**COMP3322 Modern Technologies on World Wide Web**

**Lab 4: jQuery**

Overview

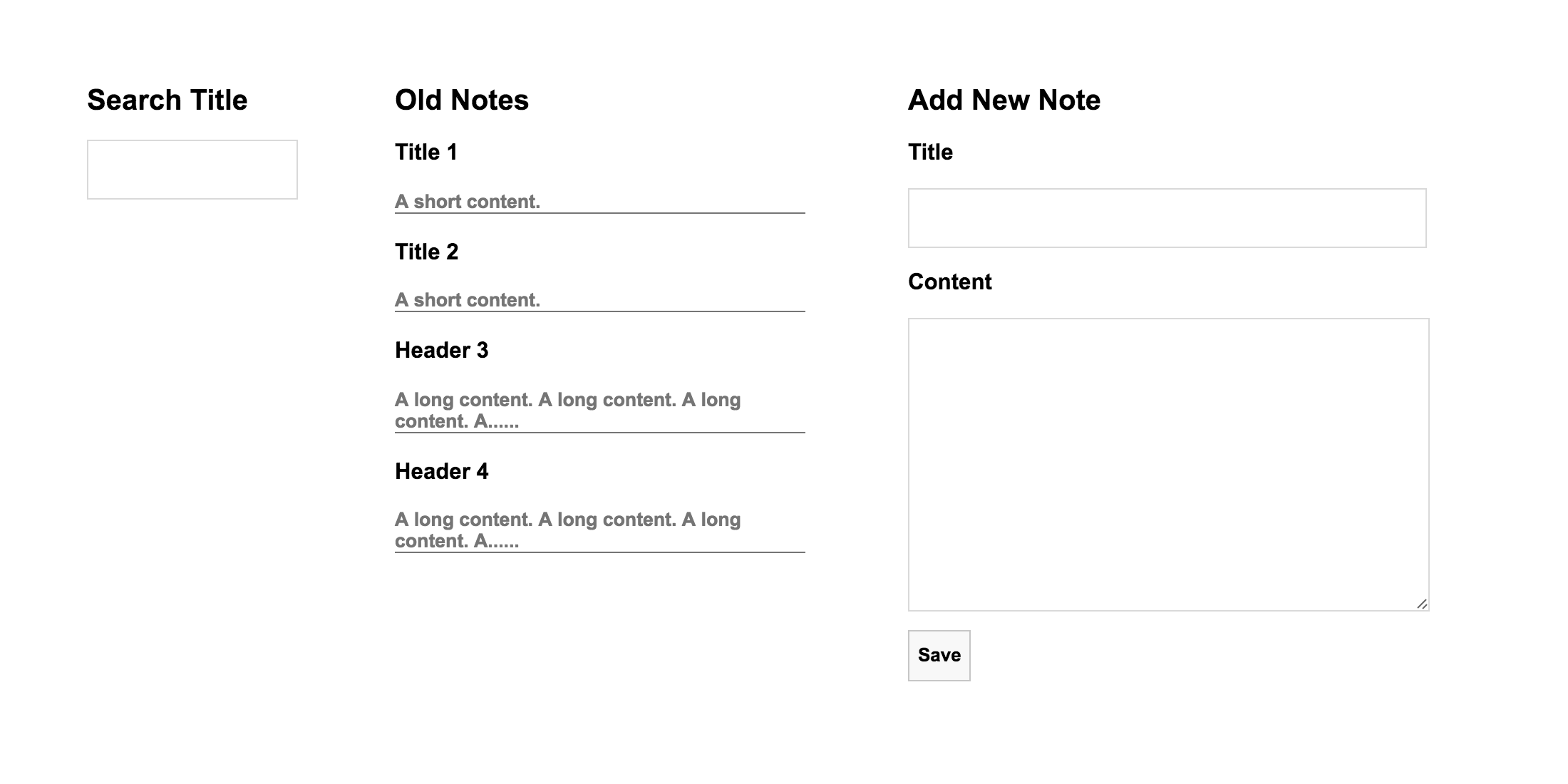
In this lab exercise, we will develop a dynamic webpage to store and update simple notes, using jQuery and PHP.

