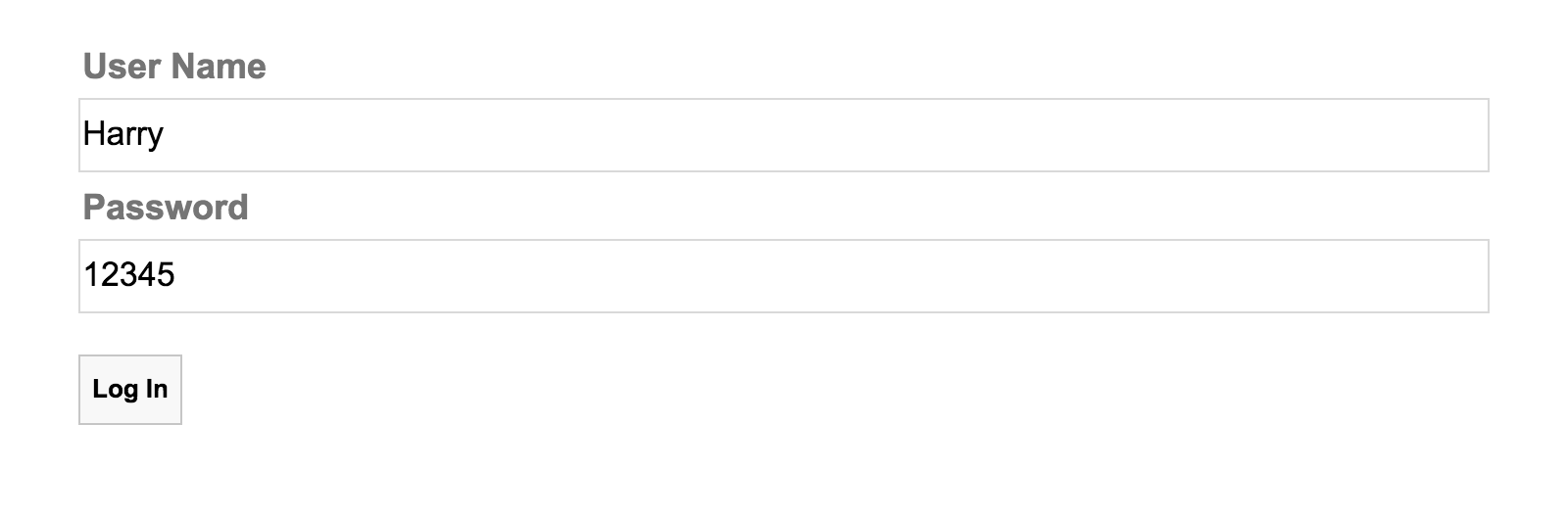
**COMP3322 Modern Technologies on World Wide Web**

