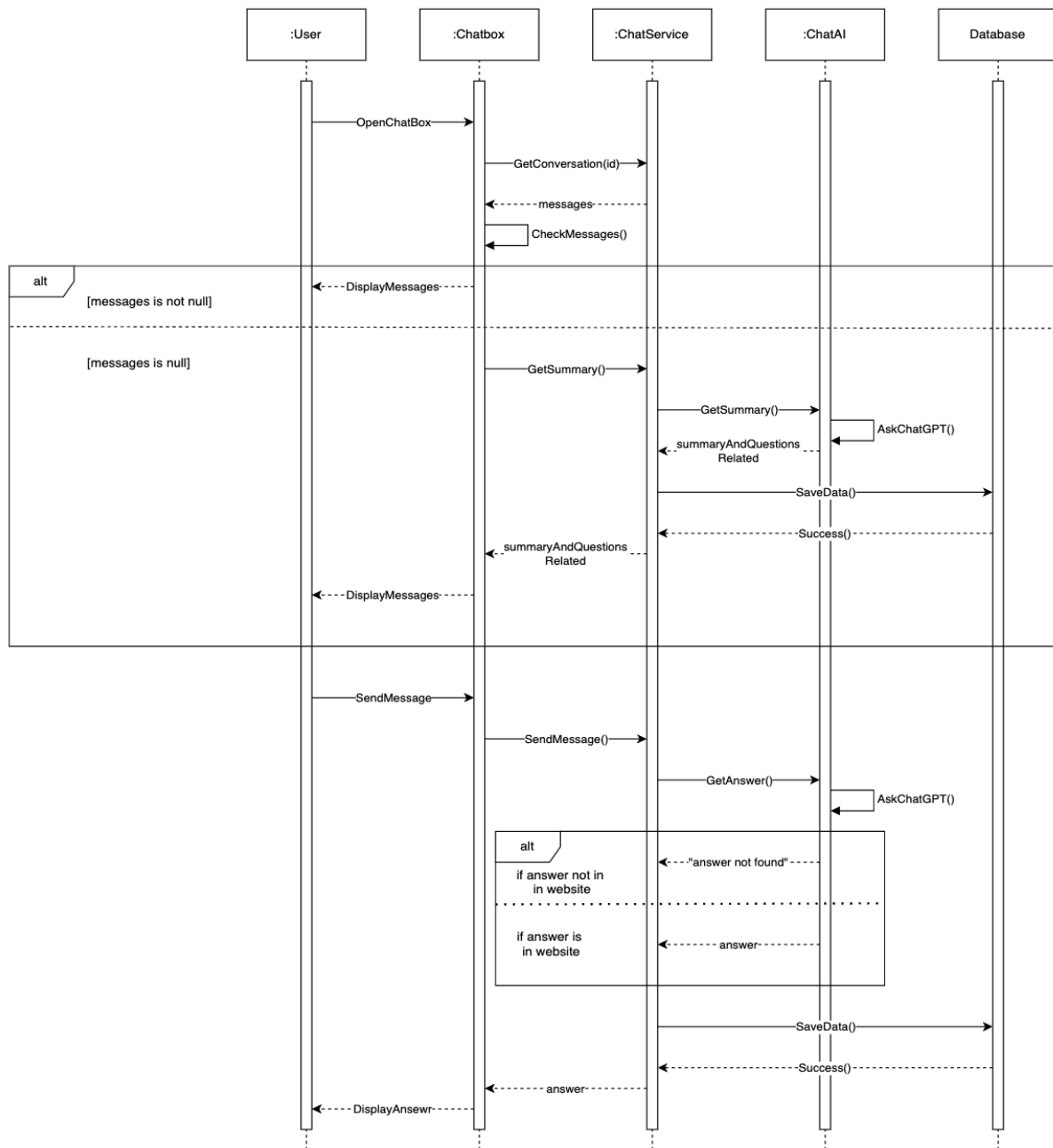


Image 2.6: Sequence diagram depicting the bookmark management system with integrated GPT chat.

2.2.5. DESIGN CLASS CHART AND DATABASE

A class is a collection of objects that are similar in behavior, structure, etc. Class diagram is a diagram that describes the types of objects in the system and the types of relationships that exist. between them. They describe a static view of the

entire or partial system, describing properties and methods in detail, giving the programmer an overview of the functionality being developed.

The class diagrams of the system are depicted in the following figures:

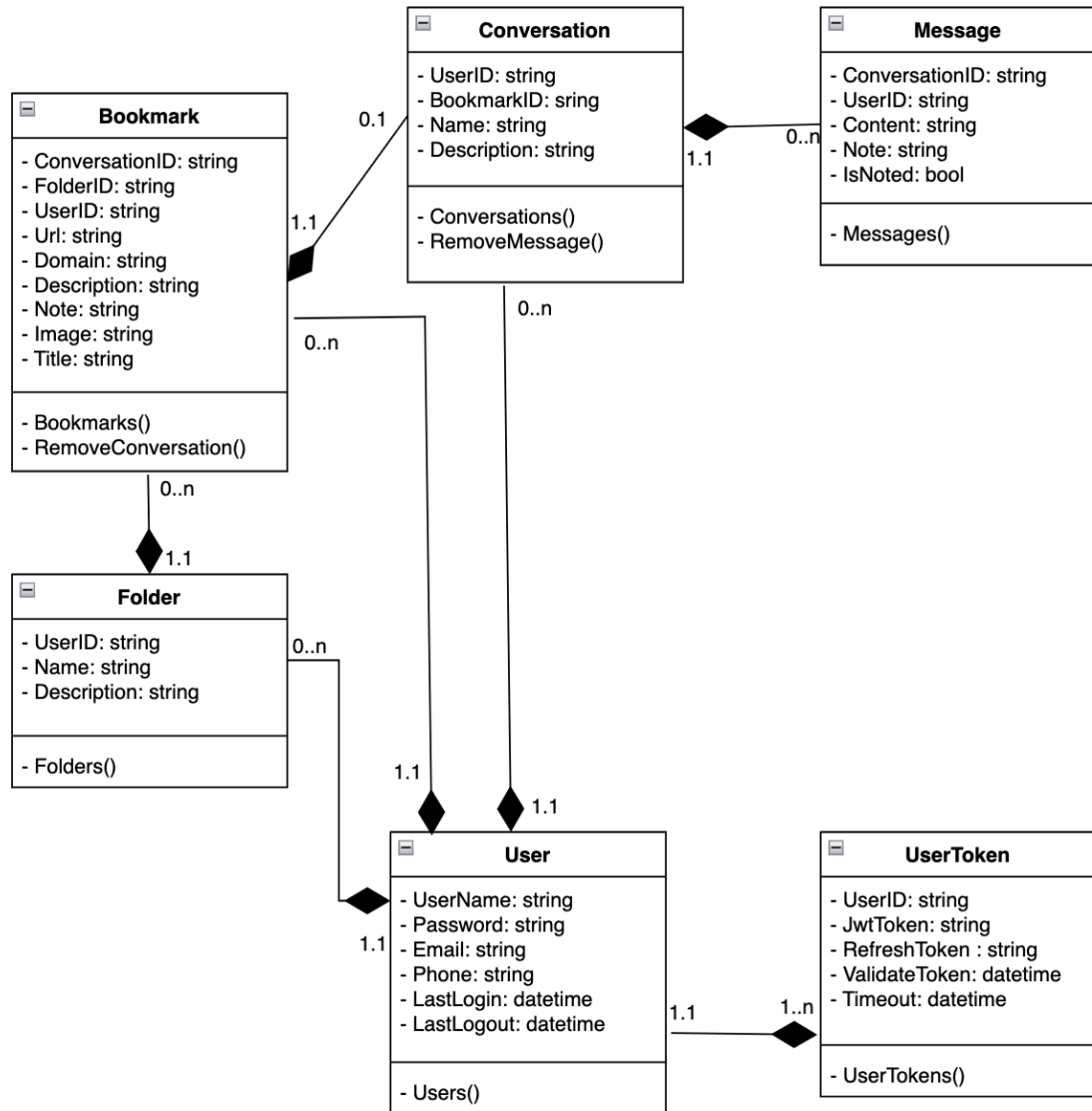


Image 2.7: Class diagram illustrating the functionality of bookmark management with integrated GPT chat.