When a user accesses the web page “index.html”, the page shows up as in Fig. 1 below. The page is divided into 3 parts.

On the left part, you can search the notes based on the title.

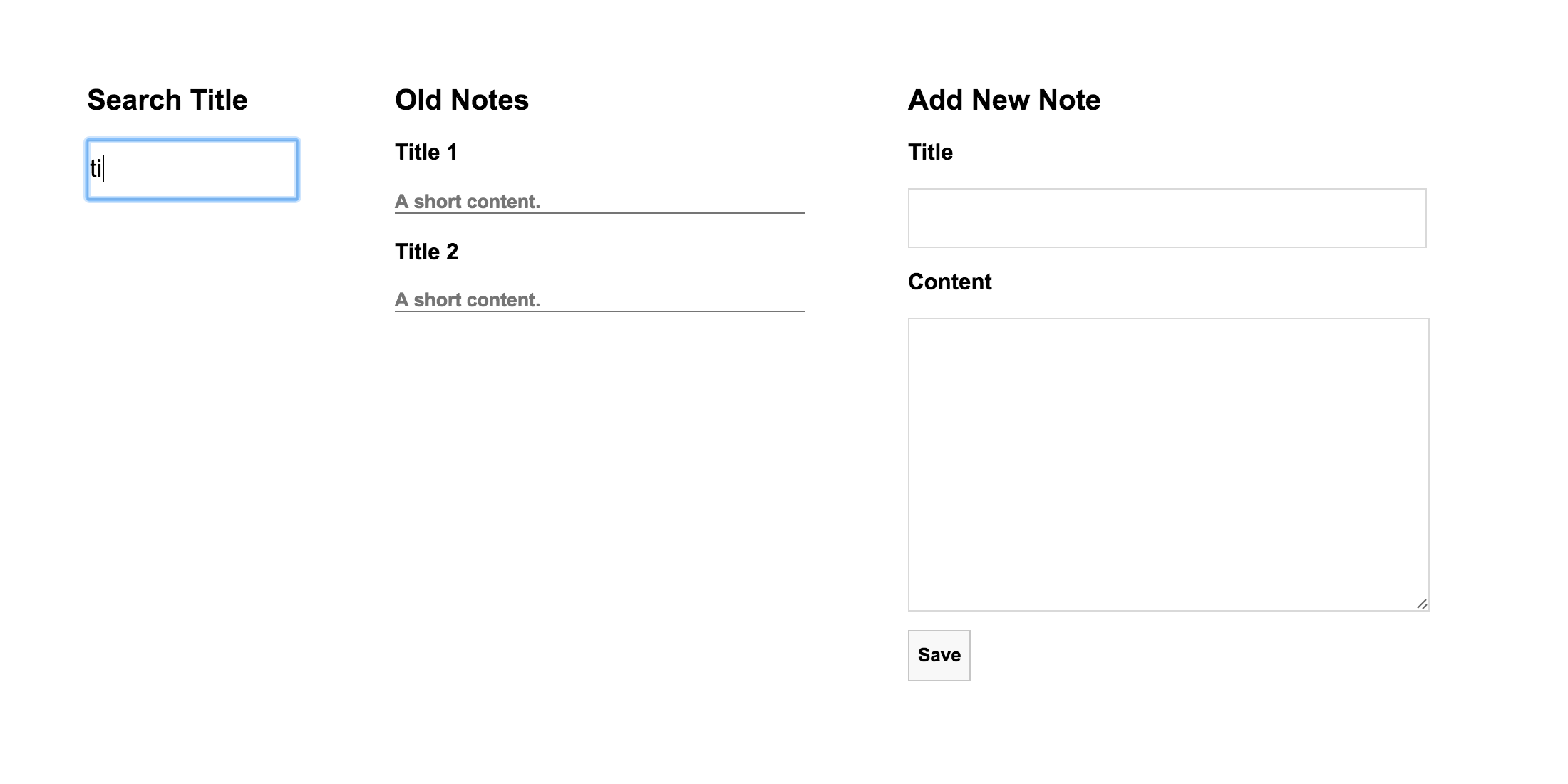
On the central part, all the notes stored in the database are shown, with their full titles and simplified content. The simplified content includes at most 10 words of the original content, plus a “……” symbol if the content is longer than 10 words.

