COMP2043.GRP Final Group Report

Webpages monitor robot

Team202218:
Ruizhe HUANG(20320222)
Yuhao MA(20321640)
Shujun JIANG(20320552)
Chi MA(20318196)
Haowei GAO(20321547)

Supervised by Jiawei Li

School of Computer Science

University of Nottingham

Submit date: 06/04/2023

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Webpages Monitor Robot project requires the team to design a program that can regularly crawl user-specified search results from selected websites and alert users when user-specified search results are updated. Unlike the GRP project that monitored Moodle in the past, this year's stakeholder asks the team to monitor updates to academic papers. After communicating with the stakeholder, the team selects Google Scholar and Bing Scholar as our main monitoring sites. In addition, the team also monitors Bilibili video website and Sina News website to meet the needs of UNNC students.

The Webpages Monitor Robot project requires the team to carry out a thorough analysis of the problem to identify the key requirements and constraints of the system. The team needs to consider the different types of web pages to be monitored, the frequency of monitoring and the methods for detecting changes to these pages. The software engineering process involves a number of stages, including requirements gathering, design, implementation, testing and maintenance.

Standard software engineering practices is used such as module design, version control, and unit testing during coding and implementation and the testing phase includes both manual and automated testing to ensure that the program meets the functional and non-functional requirements outlined in the project requirements sheet.

Ultimately, with our team's ability to work effectively, communicate clearly and follow software engineering standards carefully, a complete and reliable WeChat Mini-program is managed to be produced at the end of the project to provide users with a valuable tool to monitor website content.

1.1 Aims and Objectives

The objective of this project is to develop a software robot that can identify specific changes on designated web pages and automatically deliver the retrieved information to the user on a regular basis. After consultation with stakeholder, the prototype of the software robot is identified as a comprehensive Crawler consisting of "Focused Web Crawler" and "Parallel Crawler"[30] and would have several key functions.

As a Focused Web Crawler, its basic function is to collect information related to the keywords given by the user[23][28]. To begin with, for users who want to find the latest papers in their research field, our program can crawl on different websites selected by the user. The program should periodically retrieve papers related to the user's topic of interest. Besides, for users who want to monitor videos with specific keywords, the program should periodically search for relevant videos on Bilibili video website. For users interested in specific news topics, they can enter keywords to the robot and the robot will retrieve news titles, links, media, upload time and complete news content. All updates should be notified to the user when updated and displayed in the program.

By automating the above tasks, the project robot aims to facilitate the viewing of updates to specific web pages, thus saving the user valuable time and effort. In addition, the robot has the potential to benefit a wide range of users, including people such as researchers who rely on up-to-date information for their work or personal interests.

1.2 Project Progress

During the first semester, the team completed most of the preparation and design for the entire software. At the request of stakeholder, in the first few weeks of the new semester, the team background research was conducted and relevant literature was read enable team members to understand the latest related technologies and analyze similar software. Subsequently, requirements from stakeholder and users from informal surveys were analyzed. Afterward, three preliminary websites were identified by the team members to crawl data and monitor changes, and the initial project would produce a WeChat mini-program. Then a prototype of the mini-program was made. Through the unremitting efforts, all the above tasks were completed and backend Python code for Google Scholar, Bilibili, and Sina News were written. During the winter vacation, the progress of the project was not stopped as well. Clear divisions of labor was made, determined Taro as the front-end framework and Flask as the back-end framework, and learned relevant technical knowledge of front-end and back-end. After the start of the next semester, the development process was accelerated, when further modifications to requirements was made, monitoring of Bing Academic website was added, and finally determined three sections: Academic, Bilibili and Sina News. After completing each part, the code is integrated and debugged and corresponding tests were conducted to finally complete the project. Throughout the process, there was adequate communication with our supervisor through formal weekly meetings and cooperation through a large number of informal meetings every week while attending GRP lectures on time. In summary, this project was completed through full cooperation and communication.

1.3 Report Outline

The report consists of ten chapters. Chapter 2 introduces the background and the related work. Chapter 3 shows the requirement specificaton of the project. Chapter 4 describes the design of the whole project. Chapter 5 introduces the specific development strategies used in the software. Chapter 6 states comprehensive description of the implementation.

Chapter 7 explains the test of the whole program. Chapter 8 is mainly about the software development experience, and evaluates the teamwork and project processes.

Chapter 2

Background and Related Work

2.1 Background

With the increasing use of the Internet, users can more easily use different hyperlinks to find the resources they need. These uses of the Internet led to the invention of web surveillance crawlers [27]. But at the same time, WWW provides people with a lot of useful electronic information in the form of hypertext. This huge hypertext library is dynamic and unstructured in semantics, which makes it difficult for users to find valuable information they really care about [22] [18]. In the current scenario, the majority of users utilize bookmarks within their web browsers to keep track of websites and obtain the latest updates[24]. If users want to be informed of changes to a webpage, they have to manually refresh the page periodically to see the changes, which is time-consuming and cumbersome in today's fast-paced life. Change Detection and Notification (CDN) systems make it easy for users to get notified about changes that have occurred on webpages, without having to refresh the webpage continuously [25]. Montastic [4], Follow That Page [16] are some of the most popular CDN services which are used by numerous users to get updates about content changes that occur on webpages [24]. A webpage is a separate file composed of text, graphics, and additional features such as links to scripts and style sheets [29]. While not all search engines work in exactly the same way, they perform at least the following

basic operations [26]:

1. Search the Internet for content that matches your keywords.

2. Record the location and index of the matched content.

3. Provide a search service (search server) to help find words or combinations of words in indexed databases.

Web crawlers are used to crawl the Internet, visit web pages, download their content for indexing, and improve them according to different needs. Figure 2.1 provides an overview of the web crawling process.

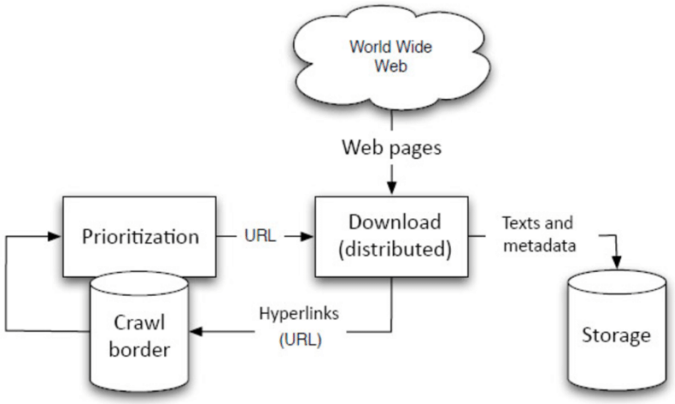


Figure 2.1: webcrawler process

For each page, it extracts elements considered significant and relevant to form a database of keywords relevant to the page being analyzed [19].

Our software utilizes crawler algorithms to crawl and monitor information changes from popular websites such as Google Scholar, Bing Scholar, Bilibili video website and Sina News, and stores the information in a database.

2.2 Competitor Analysis

In today's society, acquiring and updating data has become increasingly crucial. In order to meet the market demand, several tools have been developed to monitor updates to

website data. A majority of these tools possess the capability to monitor alterations on websites and update queries with the desired frequency as per the user's requisites. Here is an analysis of some competing products with similar features.

2.2.1 Visualping

Visualping [14] is a well-known website monitoring tool with widespread popularity and remarkable versatility. The software boasts a user base of over 1.5 million clients worldwide, carrying out an impressive 5 billion checks [1]. Its expansive array of features includes the ability to vary query frequencies from intervals of five minutes to a week, with updates for each recent change delivered through email or the user's preferred method. The tool also supports multiple modalities, such as visual text or elements, and offers a sleek and straightforward user interface. Technically, Visualping boasts multiple proxy options and the capability to perform investigations on protected pages. Nevertheless, as a commercial software, its pricing is comparatively higher than others in the market and it imposes a limit on the number of accounts per paying company, which is restricted for heavy website users. Apart from the disadvantages in terms of price, during our use of the site we found that it was not always possible to detect changes that occurred within the forms and that it occasionally suspended its service due to technical problems or upgrades, which affected his reliability, plus his email alerts were frequently delayed, which could make users miss updates.

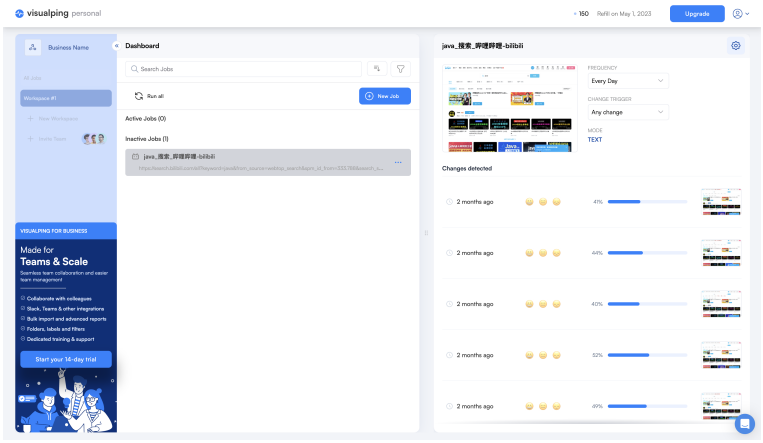


Figure 2.2: Overview of Visualping

2.2.2 Distill Web Monitor

The competitor to be discussed below is Distill Web Monitor [5], which, unlike the visualping, is used as a browser plug-in for users, it is user-friendly and easily accessible, with a simple installation process that is compatible with various web browsers. This monitor not only enables the monitoring of entire web pages but also allows local content monitoring of specific articles, paragraphs, or even single words that are updated whenever a change occurs. Additionally, the plugin supports monitoring the feed content of a website, thereby eliminating the requirement for external RSS feed tools. The minimum interval for automated shift checks is five seconds, and certain pages are capable of real-time change alerts. Despite its numerous advantages, it also has a few limitations. Its biggest disadvantage is because he is a browser plug-in, which leads to the user having to install a browser that supports this plugin in order to use it, and there can be more technical problems with the plug-in, which can be unstable or even crash. Moreover, it is not ideal for a large number of detections.

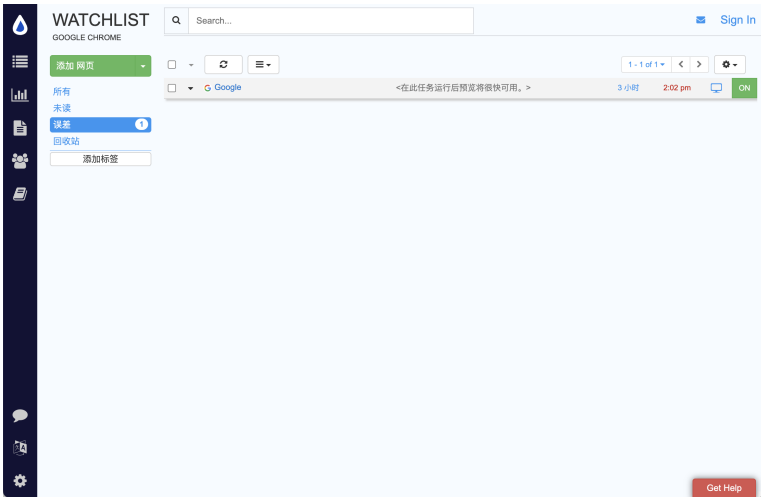


Figure 2.3: Overview of Distill Web Monitor

2.2.3 Conclusion

There are actually numerous additional tools available for this purpose, for example Uptime Robot and Uptrends, etc. Their functions are more or less the same, the difference can be in the price difference of the charges and the handling of some special cases, and

the clientele is slightly different. They are all based on the user's request for a website, and all pay a certain amount of money, The main difference between our project and them is that our main target audience is UNNC students and teachers, we have specified the site but not the keywords to be monitored, users can monitor for the keywords as they see fit, we will provide feedback on the updated search results based on the keywords monitored for. **Users can input specific keywords to monitor updates on academic, video, and news websites.** We focus more on specific websites, which reduces the error rate of monitoring and increases the stability of the results, and our program is deployed in the WeChat Mini-program platform, which suits most users and can be used directly without having to download it manually. Moreover, our program is free of charge and does not charge the user anything to perform an unlimited number of crawls.

Chapter 3

Requirement Specification

Date	Version	Reason For Changes	Brief Description
11/11/2022	version 1.0	/	Select Google Scholar, Bilibili, Sina News and Shixiseng as target sites
10/2/2023	version 2.0	based on informal research with users	Deleting Shixiseng internship section
3/3/2023	version 3.0	According to the advice given by supervisor on behalf of the stakeholder	add Bing Scholar merging it with Google Scholar to form the Academic section

Figure 3.1: Demand Update Record Form

3.1 Requirement Elicitation

The stakeholders of the project are school teachers represented by our supervisor and all the students of UNNC. First of all, in the meeting with the supervisor, we discussed the requirements of this project, analyzed the software requirements and made a preliminary decision. After that, in the informal survey and interview with friends and classmates around us, we identified the first version of the needs, and after confirming with the supervisor and getting a positive answer, we developed the first version of the project requirements.

In the first version of our requirements, we analyzed the overall architecture of the Mini-program and its various components, identified the objectives for building the WeChat Mini-program, and selected Google Scholar, Bilibili video website, Sina News, and Shixiseng (a job search website) as our targeted websites for monitoring.

After several meetings and requests from stakeholder, we updated the requirements for Version 2. We decided to remove the Shixiseng Life section and focus more on building the academic section. After the spring semester began, we completed the development of Google Scholar, Bilibili video website, and Sina News. Following this, at the suggestion of stakeholder, we updated the requirements for the third version. We decided to expand Google Scholar into an academic section and add more academic websites to be monitored, such as Bing Academic.

3.2 Requirement Validation

In this section, we discuss the reasons for the decision and the background research for each of the project's needs. As ethical discussion and reporting were not conducted within our group, we conducted several informal surveys and discussions with our friends in accordance with the rules. The majority of responses were gathered initially. The information obtained from the internet and our surveys will be summarized.

3.2.1 Wechat mini-program

In our project, we decide to use WeChat mini-program as the application scenario for our project. Being released in 2011, WeChat has been the dominant social media and payment platform in China. With the introduction of WeChat Mini Programs in 2017, WeChat has been further strengthened and integrated into the daily lives of Chinese citizens, which has become an integral part of daily life for many Chinese citizens, with over 400 million users accessing WeChat Mini Programs daily [15]. Wechat applet can be used without installation on any device, solve the constraints of user time and space, and enhance product utilization and timeliness [31]. For UNNC students, the WeChat Mini-program is also an integral part of the program. For example, we use the Mini-program Attendance as a tool for signing in to our classes, there is no need to go to the app shop to download the WeChat Mini-program, it is very lightweight. All respondents in our informal survey have used WeChat and WeChat Mini-programs.With the above in mind,

we decided to complete our project as a WeChat Mini-program.

3.2.2 Academic

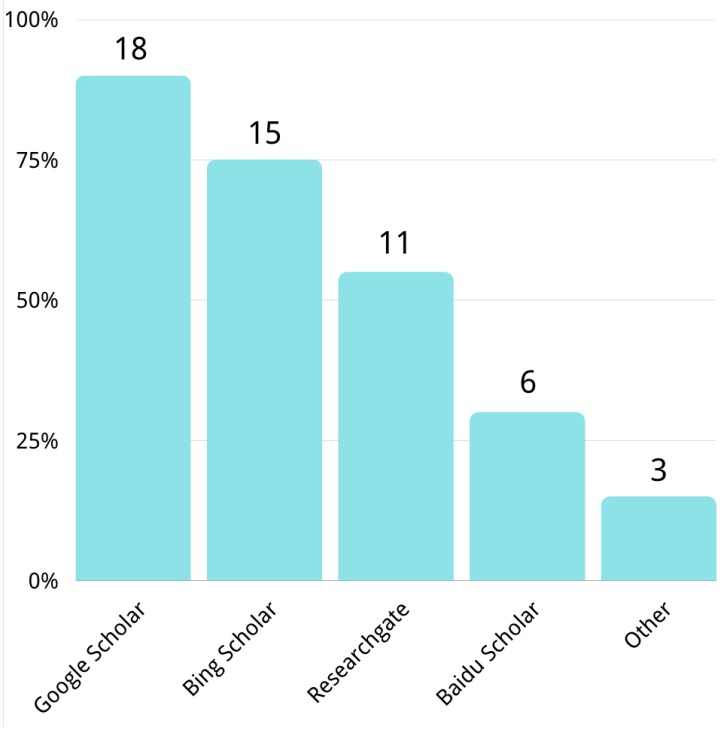


Figure 3.2: Respondents habitually use academic websites for theses (*Multiplechoicequestions*)

As the main part of our software, which is the directly requirement put forward by our stackholder, a variety of scholar websites should be considered to enrich the monitoring results. According to Figure 3.2, the proportion of Respondents habitually use academic websites of Google Scholar, Bing Scholar, Researchgate and Baidu Scholar are all bigger than 25 percent, meaning they are mainly using by people around world. Thus, they should be implemented in sequence.