The tables in the integrated management function are described in the following table:

Table 2.4: Table Description Table Bookmark.

Column name	Explain	Data Types	Note
Id	Id Bookmark	nvarchar	Primary key
ConversationID	Conversation code(chatbox)	nvarchar	Foreign key
UserID	User code	nvarchar	Foreign key
Url	Url of website	nvarchar	
Domain	Domain of website	nvarchar	
Description	Describe the website	nvarchar	
Note	Note of user when create a new bookmark	nvarchar	
Image	Icon of the website	nvarchar	
Title	Title of the website	nvarchar	

Table 2.5: Table Description table Conversation.

Column name	Explain	Data type	Note
Id	Id Conversation	nvarchar	Primary key
BookmarkID	Bookmark code	nvarchar	Foreign key
UserID	User code	nvarchar	Foreign key
Name	Name of the chat	nvarchar	
Description	Describe of the chat	nvarchar	

Table 2.6: Table Description table Message.

Column name	Explain	Data type	Note
Id	Message code	nvarchar	Primary key
ConversationID	Conversation code	nvarchar	Foreign key
UserID	User code	nvarchar	Foreign key
Content	Content of the message	nvarchar	
Note	Note of the user	nvarchar	
IsNoted	Whether the message is in the user's archive or not	boolean	

Table 2.7: Table Description table Folder.

Column name	Explain	Data type	Note
Id	Folder code	nvarchar	Primary key
UserID	User code	nvarchar	Foreign key
Name	Folder name	nvarchar	
Description	Description	nvarchar	

Table 2.8: Table Description table User.

Column name	Explain	Data type	Note
Id	Mã User	nvarchar	Primary key
Username	Name of user	nvarchar	
Password	Password of user	nvarchar	
Email	Email of user	nvarchar	
Phone	Phone number of user	nvarchar	
LastLogin	Last time Login	Datetime	
LastLogout	Last time log out	Datetime	

Table 2.9: Table Description table UserToken.

Column name	Explain	Data type	Note
Id	UserToken code	integer	Primary key
UserID	User code	nvarchar	Foreign key
JWTToken	Token	nvarchar	
RefreshToken	Refresh token	nvarchar	
ValidateToken	Validate token	nvarchar	
TimeOut	Timeout	integer	

2.2.6. INTRODUCTION OF FUNCTION

GPT chatbot integrated bookmark management system is a multi-functional website, helping users organize and interact with information on the internet conveniently and effectively. With flexible bookmark management, intelligent GPT chatbot integration, and personalized customization features, users can easily add, modify, organize, and extract information from web pages easily. convenient and quick way. At the same time, the ability to note, store and share information helps create an environment of social interaction and collaboration in research and learning on the internet.

2.2.7. CONNECTION MODEL WITH GPT API



Image 2.8: Overview of the connectivity model.

The system integrates with the GPT API to leverage the capabilities of the GPT chat for providing intelligent responses and assistance. This section outlines the connection model with the GPT API and how it is utilized within the bookmark management system.

Overview of the Connection Model:

- **API Key Authentication:** The system uses an API key to authenticate with the GPT API. This key is securely stored and managed within the system to ensure authorized access.
- **Request handling:** When a user interacts with the chatbox, the system constructs a request containing the user's query and sends it to the GPT API. The GPT API processes the request using its language model and returns a response. The system then processes the response, formats it appropriately, and displays it to the user within the chatbox interface.
- **Error handling and retries:** The system includes mechanisms to handle potential errors, such as network issues or API rate limits. In case of an error, the system retries the request or informs the user about the issue.

Implement detail:

- **API client:** The system includes an API client module responsible for communicating with the GPT API. This module handles the construction of requests, sending them to the API, and processing the responses.
- **Asynchronous Communication:** To ensure a smooth user experience, the system uses asynchronous communication when interacting with the GPT

API. This prevents the user interface from freezing while waiting for a response from the API.

- Security Considerations: All communications with the GPT API are encrypted using HTTPS to protect data in transit. API keys are stored securely and are not exposed in client-side code.

By integrating the GPT API, the bookmark management system enhances its functionality, providing users with intelligent and contextually relevant assistance as they manage and interact with their bookmarks.

2.3. CONCLUDE

This chapter conducts system analysis and design, providing an overview of the functions the system needs to fulfill. Deploy the design and construction of a Bookmarks management system with integrated GPT chat to ensure the set needs. At the same time, present diagrams showing the main functions of the system.

CHAPTER 3: EVALUATION

3.1. SOME IMAGES USED BY WEBSITE

Below are some images of the main functions of the bookmarks management system that integrates GPT chat.

3.1.1. LOGIN INTERFACE TO THE SYSTEM

Login screen to log into the system:

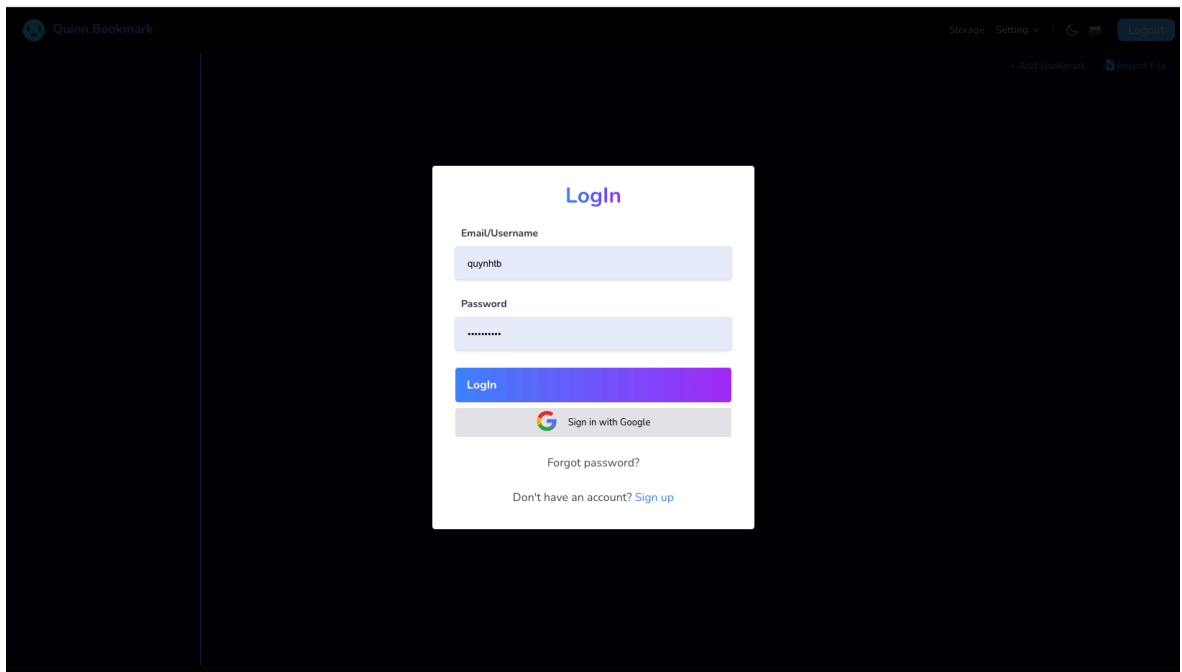


Image 3.1: Login screen.

