Abstract

Chinese characters has its long history. However, it is still evolving from time to time. There are a lot of websites working on information about Chinese characters and words. However, many of them only focus on a single area. User may still need to browse for several web pages to search for specific data they need.

This project aims to create a public sourcing database for Chinese characters and words information. The database need to be flexible enough to cater different user inputs.

This report introduces the project background, basic features and the methodology for implement this system.

The source code will be open to the public for future enhancement. We believed that the source code can benefit the public and can be used in other types of public sourcing database system.

Acknowledgements

First, I would like to thank my groupmate, Siu Lik Hang, Leo, for his hard work. I would also like to thank my advisor, Dr Vincent Lau for guiding and giving insightful feedback to me, which made this project can stick to the right track. I would also like to thank Dr. Beta Yip for giving us useful advice and comments after our presentation.
Glossary

**Chinese Character:**
In Chinese, 中文字. A character is the basic unit of Chinese language. Each character has its own shape (formed by strokes), component, meaning and pronunciation.

**Chinese Word:**
In Chinese, 中文詞語. A word is formed by 2 or more Chinese characters (usually, 2 to 4 characters). Usually, the meaning of a word is based on combining the meaning of the formation characters. However, there are also some cases that the meaning is based on other factor, such as traditions and story, instead of the meaning of the characters which form it.
## Abstract

2

## Acknowledgements

2

## Glossary

3

## Project Overview

6
- Introduction
- Project background
  - Related works
  - Website that use public sourcing model
- Objective

## Design and methodology

9
- Features
  - Data related features
  - Operation related
- Implementation details
  - Web server
  - Database
  - Coding Language
  - Encoding
  - Flexible database structure
  - Static tables
  - Security related issue
  - Performance related issue

## Testing and result

37
- Performance test

## Other Project deliverables

38

## Division of labour

39

## Difficulties, Regrets and System Limitation

40
- Difficulties
  - Flexible database structure
  - Limitation of time
- Regrets
  - Website framework
- System Limitation
  - Data validation
  - Uploading files

## Future works and Conclusion

41
- Future improvement
Better data validation 41
Better file management 41
More user management functions 42
Improve database structure 42
Develope to entertain more generic usage 42
Conclusion 43

Reference 44
Project Overview

Introduction

Chinese characters and words has a long history. There are a lot of information about Chinese characters and words. However, these information is distributed in different website. Moreover, the information may not be up to date as Chinese character and words may evolve from time to time.

This project aim to create a platform for recording the information of Chinese characters and words. We hope this platform can include as many information as possible and gather information from different sources. This platform should also be flexible so that it can handle different types of data.

Therefore, our group create a database and a corresponding website to store and show information. In order to gather information, we allow user to contribute to our website. Moreover, our system are flexible enough that user can enter different types of input. Even if there are some new kinds of information input, the system can store the information in a proper manner and user can easily search for that record later. This feature can handle the fact that information and information structure of Chinese characters and words may evolve.
Project background

Related works

Websites about Chinese information

Currently there are a lot of websites working on Chinese information. However, most of them only focus on one area of Chinese. If a user need to find some specific data of a Chinese character or words, he / she may need to browse several websites to find which websites contain those type of data.

We hope that our website can solve this problem by providing all types of information about Chinese so that user do not need to browse different websites for searching.

Example:

Chinese Character Database: With Word-formations Phonologically Disambiguated According to the Cantonese Dialect (《粵語審音配詞字庫》)
http://humanum.arts.cuhk.edu.hk/Lexis/lexi-can/

This database is developed by Chinese University. It includes many Chinese characters, with accurate information such as their script (how it writes), pronunciations and meaning. However, it mainly focus on character information and has little explanation on Chinese words.

Words.hk (《粵典》)
http://words.hk

This database is mainly about characters, words and idioms used in Cantonese. It includes pronunciation, meanings and usage of them. However, it focus on the
Cantonese side and has few information about basic information of character (such as component) and the usage in written Chinese.

**Website that use public sourcing model**

Traditionally, a website should prepare the content for the users. However, in public sourcing model, user provide contents to the website.