On the right part, you can add a new note by filling out the title and content and clicking the save button.



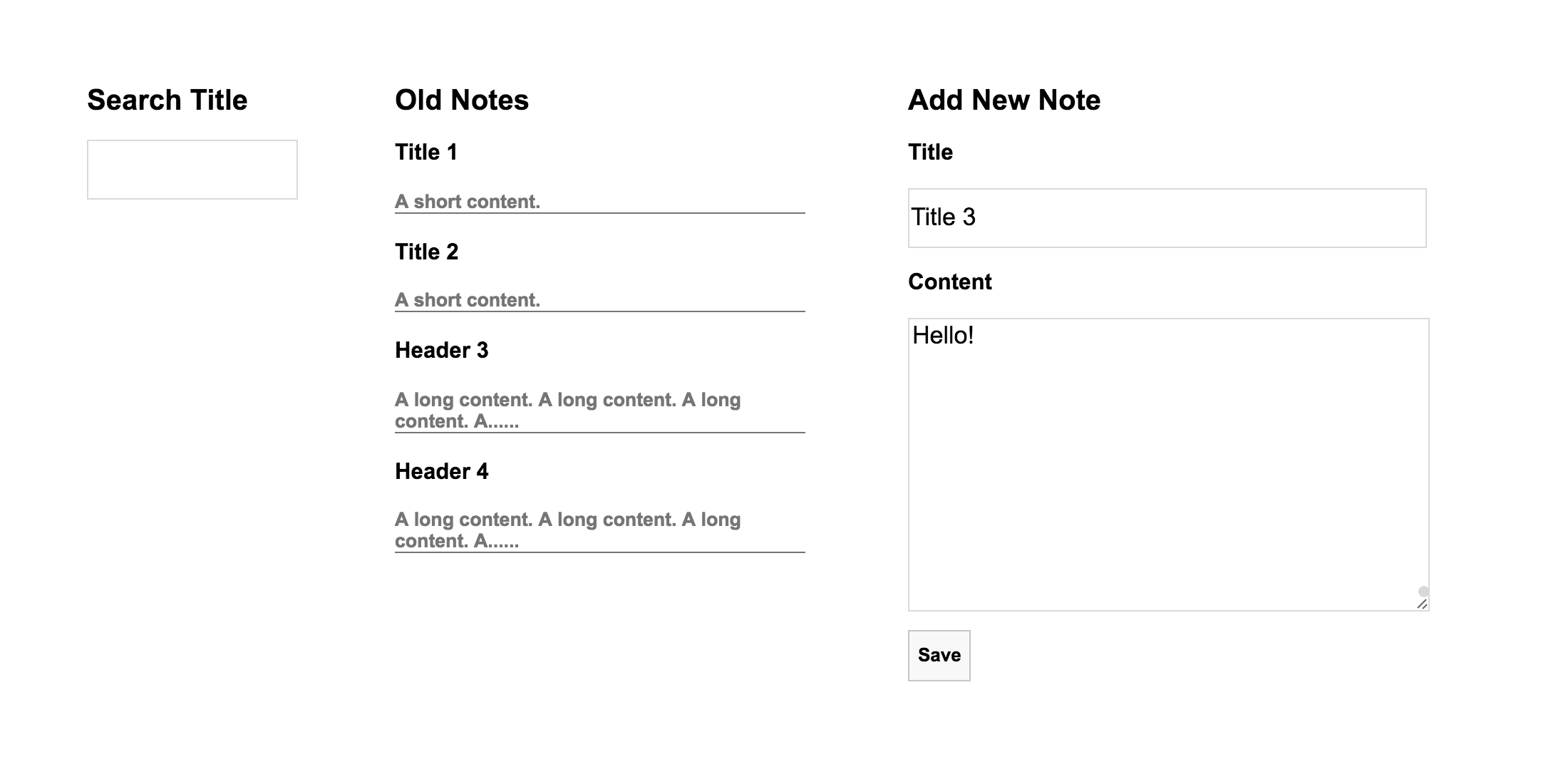
**Fig. 1 index.html web page.**

If the user types in a search string in the input box under search title, a prefix-matching search is started and only the matching notes are shown under old notes. The search is case insensitive. For instance, if we type a single character “ti” as shown in fig. 2, then the notes that are shown under old notes are “Title 1” and “Title 2”.



