Assignment 1
Web Application Development

Due date: 9:00pm Friday 11 September 2015 -- 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 Faculty's policy on plagiarism. The aim of this assignment is to develop a better understanding of building web applications using embedded PHP and MySQL.

2 Assignment Tasks

The assignment is to develop a web-based shipping system called *ShipOnline*. ShipOnline provides an online service for delivering small items for customers from Melbourne to elsewhere in Australia. A site map and three components (customer registration, login/request, and administration) of such an online service that must be completed for this assignment are specified in the following four sub-sections. Other components such as pick-up, arrangement for real shipping and payment are not required in this assignment but you are free to extend for your fun later.

2.1 Site Map

Design a home page for ShipOnline (*shiponline.php*) which provides links to the three components (see Figure 1).

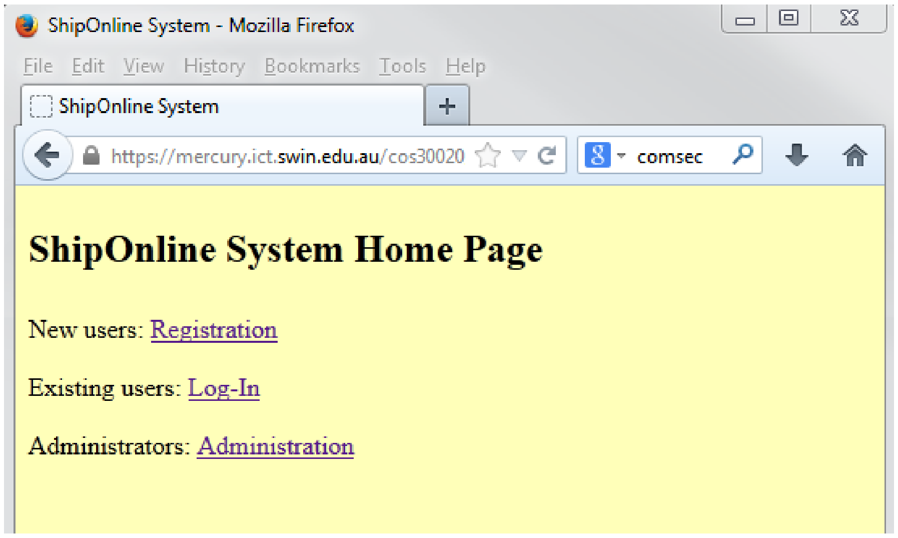
2.2 Registration

This component is designed to manage customer information and to allow a new customer to register into the system before using the system. The system maintains a *customer* table. For each customer, *name*, *password*, *email address* (used to identify the customer), and *contact phone number* are required, and a *customer number* will be generated for the customer. The specific functions of this component include

1) Design and create a MySQL table for storing information of all customers. In this customer table you need to store the generated customer number (as the primary key), name, password, email address, and contact phone number for each customer.

2) Implement the registration function (*register.php*). A simple user interface needs to be designed to take all inputs for each new customer, including name, password, re-typed password (for double checking), email address, and phone number (see Figure 2). After the *register* button is pressed, the system will check (a) all inputs are given, (b) the password is the same as the re-typed one, and (c) the email address is unique (this needs to be checked against existing customers in the customer table). If there is a problem, the corresponding error message will be displayed; otherwise, the system will (i) generate a customer number; (ii) store the generated customer number together with the inputted information as a new row in the customer table; and (iii) show a confirmation message "Dear <name>, you are successfully registered into ShipOnline, and your customer

number is <customerNumber>, which will be used to get into the system." under the *register* button on the user interface.