**Example:**

Wikipedia  
[https://www.wikipedia.org/](https://www.wikipedia.org/)

Wikipedia is a free online encyclopedia with the aim to allow anyone to edit articles [1], which support more than 299 different languages. Although it may be difficult to validate whether a user is contributing correct information, Wikipedia model has proved that public sourcing model is an effective way to collect information.

**Objective**

This project aims to create a web tool that contains as much information as possible about Chinese characters and words. User can updated the information in order to show the latest status of the characters, word and phrases. To achieve this, the website will be opened to users to contribute their knowledge to the website.

The goal of this project is to create a website that has the following features:

1. Serving as a one-stop convenient tool for searching Chinese characters/words information
2. Providing a platform for users that can contribute on updating the latest Chinese characters information.

3. Flexible database structure is used to allow enhancement on database. User can input new type of data related to Chinese characters. For example, there are only meanings and pronunciation about Chinese characters at the website originally. User is allowed to add a new column ‘big5 code’ if they wish to store the big5 code in the system.

4. Source code is opened to the public to allow improvement on our website and creating other similar database system for public interest.

Design and methodology

Features

Our system provide the following features.

Data related features

The most important part of this project is to handle the data from user. Our website provide features for user to browse, search and edit the data.

1. Dictionary

When user click on a character/word, this dictionary page will be shown. This page contain all the information related to that character or word. The information is classified in multiple tabs. User can browse by clicking on different tabs. See figure 1 for example.
We now provide 2 dictionaries, one is for Chinese characters and another is for Chinese words.
2. Internal linkage

Internal link refers to the link that point to other pages of this website. With proper internal links, user can easily browse other related page, which creates better user experience.

Auto linkage

In the pages, there are auto linkage to links to other pages.

Figure 2: Auto internal linkage

Under the dictionary tabs, the system will search for other words or characters that is related to the current page and provide a hyperlink to access that page. For Chinese words, the system will search for the characters that formed the words. For a Chinese character, system will search for words that is formed by it.

Manual linkage

When user need to create an internal link manually, he need to first type a space, then use 2 @ signs to enclose the character or words that need to be linked.
For example, when we input ‘123 @abc@’, than abc will become a hyperlink to link to page ‘abc’. You may find figure 3 to see how this function is used.

Figure 3: Compare normal record and records that contains internal links

User inputs ‘他是個好人’ in the upper record, so the record does not have the links. The lower record contains internal links because user inputs ‘他是個 @好人@’. So the lower record will create a link to the page ‘好人’.
3. Search function

User can search for the information needed using our search function. The search form is automatically generated according to the data in the system.

Figure 4: Search form example 1
The search form is auto generated by the system. Each section is represent a tab in the dictionary page, and each data field underneath is the data available under that tab. That means that all the user input is searchable.
Moreover, the search fields will change according to the data type. For example, for text fields, it will provide a text column for user. For number field, it provide two boxes for user to input a range.

4. Listview

This function allow user to choose some type of data and display all information of that data. User can compare multiple records at the same time.

Figure 6: Listview search form
<table>
<thead>
<tr>
<th>字</th>
<th>倉颉碼</th>
</tr>
</thead>
<tbody>
<tr>
<td>一</td>
<td>M</td>
</tr>
<tr>
<td>丁</td>
<td>MN</td>
</tr>
<tr>
<td>七</td>
<td>JU</td>
</tr>
<tr>
<td>丈</td>
<td>JK</td>
</tr>
<tr>
<td>三</td>
<td>MMM</td>
</tr>
<tr>
<td>上</td>
<td>YM</td>
</tr>
<tr>
<td>下</td>
<td>MY</td>
</tr>
<tr>
<td>不</td>
<td>MF</td>
</tr>
<tr>
<td>四</td>
<td>MVVS</td>
</tr>
</tbody>
</table>

Figure 7: Listview result
5. Data management
User can add, edit and delete data into the database.

Create new pages (Add a new character or word in the database)
User can create new pages by inputting the form in figure 8.

![Figure 8: Form for adding new character](image)

**Edit data**
User can make changes to existing data, delete the data and create new data for a page. The edit form will also change according to the data type. User can input text, numbers and hyperlinks in the system. He/She may also upload pictures and files to the system.
Recover data

User can view for the latest changes of that page under the dictionary page. If there are any deleted record, user may choose to recover the data. The system can add back the deleted data into the database.
6. Data upload

Data management function can only handle one page at one time. If user need to insert a lot of data, they may use data upload function.