Among them, Google Scholar whose ratio is the highest has achieved an unprecedented position as the largest academic search engine globally, with approximately 389 million documents available as of January 2018 [20]. Beside, it is also the prime requirement of our stakeholder. As a result, monitoring Google Scholar should be achieved first as the basic of this module.

Moreover, Bing scholar should be considered secondly. As the Figure 3.2 shown, the ratio

of it is also near 75 percent, resulting in the high possibility for Bing Scholar to contain essays which are not published in Google Scholar. Therefore, it should be implemented next as the supplement of Google Scholar.

3.2.3 Bilibili

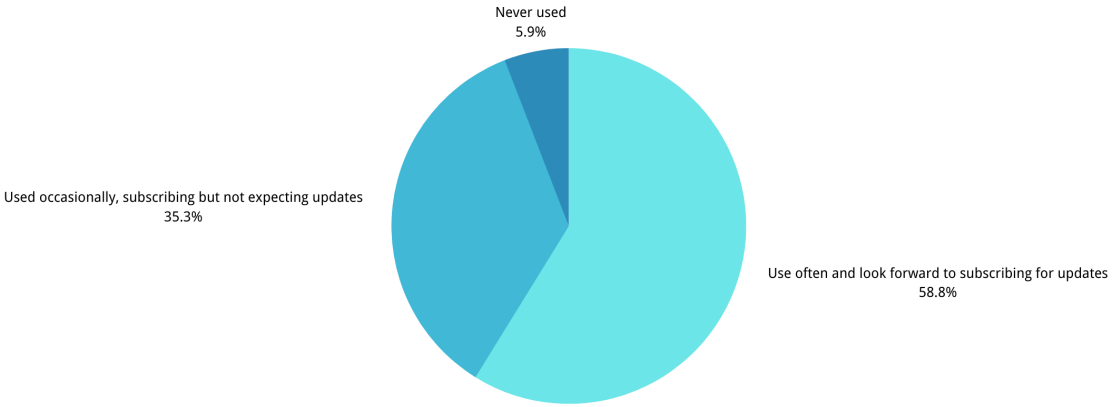


Figure 3.3: Survey respondents on the frequency of Bilibili use

Bilibili, a prominent Chinese video-sharing platform, has gained widespread recognition for its strong community engagement, vast range of content, and interactive features. The platform's demographic primarily consists of individuals in the younger age bracket [7], as evidenced by the fact that more than 90 percent of informal surveyed respondents from UNNC reported daily utilization of Bilibili for both leisure and educational purposes. Furthermore, a significant number of UNNC students are also creators on the platform, utilizing Bilibili's latest video offerings to enhance the quality of their content and stay updated with current trends. The ability of Bilibili to serve as both a platform for consumption and creation of content has contributed significantly to its popularity among users. Thus, in order to attract more teenagers to use our software and foster an immersive and engaging user experience, monitoring Bilibili should be implemented.

3.2.4 Sina News

Sina News, a prominent news platform in the Chinese digital media landscape, has secured a coveted position among the top three industry leaders, boasting a significant user base

of 146.1 million monthly active users as of August 2021 [8].Notably, Sina News maintains a high pace of updates for each category, enabling users to stay up-to-date with the latest developments in their preferred domains.

In a survey conducted to gauge users' interest in news from diverse domains, over 50 percent of respondents expressed a strong interest in news from all directions. Furthermore, more than 70 percent of the respondents underscored the significance of monitoring news websites for real-time keyword updates.

Given the platform's unparalleled reputation and immense popularity, it is imperative to gather data from Sina News through crawling to facilitate analysis and insights. As such, the decision to crawl Sina News is justifiable and can yield valuable information to benefit various stakeholders, including researchers, journalists, and media organizations.

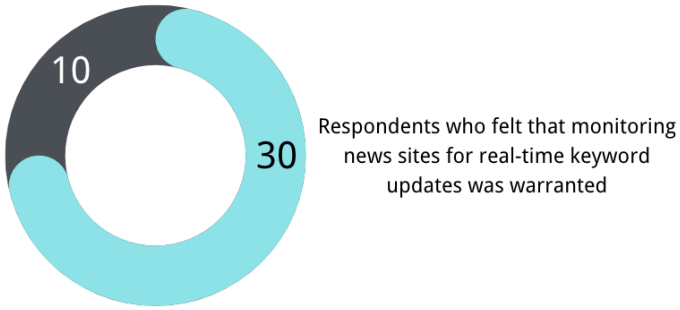


Figure 3.4: Survey respondents on monitor news sites for real-time keyword updates of interest

3.3 Software Requirements Specification

3.3.1 Introduction

1) Purpose

This project aims to develop a project that designed a robot to monitor changes on the webpages, and reports the user periodically. The robot should cover several aspects of learning life at UNNC, in order to facilitate teachers and students at the school and allow them to view the desired changes to the site at their convenience.

2) Intended Audience

The project is an Wechat mini-program for the students and teacher in UNNC.

3) Intended Use

Taking into account the needs of different users is the main aspects of development. For users who want to learn about academic papers, the program monitors Google Scholar and Bing scholar to remind the users about the latest updated papers in their field of interest. For users who want to watch videos, the software provides bilibili as a monitoring object, and for users who want to learn about news, the software provides Sina News as a monitoring object.

3.3.2 System Features and Requirements

1) Functional Requirements

1.Users can register and log in using their email address and the program can send a verification code to the user to assist with login and registration

2.Users can select different modules to use under login conditions

3.The user can enter keywords in the website module and the applet will monitor the original website for the keywords entered by the user and return the results

4.Users can receive the increased number of monitor results for their chosen time period in their mailbox

5.Users can select different time periods as the frequency of keyword crawling

6.Users can select different websites of google scholar or bing scholar in the academic section to crawl keywords regularly.

7.Users can perform regular crawls for search terms in the Bilibili section

8.Users can perform regular crawls for news search keywords in the Sina News section

9.Users can change their avatar in the personal section

10.Users can change personal information in the personal section, such as username and email

11.Users can contact the project team in the personal section

2) Non-Functional Requirements

1.The project's UI design should be simple and clear

2.The project should be able to run on platforms commonly used by users

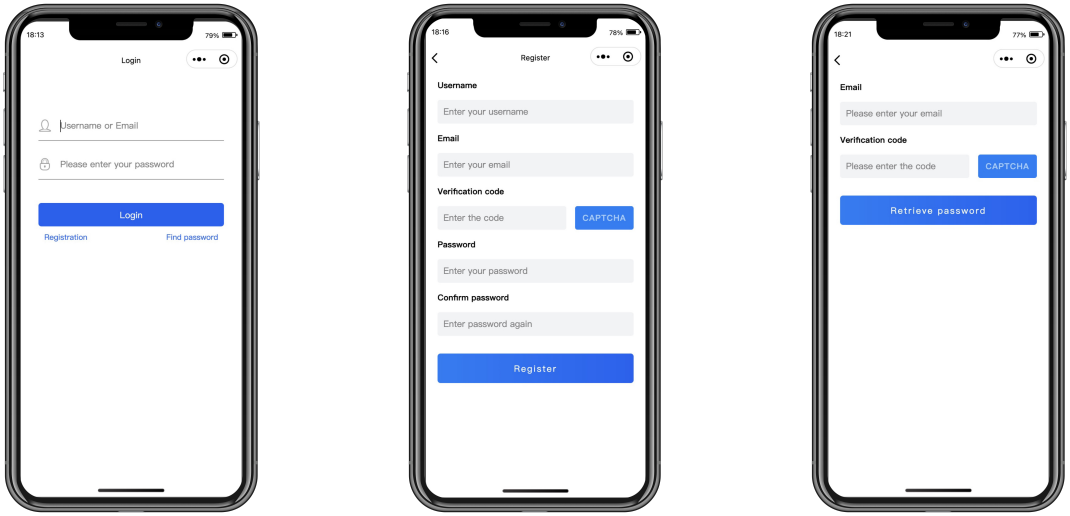
Chapter 4

Updated Design and Interface

4.1 User Interface Design

4.1.1 Login and Register

The figures shown below are the login, register and retrieve password pages of the mini-program.

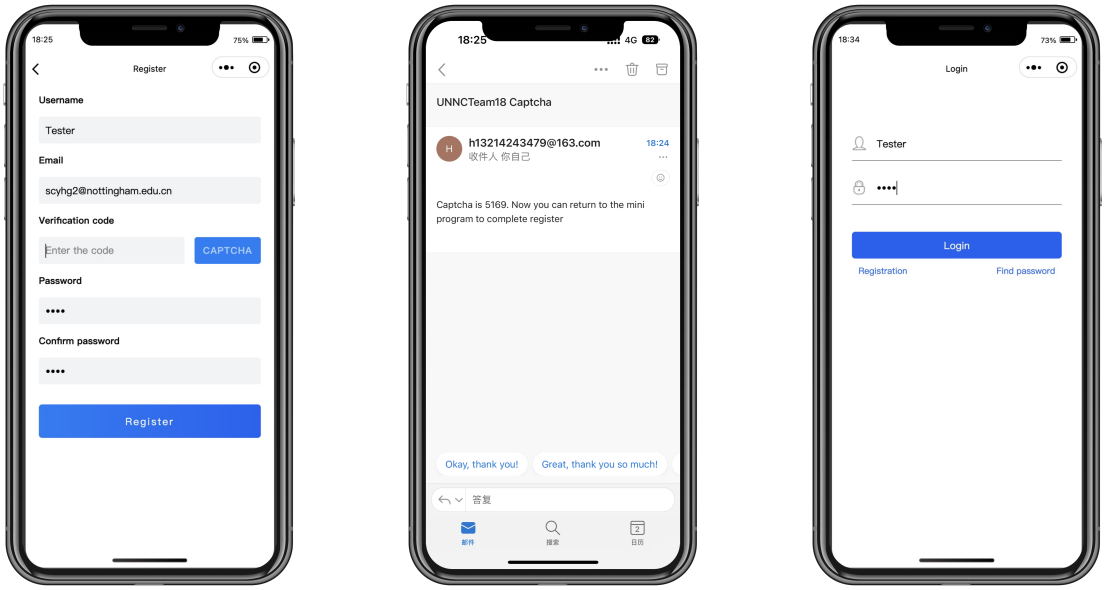


(a) Login Page (b) Register Page (c) Retrieve password Page

Figure 4.1: Login, Register and Retrieve password Page

4.1.2 Enter Verification

When the user finishes the registration, a verification email will be sent to the user's email account that he (or she) filled in. Meanwhile, the mini-program will enter the verification page to wait for the code to be entered in. If the user does not finish this process, next time when he (or she) enters the username and password, the page will be redirect to the verification page again.



(a) Request verification code (b) Verification email received (c) Switch back to Login

Figure 4.2: Email Verification

4.1.3 TabBar

The user can select the module to use through the TabBar. Academic, Bilibili and Sina modules are webpage monitoring modules, while User module is user personal information.

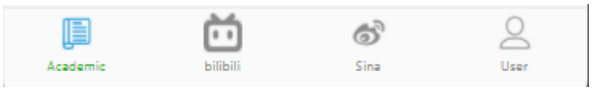
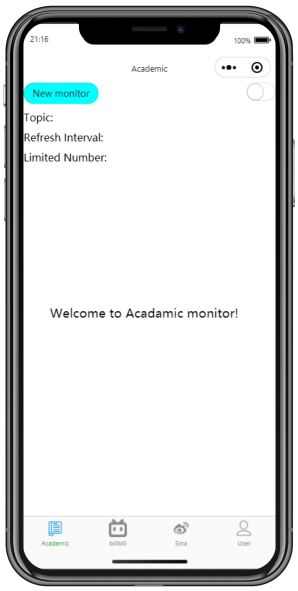


Figure 4.3: TabBar

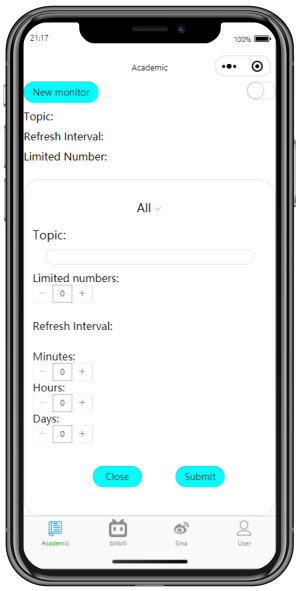
4.1.4 Monitor Page

The three monitor pages display similar results, but differ in the details. The same thing is that each page has a button to create or change the monitoring form. When the form is submitted, the information is displayed at the top left of the page. Each page also has a switch to turn monitoring on or off. There is also a button to refresh when monitoring starts. When the data is fetched from the back-end, it is rendered into a form for display.

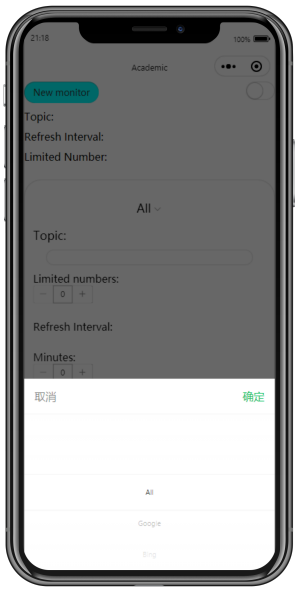
For the details of the academic page, the user has to fill in two additional blanks when creating or editing the form, one is the maximum number of papers to obtain each time, and the other is to choose which scholar to use (currently both Google scholar and Bing scholar are available, and users can also select 'All' to use these monitors simultaneously). After using different monitors to obtain data, the rendering effect will be different. The detailed academic interface is showed in Figure 4.4. Users can click on the article they are interested in, and the mini program will copy the link of the article into the clipboard. Some articles in Google Scholar have an additional green download button that copies the download link to the clipboard when clicked.



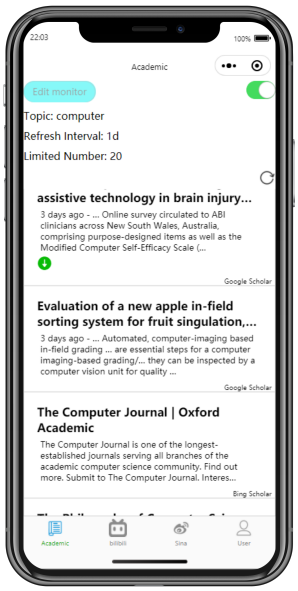
(a) First use Academic monitor



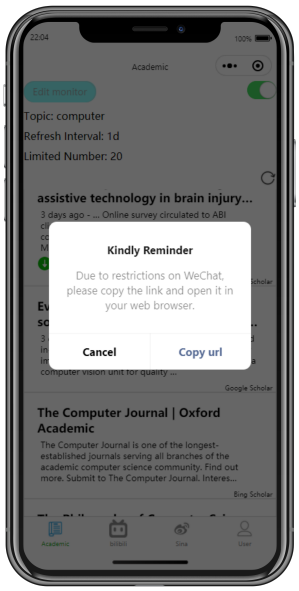
(b) Create/Edit the monitor



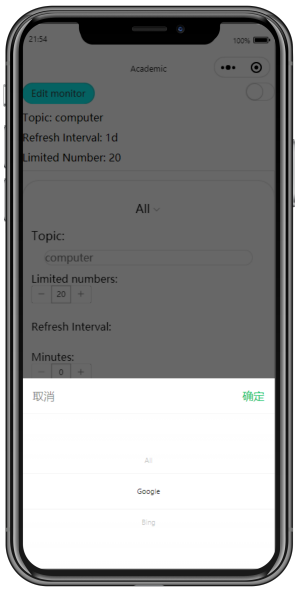
(c) Use all scholar



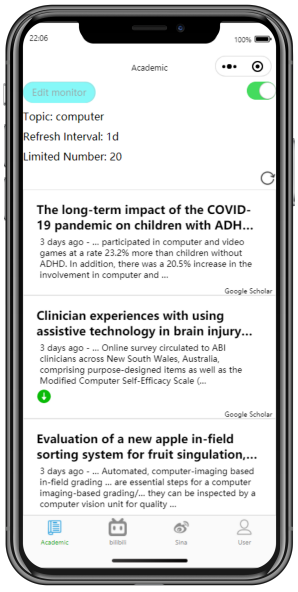
(d) Show search papers



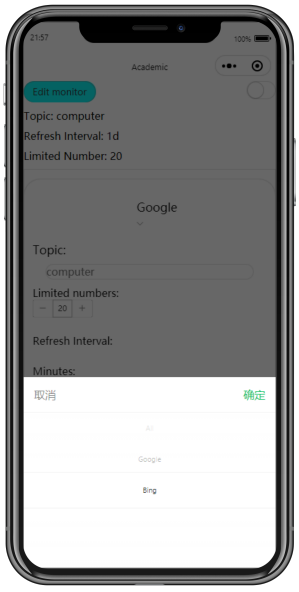
(e) Click on the paper



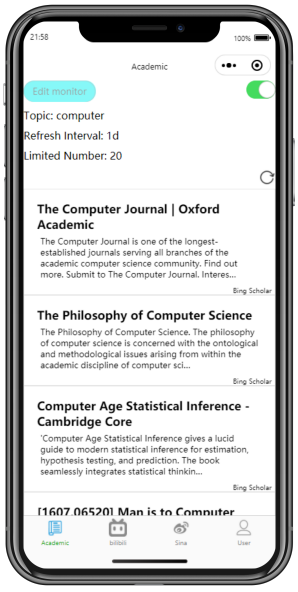
(f) Choose Google scholar



(g) Display Google results



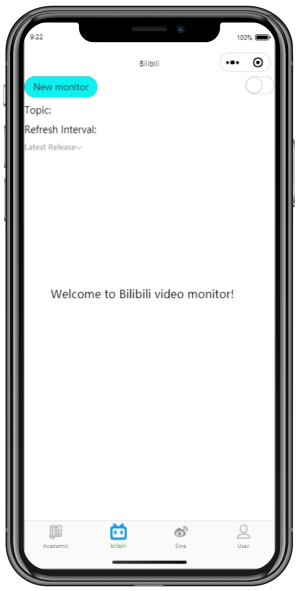
(h) Change Bing scholar



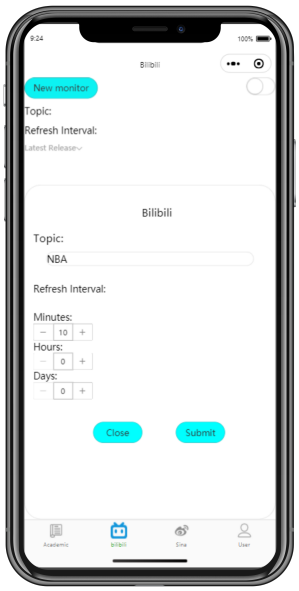
(i) Display Bing results

Figure 4.4: Academic monitor operation

On the Bilibili page, it additionally provides the ability to sort the selection. The user can choose whether the form should be sorted by the number of video hits or by the time the video was posted. The operation concerning the Bilibili module is presented in Figure 4.5.



