Assignment 1
Web Application Development

Due date: 9:00pm Friday 15 April 2016 -- Worth 12%

[Late submission will be penalized, 10% per working day for at most 5 working days]

1 Assignment Description

This is an individual assignment. Students are referred to the University's policy on plagiarism. The aim of this assignment is to develop a better understanding of building web applications using **embedded PHP** and **MySQL** only.

2 Assignment Tasks

The assignment is to develop a web-based taxi booking system called *CabsOnline*. CabsOnline allows passengers to book taxi services from any of their internet connected computing devices. Four components (registration, login, booking and admin) of such an online service that must be completed for this assignment are specified in the following three sub-sections. Other components such as payment processing, detailed processing for assigning taxi, are not required in this assignment but you are free to extend for your fun later.

2.1 Registration and Login (register.php / login.php)

These two components are used to allow a new customer to register, or an existing customer to login to the system so as to make a booking. The system maintains a customer table, recording *customer name*, *password*, *email address* (used to identify the passenger), and *contact phone number*. For a new customer, the above information is required to get into the system (as shown in Figure 1). For an existing customer, email address and password are required to login to the system (as shown in Figure 2). The specific functions include

1) Design and create a MySQL table for storing information of all customers. In this *customer* table, you need to store the email address (as primary key), customer name, password, and contact phone number for each customer.
2) For a new customer, the system will check (a) all inputs including customer name, password, re-typed password (for double checking), email address, and phone number are given, (b) the password is the same as the re-typed one, and (c) the email address is unique (i.e., has not been used by other customers in the customer table). If there is a problem, the corresponding error message will be displayed; otherwise, the system will record the inputted information as a new row in the customer table. Finally, the booking page will show up for taking the booking request. Necessary information about the customer (i.e., email address) is needed to carry forward to the booking page.
3) For an existing customer, the email address and password are expected and checked against the customer table. If a customer is matched, the booking page will show up and email address of the customer will be carried to the booking page; otherwise, login failure message will be displayed.