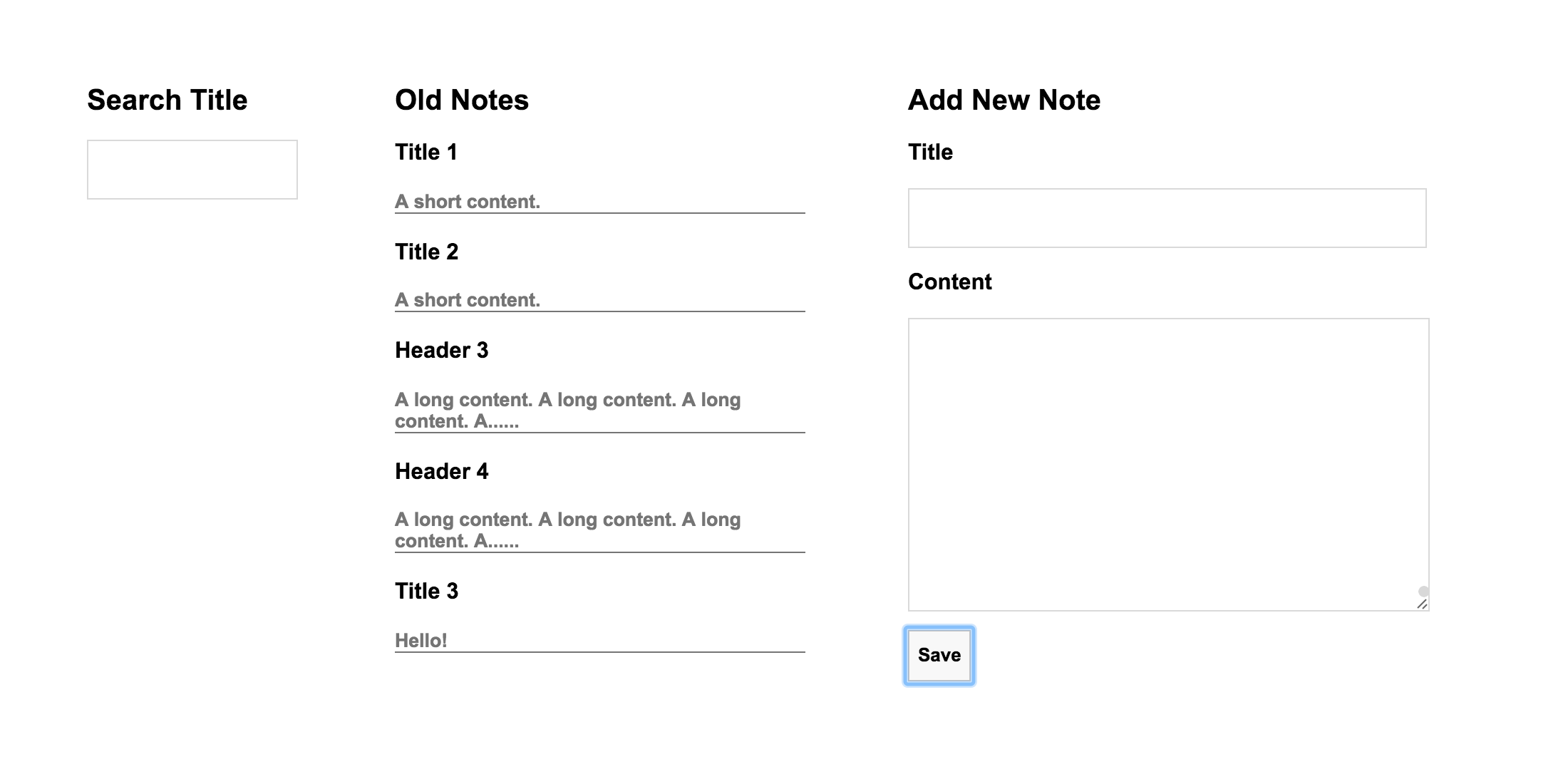
**Fig. 2 prefix-matching search**

If the user wants to add a new note, he can fill in the title and content under add new note, as shown in fig. 3.