(a) First use Bilibili monitor

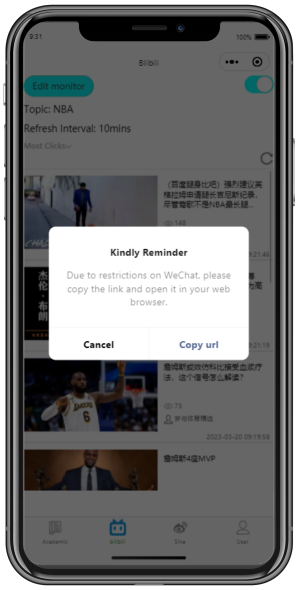


(b) Create/Edit the monitor

(c) Display queried information

(d) Change sorting

(e) Show in another sort

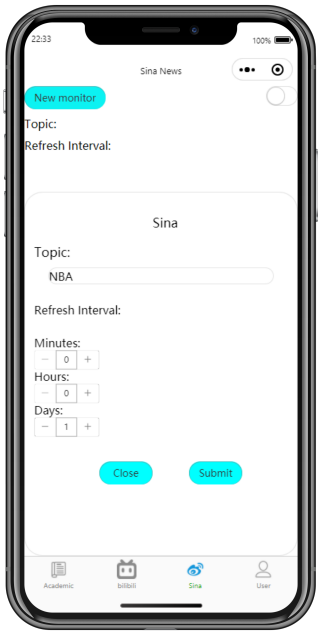


(f) Click on video item

Figure 4.5: Bilibili monitor operation

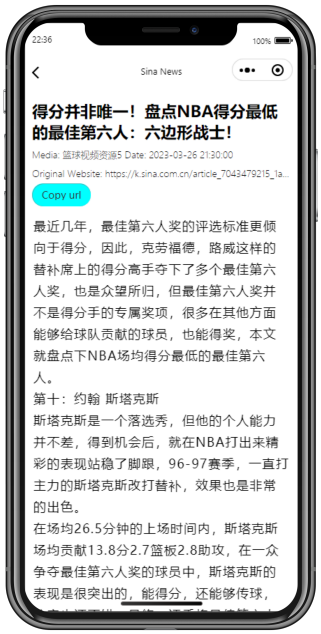
On the Sina page, users can click on the news queried by monitor to jump to the news details page. The Figure 4.6 shows the operation of Sina page.

(a) First use Sina monitor



(b) Create/Edit the monitor

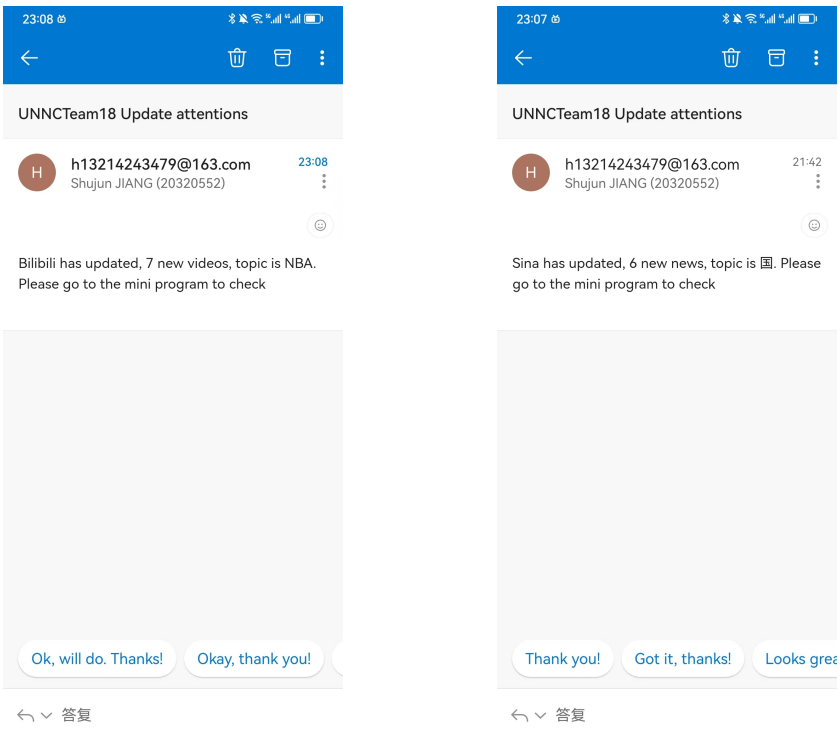
(c) Display queried information



(d) Display the selected news

Figure 4.6: Sina monitor operation

Upon the scheduled refresh time specified by the user, the mini-program will detect any newly available data on the monitored webpage, and subsequently dispatch an email notification to the user.

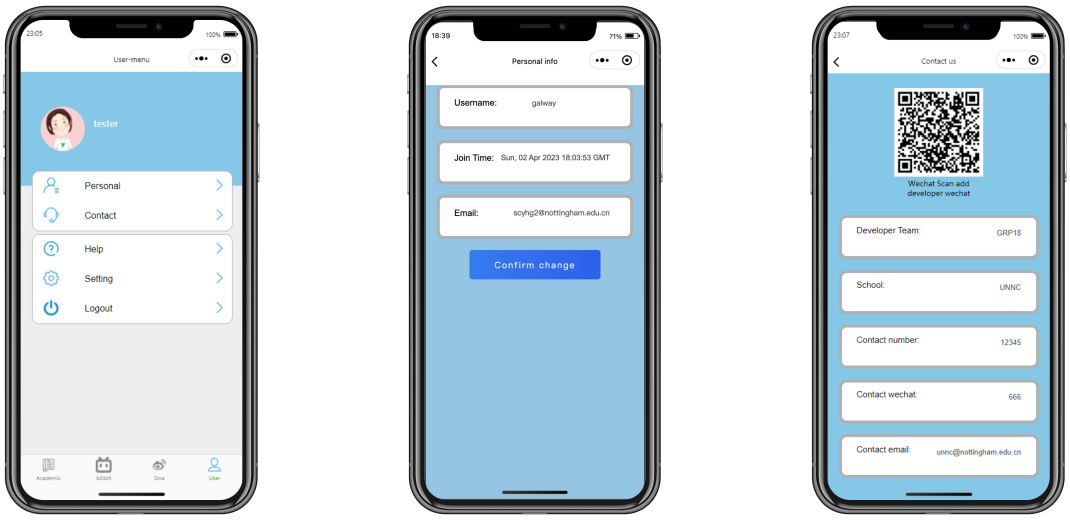


(a) Receive Bilibili email (b) Receive Sina email

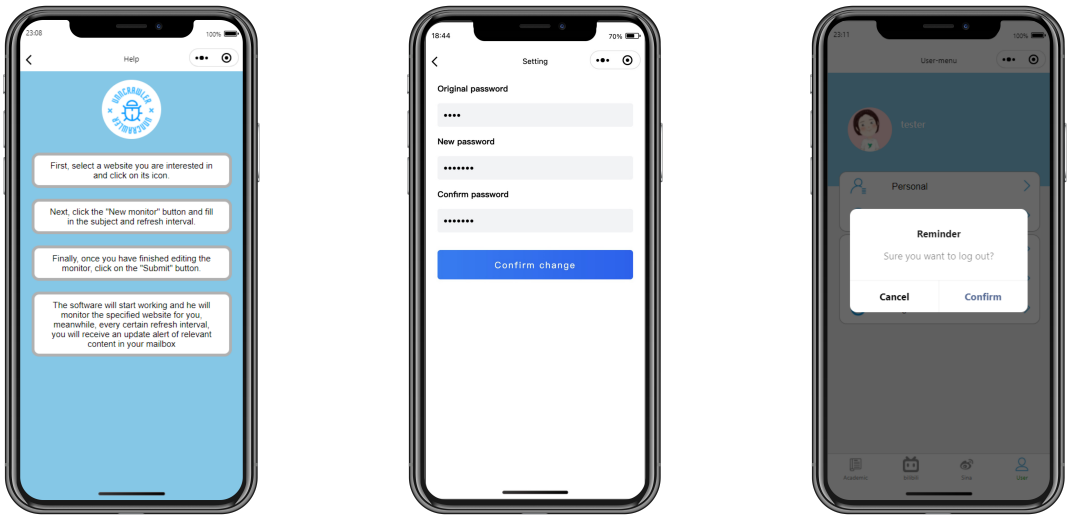
Figure 4.7: Receive email alerts

4.1.5 User Page

On the user page, users can change their avatar and other personal information, change the email and password associated with their account, and contact with our team. If the user does not understand the use of the mini program, the user can also click "Help" to view the use instructions. Figure 4.8 shows the User page in action.



(a) User page (b) Personal information (c) Team information



(d) Mini program instructions (e) Change password (f) Log out

Figure 4.8: User page operation

4.2 Database Design

There are aspects of database construction: "crawl_info" database, "extract_index" database and "user_info" database.

The "crawl_info" database is specifically designed to store the information extracted from Bilibili, Sina, Google Scholar, and Bing Scholar websites. Four separate models have been created, one for each website, to efficiently organize and manage the data retrieved from these sources. One challenge is to avoid errors caused by illegal characters that cannot be stored in the database. A function is written to detect the characters that do not belong to the utf8mb4 character repertoire and remove them from the string. Another challenge is to protect the database from storing too much data, especially Bilibilil. A script is written to supervise the "crawl_info" databases, once there are more than 1000 pieces of data stored, the script automatically runs to remove the excessive data from the database. In addition, the structures of Google Scholar and Bing Scholar are extremely similar, making them convenient for developers to add databases to other academic websites such as Baidu Scholar and Research Gate if needed.

The "extract_index" database is designed to store the ID of the last data extracted from the "crawl_info" database, in order to prevent returning the same data twice. Four models

are created for the corresponding models in the "crawl_info" database.

The "user_info" database is designed to store all data about users. One model named "user" is to store the information of users who have registered. Another model called "email_captcha" is to store the emails which will be used to register and corresponding captcha.

4.3 Crawling Algorithm Design

There are three crawling algorithms in the system as the underlying code for Bilibili, Sina and Academic respectively.

The first aspect is related to Bilibili. Initially, the program directly requested web page content through the request module and parsed out the titles, URLs[17], uploaders, upload times, pictures and views that the front end needed to display through the BeautifulSoup module. However, two months later, due to the upgrade of website anti-crawling measures, Bilibili's crawling algorithm underwent a major change. Without carrying the request header, the original request was blocked by the upgraded anti-crawling measures of the website. The web page requested with header has changed from a static web page to a dynamically loaded web page that BeautifulSoup module cannot parse out the data that the front end wants. We solved this problem by using lxml module to parse web page data and randomly changing request headers and carrying cookies when requesting to solve website anti-crawling measures. Finally, we used threading.Timer() to implement timed crawling of web page video content. In the program, each time the program crawls, it needs to check whether the video has changed. We cleverly solved this problem by comparing the release time of the video with the current time and greatly accelerated each crawl speed.

The second aspect pertains to Sina and comprises three distinct algorithmic components. Initially, news items containing the user's keywords within a specified time frame are identified using the Sina search interface. This interface manages a vast repository of data

"objects" [21] and retrieves the URLs of relevant news items in JSON format, which are temporarily stored in an array. As Sina News republishes articles from other media outlets, their URLs cannot be directly searched and are subject to a unique anti-acquisition mechanism. Unretrievable URLs are removed from the array. Subsequently, the array of URLs is iterated to extract their title, source, upload time, and content, which are compiled into a news item tuple. If the format of the news content does not conform to Sina News and its main content is inaccessible, users are notified that the news is provided by a third party and may include multimedia content accessible via the link. Ultimately, all news items are consolidated into an array and transmitted to the front-end. To expedite crawling, multi-threading technology is employed, with each thread executing the first and second operations to crawl news on a page concurrently. The upload time serves as a constraint, and only news items uploaded within the freshness interval are crawled to preclude redundancy.

Thirdly, we have the academic aspect of the project, which includes both Google Scholar and Bing Scholar. The targets are only titles and URLs of essays and links of PDF if available. The websites are both designed with ten essays located on a single page. Primly, different pages are separated due to different urls. Next in each page, the xpath of titles, links, abstracts and pdf (Google Scholar) are extracted and the rules of them are recognized. Finally, the contents of them are crawled as the rules. In order to accelerate the crawling speed, multi-thread are applied and each thread crawls essays on one page. Unlike the other two, the biggest speciality is the low update frequency, resulting in the senselessness of treating the upload time as the limiting factor in deciding whether to crawl, especially for some rare topics. For example, "openpose", "Fast RNN" and so on. Therefore, another limitation "number" is brought in to solve the complexity. In addition, the structure of their code is extremely similar, making it's convenient for developers to add new crawlers to other academic websites such as Baidu Scholar and Research Gate if needed.

4.4 Framework Design

To develop a web-based software, it is crucial to design an intact framework for the whole software. A reasonable software framework can clearly separate the responsibility and assignment in development process and raise the development efficiency. The entire software is divided into a front-end and a back-end. The responsibility of the former one is to interact with user and back-end programs. Specifically, translating user requirements to the back-end and showing data passed from the back-end to users. The latter one is responsible for processing the database. To be precise, data passed from the front-end and the underlying code is stored into the database, from which data needed to match users' requirements is extracted and passed back to the front-end.

Chapter 5

Development Strategies

5.1 Version Control - GitLab

It is vital to apply global management towards survey information and program implementation efficiently, since the project is developed as a team. Some strategies are also required to reserve versions of codes and development models for further reviewing and debugging. Thus, GitLab should be used to assist facilitation collaboration by operations such as merging versions, tracking developments, reverting changes etc as a distributed version control system.

5.2 Flask

Flask is chosen as the structure of back-end for the following reasons for the simplicity to implement in order to ensure the efficiency of development which is one of the most vital aspects of the subject. Besides, Flask is able to decrease the size of the software. The aim of the project is to complete a mini program and run it on a serve. Therefore, the size of the program has to be depressed. Moreover, the interaction with the database is not complicated. Thus, the disadvantage of flask (poor at dealing with databases) can be ignored.

5.3 React

React is a popular and widely-used JavaScript library for building user interfaces. Developed by Facebook. It allows developers to create reusable UI components and build complex UIs using a declarative syntax, making it easier to reason about the code and debug issue. One of the key reasons for choosing React as our front-end language for development is its component-based architecture. With React, we can break down complex UI into smaller, reusable components that can be easily managed and maintained. This makes it easier to scale applications and improve their performance.

5.4 Taro

Taro is a powerful open-source framework that enables developers to build cross-platform applications using a single codebase [11]. It supports the development of web, native, and mini-programs, making it an excellent choice for creating WeChat applets. WeChat applets are lightweight applications that run within the WeChat ecosystem and provide users with a seamless and convenient way to access various services and content. Taro offers a highly efficient development process, reducing development time and costs while increasing productivity. Its codebase is easy to maintain and update, making it an ideal choice for projects that require ongoing development and support. In addition, our team chose the "Ossaui" [6] library as the UI library to be used in the development and because Taro's official route jumping method is not perfect, the team chooses to use the "tarojs-router-next" [12] small program routing library to realize route jumping.