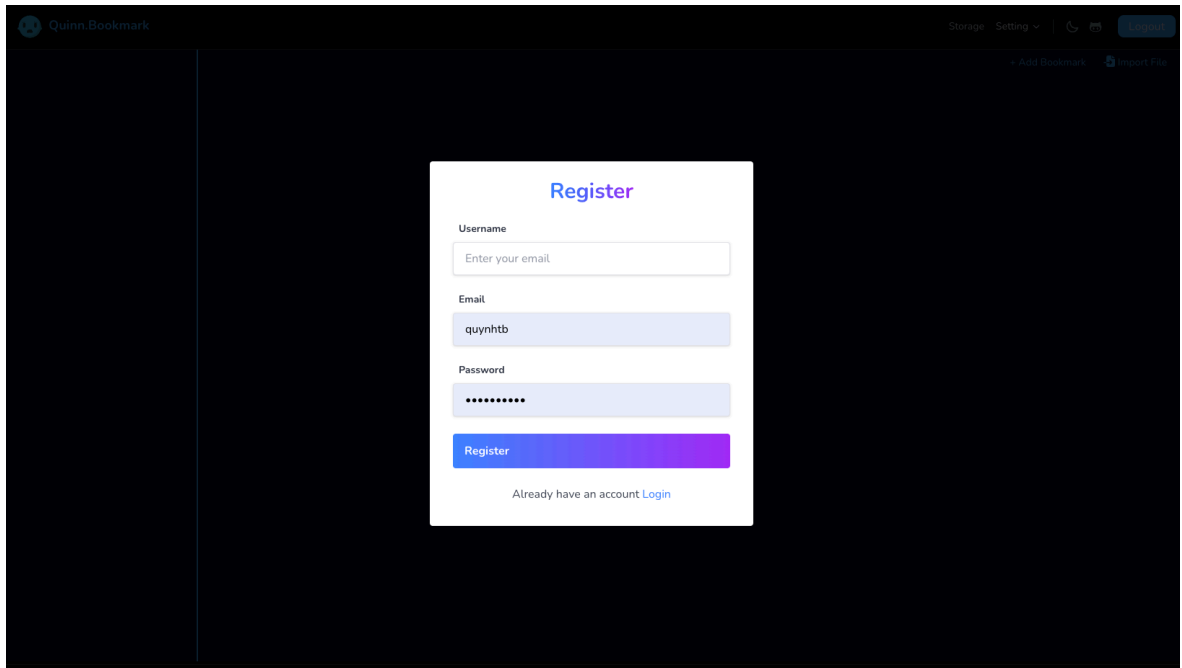


Image 3.2: Register screen.

3.1.2. BOOKMARKS MANAGEMENT FUNCTIONAL INTERFACE

Bookmarks management system with integrated chatGPT provides a bookmark list management screen.

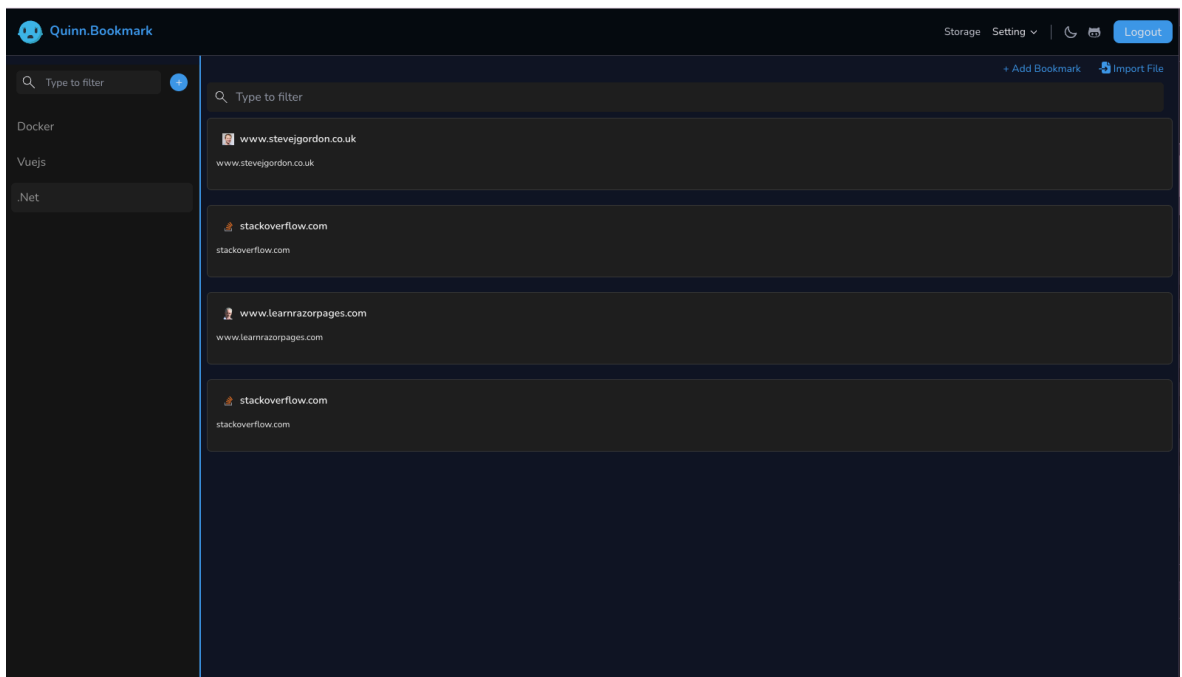


Image 3.3: Directory list and list of bookmarks within each respective directory.

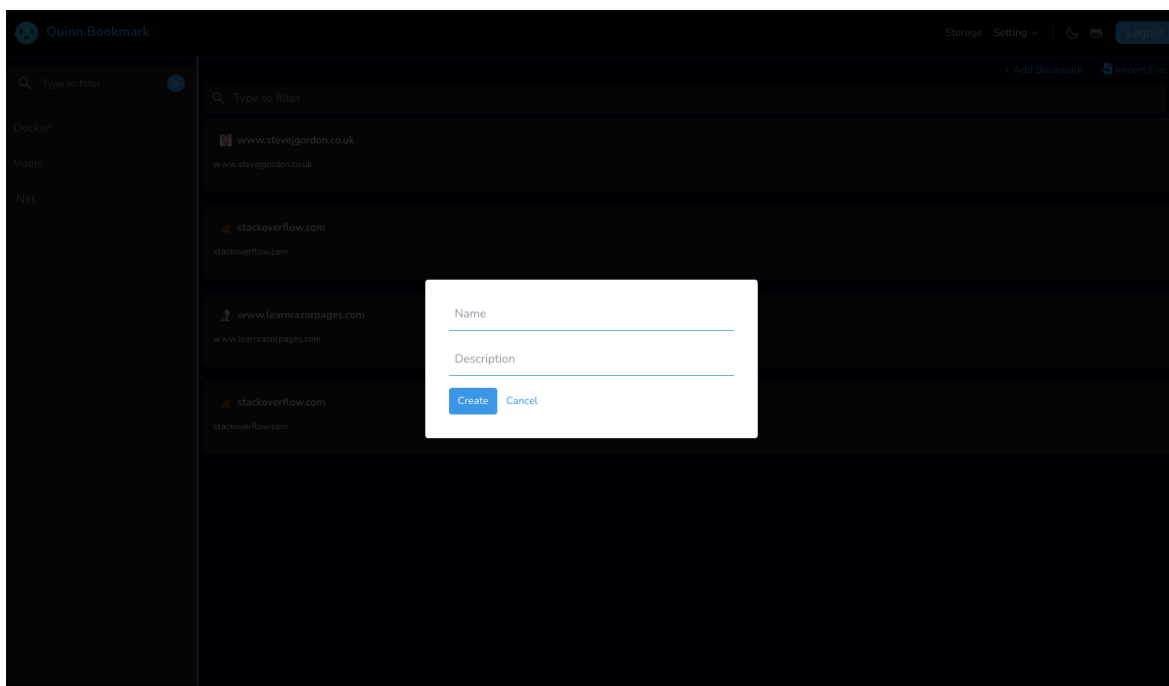


Image 3.4: UI create new folder.