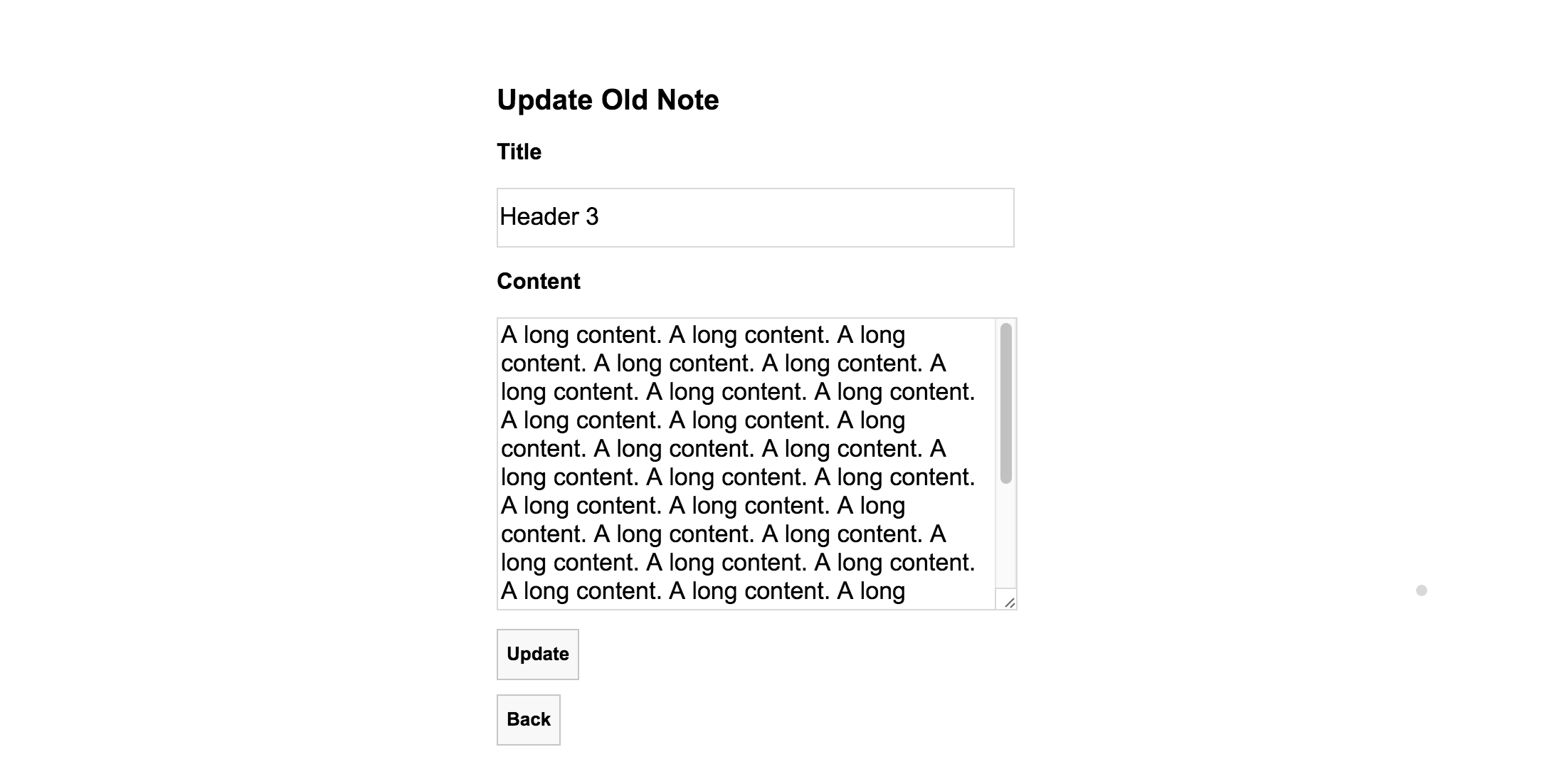


**Fig. 3 Add new note.**

After he clicks the save button, we can see the new note is successfully inserted into the database and shown under old notes in fig. 4.