5.5 Apipost

Apipost is a comprehensive and powerful API testing and documentation platform designed for developers and testers to streamline the API development process [3]. The platform allows users to create and execute test scenarios, generate API documentation, and monitor API performance, all from a single, user-friendly interface. With Apipost,

users can quickly identify and resolve issues in their APIs, optimize performance, and improve the overall quality of their API-based applications. Whether you're working on a small project or a large enterprise-scale application, Apipost provides the tools you need to ensure your APIs are reliable, scalable, and secure.

5.6 Weixin DevTools

Weixin DevTools is a comprehensive development environment provided by Tencent for developers to create WeChat Mini Program. It is an essential tool for developers looking to create WeChat Mini Programs. Its comprehensive features and user-friendly interface make it an invaluable resource for streamlining the development process and ensuring a high-quality end product.

5.7 Agile

Agile methodology is a combination of software development methods attempting to offer an answer to the eager business community asking for lighter weight along with faster and nimbler software development processes. This is especially the case with the rapidly growing and volatile Internet software industry as well as for the emerging mobile application environment [2]. Besides being a requirement of courses, Agile provides excellent systems for us to develop the project in an effective way.In terms of development, the roles of each team member are not fixed. At the same time, the team members can flexibly adjust their work content according to the project progress. For example, team member Yuhao MA was originally responsible for developing the monitoring function for Bing Scholar according to the team plan, but because he had exams during his vacation, this task was handed over to another member of our team, Shujun Jiang. After Yuhao MA finished his exams, he took on the additional task of developing the login and registration pages.

Chapter 6

Implementation

6.1 Hierarchy Overview

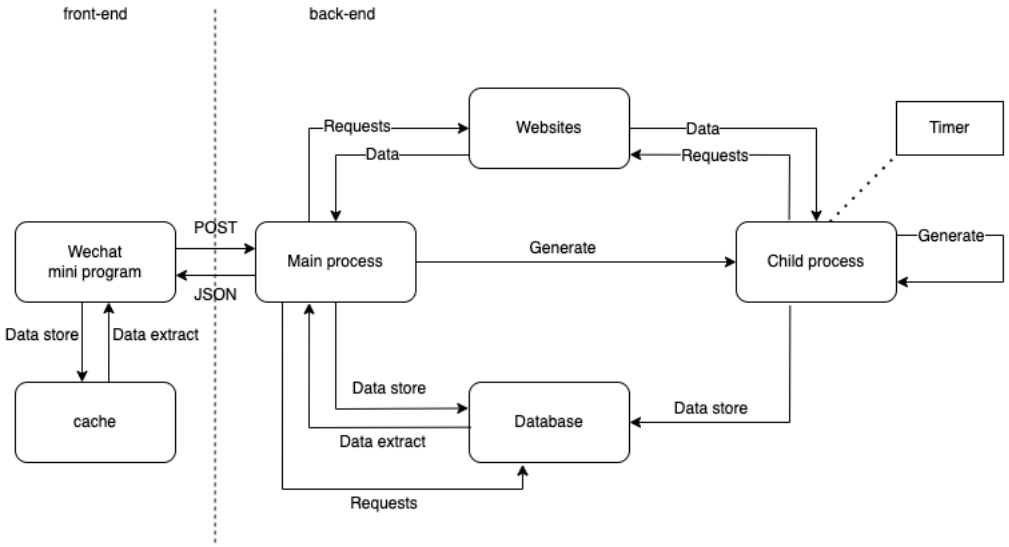


Figure 6.1: Hierarchy Overview

The entire workflow of Web Monitoring Robot is shown in Figure 6.1.

The front-end serves the purpose of rendering and interacting with data from the back-end, presenting it in a user-friendly manner for better comprehension. In addition, caching is utilized to temporarily store information from websites monitored by the user, enabling quick loading times during the next use of the application, thereby avoiding the need for repeated data requests to the back-end, thus decreasing response time.

The back-end utilizes underlying code to extract information from specific websites and store it in a database. Upon receiving a request from the front-end, data is retrieved from the database and transmitted to the front-end. Additionally, sub-threads are employed during the crawling process to enhance the speed of information retrieval.

6.2 Front End

The WeChat Mini programme is chosen as the front-end display. This section talks about the implementation of the WeChat Mini programme.

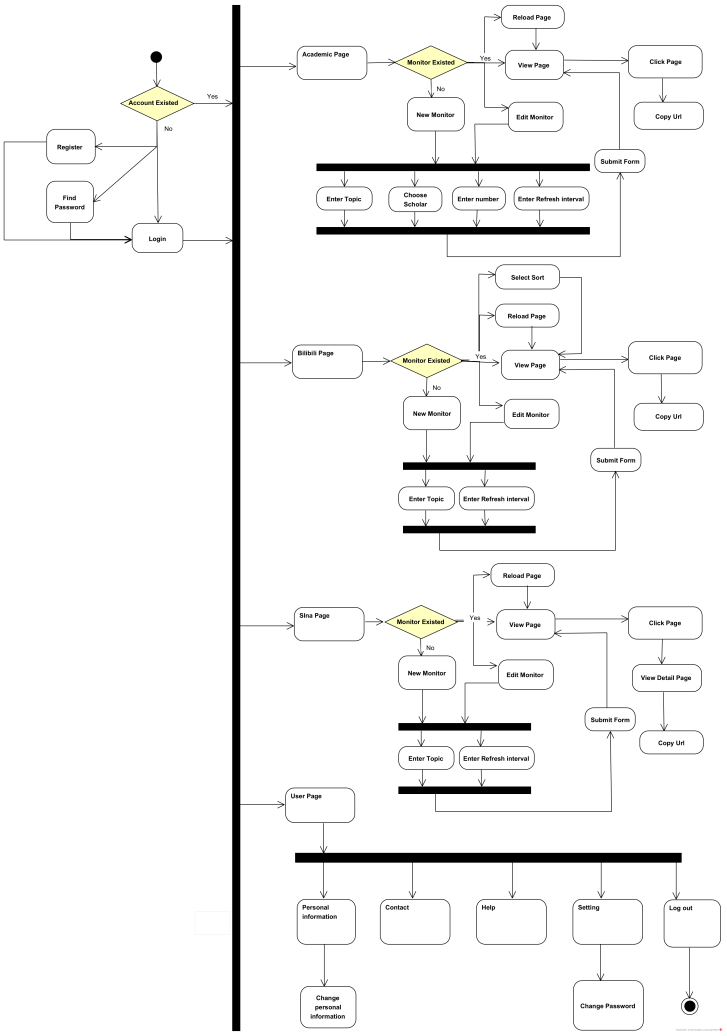


Figure 6.2: Acativity diagram of Wechat mini program

6.2.1 Register in WeChat Mini-program

Upon the user's first use of the WeChat mini-program, the initial step required is registration. The user is prompted to input their username, email, verification code, login password, and password confirmation, following which they may click the registration button to complete the process. The following content gives a detailed description of these steps with the help of Taro official documentation[11].

Step 1:After inputting their email, the user may click the "Get Verification Code" button, triggering the client to utilize Taro.request() to send a request to the backend in order to get the verification code. Taro.request() is an API utilized to initiate HTTPS network requests.

Step 2: After clicking the "Register" button, the client will send the contents of the form to the backend through Taro.request().

Step 3: After receiving the request, the backend will verify whether the data is legal. If it is legal, the user information will be stored in the database and true will be returned. Otherwise, false and the corresponding error message will be returned.

Step 4: The front-end will perform the corresponding operation according to the returned Boolean value. If it is true, it will jump to the login page through Taro.navigateTo(). Otherwise, it will prompt the corresponding error message through Taro.showModal(). Taro.navigateTo() is an API used for page navigation. It retains the current page and jumps to a page within the application, but cannot jump to a tabbar page.Taro.showModal() is an API used to display modal dialogs.

6.2.2 Login in WeChat Mini-program

After successfully registering, users can log in. users can log in with their username or email. The following content gives a detailed description of these steps with the help of Taro official documentation.

Step 1: The front-end sends the login information to the backend through Taro.request().

Step2: After receiving the request, the backend will verify whether the username and password are correct. If they are correct, true and information with a cookie will be returned. Otherwise, false and the corresponding error message will be returned.

Step3: The front-end will perform the corresponding operation according to the returned Boolean value. If it is true, it will jump to the monitoring page within the mini-program through Taro.switchTab() and calls Taro.setStorageSync() to store the cookie in the cache within the app. Otherwise, it will prompt the corresponding error message through Taro.showModal().Taro.switchTab() is an API used to navigate to a tabBar page while simultaneously closing all non-tabBar pages.Taro.setStorageSync() is an API utilized to store data in a designated key within the local cache.

Cookies serve as a unique identifier for users. Once the user has logged into the mini-program, each subsequent request sent to the backend must be accompanied by the Cookie.

6.2.3 Data Interactions and Rendering

The data interaction in front end is accomplished through the utilization of state and set State, while rendering is executed through the implementation of the render method. Whenever a change in state occurs, a re-rendering of the page is triggered. Render is responsible for describing what the UI should look like, while state is responsible for storing and updating the data that determines the behavior and rendering of the UI. Each module's state contains the corresponding front end information and data retrieved from the back end. For instance, the state of the Bilibili module stores user-generated monitoring information as well as video data returned from the back end.

6.2.4 Data catching

WeChat provides two useful interfaces, Taro.setStorageSync() and Taro.getStorageSync(), for caching data and retrieving cached data. During the initial login, user information is cached in WeChat by calling Taro.setStorageSync(). Subsequently, when the user enters the mini program again, they can skip the login process and directly access the main page. Monitored user information is also stored in the cache, so that when the user reopens the mini program, the previously monitored information can be retained.

6.3 Back End

6.3.1 Database Loading & Construction

The basic implementation of the algorithm in the back-end is to achieve the operation to the database (add, delete, update, search), in order to deal with data sent by underlying code and the front-end and sent data to the front-end to match users' requirements. The database is stored in a .db file created by the sqlite3.exe (Figure 6.3) and the front-end is required to access the basic route automatically to initialize the database the first time user opening the mini program. As shown in Figure 6.4, each of '_extract' stores the information crawled down from the corresponding website; each of '_result_count' stores the IDs of the last data extracted to avoid repeatedly extracting the same data; 'user' stores the information of registered users; 'email_captcha' stores the emails which will be registered and the corresponding captcha.

The amount of data has to be limited to match the efficiency of the software and the space saving requirements of running on the serve. In addition, SQLite3 is suitable for small batches of data. Thus, a script is added to monitor the amount of data in each '_extract' database and delete redundant data while the amount exceeds a threshold

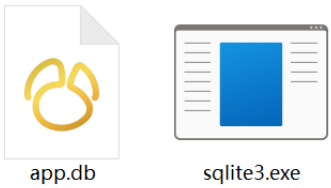


Figure 6.3: Loading Database

Name	Has Indexes	Has Triggers	Root Page
bili_extract	No	No	7
bili_result_count	No	No	9
email_captcha	No	No	4
findPwd_captcha	No	No	5
gs_extract	No	No	6
gs_result_count	No	No	11
sina_extract	No	No	8
sina_result_count	No	No	10
user	Yes	No	2

Figure 6.4: Database Construction

6.3.2 Multi-Threading

The advanced implementation is to apply multi-threading in the Flask structure. Multi-threading has to be considered to achieve the simultaneous continuous monitoring of the three websites. However, flask allocates different sessions to each thread resulting that the operation "db.session", "mail.send()" causes errors in child threads.

According to the official document of Flask [13], attributes are contained in the Flask application object, such as config, that are necessary to access within views and CLI commands. However, Circular import issues are frequently occur when importing the app instance in your project within the modules. As a result, there won't completely be instance of app to import when writing blueprints or extensions that can be reused and using the app factory pattern.

Application context is applied to solve the problem. Instead of referring to an app directly, the "current_app" proxy can be used, which points to the application handling the current activity. When handling a request, an application context is automatically pushed by Flask

6.3.3 Sending Email

Another advanced implementation is to send email to the particular user when update happen.

It is Simple Mail Transfer Protocol (SMTP) that is needed to achieve this goal with Flask_Mail. According to the official website [9], as the technology supporting email interaction, SMTP is the protocol that allows users to send and receive emails. Otherwise, email interaction would fail to exist since SMTP determines which servers will receive relay messages.

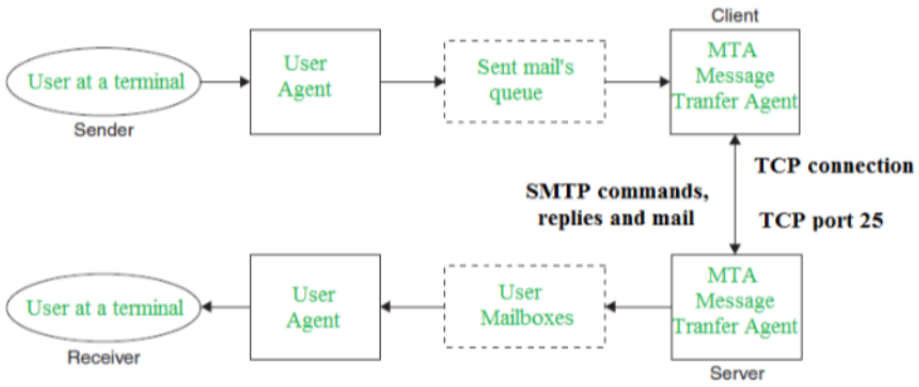


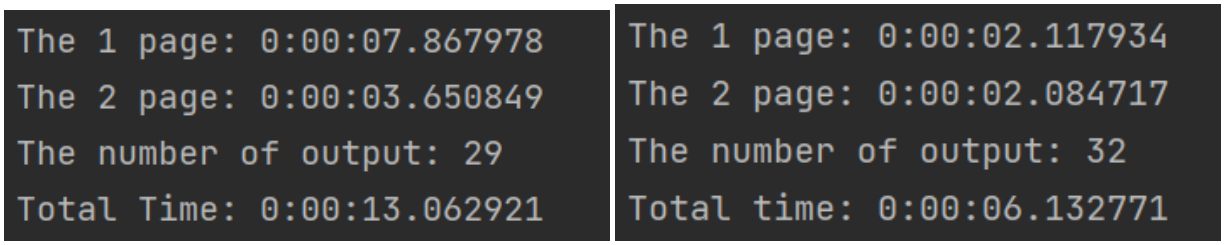
Figure 6.5: Model of STMP [10]

After the SMTP module of our host email server being opened and a series of verification, a password would be sent. Then, the password should be written in to "config" to complete the configuration of the host eamil server along with other information of it. Finally, the back-end is able to sent email to addresses according to requirements.

6.4 Underlying code

The most crucial criterion of the monitoring software is the speed of picking up messages on the websites. For example, the Google crawler runs on a distributed network of thousands of low-cost computers, so fast parallel processing is possible. This is why and how Google [23] returns results in a fraction of a second. Therefore, multi-threading is applied in the underlying code to improve the efficiency of crawling, where each item is processing

in one thread. Taking bilibili as an example, as Figure 6.6a shown, crawling each page of videos cost 7.86 seconds, and the total time of 29 items is 13.06 seconds. However, after applying multi-thread (Figure 6.6b), the time spend on each page decrease to 2.11 seconds, as more than one third as before and the total time of 32 items is 6.1 seconds, as half as before. In order to prevent our IP being blocked by the anti-crawling of these websites, a pause for random seconds between 0.5 to 1 is added after crawling five items.



(a) Single Threading (b) Multi Threading

Figure 6.6: Contrast between single and multi threading

Chapter 7

Test

7.1 Component Testing

- Account registration. (Pass)
- Account log in and out. (Pass)
- Retrieve password. (Pass)
- Change password. (Pass)
- Database storage and retrieval. (Pass)
- Upload profile pictures. (Pass)
- Mail reminder. (Pass)
- TabBar switch. (Pass)
- Taro component rendering. (Pass)
- Create new monitor. (Pass)
- Keyword matching. (Pass)
- Target answer retrieving. (Pass)
- Target answer presentation. (Pass)
- Pause and resume of search. (Pass)
- Refresh search results. (Pass)
- Change keywords. (Pass)
- Details page jump. (Pass)
- Copy target url to clipboard. (Pass)

7.2 Integration Testing

The wechat mini program has four modules, which can be selected by TabBar. The mini program could successfully detect the user's input and send it to the back-end program in the form of json via Taro.Request. Back-end program could create a thread to receive the request and handle it into functions as expected. The back-end program assembles the output of the functions into a tuple and assuredly storage it into database. After the refresh time passing or the refresh button being triggered, the front-end will issue the request smoothly. The back-end successfully retrieves data from the database and returns it to the front-end.

User account could successfully log in to the program using the correct password. The query of changing the password, retrieving the password, and logging out could be executed properly.