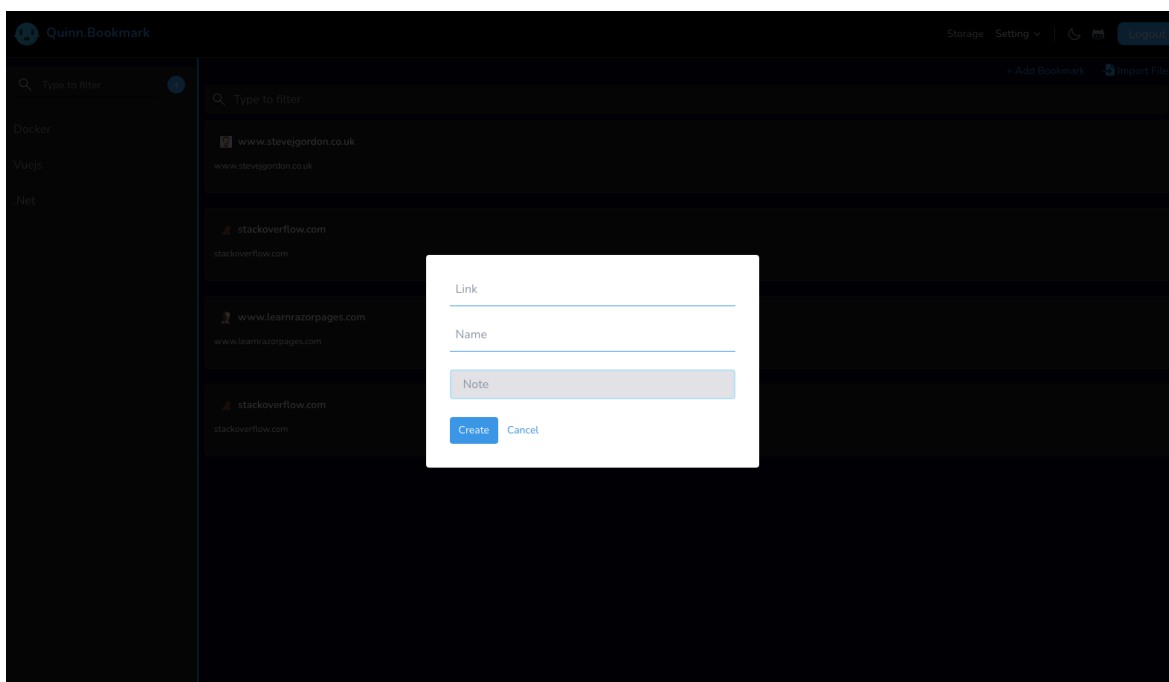


Image 3.5: UI create new bookmark.

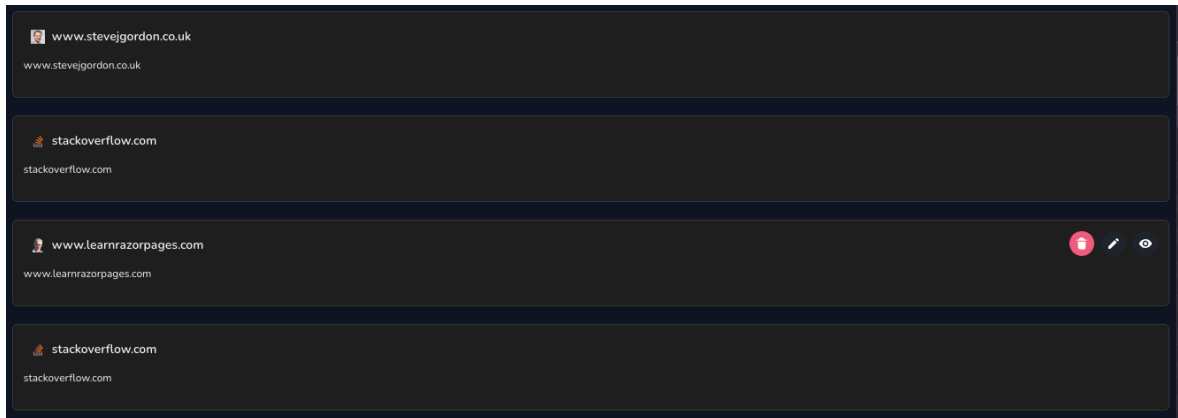


Image 3.6: Function bar on each bookmark.

The image above is the interface for managing the user's folder list and bookmark list:

- After the user successfully logs in, the system screen will display a list of folders and a list of bookmarks for each user's corresponding folder;
- Search folders and bookmarks: Users select the search bar, enter the keywords they want to search, the corresponding results will be displayed as a list;
- Create a new folder: Users click the button with the "+" symbol next to the folder search bar to open the dialog to add a new folder. Enter the necessary information, press the "Create" button to create a new folder or press "Cancel" to cancel the operation;
- Create a new bookmark: The user clicks the "Add bookmark" button in the upper right corner of the screen to open the dialog to add a new bookmark. Enter the necessary information, click the "Create" button to create a new bookmark or click "Cancel" to cancel. operation;
- Delete bookmarks: The user places the mouse cursor over each bookmark, the toolbar will display, select the button with the trash can icon on the toolbar, the corresponding bookmark will be deleted.