ShipOnline System - Mozilla Firefox
File Edit View History Bookmarks Tools Help
ShipOnline System
https://mercury.ict.swin.edu.au/cos30020
comsec

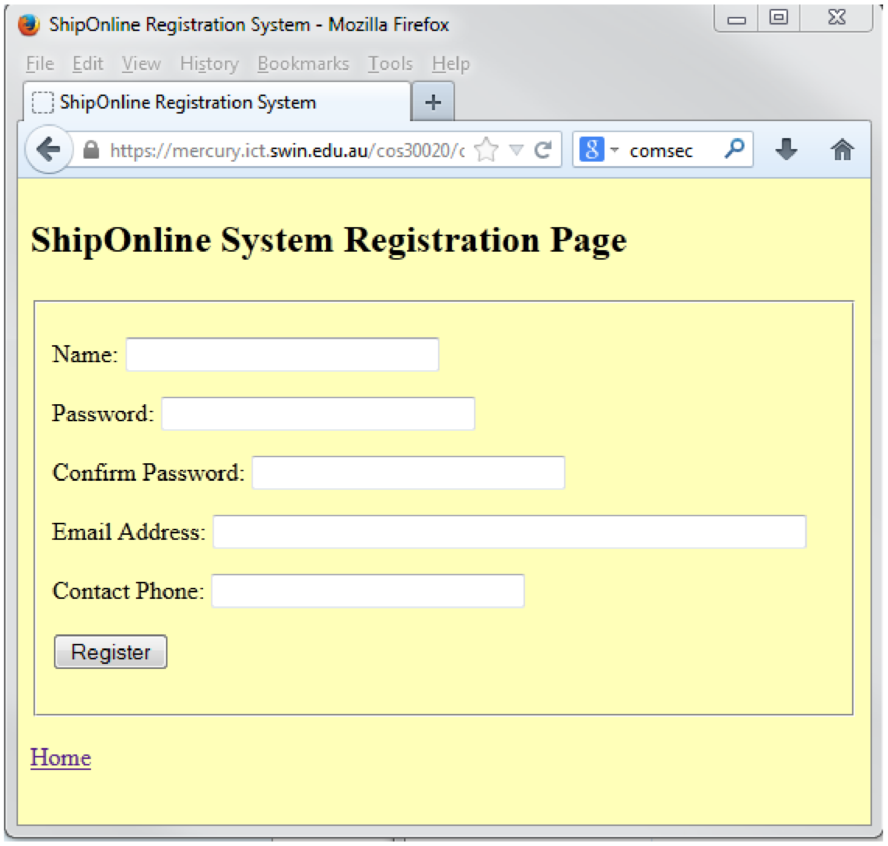
ShipOnline System Home Page

New users: Registration

Existing users: Log-In

Administrators: Administration

Figure 1: Home Page of ShipOnline


ShipOnline Registration System - Mozilla Firefox
File Edit View History Bookmarks Tools Help
ShipOnline Registration System
https://mercury.ict.swin.edu.au/cos30020/c
comsec

ShipOnline System Registration Page

Name:

Password:

Confirm Password:

Email Address:

Contact Phone:

Register

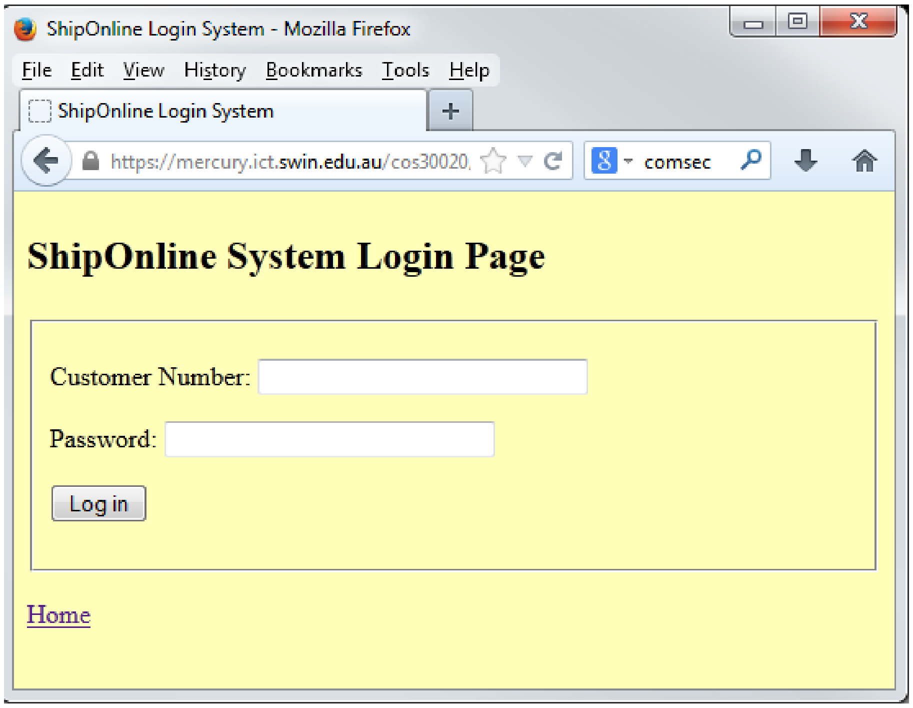
Home

Figure 2: Registration Page of ShipOnline

2.3 Login and Request

This component is designed to allow a registered customer to login and lodge a shipping request. The system needs to maintain another table - *request* table. For each request, you need to store the *customer number*, the generated *request number* and *request date*, the customer's inputs including *item description*, *weight*, *pick-up address* and *suburb*, preferred *pick-up date and time*, *receiver name*, and *delivery address*, *suburb* and *state*. The specific functions of this component include

1) Design and create a MySQL table for storing information of all requests. For each request, one row is created to include the *customer number*, the generated *request number*, the generated *request date*, *item description*, *weight*, *pick-up address* and *suburb*, preferred *pick-up date and time*, *receiver name*, and *delivery address*, *suburb* and *state*. Either the generated *request number* or the combined *customer number* and the generated *request number* can serve as the *primary key*, and the *customer number* is a *foreign key* that references the customer table.
2) Implement the login function (*login.php*). As shown in Figure 3, a simple user interface needs to be designed to take *customer number* and *password*. After the *login* button is pressed, the login information will be checked with the *customer* table. If either the customer number or the password is incorrect, display an error message under the *login* button; otherwise, redirect to request page (*request.php*) with the *customer number* as the parameter.


ShipOnline Login System - Mozilla Firefox
File Edit View History Bookmarks Tools Help
ShipOnline Login System
https://mercury.ict.swin.edu.au/cos30020,
comsec

ShipOnline System Login Page

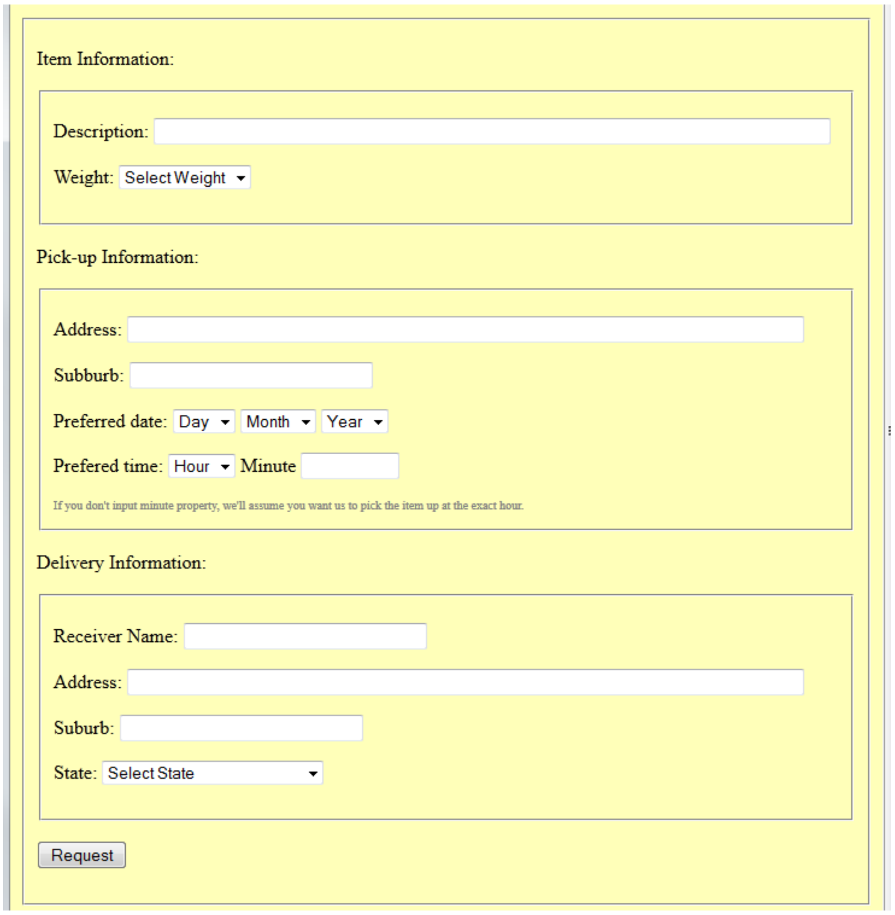
Customer Number:

Password:

Log in

Home

Figure 3: Login Page of ShipOnline



Item Information:

Description:

Weight: Select Weight

Pick-up Information:

Address:

Subburb:

Preferred date: Day Month Year

Prefered time: Hour Minute

If you don't input minute property, we'll assume you want us to pick the item up at the exact hour.

Delivery Information:

Receiver Name:

Address:

Suburb:

State: Select State

Request

Figure 4: Request Page of ShipOnline

3) Implement the request function (*request.php*). As shown in Figure 4, a simple user interface needs to be designed to take the customer's inputs for a request, including item information (*item description, weight*), pick-up information (*pick-up address* and *suburb,* preferred *pick-up date and time*) and delivery information (*receiver name*, and *delivery address, suburb* and *state*). You need to use a drop-down list to take the input for *state*. Pricing information must be displayed on this page. For comprehensive pricing information, you may refer to http://auspost.com.au/apps/domestic-parcel.html. However, to make it simple and easy to calculate, pricing for *ShipOnline* could just be "$10 for 0-2 kg and $2 for each additional kg". After the request button is pressed, the system will check (a) all inputs are given, (b) the *weight* is an integer number between 2 and 20 kg (you don't need to validate this if you choose to use a drop-down list with values 2 - 20), (c) the preferred *pick-up date* and *time* are at least 24 hours after the current time, and (d) the preferred *pick-up time* should be between 7:30 and 20:30. If there is a problem, the corresponding error message will be displayed under the *request* button; otherwise, the system will generate a *request number* and *request date* for the request, add them together with the forwarded customer number and the customer's inputs for the request in the *request* table, calculate the cost (Don't store the cost which can be calculated based on the weight), and display "Thank

you! Your request number is <request_number>. The cost is <cost>. We will pick-up the item at <pickupTime> on <pickupDate>." under the *request* button.

4) In addition, the system will also find the customer *name* and *email address* from the *customer* table and send a confirmation email to the customer with the following information;
   Recipient: the provided <email_address>
   Subject: "shipping request with ShipOnline"
   Message: "Dear <name>, Thank you for using ShipOnline! Your request number is <request_number>. The cost is <cost>. We will pick-up the item at <pickupTime> on <pickupDate>."
   When sending email from php using the mail() function, you should specify an envelope sender who will receive the bounce messages as shown below:
   mail($to, $subject, $message, $headers, "-r 1234567@student.swin.edu.au");

2.3 Administration

This component allows you as the owner of *ShipOnline* to view requests on a particular *request date* or *pick-up date* so as to have an idea of the workload/revenue and to make necessary arrangement. Note, authentication is not required though it would be necessary in a real application. The specific functions of this component include

1) As shown in Figure 5, design the user interface (*admin.php*) with a group of two radio buttons for either a *request date* or a *pick-up date* and an input field for a particular date as input. Once the *show* button is pressed, the system will react as follows depending on the type of the request.