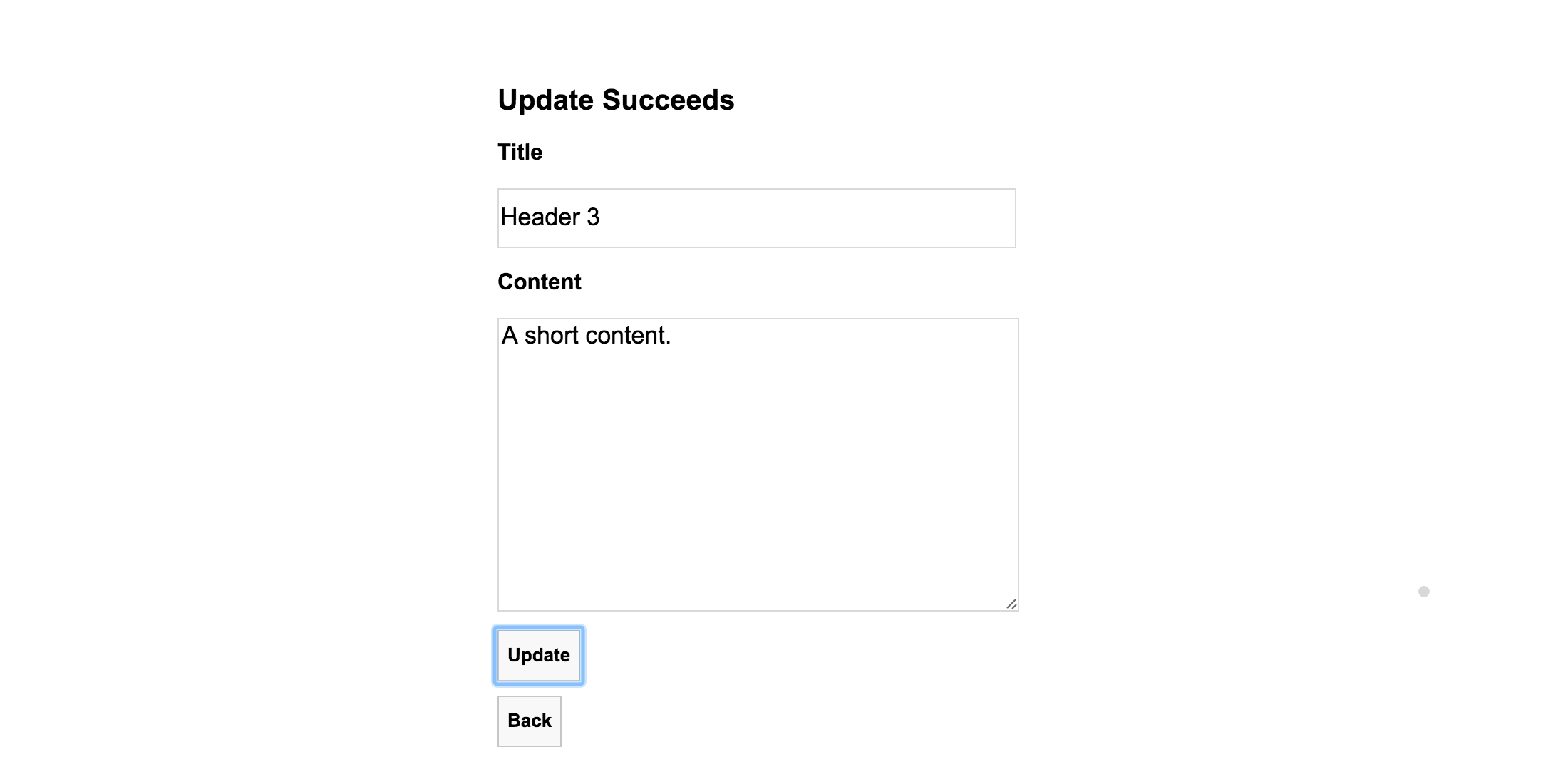
 **Fig. 4 The new note is inserted to the database.**

If the user clicks the title of the of an old note, he is directed to fig. 5. In the web page shown in fig. 5, he can read the full content of the old note and update both the title and the content of the old note.