To use data upload function, user first need to prepare a file with the correct format:
1. The file type should be in unicode txt format
2. The first column is the page that you want to update.
3. At the end of each row, an indicator word END must be added.
4. On each row, the data is separated by a tab.
5. For each file, user can only choose to update either character or word dictionary, cannot update both dictionary at the same time.
After uploading the file, the system will display the result on the screen. User just need to choose what is the meaning of that column and then click 'send' button. The data will be uploaded to the database.

Figure 11: Example upload file

Figure 12: Upload function
7. Structure management

Users can edit the structure of the dictionary. They can add new tab in the dictionary page or add new kind of data under a tab.

Figure 13: Form for adding new tab and new data type

When adding new tab, user can decide the name of that tab. For adding new data type, besides naming, user also need to specifies what kind of data should be store. Currently, user can choose between text, numbers, multimedia, file and external hyperlinks.

For deleting or renaming a tab or data type, since it may have big influence to all the data. Only administrator can perform such operation. The additional option besides the tab and data type only appears when an administrative user has logged in.
Operation related

Beside the data related function, we also have other functions which help to increase operational efficiency and improve user experience.

1. User management

Everyone can use our website to search and browse for information. However, user need to login before they can edit the data in the database. As a result, system can record who has edited the record. Furthermore, the system can enable additional functions to the current user if he/she is an administrator.

To enable this function, the system provide functions such as user login, apply user account and change password function.
A page is created for each user. One can view the data of a particular user and link to the pages that he or she has recently edited.

![User information page](image)

**Figure 15: User information page**
2. Communication

Discussion is useful when sometimes the information is unclear. User can communicate to each other under each page to discuss about the correctness of the data.

Also, every user can send message to the administrator to ask for help.

Figure 16: Discussion area under dictionary page
Implementation details

Web server

We use the web server provided by the faculty for hosting the web pages.

Database

We use MySQL database for building our database system.

Coding Language

For the server side script, we mainly use php and mySQL to handle the data and output.

For client side script, we use basic html for displaying data and javascript for event handling.

Encoding

Since our database need to handle Chinese input, the encoding is important. Otherwise, the input and output will be affected. For both database fields, php sql setting and html character set, we use unicode (utf8) for encoding.
Flexible database structure

Since we aim to collect different types of data from the public, we need to structure all these data well in order to allow user to search for these data easily in the future.

To achieve this purpose, we adopted a flexible database structure. In the database point of view, users can create new tables and new column for some new types of data instead of only inserting new records.

In our database structure, tables can be classified in 3 types.

Base tables

Base tables are used to define what records are available in the database. If a character or word does not appear in the base tables, the webpage will not create a page for that character or words.

In the system, we now have 2 base tables, called base_1 and base_2. The former is used to handle Chinese character and the latter is for Chinese word.

Table name: base_1

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>varchar(10)</td>
<td>Store the character that have a page in the website.</td>
</tr>
</tbody>
</table>
Table name: base_2

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>varchar(100)</td>
<td>Store the word that have a page in the website.</td>
</tr>
</tbody>
</table>

Attribute tables

Attribute tables are used to store the data related to the bases. They are the part that contribute to the flexibility of the system. When user add new column, create new tables or insert new record, the changes are reflected in the attribute tables.

In fact, for one attribute table, it is represented as a tab in the dictionary pages. Each column in that attribute table is the data type. Therefore, when user adding new tabs or data type, they are creating new attribute tables or adding new columns.

This table show the structure of an attribute table.