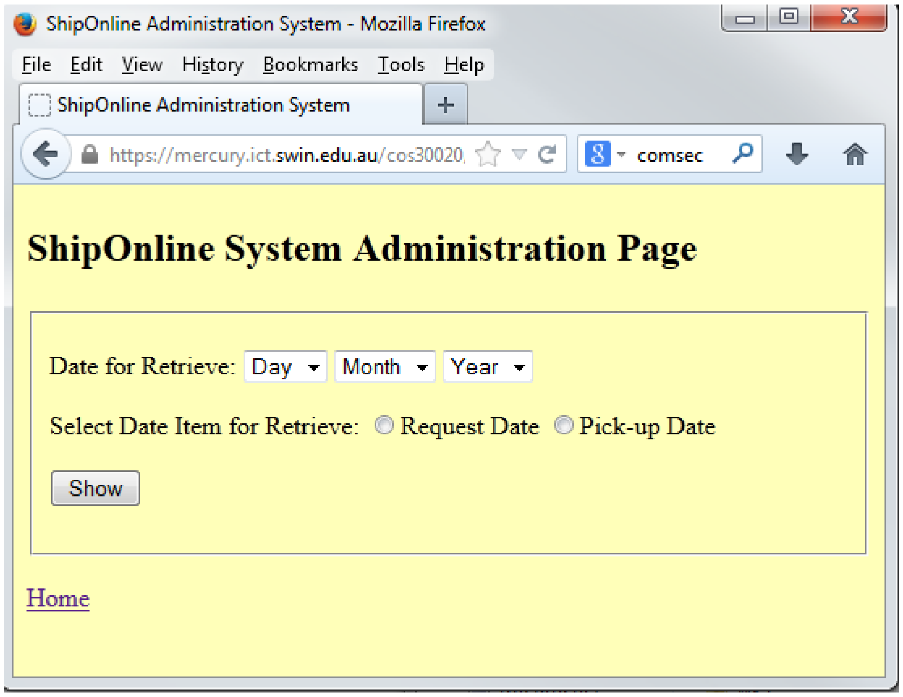

ShipOnline Administration System - Mozilla Firefox
File Edit View History Bookmarks Tools Help
ShipOnline Administration System
https://mercury.ict.swin.edu.au/cos30020
comsec
ShipOnline System Administration Page
Date for Retrieve: Day Month Year
Select Date Item for Retrieve: Request Date Pick-up Date
Show
Home

Figure 5: Admin Page of ShipOnline

2) If a *request date* is given, the system will find all the requests on the *request date* in the *request* table in the MySQL database and show them in an html table under the *show* button. For each returned request, select the *customer number*, *request number*, *item description, weight, pick-up suburb,* preferred *pick-up date*, *delivery suburb* and *state*. Under the html table, also calculate and show *the total number of requests* on the given *request date*, and *the total revenue* calculated from the total costs of all requests on the given *request date*.
3) If a *pick-up date* is given, the system will find all the requests on the *pick-up date* in the MySQL database (*both the request and the customer tables*) and show them in an html table under the *show* button. For each returned request, we need to show the *customer number*, *customer name* (from the *customer* table), *contact phone* (from the *customer* table), *request number, item description, weight, pick-up address* and *suburb,* preferred *pick-up time*, *delivery suburb* and *state*. The requests in the html table are sorted by the pick-up suburb, then the delivery state, then the delivery suburb. Under the html table, also calculate and show *the total number of requests* on the given *pick-up date,* and *the total weight* of all requests on the given *pick-up date*.

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 files and 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:

- five PHP files: shiponline.php, register.php, login.php, request.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 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 tables should be created 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.

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!

Assignments that fail to follow "submission requirements" will NOT be assessed.

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 home page	2
2.2.1 create customer table	2
2.2.2 user interface; input normal checking; check email uniqueness; generate customer number; insert to table; display message	8
2.3.1 create request table	2
2.3.2 user interface; check customer number/password; redirect to request	5
2.3.3 user interface; input checking, generate info, insert to table, calculate cost, display info	8
2.3.4 find additional info, generate and send email	3
2.4.1 user interface	2
2.4.2 retrieve requests; show the table; total number of requests, total revenue	7
2.4.3 retrieve requests (both tables), sort, show the table, aggregated info	6
Total	48