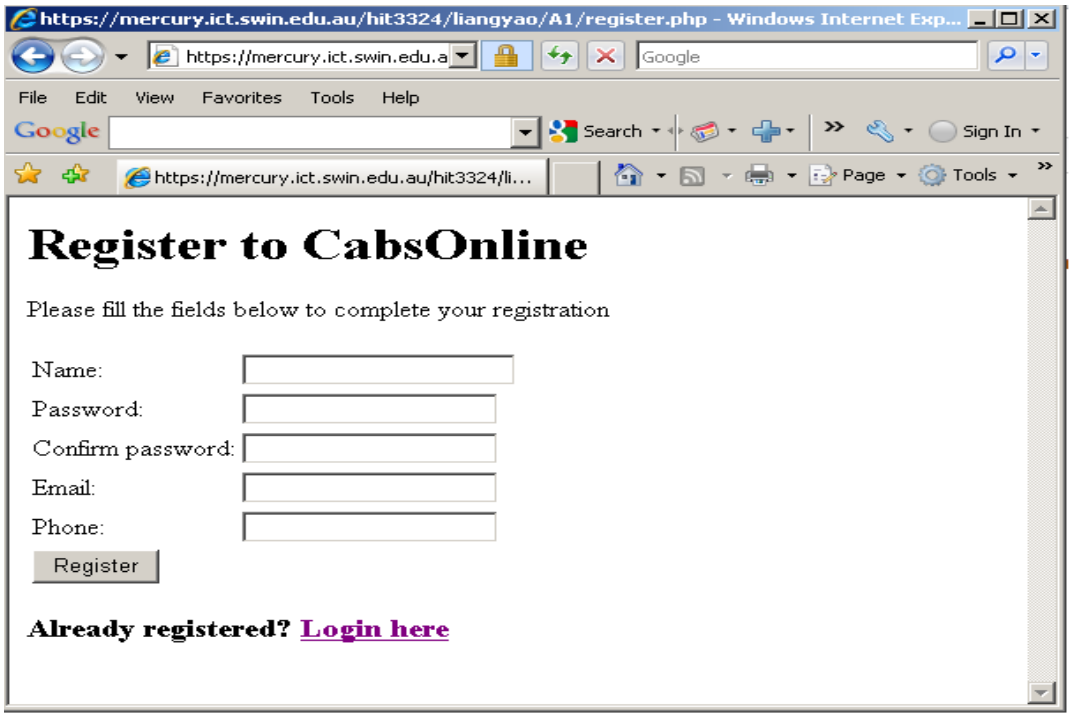

Figure 1: Registration for New Customers

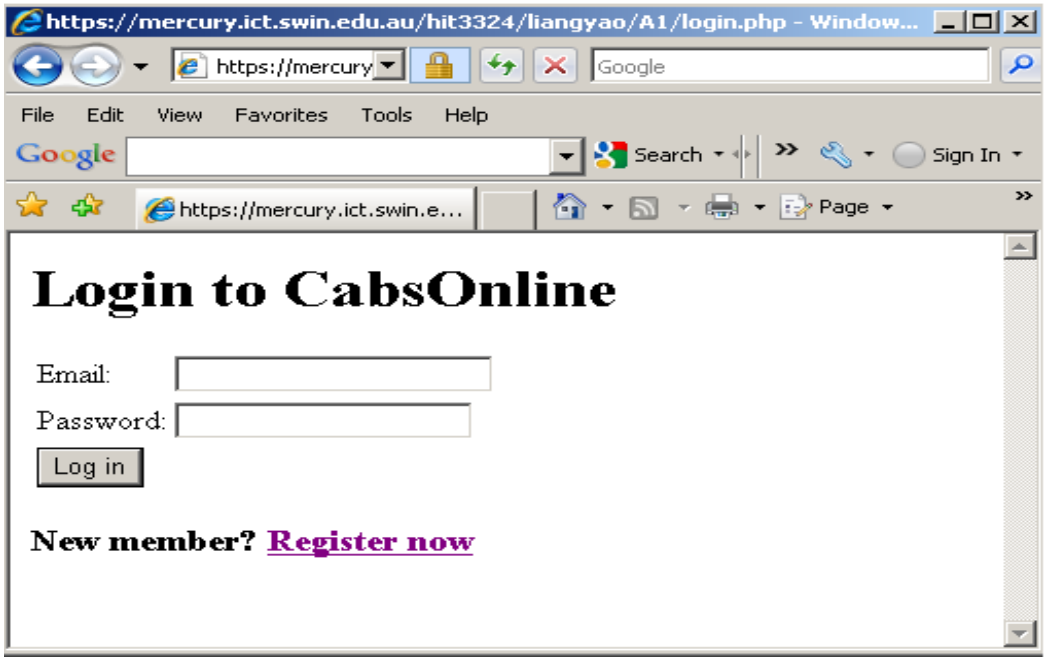

Figure 2: Login for Existing Customers

2.2 Booking page (booking.php)

After successful login of a customer, this component is used to allow the customer to put in a taxi booking request in Melbourne and surrounding areas. As shown in Figure 3, the inputs for such a request include *passenger name* (Note, this may not be the same as the

customer name), *contact phone of the passenger*, *pick-up address* (including *unit number* if applicable, *street number*, *street name*, and *suburb*), *destination suburb* (Note, no need to give the detailed address for destination), and *pick-up date/time*. Some other details such as number of passengers, car type, etc. may not be required for this assignment. Once you get these inputs, you also need to generate a **unique** *booking reference number*, *booking date/time* (i.e., the current date/time when the booking is made) and *status* with initial value "unassigned" for the request, add the request as a row in a new MySQL table called the booking table. The specific functions include

1) Design and create another MySQL table for storing all booking requests. In this *booking* table, you need to store the generated booking number (as the primary key), the *email address* of the customer (as a foreign key that references the primary key in the customer table), the customer's inputs including passenger name, contact phone of the passenger, pick-up address (Note, each component of the address must be stored as a field in the table and NULL value may be taken for unit number if not applicable), destination suburb, and pick-up date/time, the generated booking date/time, and the generated status.
2) Design a form to take inputs for a booking request. All input items except unit number must be provided and the pick-up date/time must be at least 1 hour after the current date/time; otherwise, an error message will be displayed.
3) For the booking request, the system will generate a unique booking reference number, booking date/time and a status with initial value "unassigned", and add them together with customer's inputs into the booking table. Upon completion, the system will display the confirmation information "Thank you! Your booking reference number is <booking_reference_number>. We will pick up the passengers in front of your provided address at <pickup_time> on <pickup_date>."