7.3 Compatibility Testing

- Compatibility of Operating System (Pass)

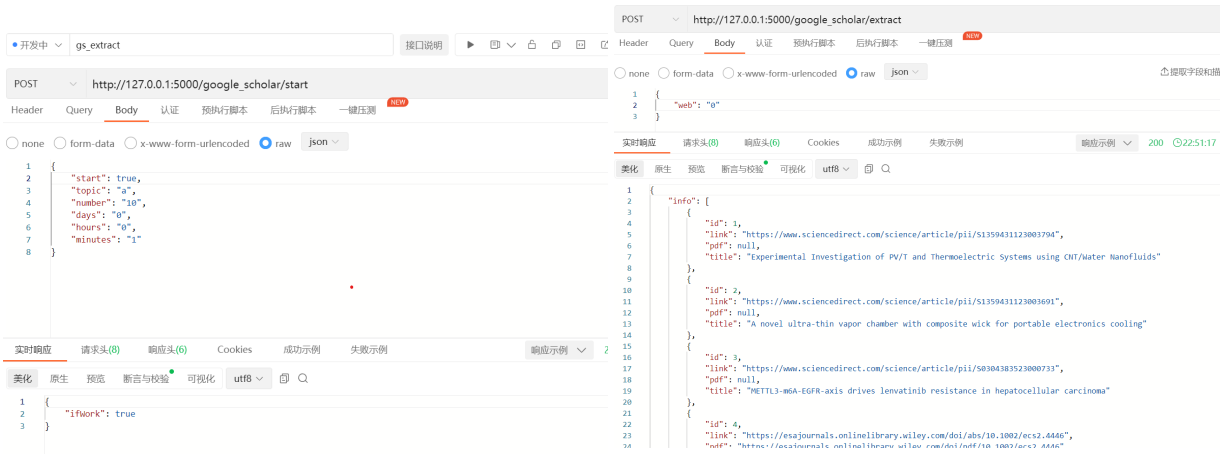
  Note: The mini program is available for any terminal that can use wechat and wechat mini program.

- Compatibility of different models (Pass)

  Note: It can be used for different machine models and different devices (such as iPhone5-14Pro Max, Nexus5-6, Windows 480*800, Mac 13, 15, 21, ipad and ipad pro). However, the rendering will be slightly inconsistent, the recommended best use model is iPhone13-14 (428*926).

7.4 API Testing

API tests detect points of interaction between the front end and the back end. The purpose of an API test is to check that data can be exchanged and transferred correctly.



(a) In start (b) In extract

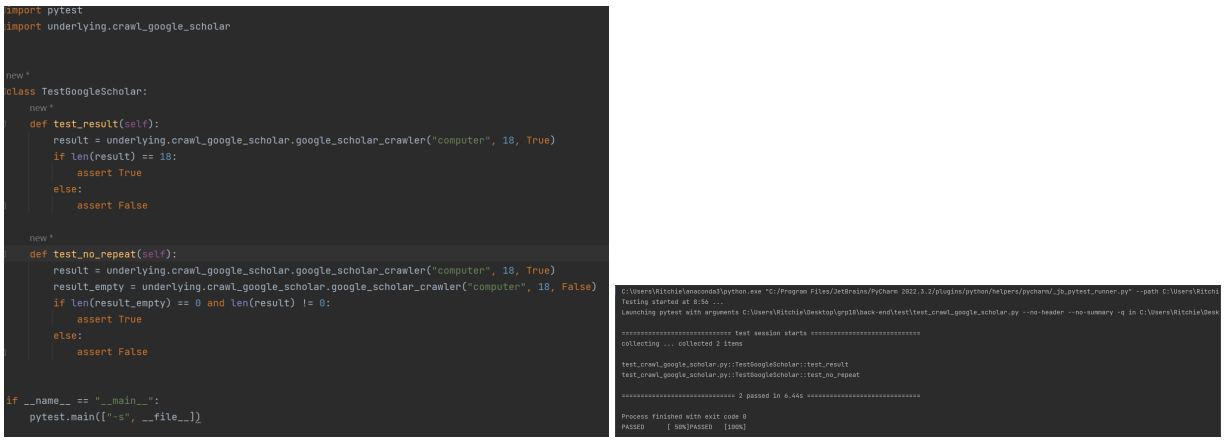
Figure 7.1: Google scholar's API test

The ApiPost tests results show that all apis in the small program can effectively exchange data successfully.

7.5 Performance Testing

7.5.1 Underlying code

The requirement of the underlying codes are to return information according to conditions and duplicated information cannot appear. The performance tests are applied to test the results. Taking Google Sholar as an example, the first test (Figure 7.2a) is to prove the number of the result match the condition (the former one), the second (Figure 7.2a) is to prove no repeated essays are crawled (the latter one). Both of them are passed as shown in Figure 7.2b, indicating that the underlying code is available.



(a) Test cases (b) Test results

Figure 7.2: Underlying code performance testing

7.5.2 Back-end

The requirement of the back-end is to receive requests sent from the front-end and return results in this session without repeatedly showing. Taking Google Scholar as an example, as Figure 7.1a and Figure 7.1b indicated, the back-end is able to receive requests and return results normally. What is still needed to be test is the back-end should reject to return duplicated essays. Figure 7.3 shows that, if the same extract request sent again, back-end will return an blank array to show no more new essays.

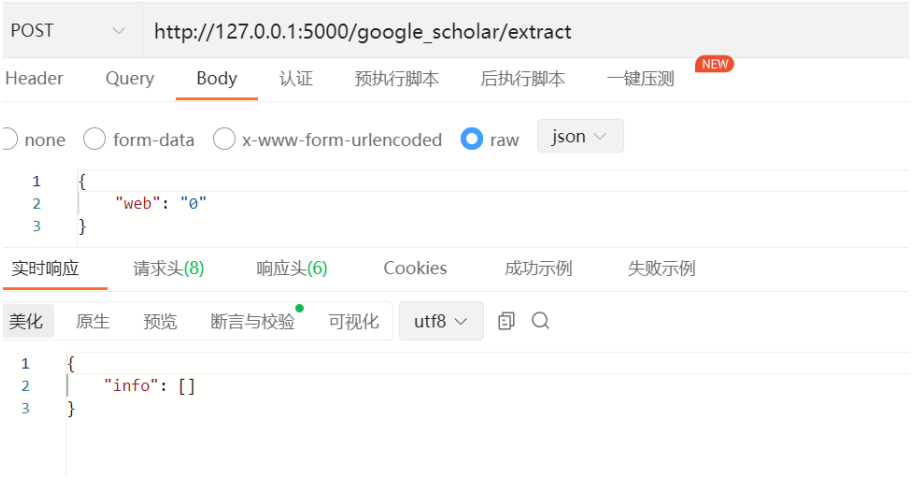


Figure 7.3: Back-end performance test

7.5.3 Front-end

The front-end work is tested on Wechat DevTools. The tool provides performance monitoring testing and auditing modules. They graphically show the response time of the application and score their performances and best practices to help us effectively evaluate the front-end effort.

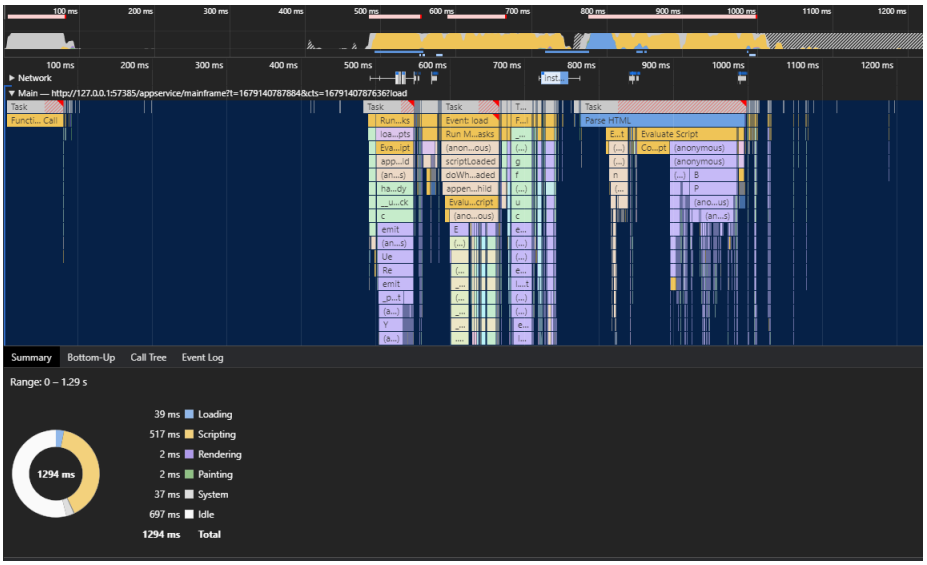


Figure 7.4: Performance test in Wechat DevTools's performance model

As shown in Figure 7.4, the small program has completed loading, rendering, painting and other operations in a very short time, proving its good performance.

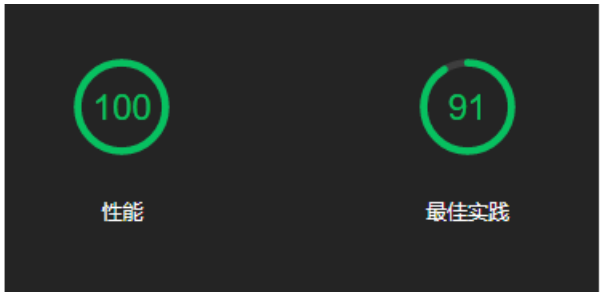


Figure 7.5: Performance score in Wechat DevTools's audis model

Figure 7.5 shows how Wechat DevTools rated the programs performance and best practices. This score is enough to prove the excellent performances of the software. Wechat DevTools believes that best practices are lacking because some redundant components are declared in the front-end wxml file. It has been checked that these components, which are considered redundant, were created (inevitably) when Wechat DevTools compiled the Taro framework source code we used, and have no impact on the normal operation of the small program.

7.6 Concurrency Testing

Concurrency is important for this applet, which relies on concurrency to allow the three monitoring modules to work simultaneously. As shown in the figure 7.6, the applet is able to perform three monitoring functions simultaneously and get the data correctly and render the results within the specified time. This test proves that the concurrency of the small program is guaranteed.

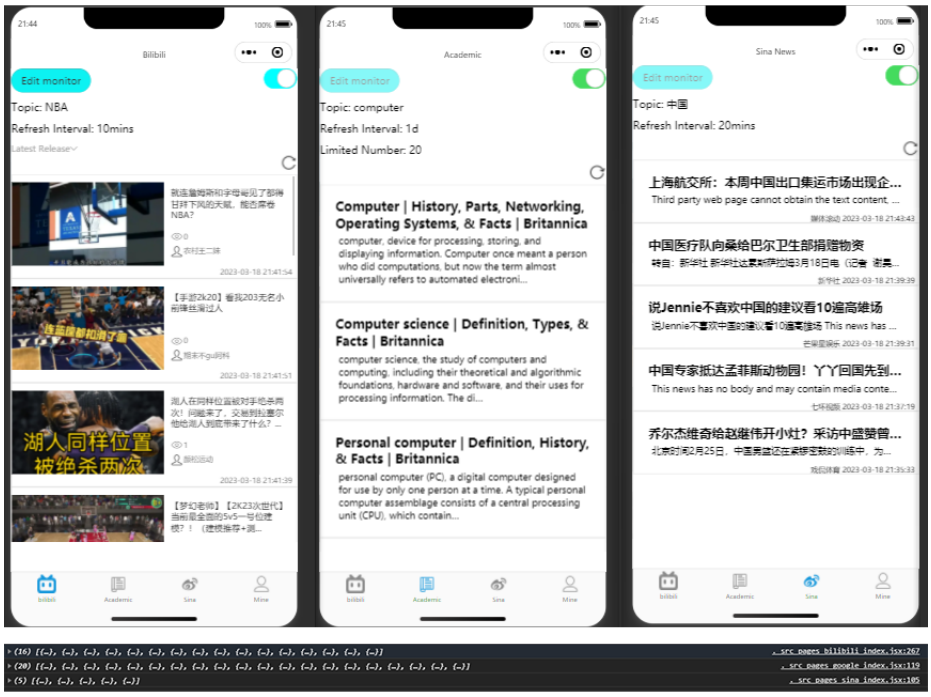


Figure 7.6: Three modules in the program run simultaneously

7.7 Pressure Testing

7.7.1 Back-end

To avoid being blocked by the website, pressure tests have to be applied to test the acceptance of each website. The methods to conduct the pressure testing is to continue to crawl each website under a particular topic and refresh interval for a period of time. Take Bilibili (updating speed is the highest) as an example, the Bilibili module in back-end runs with "music" as topic and "0 days, 0 hours, 5 minutes" as refresh interval and persists about 12 hours when each thread can return around 100 videos, proving that the frequency of crawling is acceptable. In addition, running the module with a particular topic and setting the interval to "1 day" (Figure 7.7). The code is still available, proving that the underlying code is creditable and stable.

Figure 7.7: Back-end pressing test

7.7.2 Front-end

The pressure test of the front end focuses on the correct rendering display and routing given a large amount of data. Among them, the modules that are most likely to fail due to the large amount of data are the module that monitor Bilibili videos and the module that monitor Sina news.

Because users being highly active, the Bilbili module may query a large amount of data in a short time by inputting popular keywords, which leads to the front-end need to render

a large number of pictures. As shown in figure 7.8, the test proves that the program can render a large number of images correctly. The Bilibili module passes the stress test.

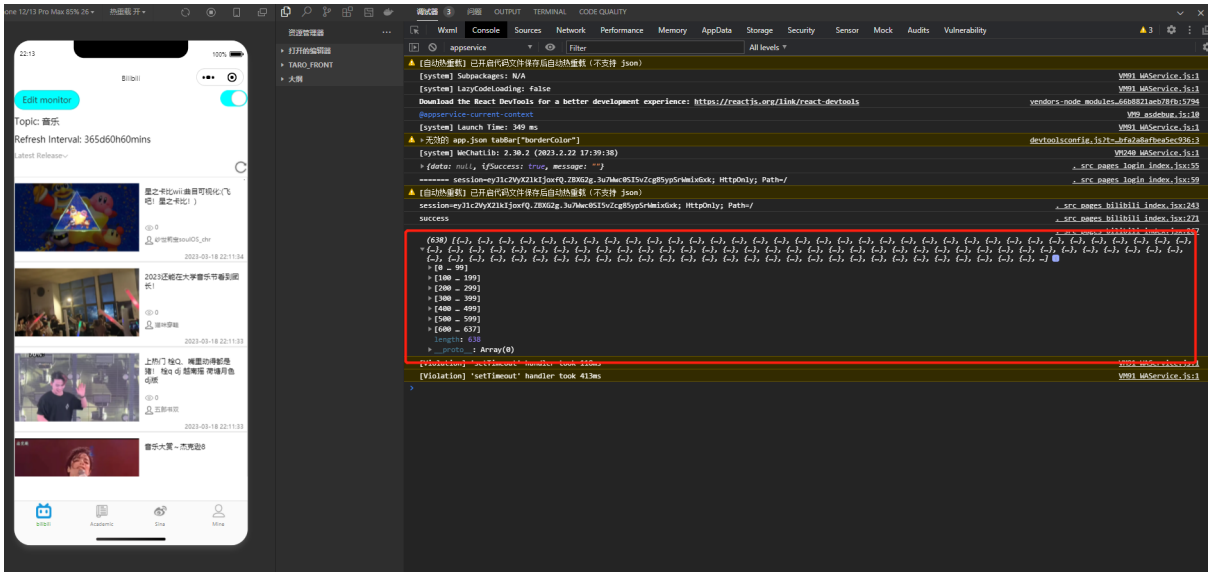


Figure 7.8: Pressing test for Bilibili module

When the Sina module getting a large amount of data, the Sina front end needs to automatically create a large number of jump routes. The stress test requires that the module render the list of stories correctly, and ensure that each story in the list jumps and shows the content. The figure 7.9 proves that the Sina module passes this stress test and the program has the ability to handle large amounts of data.

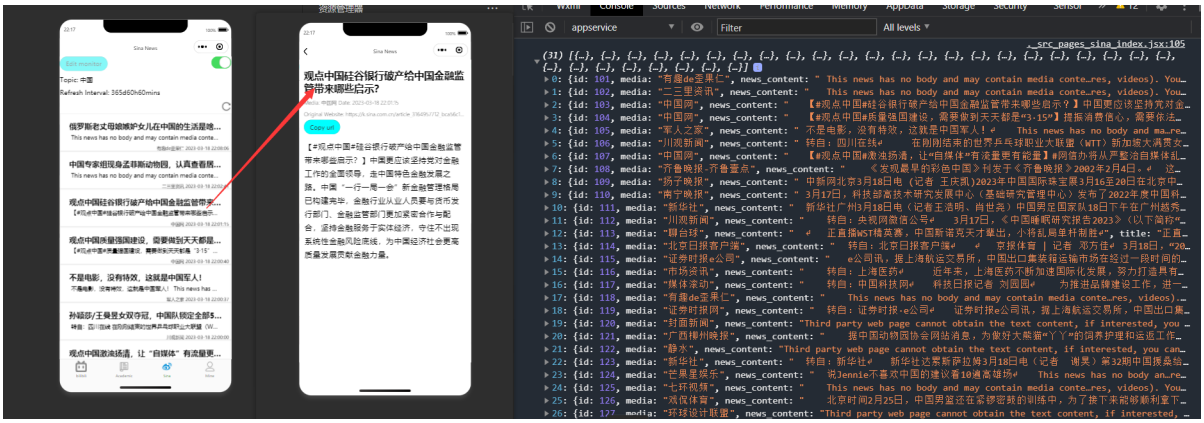


Figure 7.9: Pressing test for Sina module

7.8 Summary

The test results meet expectations. The functions of each component is fully verified, providing a prerequisite for integration. The reliability, stability, accuracy and concurrency of the small program are verified. These tests show that the small program can correctly complete all the expected functions at the present stage, and it is a product that can go online and run under the condition of server support. It also makes a good foundation and reserve interface for potential requirements that may be added in the future.