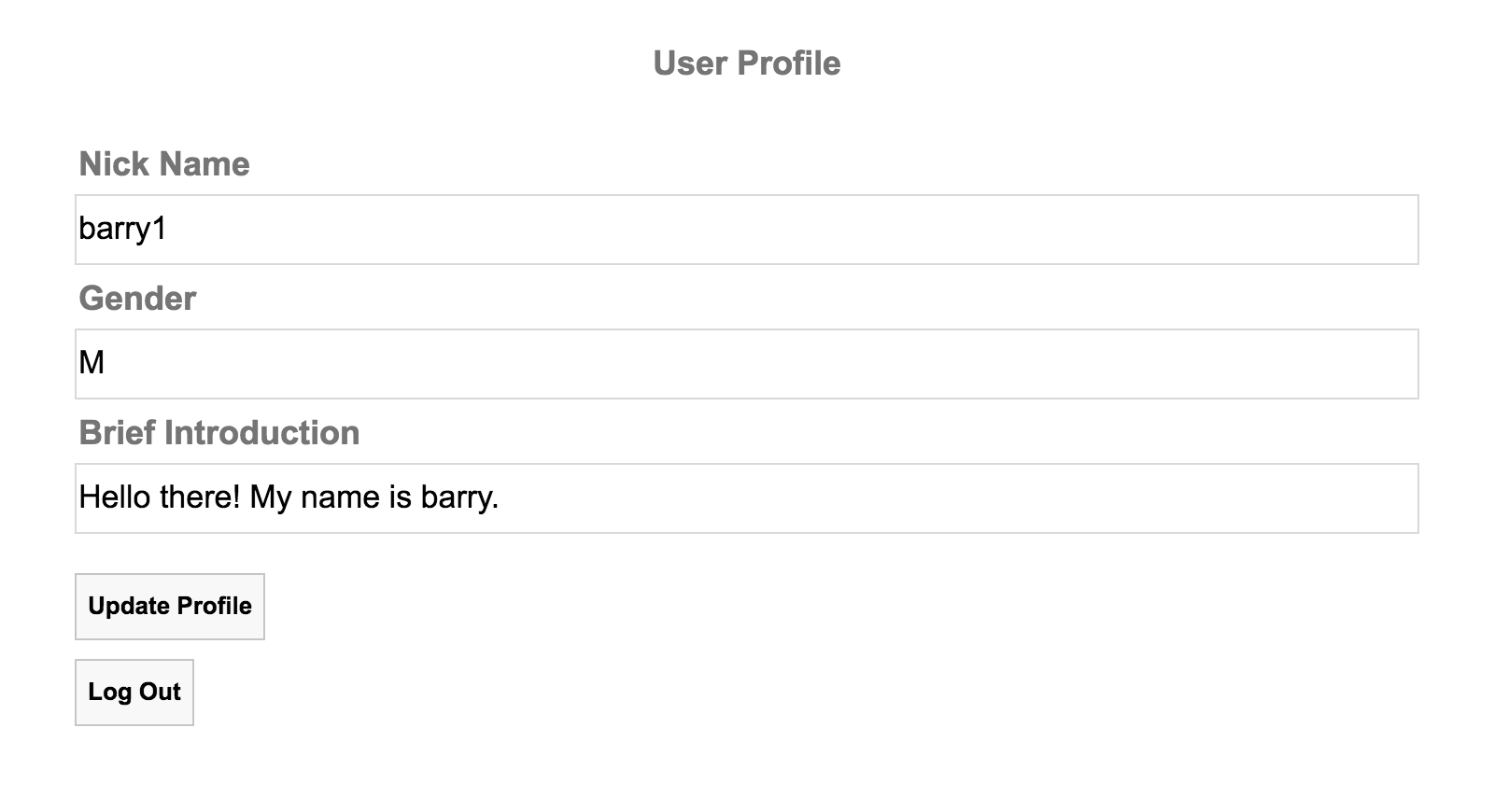
**Lab 3: PHP, JavaScript, Ajax, Database**

1. Introduction

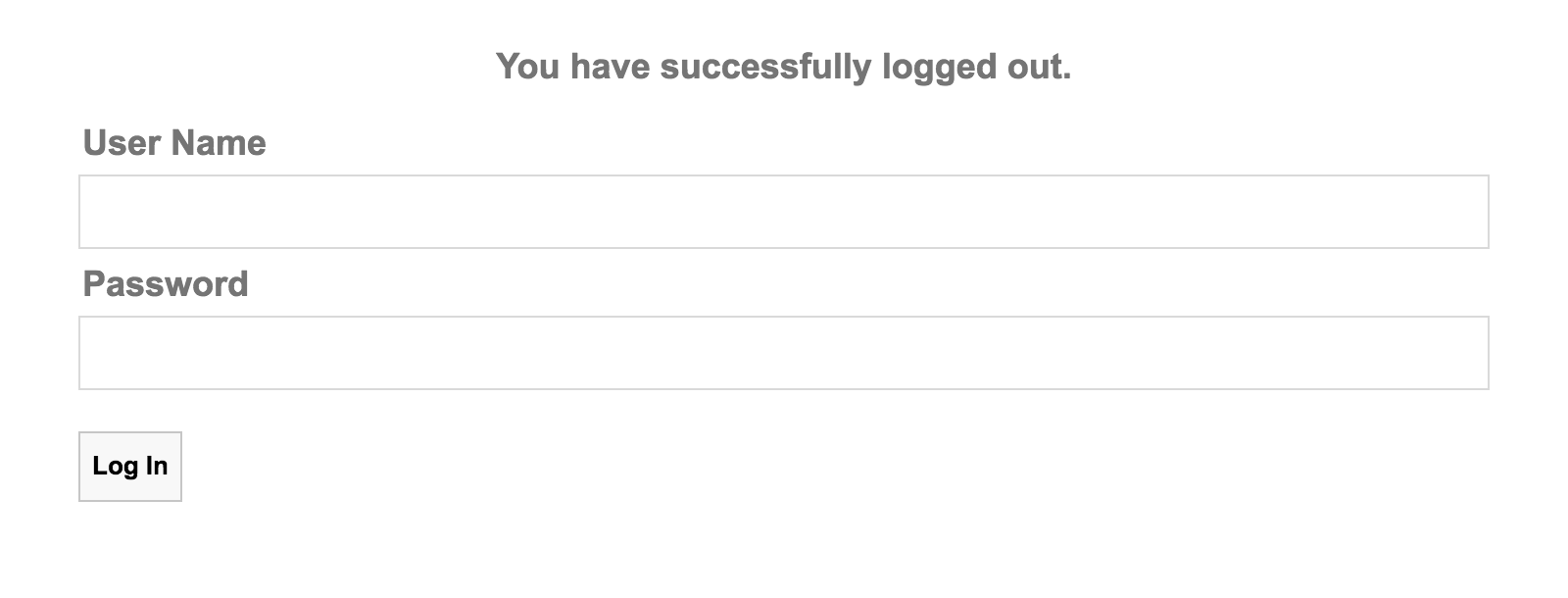
In this lab exercise, we will develop a dynamic webpage to change a user’s profile. When a user opens the web page, the user sees fig. 1 and he needs to enter proper user name and password to log in to the system. After successful logging in, the user sees fig. 2, where he can update his user profile and save the updated information to the backend database. After the user finishes updating his profile, he can log out the system. This leads the user to fig 3