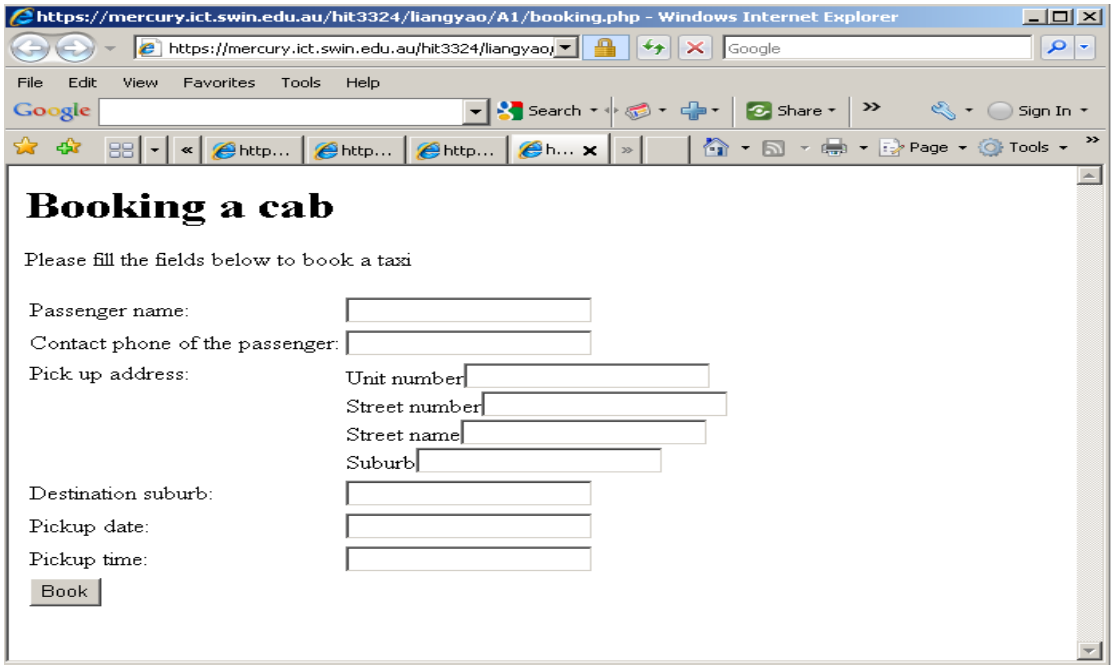

Figure 3: Issuing a Booking Request

2.3 Admin page (admin.php)

This component allows administrative people of *CabsOnline* to view those taxi booking requests that need to be assigned as soon as possible and to assign taxi for a particular booking request. Note authentication is not required though it is necessary in the real application. If you provide this function, you must provide the password and explain it in the *readme* document. The specific functions of this component include

1) The system takes two types of requests. The first is to search those booking requests with the pick-up time within 2 hours from now and is realized by a single button *List All* to show pick-up requests. The second is to assign taxi for a particular booking request and is implemented by a text box allowing input for a specific booking reference number and followed by a button *update* to assign taxi. Figure 4 shows the interfaces for these two functions.
2) For a request for showing pick-up requests, the system executes a query on the MySQL database for finding those "unassigned" booking requests with a pick-up time *within 2 hours* from now only. A table is used to display all the found requests, with each request displayed as a row as shown in Figure 4. Each row contains the booking reference number, customer name, passenger name, passenger contact phone, pick-up address, destination suburb, and pick-up date/time. As shown in Figure 4, a pick-up address is formed as a *single string* with "/" separating unit and street numbers if applicable, ", " separating street name and suburb, and only a space, i.e., " ", separating street number and street name.
3) For a request for assigning taxi, make an update to the MySQL database to change the status of the booking request that matches the given booking reference number from "unassigned" to "assigned", and display confirmation information "The booking request <bookingRefNumber> has been properly assigned". If no unassigned booking request is matched for update, an error message will be displayed.