**Fig. 5 Update old note.**

If the user modifies the content of this old note and click update button, the header is replaced by “Update Succeeds” as shown in fig. 6.



**Fig. 6 Update succeeds.**

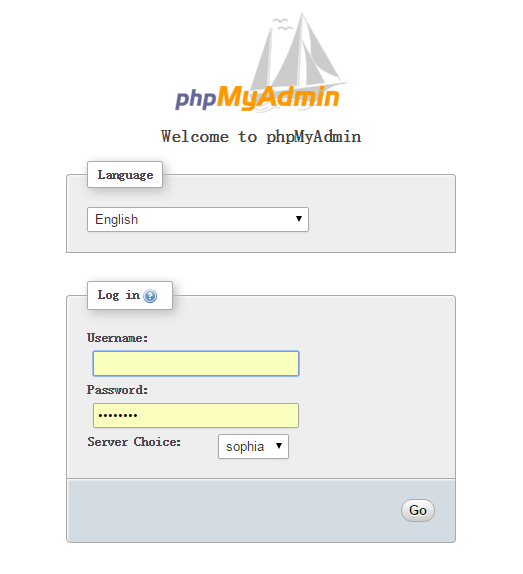
Finally, when the user finishes updating the old note, he can click the back button to go back to the web page shown in fig. 1.

Lab Exercise 1: Set up the Database

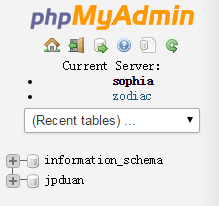
User profiles are permanently stored in a database. So the first thing that we need to do is to set up the database for storing user profile.

**[Step 1]**  Go to <https://intranet.cs.hku.hk/common/mysqlacct/register.php> and register a MySQL account, if you have not done so. It takes about one working day for CS technical staff to activate your account.

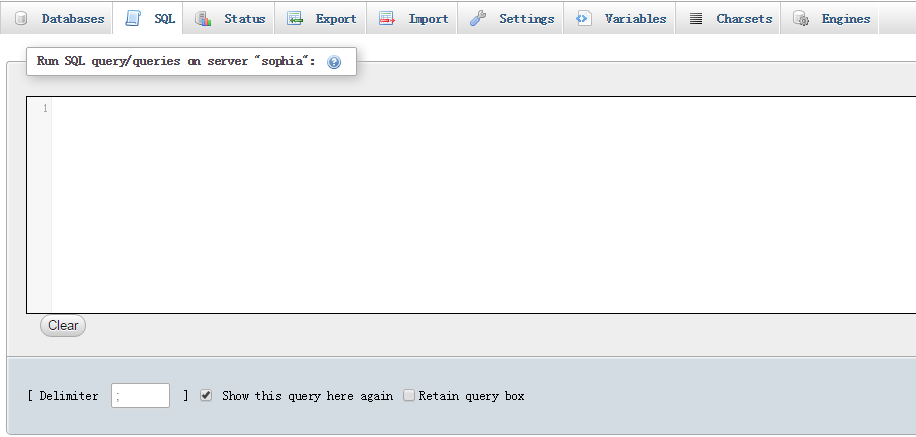
**[Step 2]** Go to https://i.cs.hku.hk/phpmyadmin/. Log in using your MySQL account. Choose the server **Sophia**. We will use the MySQL database hosted on sophia.cs.hku.hk.



**[Step 3]** After logging in, select your database on the left column. Your database name should be the same as your account name.



**[Step 4]** Click the SQL tab and you will see an input field for executing SQL code.



**[Step 5]** Create **notes** table by executing the following SQL code. The **notes** table contains the **id**, **title** and **content** field. (All SQL statements are in sql.txt in lab4\_materials.zip, which you can copy and paste.) Please insert at least 4 notes in the database.