**Fig. 1 Enter user name and password to log in.**

****

**Fig. 2 Update user profile.**

****

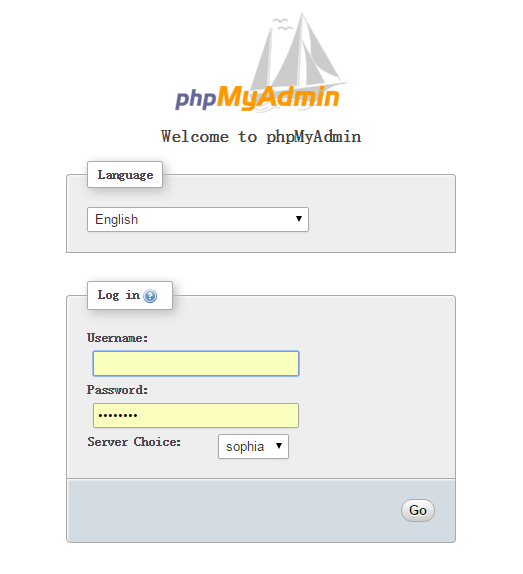
**Fig. 2 Log out.**

2. Set up the database

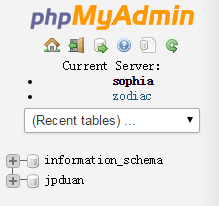
User profile is permanently saved on the database. So the first thing that we need to do is to set up the database for storing user profile.

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

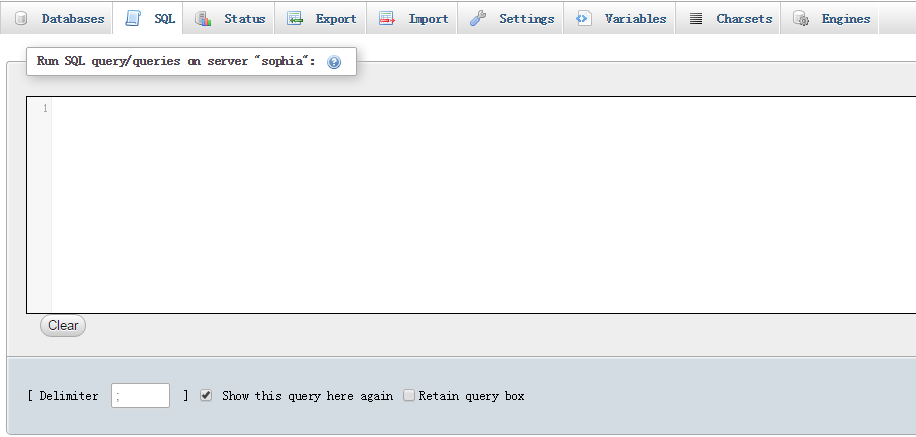
**2.2.** Go to https://i.cs.hku.hk/phpmyadmin/. Log in using your MySQL account. Choose the server **Sophia**. We will use the MySQL database hosted on sophia.cs.hku.hk.