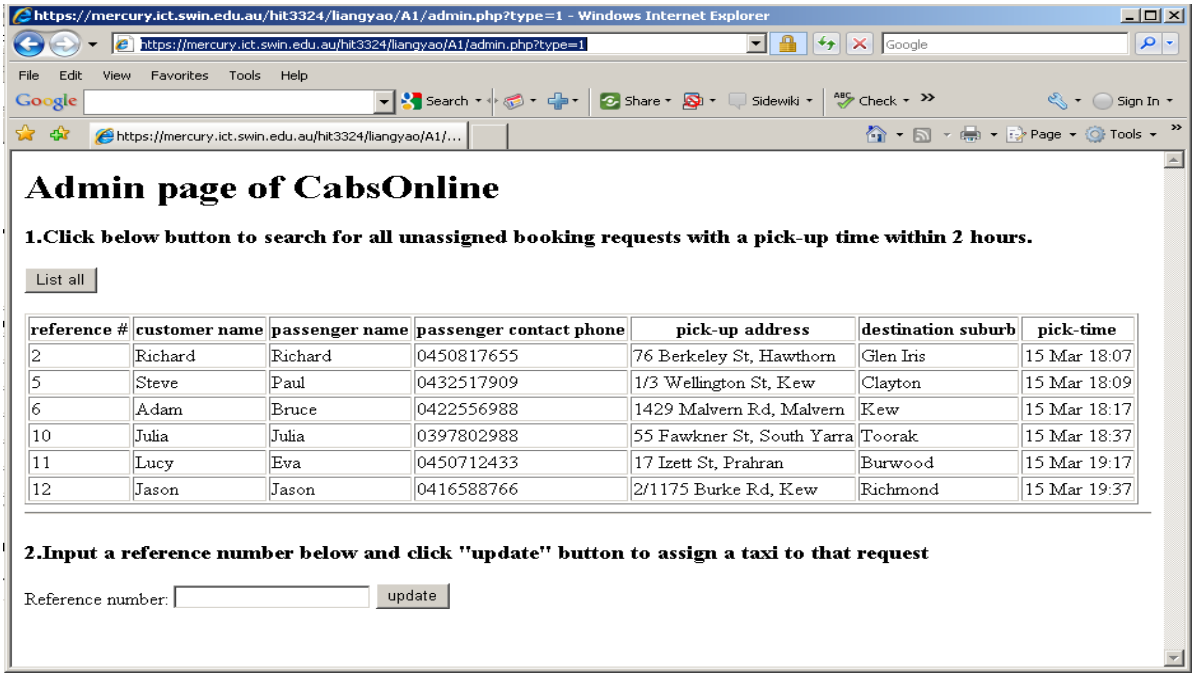

Figure 4: CabsOnline Administration

3 Submission Requirements

You should ensure that all files used for the assignment sit in a directory called "Assignment1" (use this name exactly, it is case sensitive and no space between "Assignment" and "1") within your Mercury account. The directory should contain no other sub-directories (i.e., all files are placed directly under the "Assignment1" directory).

All files required by the assignment description must be submitted as one single ZIP file by **using the electronic submission system (ESP)**. The files should include:

- four PHP files: register.php, login.php, booking.php and admin.php
- any additional PHP files that you use;
- a text file that includes the MySQL commands that you used to create the database and the two tables;
- a file readme.doc that includes
    - a list of all the files in the system;
    - brief instructions on how to use the system.

For each submitted file, we require the minimum comments including student information and the main function for the file.

The MySQL database that you use should be constructed in your Mercury account. **After submission, you are not allowed to change any of the submitted files in the Assignment1 directory on your Mercury account; time stamps will be checked.**

Assignments that fail to follow "submission requirements" will <u>NOT</u> be assessed.

If you use your PC/laptop for the assignment, you must make sure your completed assignment is loaded under your mercury account and works fine. We strongly recommend that you give sufficient time to do so!

4 Marking Scheme

Work will be assessed based on the quality and presentation. The assignment will be marked out of 48 and will contribute 12% towards assessment of the unit.

Assessment item	Marks
Minimum comment; readme.doc and quality of code	3
2.1.1 create customer table	2
2.1.2 user interface; input normal checking; check email uniqueness; save to database; redirect to booking; carry forward email	11
2.1.3 user interface; check email/password; redirect to booking; carry forward email	5
2.2.1 create booking table	3
2.2.2 user interface; check pick-up date/time	4

2.2.3 generate booking number; booking date/time; initialize status; save to database; display confirmation info	6
2.3.1 user interface	2
2.3.2 retrieve booking requests; process address; display result table	8
2.3.3 update database; display info	4
Total	48