Table name: attribute_1

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique_key</td>
<td>int(11)</td>
<td>The unique identifier of a record. It is the primary key and will auto increase by 1 upon insert.</td>
</tr>
<tr>
<td>Base</td>
<td>varchar(100)</td>
<td>The base that this record belongs to.</td>
</tr>
<tr>
<td>Column_1</td>
<td>varchar(1000)</td>
<td>Store text data for this record.</td>
</tr>
<tr>
<td>Column Name</td>
<td>Data Type</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Number_1</td>
<td>double</td>
<td>Store number data for this record.</td>
</tr>
<tr>
<td>Image_1</td>
<td>varchar(100)</td>
<td>Used for storing multimedia data (such as image, audio, video) for this record. This field records the file name of that data so the webpage can display the data correctly.</td>
</tr>
<tr>
<td>File_1</td>
<td>varchar(100)</td>
<td>Used for storing file for this record. This field records the file name so the webpage can create a link to the file correctly.</td>
</tr>
<tr>
<td>Link_1</td>
<td>varchar(100)</td>
<td>Used for storing the hyperlink to external webpage.</td>
</tr>
</tbody>
</table>

For the starred field, they may or may not be included in that table.

When user request to add a column, he need to choose a format from all the starred column. The naming will be automatically done by the system. For example, the first image column created in the table will be called Image_1, the second will be called Image_2.

The case for table name is similar. The table name is auto generated by the system upon creation. The first attribute table created is called attribute_1, and the second attribute table is named attribute_2.
Static tables

These tables served the purpose of operating the webpage. The structure of these tables cannot be changed by the user.

**Map table**

This table is an important table for displaying the Chinese information correctly in the dictionary function. It maps the attribute tables with suitable description. It will translate the table name and column name into user's language. As a result, user can know the actual meanings of each column in the attribute tables instead of meaningless Column_1 and Column_2.
Table name: map

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table_name</td>
<td>varchar(50)</td>
<td>The target attribute table this record describe.</td>
</tr>
<tr>
<td>Column_name</td>
<td>varchar(50)</td>
<td>The target column this record describe.</td>
</tr>
<tr>
<td>Relation</td>
<td>varchar(40)</td>
<td>The base table this record is relate to.</td>
</tr>
<tr>
<td>Describ</td>
<td>varchar(40)</td>
<td>The description of that table / column. This description is displayed in the webpage.</td>
</tr>
<tr>
<td>Type</td>
<td>varchar(1)</td>
<td>The type of the record. There are 3 possible type. B - This record is used for describing the title of a base table. C - This record is used for describing the title of a column in the attribute table. T - This record is used for describing the title of the attribute table.</td>
</tr>
</tbody>
</table>

Audit log

This table record the changes that user has been made to the system, including modifying attribute tables’ structure and editing data. This table store the sql statement that has been run to implement the changes.

This table also contribute to the recover function. The system find the corresponding insert statement for a deleted record to insert back the record in the database.
When necessary, administrator can study the logs for restoring the database. In the most extreme case, this table can serve as a backup of the database. We can build a new database and run all the sql statement in sequence to rebuild the system.

Table name: audit_log

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unique_key</td>
<td>int(11)</td>
<td>The unique identifier of a record. It is the primary key and will auto increase by 1 upon insert.</td>
</tr>
<tr>
<td>Table_name</td>
<td>varchar(50)</td>
<td>The table that has been modified.</td>
</tr>
<tr>
<td>Column_name</td>
<td>varchar(50)</td>
<td>The column that has been modified.</td>
</tr>
<tr>
<td>Base</td>
<td>varchar(100)</td>
<td>The base (character / words) this record is relate to.</td>
</tr>
<tr>
<td>Action</td>
<td>varchar(10)</td>
<td>The action has been done. There are several type:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Add base - Add new record in the base table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Add column - Add new column in the attribute table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Add map - Insert new record in map table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Insert - Insert new record in the attribute table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Delete - Delete record in the attribute table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- New table - Create new attribute table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Del column - Delete a column in attribute tables.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Del map - Delete a record in map table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Del table - Delete an attribute table.</td>
</tr>
<tr>
<td>Column</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SQL_run</td>
<td>varchar(200)</td>
<td>The sql statement has been run for this action.</td>
</tr>
<tr>
<td>Lst_upd_date</td>
<td>timestamp</td>
<td>The date that this record is created.</td>
</tr>
<tr>
<td>IP</td>
<td>varchar(100)</td>
<td>The ip address of the machine that user used to do the changes.</td>
</tr>
<tr>
<td>User_id</td>
<td>varchar(50)</td>
<td>The user_id that made the changes.</td>
</tr>
<tr>
<td>Remark</td>
<td>varchar(100)</td>
<td>For insert and delete, remark store the record’s corresponding Unique_key in that attribute table. It is used for recover the deleted record. For other action, remark store the Chinese description of that action just for convenient.</td>
</tr>
</tbody>
</table>

The base tables, attribute tables, map table and audit log formed our flexible database system to let user manage the data and alter the structure of the attribute tables.
Other static tables

There are also other static tables in our system. Although they are not contributing to the flexible structure of the system, they are important for providing operation related functions in our system.