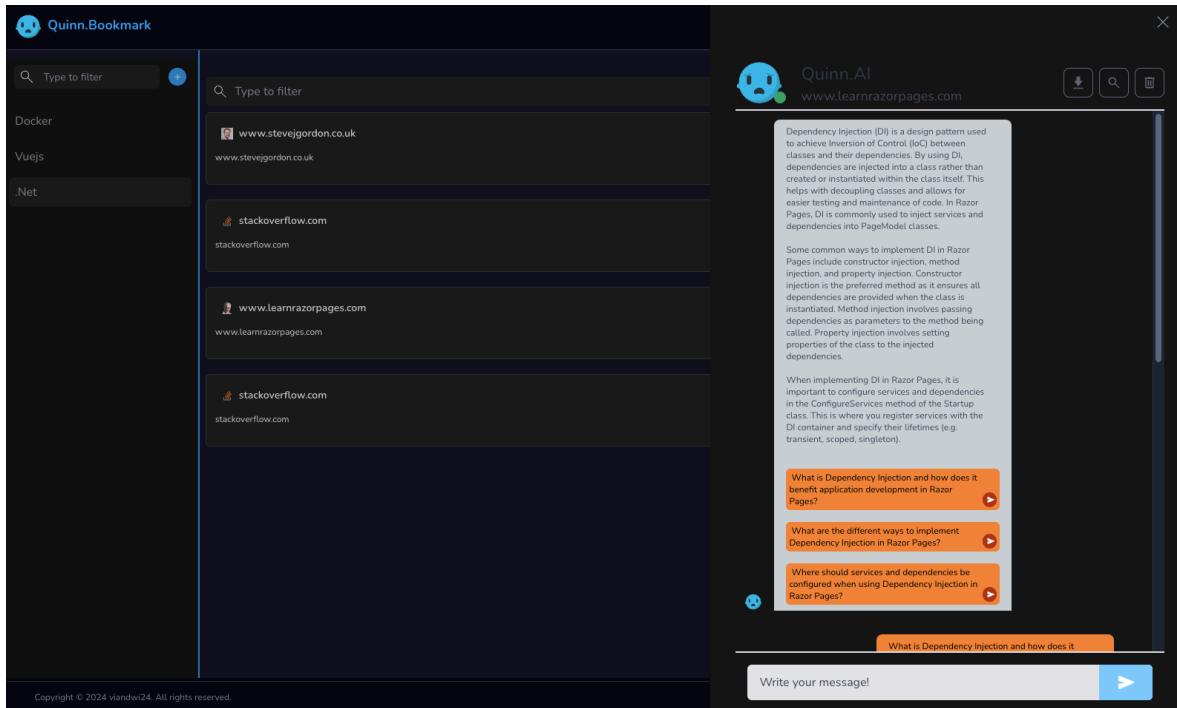


Image 3.7: Chatbox UI.

The user presses the button with the view icon on the toolbar (Figure 3.4) of the bookmark to open the corresponding chatbox for the content of that bookmark. After opening the chatbox. If the chatbox is initialized for the first time, the chatbox will automatically summarize the bookmark content and suggest questions related to that bookmark content to the user and will not if the chatbox has already been initialized (i.e. the user has opened it). chatbox of this bookmark before).

The chatbox integrates GPT chat to summarize content, suggest related questions and answer user questions related to the content of the corresponding bookmark. Provides features such as interacting with chatbox content such as: searching, downloading, deleting chats, storing answers, adding notes for each answer, quick copying.

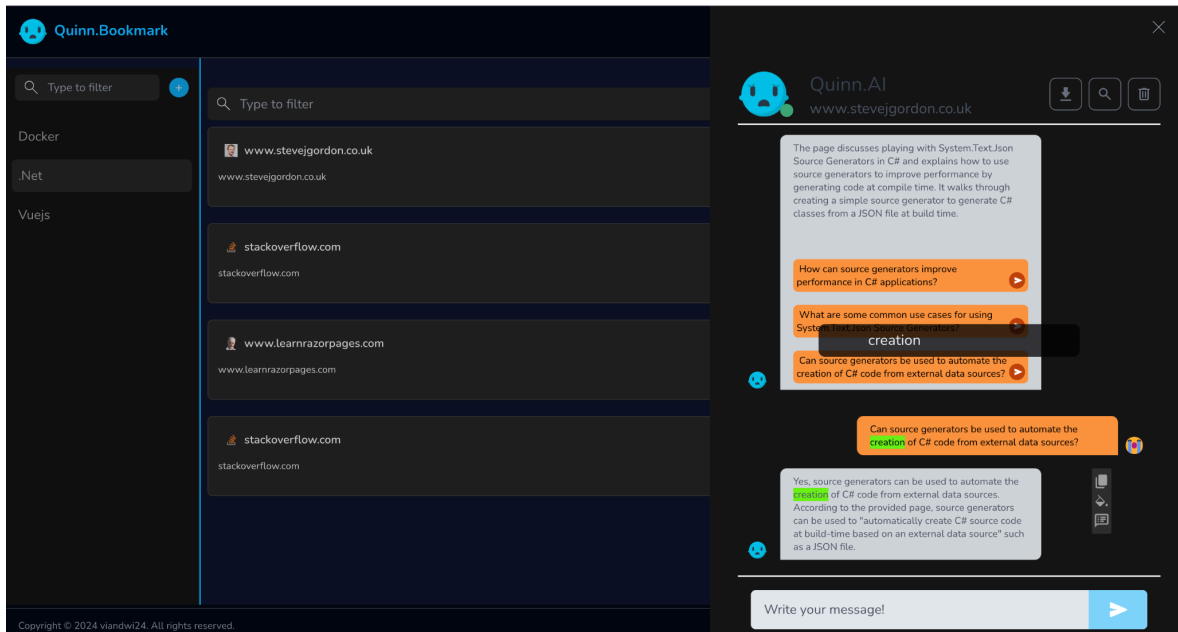


Image 3.8: Search function in chatbox.

- Search function in the chatbox: The user clicks on the pen with the search icon on the toolbar in the upper right corner of the chatbox. Enter the keyword you want to search for in the chat into the search bar. Press enter to search, press esc to exit. As a result, only messages containing keywords will be displayed and keywords in the message will be bolded (Figure 3.6);
- Function to delete chat: User presses the button with the delete icon on the toolbar to delete all messages in the chat;

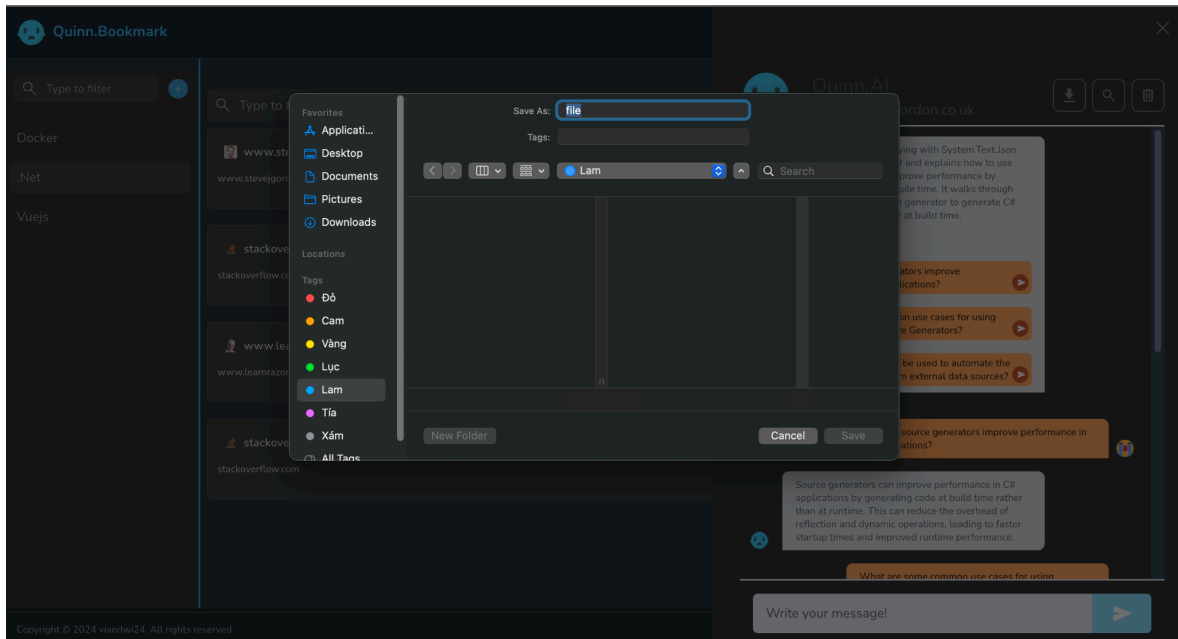


Image 3.9: Download content chatbox function.