Chapter 8

Reflection

8.1 Experience

This section provides an overview of the overall progress, describes the software development experience, the problems we encounter and how we overcome them.

8.1.1 Requirements Engineering

After discussing with stakeholders, we learned that the previous GRP team had done a similar project where they monitored Moodle. The stakeholders suggested that we monitor more academic websites this year, such as Google Scholar. After hearing their suggestion, we had a team discussion and decided to crawl Google Scholar and added three more websites: Bilibili video website, Sina News and Bing Academic to increase the diversity of our software's functionality. However, the requirements given by the stakeholders were not clear. We had to decide on the software's features and the front-end and back-end frameworks, programming languages, and databases used throughout the software development process. In terms of back-end programming language selection, there was a dispute within the team over whether to choose Java or Python. Both Java and Python had a student with relevant crawler experience. After team discussion, we finally chose Python because it has multi-threading and stronger data processing capabilities.

8.1.2 Webpages selection

There is some controversy over which academic website should be monitored between Google Scholar and Baidu Scholar. The former one has the advantage of more frequent updates and a much larger number of articles. The latter one has the advantage of lower monitoring difficulty and lower network requirements. However, unlike the second one, the structure of the first one is extremely complex, making it difficult to capture detailed data and having certain anti-crawling measures. With internal discussions and suggestions from stakeholders, although the difficulty is higher, we ultimately choose Google Scholar. The webpages monitoring robot is oriented towards students and staff of UNNC, meaning a higher demand for English papers, and frequent updating means that it is easier to test later, which is beneficial for our subsequent development. Later, in order to meet the increased demand of stakeholders, we also add Bing Scholar as a monitoring object. In the selection of video websites, we initially choose Youtube, but because its anti-crawling technology is difficult to overcome, after trying for a week, we decided to give up and choose Bilibili video website instead. Its main advantages are fast updating and more popular among UNNC students. The choice of news websites has also been controversial. Our team discussed BBC, CNN, Global Times and Sina News and finally chose Sina News. Compared with the other few, Sina News has more stable network requests and is easier to develop and test. After the final discussion, our team finally chose Sina News as the monitoring object for the news section.

8.1.3 System Development

The development stage can be divided into three stages: the autumn semester, the winter vacation, and the spring semester. During the autumn semester, the team efficiently completed the design of the back-end crawler algorithm and planned to integrate the underlying code of the crawler into the Flask framework and simultaneously develop the front-end during the winter vacation. However, due to the outbreak of COVID-19, the development work during the winter vacation did not go smoothly at first. All members of the team were infected with COVID-19 at different times, and development work had

to be postponed. Since only one person had relevant experience with the front-end framework we chose, the team responsible for developing the front-end needed to spend some time learning front-end related technologies. It was not until two weeks before school started that we officially began developing the front-end. Fortunately, the team's efficiency and enthusiasm were high. Whenever technical problems were encountered, the team would discuss them in a group chat and synchronize progress. Everyone completed their respective development work at the first meeting after school started. In the spring semester, our main tasks were to integrate front-end and back-end code, expand functionality and conduct testing. With the joint efforts of the team, these tasks were successfully completed.

8.2 Assessment of Teamwork

8.2.1 Achievement

1. Clear Division Of Responsibility

The implementation tasks basically consist of front-end development, back-end development, database creation, and testing. Each team member was responsible for a specific task. All decisions related to this task were made by the responsible person. The main benefits of a clear division of tasks are that it reduces the managers workload and allows equal attention to be paid to each part of the tasks. It also helps to speed up the process of the whole project

2. Frequent Review

During the meeting, each team member will present the results of the task and report on overall progress. Everyone is encouraged to comment on each other's results. In addition, one member summarizes all valuable suggestions and takes minutes of the meeting. Managers adjust task requirements and deadlines based on meeting feedback to improve the project completion rate.

2. Harmonious Team Environment

All team members are treated with dignity and respect, and no member is harassed. When there is a problem, the team tends to indicate it and solve it together and there is few of complaints. This helps create a work environment full of energy, positivity and growth.

8.2.2 Issues & Remedial Action

1. Lack of relevant knowledge and experience

Our project involves the integration of front-end and back-end development of a WeChat mini program, utilizing newer frameworks such as Taro, React, and Flask. Most members are not proficient in these technologies, so learning and research was necessary during the development process, causing a slow start to the project. However, the team is passionate about learning these technologies, and actively discuss issues in the group chat. With the collective effort of the team, we are able to overcome these difficulties.

8.3 Assessment of Project

8.3.1 Achievement

The software we have developed effectively meets all the listed requirements. Specifically, it successfully implements monitoring of specific websites based on user preferences, sends email notifications to users about updates related to keywords they are monitoring on the website, and provides a user-friendly interface for displaying the new content.

8.3.2 Issues & Remedial Action

Database timing delete

Captchas should be deleted after a particular time in order to ensure the safety of accounts. To achieve this, data in "find_Pwd captcha" and "email captcha" need to be delete automatically after being kept a particular of time.

One possible solution is to write a script for SQLite. The difficulty is that it's difficult to implement and errors easily occur while call the script with flask. Another solution is to write a function to access the database and delete data created out of the limitation. The weakness of this method is that all the data need to be checked every time a new data is restored, which will decrease the efficiency of the software.

8.4 Maintenance Software

Since websites update their anti-crawling strategies, the team must frequently test the effectiveness of the software and modify the monitoring algorithm in a timely manner when the website updates its anti-crawling strategy. For example, two weeks before submitting the code, the Bilibili video website updated its anti-crawling strategy, causing the software to be unable to crawl video content. The team promptly detected this issue and modified the algorithm, successfully resolving it.

8.5 Team Construction

Heterogeneous groups can be beneficial to more talented students because they must take a leadership role in the group. Many more able students have an innate understanding of their own abilities compared to other students, and when solving problems as a team, they naturally tend to lead the way. In our group, two more talented team members set the direction of the entire project and took on the more complex parts, while the rest followed that direction and were allowed to choose relatively simple tasks most of the time.

8.6 Collaborative Project Development

In the collaborative project development, our group adopts the model of separate front and back-end development, with the back-end providing the API and the front-end completing the rendering to improve the efficiency, maintainability and expandability of the product.

During the development process, some issues arose due to inconsistencies in the component library and frontend integration versions. However, through active communication within the team, these issues were efficiently resolved.

8.7 Further Development

1.Baidu Scholar and ResearchGate will be added to the academic module to enrich it's function.

2.Adding a new way to notify users through WeChat about updates.

3.Purchase a server and deploy the backend of the applet to the server.

Chapter 9

Summary

In conclusion, our team has successfully achieved all the expected functions of the web monitoring robot software, which generally meets most of the requirements. Additionally, the quality and performance of the software have been ensured through rigorous testing. By completing this project, we were able to apply what we learned in our classes to real-world practice, and we gained proficiency in developing a simple WeChat mini program using technologies such as Flask, React, and Taro. Furthermore, our team members were able to enhance their communication and cooperation skills through successfully resolving conflicts and achieving satisfactory results.

Bibliography

[1] 8 best free tools to monitor website changes. https://www.hongkiat.com/blog/detect-website-change-notification/. [Online; accessed 15-12-2022].

[2] Agile. https://arxiv.org/abs/1709.08439. [Online; accessed 12-03-2023].

[3] Apipost document. https://v7-wiki.apipost.cn/docs/1. [Online; accessed 16-12-2022].

[4] Companies use montastic to monitor services and keep people informed during incidents. https://montastic.io/. [Online; accessed 25-10-2022].

[5] Distill web monitor official website. https://chrome.google.com/webstore/detail/distill-web-monitor/inlikjemeeknofckkjolnjbpehgadgge. [Online; accessed 13-12-2022].

[6] Ossaui. https://docs.taro.zone/docs/ossa/. [Online; accessed 25-01-2023].

[7] Profiling bilibili user profiles. https://www.bilibili.com/read/cv18496411/. [Online; accessed 19-12-2022].

[8] Sina news brand upgrade launches new slogan "know the unknown, see the unknown". https://finance.sina.com.cn/tech/2021-09-23/doc-iktzscyx5808785.shtml?_zbs_baidu_bk. [Online; accessed 25-11-2022].

[9] SMTP. https://www.smtp.com/solutions/smtp-for-developers/. [Online; accessed 12-03-2023].

[10] SMTP Diagram. https://www.geeksforgeeks.org/simple-mail-transfer-protocol-smtp/. [Online; accessed 12-03-2023].

[11] Taro introduction document. https://docs.taro.zone/docs/. [Online; accessed 21-01-2023].

[12] tarojs-router-next. http://lblblib.gitee.io/tarojs-router-next/. [Online; accessed 28-02-2023].

[13] The Application Context. https://flask.palletsprojects.com/en/2.2.x/appcontext/. [Online; accessed 12-03-2023].

[14] Visulaping official website. https://visualping.io/. [Online; accessed 25-12-2022].

[15] What is a wechat mini program? https://www.vivacityapp.com/wechat-news/what-is-a-wechat-mini-program/. [Online; accessed 22-3-2023].

[16] Working hard behind the scenes, so that you don't have to. Workinghardbehindthescenes,sothatyoudonthaveto. [Online; accessed 28-10-2022].

[17] Cho, J., and Garcia-Molina, H. The evolution of the web and implications for an incremental crawler. Tech. rep., Stanford, 1999.

[18] Dhenakaran, S., and Sambanthan, K. T. Web crawler-an overview. *International Journal of Computer Science and Communication 2*, 1 (2011), 265–267.

[19] Diallo, M., Sall, O., N'diaye, M., and Djomo, B. A search engine dedicated to the environment around the congo basin area: Model and architecture. *EAI Endorsed Transactions on Ambient Systems 6*, 18 (2019).

[20] Gusenbauer, M. Google scholar to overshadow them all? comparing the sizes of 12 academic search engines and bibliographic databases. *Scientometrics 118*, 1 (2019), 177–214.

[21] Hirai, J., Raghavan, S., Garcia-Molina, H., and Paepcke, A. Webbase: A repository of web pages. *Computer Networks 33*, 1-6 (2000), 277–293.

[22] Kausar, M. A., Dhaka, V., and Singh, S. K. Web crawler: a review. *International Journal of Computer Applications 63*, 2 (2013), 31–36.

[23] Mali, S., and Meshram, B. Implementation of multiuser personal web crawler. In *2012 CSI Sixth International Conference on Software Engineering (CONSEG)* (2012), IEEE, pp. 1–12.

[24] Mallawaarachchi, V., Meegahapola, L., Madhushanka, R., Heshan, E., Meedeniya, D., and Jayarathna, S. Change detection and notification of web pages: A survey. *ACM Comput. Surv. 53*, 1 (feb 2020).

[25] Meegahapola, L., Alwis, R., Nimalarathna, E., Mallawaarachchi, V., Meedeniya, D., and Jayarathna, S. Detection of change frequency in web pages to optimize server-based scheduling. In *2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer)* (2017), pp. 1–7.

[26] Peshave, M., and Dezhgosha, K. *How search engines work: And a web crawler application.* PhD thesis, University of Illinois Springfield, 2005.

[27] Poonam P. Doshi, Emmanuel M., P. Web pattern mining using eclat. *International Journal of Computer Applications 179*, 8 (December 2017), 0975 – 8887.

[28] Shekhar, S., Agrawal, R., and Arya, K. V. An architectural framework of a crawler for retrieving highly relevant web documents by filtering replicated web collections. In *2010 International Conference on Advances in Computer Engineering* (2010), IEEE, pp. 29–33.

[29] Smith., B. E. *Creating Web Pages For Dummies (9th ed.).* John Wiley&Sons Inc, New Jersey, USA., 2008.

[30] Udapure, T. V., Kale, R. D., and Dharmik, R. C. Study of web crawler and

its different types. *IOSR Journal of Computer Engineering 16*, 1 (2014), 01–05.

[31] Zeng, K., Zhang, Y., Liu, Y., and Wen, F. Design and development of mobile learning platform based on wechat mini programs. In *Advances in Artificial Systems for Logistics Engineering* (2021), Springer, pp. 96–106.

Appendix A

Timetable

Date	Task	DeadLine
10/21/2022	Decide language, website	10/24/2022
10/24/2022	Complete all underlying codes	11/04/2022
11/04/2022	Interim report	12/08/2022
11/04/2022	Draw user case diagram	11/08/2022
11/08/2022	Draw activity diagrams	11/15/2022
11/15/2022	Complete prototype	11/25/2022
11/29/2022	First edition of interim report	
12/09/2022	Allocate task(front end and bach end)	
12/09/2022	First edition of individual module	02/19/2023
02/19/2023	Test individual module	03/08/2023
03/08/2023	Final report	04/06/2023
03/08/2023	Combine four modules	03/19/2023
03/19/2023	Test and maintain the software	04/06/2023
03/19/2023	Deal with new requirement of our supervisor	03/22/2023
03/26/2023	First edition of final report	
03/29/2023	complete videos(pre, ad, demo)	04/11/2023
04/19/2023	Open Day	

Appendix B

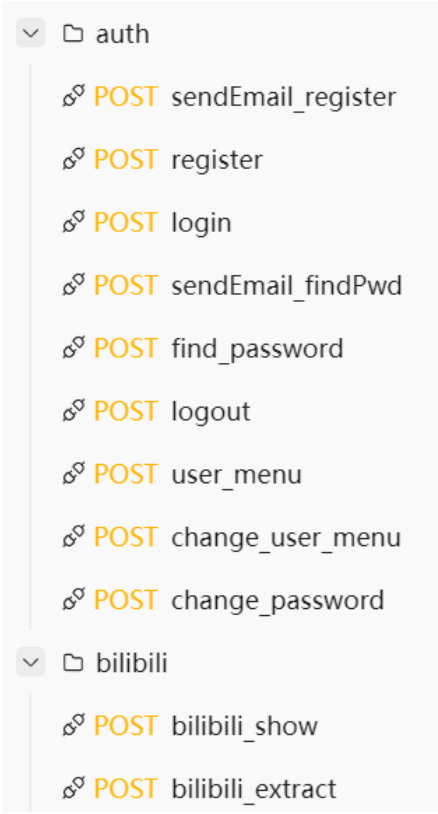
Test cases

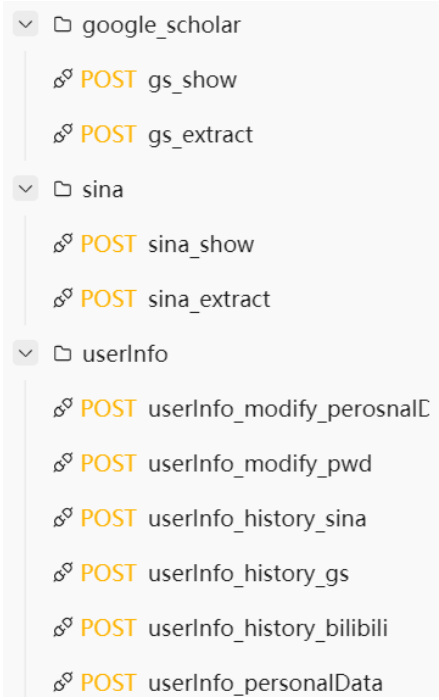
B.1 Front end

Test Case ID	Test Scenerio	Test Steps	Test Data	Expected Results	Actual Results	Pass/Fail
1	Screen compatibility test	Open Wechat Mini-program	iPhone13	run normal	as expected	Pass
2	Screen compatibility test	Open Wechat Mini-program	Nexus6	run normal	as expected	Pass
3	Screen compatibility test	Open Wechat Mini-program	ipad	run normal	as expected	Pass
4	Screen compatibility test	Open Wechat Mini-program	Apad	run normal	as expected	Pass
5	Screen compatibility test	Open Wechat Mini-program	Mac	run normal	as expected	Pass
6	Screen compatibility test	Open Wechat Mini-program	Windows	run normal	as expected	Pass
7	Compatibility test	Open Wechat Mini-program	Andorid	run normal	as expected	Pass
8	Compatibility test	Open Wechat Mini-program	iOS	run normal	as expected	Pass
9	Login test	Input wrong data then click login	Wrong username/password	Error	as expected	Pass
10	Login test	Input correct data then click login	Correct username and password	Login successfully	as expected	Pass
11	Register test	set new name, email and password	An correct email, name and password	successfully registered	as expected	Pass
12	Register test	use wrong email or registered email	A registered email or wrong email	Error	as expected	Pass
13	Register test	Click on verification code button	User action	starting counting down	as expected	Pass
14	Find password	Get vertification code from email to get back password	User action	return password	as expected	Pass
15	View tabbar	Get into module and change to other modules	User action	successfully turned to other modules	as expected	Pass
16	View module data	Check that the component was successfully rendered	User action	successfully rendered	as expected	Pass
17	Create monitor	Create Sina monitor	Topic, time interval	successfully created	as expected	Pass
18	Create monitor	Create Sina monitor	miss necessary data	Error	as expected	Pass
19	Create monitor	Create Bilibili monitor	Topic, time interval	successfully created	as expected	Pass
20	Create monitor	Create Bilibili monitor	miss necessary data	Error	as expected	Pass
21	Create monitor	Create Academic All monitor	Topic ,limited number and time interval	successfully created	as expected	Pass
22	Create monitor	Create Academic All monitor	miss necessary data	Error	as expected	Pass
23	Create monitor	Create Academic Google monitor	Topic ,limited number and time interval	successfully created	as expected	Pass
24	Create monitor	Create Academic Google monitor	miss necessary data	Error	as expected	Pass
25	Create monitor	Create Academic Bing monitor	Topic ,limited number and time interval	successfully created	as expected	Pass
26	Create monitor	Create Academic Bing monitor	miss necessary data	Error	as expected	Pass
27	Switch test	Turn on Sina monitor	User action	start/continue monitor	as expected	Pass
28	Switch test	Turn off Sina monitor	User action	stop monitor	as expected	Pass
29	Switch test	Turn on Bilibili monito	User action	start/continue monitor	as expected	Pass
30	Switch test	Trun off Bilibili monitor	User action	stop monitor	as expected	Pass