Log in related

To enable user management functions, we need to have tables related to user account. User_info table stores all information about a user.

Table name: user_info

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User_id</td>
<td>varchar(50)</td>
<td>The unique identifier of a user. User can login using this username.</td>
</tr>
<tr>
<td>Field</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Email</td>
<td>varchar(100)</td>
<td>The email address of the user.</td>
</tr>
<tr>
<td>Status</td>
<td>varchar(1)</td>
<td>The status of the user.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>A - This account is active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L - This account is locked.</td>
</tr>
<tr>
<td>Date_create</td>
<td>timestamp</td>
<td>The date of creating this user account</td>
</tr>
<tr>
<td>Passwd</td>
<td>varchar(100)</td>
<td>The password of the user. When user login, system will compare the input password with this field to check if the password is corrected. For security purpose, the value of this field is hashed. No one can get the actual password of the user using this field.</td>
</tr>
<tr>
<td>Role</td>
<td>varchar(10)</td>
<td>The role of the user. Currently, there are 2 roles:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>User - normal user</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Admin - Administrator</td>
</tr>
<tr>
<td></td>
<td></td>
<td>According to user’s role, the system has different behaviour. For example, some function may only available to administrator.</td>
</tr>
<tr>
<td>Lst_upd_date</td>
<td>timestamp</td>
<td>The last update date of this record. When user change his/her password, this field will be updated.</td>
</tr>
</tbody>
</table>
access_Log table record the date and time a user try to login the system.

Table name: access_Log

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User_id</td>
<td>varchar(50)</td>
<td>The unique identifier of a user.</td>
</tr>
<tr>
<td>Login_result</td>
<td>varchar(20)</td>
<td>The result of login.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Success - The user login successfully.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Fail - The user failed to login the system. (Typing wrong username or password).</td>
</tr>
<tr>
<td>Access_date</td>
<td>timestamp</td>
<td>The date of this login attempt.</td>
</tr>
<tr>
<td>Access_ip</td>
<td>varchar(100)</td>
<td>The ip address of this user who try to login.</td>
</tr>
</tbody>
</table>

Communication related

Both these tables, admin_log and comment stored the message for users. The former saves all the message sent to administrator, and the latter saves the message in the discussion area under the dictionary page.

Upon loading the dictionary page, the system will retrieve the message related to that page.

Table name: admin_log

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int(11)</td>
<td>The unique identifier of the message.</td>
</tr>
<tr>
<td>Issue</td>
<td>varchar(100)</td>
<td>The type of the message.</td>
</tr>
</tbody>
</table>
The message user has input.  

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>int(11)</td>
<td>The unique identifier of the message.</td>
</tr>
<tr>
<td>page</td>
<td>varchar(100)</td>
<td>The page that this message is related to</td>
</tr>
<tr>
<td>post_name</td>
<td>varchar(100)</td>
<td>The user who post this message.</td>
</tr>
<tr>
<td>time</td>
<td>date</td>
<td>The time when the user send the message.</td>
</tr>
<tr>
<td>content</td>
<td>varchar(1000)</td>
<td>The message user has input.</td>
</tr>
</tbody>
</table>

Security related issue

**SQL injection**

As a website which its function highly depends on interaction with the database, SQL injection attack is a threats. One of the common approaches to solve this problem is to use SQL parameters. However, since our table names and column name is not static, it is difficult to use the parameter.

Our workaround solution is to check the user input when passing the data. All special character, such as semicolon ( ; ) and comment characters ( --) will be
removed in the input string. In this case, it is more difficult to use SQL injection to attack our system.

**Session control**
To handle situation when user forgot to logout, time limit for each session is set. If user does not use the system for 30 minutes, the session will automatically expired.