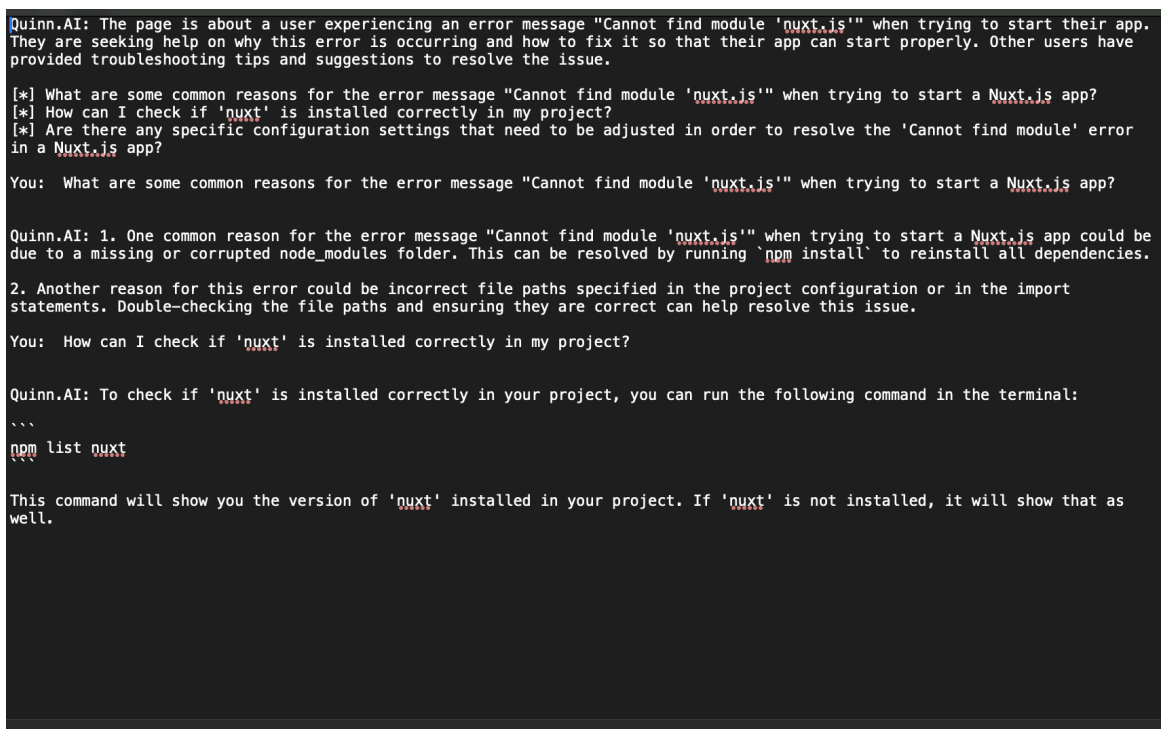


Image 3.10: Content of chatbox in file downloaded.

- Function to download chat content: The user presses the button with the download arrow icon on the toolbar, selects the folder to save the file and clicks "Save" to complete the download (Figure 3.7). The content of the file is downloaded. download is displayed as shown in Figure 3.8.

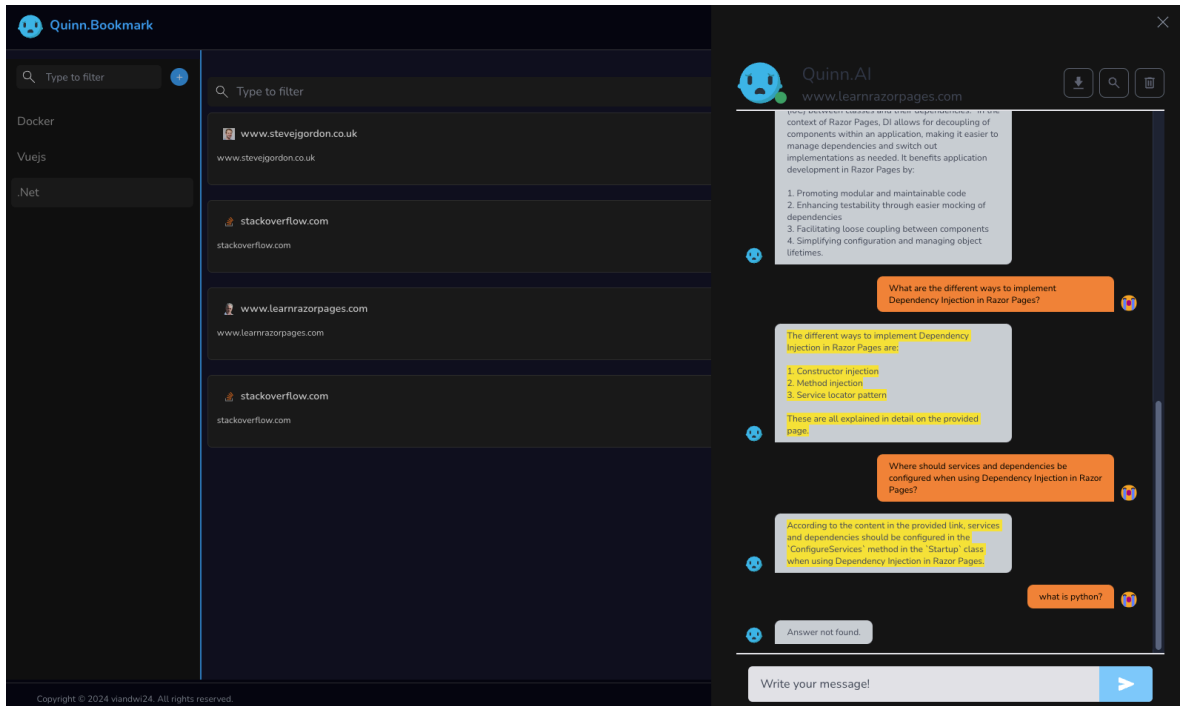


Image 3.11: Chatbox screen displaying responses to questions unrelated to bookmark content.

If the system does not find an answer related to the bookmark content, the chatbox will display the response "Answer not found".