Test Case ID	Test Scenerio	Test Steps	Test Data	Expected Results	Actual Results	Pass/Fail
31	Switch test	Turn on Academic monitor	User action	start/continue monitor	as expected	Pass
32	Switch test	Turn off Academic monitor	User action	stop monitor	as expected	Pass
33	Click Sina item	Click Sina searched item	User action	turn to news page	as expected	Pass
34	Copy Sina item url	Click news url button	User action	clipboard has the web address	as expected	Pass
35	Click Bilibili item	Click Bilibili searched item	User action	clipboard has the web address	as expected	Pass
36	Click Google item	Click Google searched item	User action	clipboard has the web address	as expected	Pass
37	Click Google download button	Click Google button	User action	clipboard has the pdf address	as expected	Pass
38	Click Bing item	Click Bing searched item	User action	clipboard has the web address	as expected	Pass
39	Change profile	Click profile	Photo	change profile successfully	as expected	Pass
40	Change username	Click 'Personal' and enter new name	new name string	change new name successfully	as expected	Pass
41	Change email	Click 'Personal' and enter new email	new email address	change new email successfully	as expected	Pass
42	Change password	Click 'Personal' and enter new password	new password	change new password successfully	as expected	Pass
43	Log out check	Click log out	User action	Log out successfully	as expected	Pass
44	Cache test	Exit program without log out, then restart program	User action	All components retain the last exit information	as expected	Pass

B.2 Back end





B.3 Underlying code

Unit Test						
Number	Underlying Code	Test Case		Excepted Output	Test Output	Result
		Topic	Number			
1	crawl_google_scholar	computer	18	18, 0	18, 0	PASS
2	crawl_google_scholar	a	39	39, 0	39, 0	PASS
3	crawl_google_scholar	science	9	9, 0	9, 0	PASS
4	crawl_google_scholar	FAST-RNN	26	26, 0	26, 0	PASS
5	crawl_google_scholar	Network	44	44, 0	44, 0	PASS
6	crawl_bing	computer	18	18, 0	18, 0	PASS
7	crawl_bing	a	39	39, 0	39, 0	PASS
8	crawl_bing	science	9	9, 0	9, 0	PASS
9	crawl_bing	FAST-RNN	26	26, 0	26, 0	PASS
10	crawl_bing	Network	44	44, 0	44, 0	PASS
		Topic	Fresh_interval			PASS
11	crawl_bilibili	音乐	2, 23, 19	TRUE	TRUE	PASS
12	crawl_bilibili	music	2, 12, 33	TRUE	TRUE	PASS
13	crawl_bilibili	nba	0, 59, 59	TRUE	TRUE	PASS
14	crawl_bilibili	LU3	1, 44, 25	TRUE	TRUE	PASS
15	crawl_bilibili	Chat*GPT	0, 0, 20	TRUE	TRUE	PASS
16	crawl_sina	中国	2, 23, 19	TRUE	TRUE	PASS
17	crawl_sina	China	2, 12, 33	TRUE	TRUE	PASS
18	crawl_sina	Rus#sia#	0, 59, 59	TRUE	TRUE	PASS
19	crawl_sina	nba	1, 44, 25	TRUE	TRUE	PASS
20	crawl_sina	Chat*GPT	0, 0, 20	TRUE	TRUE	PASS

Pressure Testing						
Number	Underlying Code	Test Case		Excepted Output	Test Output	Result
		Topic	Number			
1	crawl_google_scholar	computer	230	230	230	PASS
2	crawl_google_scholar	a	315	315	315	PASS
3	crawl_google_scholar	science	160	160	160	PASS
4	crawl_google_scholar	FAST-RNN	253	253	253	PASS
5	crawl_google_scholar	Network	288	288	288	PASS
6	crawl_bing	computer	230	230	230	PASS
7	crawl_bing	a	315	315	315	PASS
8	crawl_bing	science	160	160	160	PASS
9	crawl_bing	FAST-RNN	253	253	253	PASS
10	crawl_bing	Network	288	288	288	PASS
		Topic	Fresh_interval			PASS
11	crawl_bilibili	音乐	15, 21, 16	TRUE	TRUE	PASS
12	crawl_bilibili	music	20, 19, 25	TRUE	TRUE	PASS
13	crawl_bilibili	nba	21, 52, 34	TRUE	TRUE	PASS
14	crawl_bilibili	LU	25, 43, 09	TRUE	TRUE	PASS
15	crawl_bilibili	ChatGPT	29, 13, 53	TRUE	TRUE	PASS
16	crawl_sina	中国	15, 21, 16	TRUE	TRUE	PASS
17	crawl_sina	China	20, 19, 25	TRUE	TRUE	PASS
18	crawl_sina	Russia	21, 52, 34	TRUE	TRUE	PASS
19	crawl_sina	nba	25, 43, 09	TRUE	TRUE	PASS
20	crawl_sina	ChatGPT	29, 13, 53	TRUE	TRUE	PASS

Appendix C

Test Report

C.1 Introduction

C.1.1 Revision History

This is the first beta testing of the mini-program.

C.1.2 Testing Personnel

Our development team invited students majoring chemical engineering from UNNC to join the beta test of the mini-program with us.

C.1.3 Purpose and Scope

This document is used for beta testing of the mini-program. The users would be team invited students and GRP development team.

C.1.4 List of Reference Documents

- Chapter 4: Updated Design and Iterfaces
- Chapter 7: Test
- Appendix B: Test cases

C.2 Beta Test Scope, Strategy, and Timeline

Define the functional scope of the test for this test, the usage scenario, and the test timeline.

C.2.1 Beta Test Customers, Environment, and Scope

Testers can use any device that can carry a front and back end, test all the features out of order, and give feedback after one week.

C.2.2 Testing Criteria

Beta Ship Criteria

1. Users correctly sets up the front and back ends according to the "User manual".
2. Users can open the application successfully
3. The user can sign up and login in the application.
4. Users can create monitor and receive email alerts if monitor find new information.
5. Users can complete the page jump and copy the link of the searched content allowed by the applet.
6. Users can modify their personal information.
7. Users can close the mini-program (without logging out) and open it again without logging in and restore the working status of each monitor and refresh the search results.

Beta Install and Criteria to Begin Testing

1. The test machine is able to run python, npm and yarn instructions.
2. The test machine is properly equipped with the front and rear ends.
3. The test machine was able to run and render the front-end.

C.2.3 Testing Scope

Features to be Tested

Roles	Unit	Activities
GRP tester	Academic Monitor	Create\edit monitor
		Refresh search results
		Monitor switch test
		Click paper link
		Click download link
	Bilibili Monitor	Create\edit monitor
		Refresh search results
		Monitor switch test
		Click video link
	Sina Monitor	Create\edit monitor
		Refresh search results
		Monitor switch test
		View news detail page
		Click news link
	User page	View personal information
		Change personal information
		Change password
		View other program details
	Document from Wechat	Integration of development logs
		Fix any errors found
Normal User	Academic Monitor	Create\edit monitor
		Refresh search results
		Monitor switch test
		Click paper link
		Click download link
	Bilibili Monitor	Create\edit monitor
		Refresh search results
		Monitor switch test
		Click video link
	Sina Monitor	Create\edit monitor
		Refresh search results
		Monitor switch test
		View news detail page
		Click news link
	User page	View personal information
		Change personal information
		Change password
		View other program details
	Document	Submit problem feedback

Figure C.1: Test functions and personnel responsibilities

Other items to be evaluated

1. Interface design (good or bad?)
2. Interaction logic (simple or complex?)
3. Learning complexity (Is it hard to learn to use?)

Overall Test Strategy and Timeline

Time	Activities
Day 1	Software installation
Day 2-6	User registration
	Functionalities testing
Day 7	Feedback to team

Figure C.2: Timeline

C.3 Test Descriptions

Describe the testing performance of the functionality being tested across various aspects, including functionality implementation, performance, and stress, among others.

C.3.1 Configuration Tests

Configuration testing is usually performed prior to the deployment of a software system to ensure that the system can operate correctly in the target hardware and software environment.

C.3.2 Functional Tests

Ensure that the software can properly operate all designed functions as described in the requirements document and User manual.

C.3.3 Load and Performance Tests

Evaluate the performance of the software under different load conditions.

C.3.4 Stress Tests

Evaluate the stability and reliability of the system under high load conditions.

C.3.5 Recovery and Error Handling Tests

Evaluate the performance and ability of the system to handle exceptional situations. Recovery testing refers to testing the responsiveness of a system to evaluate its performance and reaction time when it encounters errors, abnormal situations, or improper operations. Error handling testing refers to the testing of the error handling function of a system to evaluate its handling capability and correctness when it encounters errors, exceptional situations, or improper operations.

C.3.6 Tools and Test Equipment Required

Any device that can operate according to the user manual and build the front and back-end could be used. It is recommended to use a virtual machine to remove programs and dependencies after testing.

C.4 Problem Recording, Issues Management and Escalation, Rework ,and Resolution

C.4.1 How issues will be recorded and who records them

- By screenshots.
- Send to wechat test group.
- Better offer detailed steps so that we can recur the problem.

C.4.2 Who will deal with the detected problems

- The team responsible for the development and testing of that part of the code will fix the issues.

C.4.3 What issues are serious enough that they should be raised to management immediately, and what is the escalation procedure for them

- Database problem.
- Software running problem.

C.4.4 How enhancement requests will be logged and handled

- The enhancement requirements will be gathered and documented so that the team can continue developing in the summer vacation and later.

C.4.5 Record development logs.

The team records any errors that occur during the test development phase and how they are handled.

C.5 Exit Criteria

The software has completed all the above tests, confirmed that it has met all the current requirements, ensured its usability, and demonstrated excellent product logic.

Appendix D

User Manual

D.1 Environment Requirements

- Python 3.9
- Flask
- Taro
- React

D.2 Installation instructions

D.2.1 Back-end

Running "pip install -r back-end/requirements.txt" in terminal of folder "grp18".

D.2.2 Front-end

1.Download latest node.js version 18. You can go to download node.js in "`https://nodejs.org/en/download`".

2.Open a terminal and type "npm install -g yarn" to install the yarn package manager.

3.Enter "yarn global add @tarojs/cli" in the terminal to install the CLI.

4.From the terminal, go to the grp18/front_end/mini_program folder.

5.Running "npm i" to download node modules.

6.Download Weixin dev tool in "`https://developers.weixin.qq.com/miniprogram/en/dev/devtools/download.html`".

D.3 Running Wechat Mini-program

D.3.1 Back-end

From the terminal, go to the grp/back-end folder and run "flask run".

D.3.2 Front-end

1.From the terminal, go to the grp18/front_end/mini_program folder and run "npm run build:weapp – –watch".

2.Import the project into WeChat Developer Tools to use it.

D.4 Wechat Mini-program User Manual

This software is run in the Wechat Mini-Program. The following is the step-by-step operation manual after entering the UNNCrawler mini program.

D.4.1 Assumptions and dependencies

Users have Wechat and an email address that can view the verification code.

D.4.2 Getting Started

This section includes the details of registration, logging on process and how to change personal information and password.

Login Page

This page is the first page the user enters when opening the Mini-program. It can jump to the main page, the registration page and the password retrieval page.

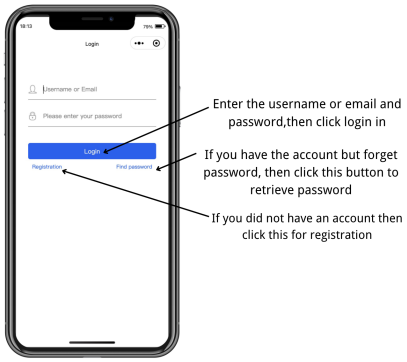


Figure D.1: Login page

Register Page

The following is an example of a Register Page. If you would like to create a new account, you can do so by clicking Register on the login page and the program will then jump to the registration page.

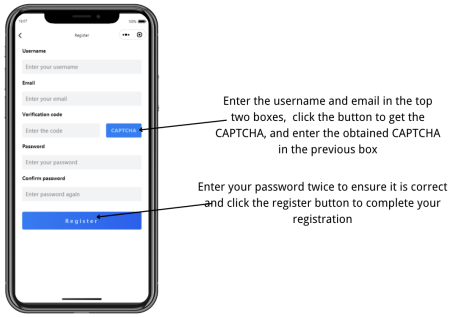


Figure D.2: Register page

Retrieve Page

If you click ”forget the word” in the Login Page then the program will jump to retrieve page.

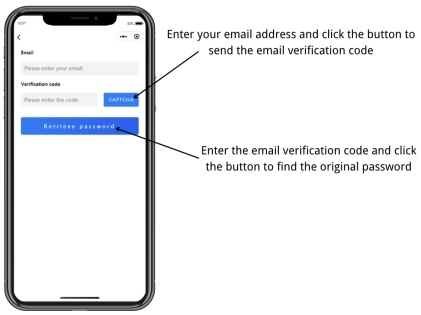


Figure D.3: Retrieve page

D.4.3 Home Page

After the user has successfully logged into their account, the user is taken to the main part of the Mini-program. There are four sections in the main part: Academic Page, Bilibili Page, Sina News page and User Page.

Academic Page

Firstly, there is AcademicPage, where users can monitor thesis search keywords, either on Google Scholar, Bing Scholar or both sites.

In figure D.4 you can add a new monitor by clicking on the "New monitor" button in the top left corner, the button in the top right corner indicates whether the monitor is currently active or not, and you can switch between sections in the tab bar in the following column.

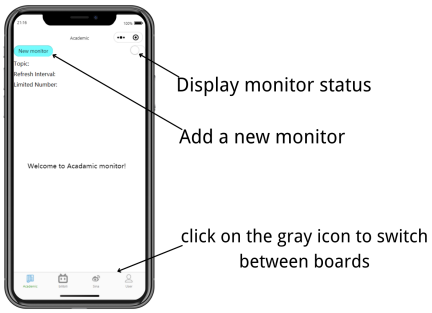


Figure D.4: Academic page

When the user clicks on the New monitor button, the monitoring information will appear in the interface, as in figure D.5. The user can select the sites to be monitored, select the topic to be monitored, limited numbers, and also can specifically adjust the refresh

intervals, click the submit button to start monitoring, click close to cancel the creation of the latest monitor.

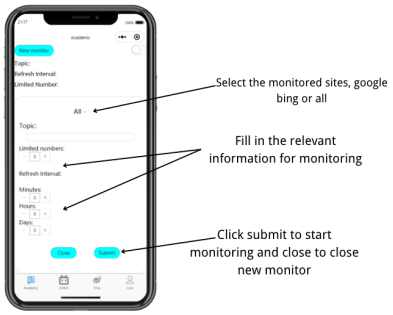


Figure D.5: Overview of Academic monitor setting

When the user clicks submit, monitoring begins and the user will see the same page as in Figure D.6, the status bar in the upper right corner becomes open, the new Monitor button in the upper left corner becomes edit monitor, the user can click on this button to modify the monitor related information, the main part of the page is the search results,the green arrow means that the user can click to download PDF. Users can also directly click on the paper they are interested in. As shown in Figure D.7, users can click the Copyurl button to copy the link and then open it in the browser.