**Performance related issue**

**Indexing**
When there are more data in the tables, the performance of our data is worse. After creating index on the field Base in the attribute tables, the performance improve significantly. For details, please find the ‘Testing and result’ part.

**Testing and result**

**Performance test**
To test the performance of our system, we conduct the following testing.

**Test 1:**
Before indexing the attribute tables on the Base field, upload a file with 5283 rows, 1 input column.
Upload time: around 20 seconds
Use listview function after uploading: around 16 seconds
Use search function after uploading: around 20 seconds
Loading one dictionary page: around 10 seconds
Test 2: Create index on the Base field of table attribute_3, than upload the same file
Upload time: around 20 seconds
Use listview function after uploading: around 4 seconds
Use search function after uploading: around 2 seconds
Loading one dictionary page: within 1 second

The result show that index on the ‘Base’ field of the attribute tables helps increase the performance.

Other Project deliverables

Since we decide to makes our project open source, we have prepare the following deliverables:
1 . Source code of the web application
2 . Source code for setting the database.
3 . An MIT license to declare the copyright issues
4 . A readme.txt to introduce our project
5 . A detail user guide for teaching user how to setup and use our system.

We hope that by opening our source codes to the public, others can benefit from our project and help to improve our system.
**Division of labour**

To develop this system, Leo and I each responsible for some of the functions.

Please find below tables for our tasks

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Wong Tsz Hing, Chris</th>
<th>Siu Lik Hang, Leo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interface design</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Web interface</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dictionary function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internal linkage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Search function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Listview function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data management function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data upload function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Structure management function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User management related function</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Communication related function</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Difficulties, Regrets and System Limitation

Difficulties

Flexible database structure

When designing the database to be flexible, we face a lot of difficulties at the beginning. We have modified the table structure many times, trying to store all the useful information to make the structure works. During coding, the uncertainty of the tables and column name are sometimes confusing. Fortunately, our design function well at the end.

Limitation of time

Both my teammate and I have to handle other courseworks as well as this project. As a result, we cannot spend too much time on this project, which means that the schedule of the project is really tight.

Regrets

Website framework

At the start of the project, we have not choose any framework for our website. The outcome is that our website’s interface is not attractive enough. However, when we discover this problem, it is already too late. It is difficult to integrate all the current codes into a framework.

We should choose a framework at the beginning of the project.
System Limitation

Data validation

For our system, strict data validation has not been implemented yet. For example, no validation is done during user upload. It is possible that user may input wrongly in the database, like inputting texts in number fields.

Uploading files

Currently, when user upload a file, the file will be stored in the upload folders. However, when user delete the file record, the changes will only reflect in the database. The actual file in the upload folder will not be removed. Therefore, when user recover a record, that record can be re-link to that file. Nevertheless, these files will occupied the server space and administrator may need to manage those files manually.

Future works and Conclusion

Future improvement

Better data validation

Stricter data validation should be imposed in edit function and upload function to increase the quality of user input.

Better file management

The rules of file management should be developed so that the system can have a balance between serverspace and needs for recovery data.
More user management functions

Currently, the user management function has not yet completed. For example, there is no functions handling cases of locking a user account or user forget his own password. More functions are needed to complete user management features.

Improve database structure

The flexible database structure can be improve. One direction is that we can add a status field in the attribute tables. With this fields, we can define the status of a record, such as ‘approved’ or ‘pending for approval’, so more control can be done on user input.

Develop to entertain more generic usage

After finish developing our flexible database structure, we find that it can serve for a more generic purpose other than only storing Chinese information.

Imagine the situation if we use base table to store the name of animals. The attribute tables can use to classified the data into habit, information of their bodies, type, where they live. After inserting the data, user can easily use our system to do searching like Finding an animals which its type is mammal and live in Africa. We believe that we can use the system to serve a more general purpose when modifying some of the codes.
Conclusion

We have developed a flexible database system with a corresponding web application. With this application, we can collect different type of data about Chinese character and words from users. All the user input are classified well and can be used for searching. This application is suitable for handling the evolution of Chinese data.

We will open the source code to the public. We hoped that our system can benefited the public, and the system can be enhanced by the public at the same time. We believe that our system can also be used for a more general purpose, such as building a public sourcing database of other objects.
Reference