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



**2.4.** Click the SQL tab and you will see an area for executing SQL code.



**2.5.** Create **users** table by executing the following SQL code. The **users** table contains the **userName** field and **password** field. The **users** table is used to handle log in.

CREATE TABLE users (

userName varchar(20) NOT NULL,

password varchar(20) NOT NULL,

PRIMARY KEY (userName)

)

INSERT INTO users(userName, password)

VALUES (

'Harry',

'12345'

);

INSERT INTO users(userName, password)

VALUES (

'Barry',

'23456'

);

**2.5.** Create **profiles** table by executing the following SQL code. The **userName** field is the primary key. It uniquely identifies a user. The **nickname**, **gender** and **briefIntro** fields are user’s profile that could be dynamically modified by the user.

CREATE TABLE profiles (

userName varchar(20) NOT NULL,

nickName varchar(20) NOT NULL,

gender varchar(1) NOT NULL,

briefIntro longtext,

PRIMARY KEY (userName)

)

INSERT INTO profiles(userName, nickName, gender, briefIntro)

VALUES (

'Harry',

'Rock Star',

'M',

'Hello there!'

);

INSERT INTO profiles(userName, nickName, gender, briefIntro)

VALUES (

'Barry',

'Chosen 2',

'M',

'Hello there!'

);

**3. Complete the lab material**

Download lab3 material from Moodle and implement all the required functionality.

**3.1.** We enter the web site using the following address “http://xxxx/index.html”. The index.html file is the entry point of the web site. Open index.html, we can see that when index.html is loaded by the browser, it calls **start()** function to render the entire webpage.

The **start()** function is defined in script.js file. It sends an HTTP GET request to **handleLogin.php** file, and replaces the inner HTML text of **<div id=”content”>** element in index.html file with the HTTP response.

**Todo:** Please complete the **start()** function in script.js.

**3.2** The handleLogin.php should check whether the user has logged in before, by checking the cookie variable contained in the HTTP request. If the cookie variable is set, handleLogin.php should retrieve the profile for the user from the database and returns HTML text to render fig. 2.

If the HTTP request does not contain a cookie variable, then the user has not logged in before. The handleLogin.php returns HTML text to render fig. 1.

When the user sees fig. 1, he tries to enter correct user name and password to log in. This will generate an HTTP request with GET method. The GET method contains both the user name and password input by the user.

handleLogin.php should check whether the user enters valid user name and password. If user name and password are accepted after comparing the data in the database, handleLogin.php sets up the cookie variable, retrieves the current user profile information and responds an HTML text to render fig 2.

If user name and password are invalid, handleLogin.php responds an HTML text for rendering fig 1.

**TODO**: Implement all the missing part of handleLogin.php. There are totally 12 TODO items in handleLogin.php

**3.3**. In fig. 1, when the user clicks the log in button, an HTTP GET method will be generated and sent to handleLogin.php. The get method should contain the input user name and password as parameters.

**TODO**: Please implement the login() function in script.js file.

**3.4**. In fig. 2, when the user clicks the update profile button, an HTTP GET method will be generated and sent to handleUpdate.php. The get method contains user name, and the updated nick name, gender and brief introduction as parameters.

**TODO**: Please implement the client-side updateProfile() function in script.js file.

**3.5**. On the server side, when handleUpdate.php receives the HTTP GET method, it should update the profiles table with the updated values contained in the GET method.

**TODO**: Implement the missing part of handleUpdate.php file.

**3.6** In fig. 2, when the user clicks the log out button, an HTTP GET method will be generated and sent to handleLogout.php.

**TODO**: Please implement the logout() function in script.js file.

3.7 On the server side, when handleLogout.php receives the HTTP GET method, it resets the cookie associated with the user. Then it should returns an HTTP response text to render a log in web page.

TODO: Please finish the missing part of handleLogout.php file.

Submission

Please finish this lab exercise before 23:59 Wednesday November 4. Upload the following files to i.cs.hku.hk web server under **public\_html/lab4**:

index.html, queryLectures.php, updateLectureState.php, jquery-1.11.3.min.js, style.css, yes.png, no.png,

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