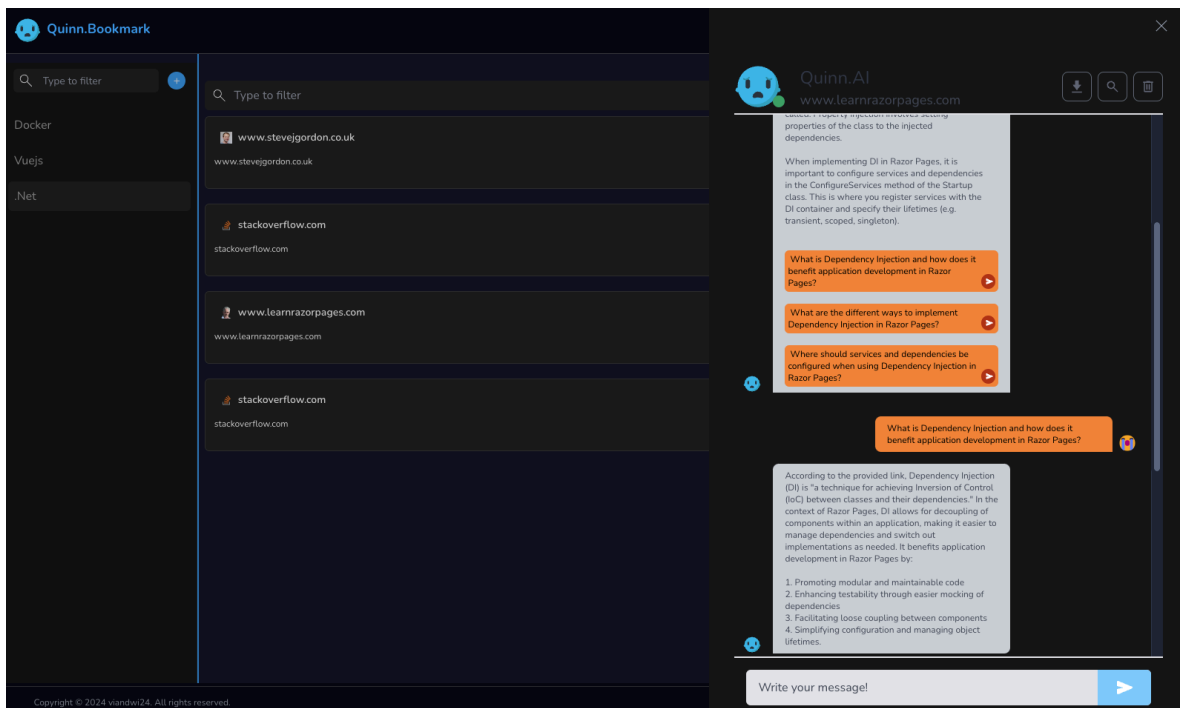


Image 3.12: The chatbox screen displays the answer to the user.

Users can ask questions to the chatbox in 2 ways: using questions suggested by the chatbox or entering the question they want to ask into the message input bar and pressing the send button:

- Using questions suggested by the chatbox: Users click on the button with the send message icon at the end of the question they want to ask in the list of questions suggested by the chatbox, then wait and receive an answer from the chatbox;
- Enter a question: The user enters a question by scratching the message input bar, entering the question they want to ask and pressing the send button next to it. Then wait and receive an answer from the chatbox.

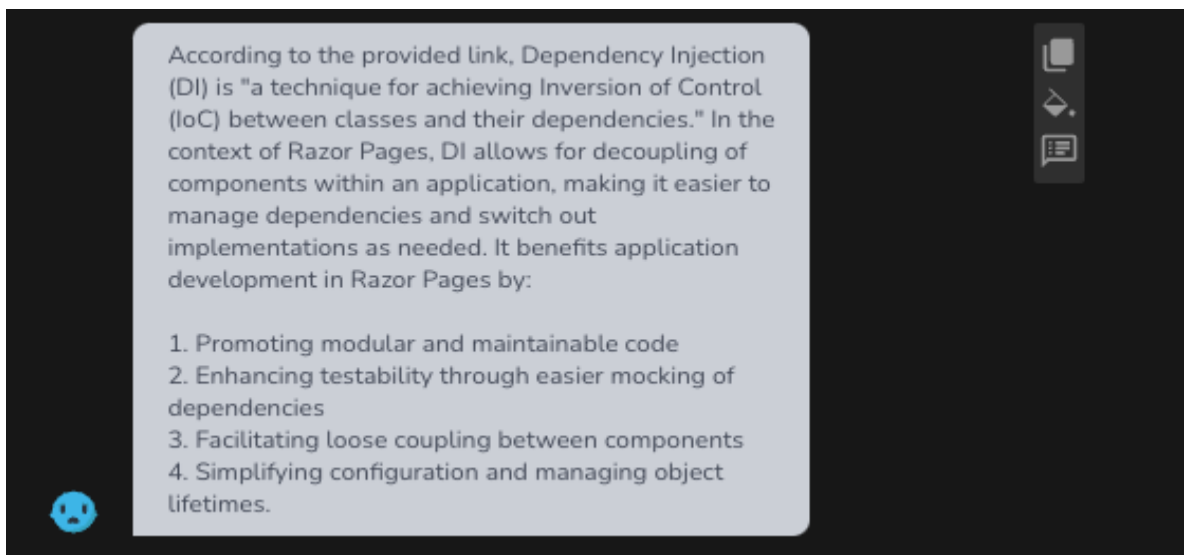


Image 3.13: Tools that users can interact with the responses.

When the user puts the cursor over each message interface, the toolbar will appear, including functions corresponding to each icon: copy chat, store and create notes.

- Copy message: Click the button with the copy icon on the toolbar (Figure 3.8) to copy the message content;

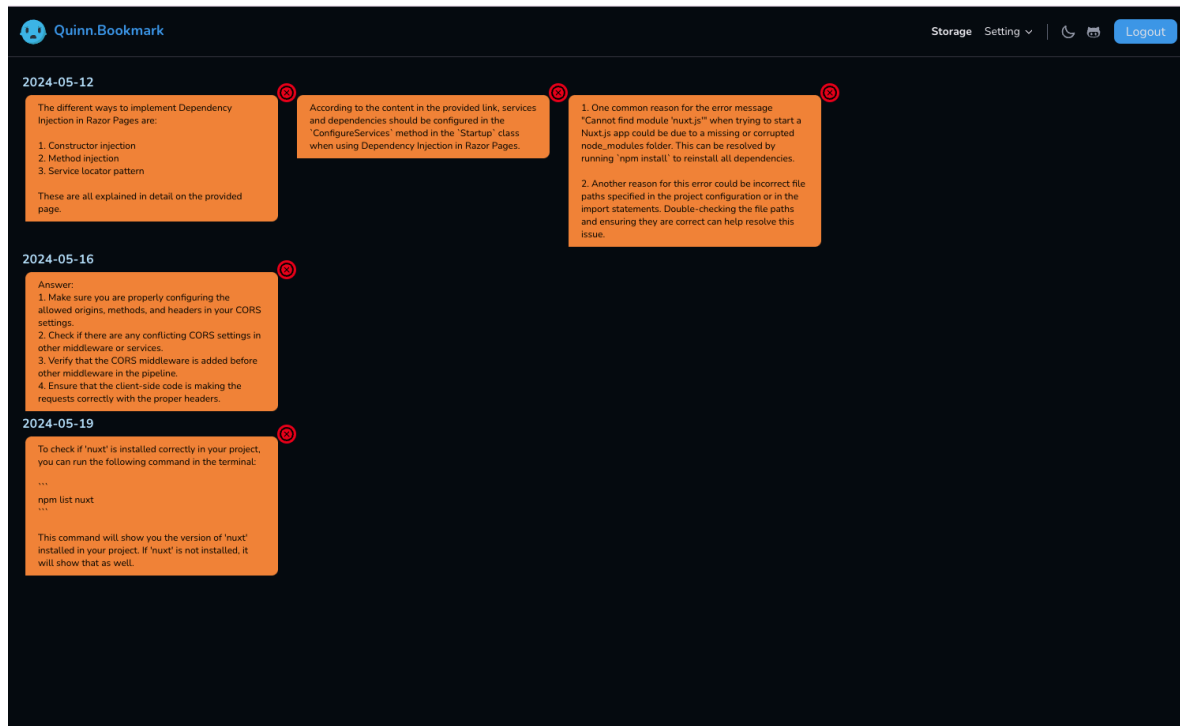


Image 3.14: Page containing stored messages.