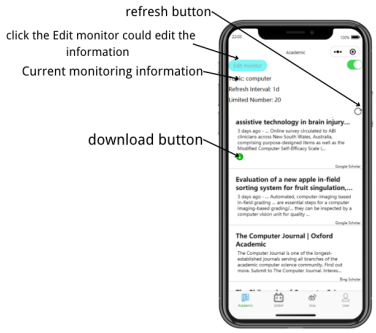


Figure D.6: New monitor

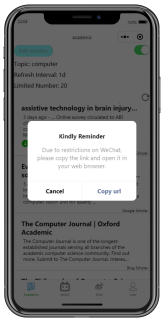


Figure D.7: Copy url

Bilibili Page

The Bilibili section is used in much the same way as Academic, except for the sorting of the search results. The button below the new monitor in figure D.8 toggles the sorting of the search results and allows the user to choose between most clicked or latest posted. Figure D.9 shows the display of the monitoring results. User can also click on the section of interest like figure D.10 , to get the URL and copy the link to open it in the browser.

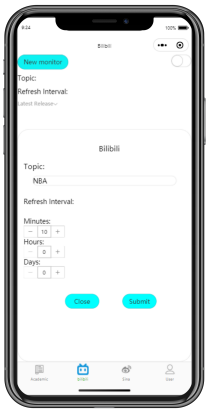


Figure D.8: New monitor

Figure D.9: Results

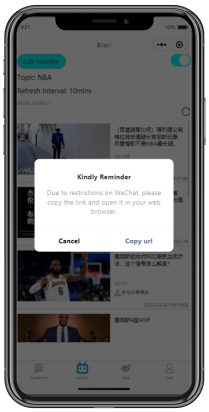


Figure D.10: Copyurl

Sina News Page

The Sina section is mostly used in the same way as the academic section, except that the search results are sorted by time only, with the time of publication indicated in the bottom right hand corner of the search results as Figure D.12, and the user can click directly on the text section to browse, and the details can be copied to the browser by clicking the copy url button like Figure D.13.

Email Notification Page

Upon the scheduled refresh time specified by the user, the mini-program will detect any newly available data on the monitored webpage, and subsequently dispatch an email notification to the user like Figure D.14.

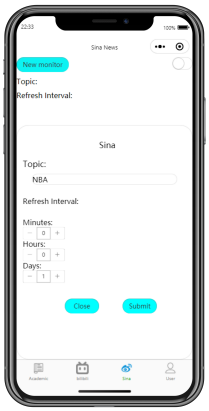


Figure D.11: News monitor

Figure D.12: Results display

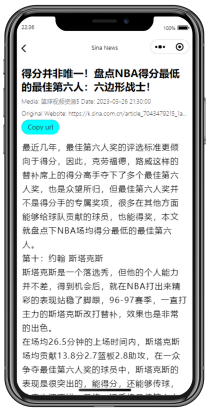


Figure D.13: Results display

Figure D.14: Email Notification Page

User Page

In the user page, the user can perform a variety of actions, in order to view personal information, contact the developer, manual, settings and logout, as shown in Figure D.15. And when the user clicks on the avatar, it can be switched to an avatar of the user's choice.

This is shown in Figure D.16 when the user clicks on the personal column, they can view their personal information and directly change the information they want to change, and then click the confirm change button to save the changes.

This is shown in Figure D.17 when users click on the Contact column, they can view information about the developer and contact the developer directly if they have questions

about the project.

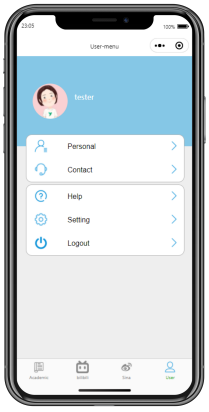


Figure D.15: User Menu

Figure D.16: PersonalPage

Figure D.17: Contact Page

This is shown in Figure D.18 when the user clicks on the Help column, the user can view a short guide on the use of the program.

This is shown in Figure D.19 when the user clicks the Settings column, they can change the account password and then click on the Confirm Change button to save the changes.

This is shown in Figure D.20 when the user clicks the Logout column, a reminder will pop up, and the user can click confirm to log out of the account, or cancel to cancel the logout.

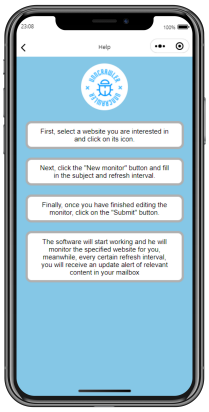


Figure D.18: Help Page

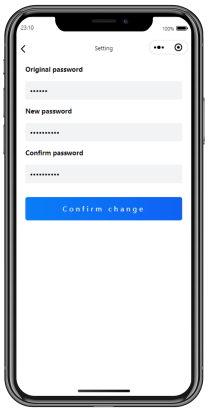


Figure D.19: Change Page

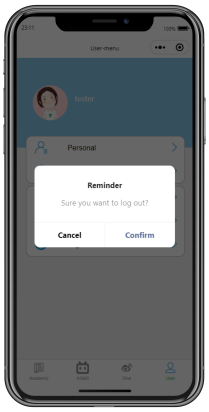


Figure D.20: Logout Page

Appendix E

Minutes

E.1 Summary of Team18 First Formal Meeting

Date (MM/DD/YY):	10/21/22
Present member:	All
Absent Member and Reason:	
Content:	language： python Web(initial)：Douban AMS(https://journals.ametsoc.org/view/journals/aies/aies-overview.xml), google scholar design: divide into several modules
Task and deadline:	Decide which website to choose (10/24)
If supervisor attend:	Y

Question	Answer
Ethic form	No ethic problem
how to deploy our software	any platform is OK
which language should we choose	any language
which website can we choose	free to choose and supervisor will discuss with us if it's suitable

E.2 Summary of Team18 Second Formal Meeting

Date (MM/DD/YY):	10/28/22
Present member:	Yuhao MA, Chi MA, Shujun JIANG, Haowei GAO
Absent Member and Reason:	Ruizhe Huang was absent from the meeting because he went home.
Content:	Web(v1): news: http://english.sina.com/ essay: https://scholar.google.com video: https://www.bilibili.com economy: http://www.stcn.com/article/list/djjd.html (alternative offer) Role: Technical lead: Yuhao MA Implementer: Chi MA Testing: Shujun JIANG Documentation: Haowei GAO Project manager: Ruizhe HUANG Every minute should have a moderator to gather the topics to be discussed before the meeting and controls the meeting pace. The moderator also needs to report on the progress of the previous week's work. Task1: Collect relevant papers and literatures about network monitoring robots and crawls. Task2: Look for projects or successful commercial products in development related to network monitoring robots and get background information about the technologies they use. Collect information about the market situation, prospects, etc of this product. Task3: About the crawler function development experiment on Google Scholar. Task4: About the crawler function development experiment on Sina English news Web crawler.
Task and deadline:	1: Task1 primarily responsible —— Chi MA 2: Task2 primarily responsible —— Haowei GAO 3: Task3 primarily responsible —— Yuhao MA 3: Task4 primarily responsible —— Shujun JIANG
If supervisor attend:	Y

E.3 Summary of Team18 Informal Meeting

Date (MM/DD/YY):	11/1/2022
Present member:	All
Absent Member and Reason:	
Content:	build project (gitlab, private) prepare a new website (https://www.shixiseng.com/)
Task and deadline:	obtain all the required data, combine the code with flask (Y.MA JIANG HUANG) (ddl: 11/8) read DMS ppt and discuss UML diagrams (HUANG C.MA GAO)(11/4) look up for comparing algorithm (to deal with new web)(Y.MA JIANG)
If supervisor attend:	N

E.4 Summary of Team18 Third Formal Meeting

Date (MM/DD/YY):	11/4/2022
Present member:	ALL
Absent Member and Reason:	
Content:	build project (gitlab, private) prepare a new website (https://www.shixiseng.com/) Y.MA: Bilibili. Obtain name, link. Sort as title, amount of play, time, number JIANG: Xinlang，Obtain title, content except picture. Sort as title, time, number HUANG: Google Scholar, Obtain title, PDF(link). Sort as title, time,number C.MA: relevant, software, essay and summarize them as a report GAO: web robot, analysis for competitive products analysis for competitive products What to do next：Combine code to FLASK frame Think about the final presentation function of the software Analyze the example of interim report and final report
Task and deadline：	obtain all the required data, combine the code with flask (Y.MA JIANG HUANG) (ddl: 11/8) read DMS ppt and discuss UML diagrams (HUANG C.MA GAO)(11/4) look up for comparing algorithm (to deal with new web)(Y.MA JIANG) JIANG, HUANG: change the loop to non-blocking (ddl:11/8) JIANG, HUANG, Y.MA: ignore repeated content(ddl: 11/8) ALL:read GRPHandbook about interim report (All)(ddl:11/8)
If supervisor attend:	Y

Question	Answer
Which UML diagram should be drawn	it should be as detailed as possible
Interim report	Should contain most of the functions of our software

E.5 Summary of Team18 Informal Meeting

Date (MM/DD/YY):	11/8/2022
Present member:	All
Absent Member and Reason:	
Content:	1. Read the example report given by supervisor and learn how to write report
Task and deadline:	Find resource online to complete the “Background” part(C.MA, GAO) Think about technical aspect for the interim report(Y.MA) Think problems so far(HUANG) FOR ALL: Draw draft about UML Diagram
If supervisor attend:	N

E.6 Summary of Team18 Fourth Formal Meeting

Date (MM/DD/YY):	11/12
Present member:	All
Absent Member and Reason:	
Content:	HUANG: non-blocking loop，separating the time and number JIANG: non-blocking loop Y.MA: monitoring the website，avoid obtain repeated contents C.MA: find essay about crawler GAO: improve analysis for competitive products
Task and deadline:	HUANG: google scholar activity diagram, complete user case diagram C.MA GAO: Shixiseng activity diagram Y.MA: bilibili activity diagram JIANG: xinlang activity diagram (ddl: 11/15) Interim report: HUANG: 1 8 JIANG: 7 GAO: 2 3 Y.MA: 2 3 C.MA: 5 4,6 write after creating prototype 8 appending meeting records
If supervisor attend:	N

Q	A
Can we choose database which is not provided by university	Yes
Can we write our report with markdown	No, we have to use latex, because .md file have to be opened with other softwares
How to deal with user requirement	We can find it online, but need to correspond to our project, if we cannot find proper one,

	we can draw up by ourselves(e.q. informal talk)

E.7 Summary of Team18 Fifth Formal Meeting

Date (MM/DD/YY):	11/18
Present member:	All
Absent Member and Reason:	No
Content:	HUANG: prototype draft part1 Interim reporting Framework JIANG:Sina UML diagram Y.MA: bilibili UML diagram C.MA: Internship UML diagram GAO: prototype draft part2
Task and deadline:	Interim report plan: HUANG: 1 8 JIANG: 7 GAO: 2 3 Y.MA: 2 3 C.MA: 5 4,6 write after creating prototype 8 appending meeting records (ddl: 11.25) Prototype (ddl: 11.22)
If supervisor attend:	Y

Q	A
Do images in prototype need to be approved by the copyright holder?	Yes required
Write in the interim report whether the final presentation, e.g. a web page or applet, can be changed at a later stage of design?	Yes, you can include it in the interim report and add the option requirement
The aim of the interim report	As an aeeseement, Summarize the current progress. Not everything in the report can be changed at will

E.8 Summary of Team18 Sixth Formal Meeting

Date (MM/DD/YY):	11/25
Present member:	ALL
Absent Member and Reason:	No
Content:	report our process: final version of Prototype Interim report
Task and deadline:	The first edition of interim report(ddl: 11/29)
If supervisor attend:	Y

Q	A
If we can collect uses' email address for us to send them emails when uploading	Use free, don't need to fill the ethic form

E.9 Summary of Team18 Informal Meeting

Date (MM/DD/YY):	11/29
Present member:	ALL
Absent Member and Reason:	No
Content:	Introduction: Added content and fixed syntax errors User Interface: Added the user mailbox login page
Task and deadline:	
If supervisor attend:	

E.10 Summary of Team18 Seventh Formal Meeting

Date (MM/DD/YY):	12/9
Present member:	ALL
Absent Member and Reason:	
Content:	1. Telling supervisor that the first edition of the interim group report has been finished and ask him to give us some advice 2. Discussing members who are in charge of front-end code and back-end code **Result: front-end : Haowei GAO, Yuhao MA, Shujun JIANG, Chi MA** **back-end: Ruizhe HUANG, (Assist: Yuhao MA, Shujun JIANG)** 3. constructor of front-end: React back-end: Flask
Task and deadline:	1. Think and write **individual plan** (ddl: 12/16) 2. study how to develop software with particular constructor
If supervisor attend:	Y

E.11 Summary of Team18 Eighth Formal Meeting

Date (MM/DD/YY):	12/16
Present member:	ALL
Absent Member and Reason:	
Content:	The problems of the Interim Group Report: 1. retract 2. grammar 3. tense 4. index of figures 5. name of figures 6. figures should correspond with texts
Task and deadline:	
If supervisor attend:	

E.12 Summary of Team18 Ninth Formal Meeting

Date (MM/DD/YY):	12/30
Present member:	Ruizhe HUANG, Shujun JIANG
Absent Member and Reason:	
Content:	Solve problems in shared databases with our supervisor: Solution: the databases can only be shared under "edurom"
Task and deadline:	
If supervisor attend:	Y

E.13 Summary of Team18 Tenth Formal Meeting

Date (MM/DD/YY):	1/13
Present member:	ALL
Absent Member and Reason:	
Content:	Problems: Ruizhe HUANG: 1. cannot make sure how cookie can be reserved: there are codes to control the cookie (searching online) 2. problems in returning value: multi-thread Tools to test front-end code: “微信开发者工具”
Task and deadline:	Front-end: Y.MA: Bilibili S.JIANG: Sina H.GAO: Google Scholar C.MA: User Menu, login and register pages DDL: before the exam(2/9)
If supervisor attend:	N

E.14 Summary of Team18 Eleventh Formal Meeting

Date (MM/DD/YY):	2/19
Present member:	ALL
Absent Member and Reason:	
Content:	1. make sure DDL final report: 4.6 individual report: 4.24 software complete ddl:4.6 open day: 4.19 2. Meeting time(formal, informal) formal: 10-11 Wednesday informal: 15-17 Saturday 3. Process R.HUANG: back-end complete except multi-thread C.MA: some problem Y.MA: no S.JIANG: exterior, need function H.GAO: complete 4. problems and solution R.HUANG: multi-thread in flask ask jiawei for help(or multi-thread in front-end) C.MA: input fill cannot be seen change modules Y.MA: No S.JIANG: how to range css, layer solved H.GAO: Np
Task and deadline:	
If supervisor attend:	NO

E.15 Summary of Team18 Twelfth Formal Meeting

Date (MM/DD/YY):	3/3
Present member:	ALL
Absent Member and Reason:	
Content:	serve: contact with IT service, if cannot, rent a serve by ourselves and connect VPN Task of final report HUANG: 4.2 4.3 4.4 5.2 5.4 5.5 6.3 6.4 Y.MA: 5.1 5.6 C.MA: 8.1 9.1 JIANG: 4.4 Appendix 3 GAO : 1-3 7 Each: 4.1 6.2 after complete software: 5.3 6.1 8.2 8.3 9.2 10
Task and deadline:	
If supervisor attend:	

E.16 Summary of Team18 Thirteenth Formal Meeting

Date (MM/DD/YY):	3/8
Present member:	
Absent Member and Reason:	
Content:	Problem of the final report: 1. Indenting : 2. interval between the subtitle and the text: 3. add reference:
Task and deadline:	improvement: academic: the result of multiple academic websites: Google, Bing, Baidu, ResearchGate Adjusting according to request(point front end) R.HUANG: Bing crawler code S.JIANG: Design front end about Bing, (switch between google and bing)
If supervisor attend:	

E.17 Summary of Team18 Fourteenth Formal Meeting

Date (MM/DD/YY):	3/12
Present member:	ALL
Absent Member and Reason:	
Content:	
Task and deadline:	complete the first edition of the report (ddl: 3/18)
If supervisor attend:	

E.18 Summary of Team18 Fifteenth Formal Meeting

Date (MM/DD/YY):	3/19
Present member:	
Absent Member and Reason:	
Content:	order of citation interval between subtitle and text
Task and deadline:	
If supervisor attend:	

E.19 Summary of Team18 Sixteenth Formal Meeting

Date (MM/DD/YY):	3/26
Present member:	ALL
Absent Member and Reason:	
Content:	1. Introduction, Requirement simplify 2. "List of table", AppendixB, "vi" delete 3. Requirement, 3.2.5, 3.2.6 delete 4. 3.3, 3.4, 3.5 combine 5. Time table 6. rest meeting record 7. check grammar error
Task and deadline:	
If supervisor attend:	

E.20 Summary of Team18 Seventeenth Formal Meeting

Date (MM/DD/YY):	3/29
Present member:	
Absent Member and Reason:	
Content:	1. clone the project in git lab and try to run. 2. software: marked, Assessed(pdf), user manual(pdf). 3. draft of presentation, demo, ad
Task and deadline:	
If supervisor attend:	