CREATE TABLE notes (

id int NOT NULL,

title varchar(20) NOT NULL,

content longtext,

PRIMARY KEY (id)

)

INSERT INTO notes( id, title, content )

VALUES (

1, 'Title 1', 'A short contet.'

)

**Lab Exercise 2: Implement the Dynamic Webpage**

Complete all the **TODO** items in script.js, handleSimplifiedDisplay.php, handleNewNote.php.

**[Step 1]** The web page is accessed by retrieving index.html. (e.g., by accessing [**http://i.cs.hku.hk/~[YourCSID]/lab4/index.html**](http://i.cs.hku.hk/~%5bYourCSID%5d/lab3/index.php)). index.html only contains static content. All the dynamic content is generated by calling JS functions in script.js file. In script.js file, when the web page is loaded for the first time, it needs to retrieve all the old notes from the database and set up event handlers for different elements. Please complete TODO task 1-5 in **script.js**.

**[Step 2]** When the web page is loaded for the first time, or when the the user inserts a new note to the database, or when the user finishes editing old note and returns to the initial web page, the old notes section in fig. 1 needs to be rendered to reflect the latest change in the database. This is accomplished by calling **loadSimplifiedDisplayOn()** function in script.js. Please complete TODO tasks 1-7 marked in the **loadSimplifiedDisplayOn()** function in **script.js.**

**[Step 3]** **loadSimplifiedDisplayOn()** function contacts **handleSimplifiedDisplay.php** to retrieve all the old notes encoded as a JSON string. The response JSON string sent by **handleSimplifiedDisplay.php** only contains simplified content as mentioned in the overview section of the handout. Please complete TODO tasks 1-7 marked in the **handleSimplifiedDisplay.php** file.

**[Step 4]** When the user fills in the title and content of a new note and clicks the save button under “Add New Note” of fig. 1, the **saveNewNote()** function is called to insert a new note to the database. The **saveNewNote()** function contacts **handleNewNote.php** file for the insertion and renders the web page under “Old Notes” to reflect the change. Please complete TODO tasks 1-3 marked in **saveNewNote()** in script.js.

**[Step 5]** The **handleNewNote.php** is responsible for inserting the new note to the database. Please complete TODO tasks 1-2 in **handleNewNote.php**.

**[Step 6]** When the user clicks the title of an old note under “Old Notes”, he is directed to the web page shown in fig. 5. The client side is handled by **getFullDisplay()** function in script.js. The **getFullDisplay()** function contacts **handleFullDisplay.php** file to retrieve the title and the complete content of the note and render the webpage shown in fig. 5. Please complete TODO tasks 1-6 marked in **getFullDisplay()** function in script.js. You don’t need to implement **handleFullDisplay.php** and we provide all the required code in **handleFullDisplay.php**.

**[Step 7]** In fig. 5, there are 2 buttons, **update** and **back**. By clicking the **update** button, the **updateNote()** function in **script.js** updates the title and content of the note in the database. By clicking the **back** button, **getBack()** function renders the web page shown in fig.1 again. Please complete TODO tasks 1-4 marked in **updateNote()** function in **script.js** file and complete TODO tasks 1-2 marked in **getBack()** function in script.js file.

**[Step 8]** In fig. 1, when you type in a search string under “Search Title”, a prefix-matching search is performed and only matching notes are shown under “Old Notes”. This is handled by the **search()** function in **script.js**. Please complete TODO tasks 1-3 marked in **search()** function in **script.js.**

**Note:** You are encouraged to upload your files to i.cs.hku.hk server for testing. Pleas ensure cookie is enabled on your browser used to test the page.

Submission

Please finish this lab exercise before 23:59 Monday Oct 17. Upload the following files to i.cs.hku.hk web server under **public\_html/lab4**:

**Index.html, script.js, handleFullDisplay.php, handleNewNote.php, handleSimplifiedDisplay.php, handleUpdate.php, jquery-3.1.1.min.js, style.css**

**Please make sure that** [**http://i.cs.hku.hk/~[YourCSID]/lab4/index.html**](http://i.cs.hku.hk/~%5bYourCSID%5d/lab3/index.php) **is accessible.**