- Archive messages: Click the button with the colored icon on the toolbar (Figure 3.8) to save the message to the user's message archive. Any messages that are archived will be marked in yellow in the chat. (as shown in figure 3.6). After the message is stored, the user clicks "Storage" on the menu bar to access the storage page (Figure 3.9);

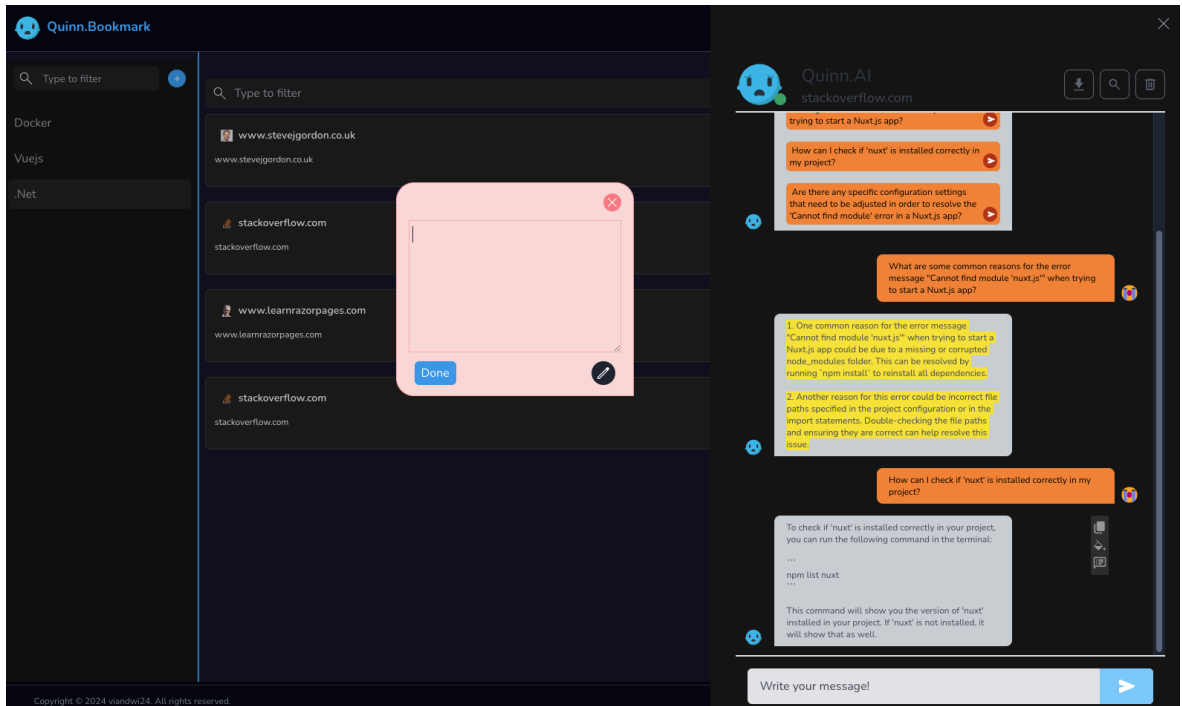


Image 3.15: Adding note.

- Note information on the chatbox answer: Click the button with the note icon on the toolbar (Figure 3.8) to open the note box. Click the “Done” button to save the note, or close the note box to cancel.

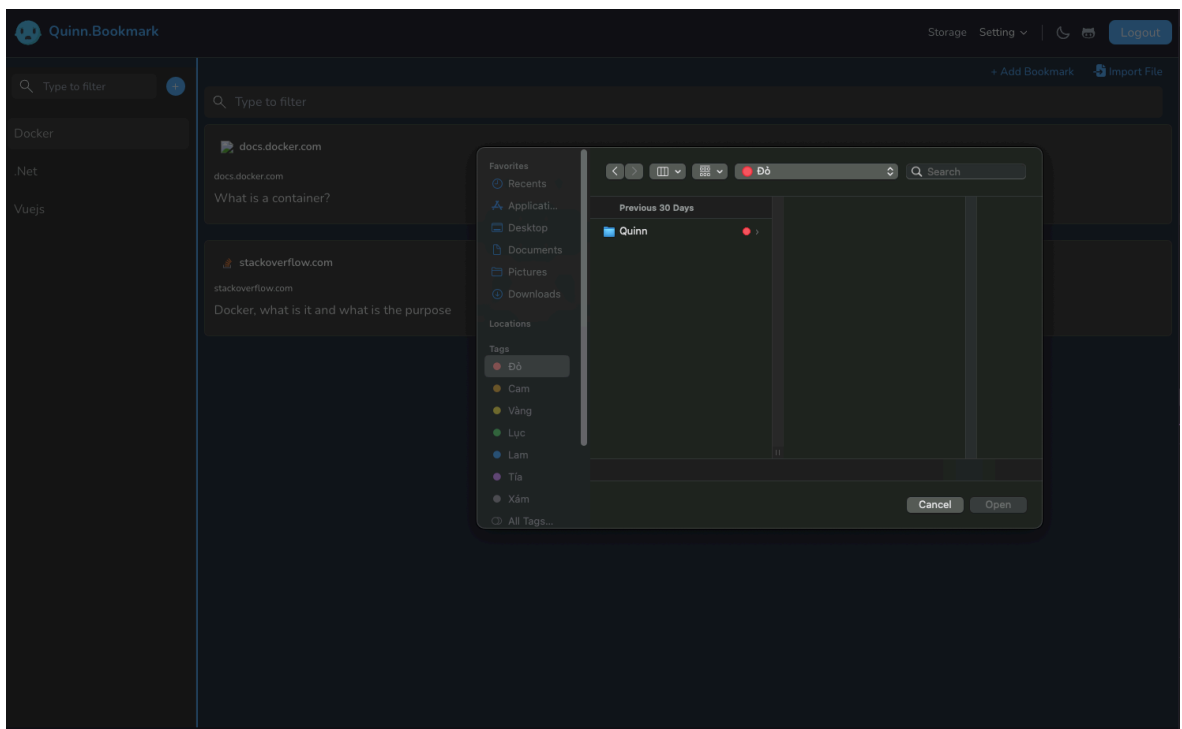


Image 3.16: Import bookmark list.

When the user clicks on the "Import file" button in the upper right corner of the screen, then selects the exported bookmark file from the Chrome browser to import that list file into the system. If the file is valid, the bookmark list will be displayed. displayed on the screen.

3.2. CONCLUDE

This chapter provides images of the website deployment interface in practice. Here, readers will have a more specific view on how to use a bookmark management website that integrates GPT chat when deployed in practice.

CONCLUSION AND DEVELOPMENT DIRECTIONS

1. RESULT

During the period of studying and researching the theoretical basis and implementing technology applications, the project achieved the following results:

Theoretically, the project applies Asp.Net core, Entity Framework, NuxtJs and Vuejs technology to build a bookmark management system that integrates GPT chat.

In terms of practical application, Built and deployed a bookmark management system that integrates GPT chat.

However, the project still has the following problems:

- The processing speed of the Website System is still slow, because the processing method on the server is not yet optimized;
- Some features are not yet finalized & developed;
- Practical applicability The Website system still does not meet the required amount of actual data.

2. DEVELOPMENT

Some research and development directions of the topic are as follows:

- Increase the processing speed of the Website System by optimizing the code;
- Implement features that do not yet exist, and at the same time perfect existing functions to fully meet practical needs;
- Increase server load capacity to handle large traffic volumes.

REFERENCES

- [1] SignalR, Link: <https://learn.microsoft.com/vi-vn/aspnet/core/signalr/javascript-client?view=aspnetcore-7.0&tabs=visual-studio>
- [2] The Component Object Model, Link: <https://learn.microsoft.com/en-us/windows/win32/com/the-component-object-model>
- [3] .NET Core, Link: <https://dotnet.microsoft.com/>
- [4] Language Integrated Query (LINQ), Link: <https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/>
- [5] EF Core and ORM, Link: <https://learn.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/work-with-data-in-asp-net-core-apps#ef-core-or-micro-orm>
- [6] Nuxt, Link: <https://nuxt.com/docs/getting-started/installation>
- [7] Vue 3, Link: <https://vuejs.org/guide/introduction.html>