COIT12207 Assignment 2 Specification

Problem

A local computer repair business has come to you for help. They want to create a website they can use to allow customers to log jobs online. They want a system in which they can allow customers to upload their job details without having to come in or email. Anyone can log a job, but only the business personnel should be able to see logged jobs. Business personnel may also need to update and delete jobs as appropriate.

Requirements

Your implementation should be developed on your computer using XAMPP.

You are required to implement a PHP Web Based front end system. This system will connect to the database back end which stores all the job details. The SQL script to create all required tables is supplied. You must run this script from within phpMyAdmin to create the required databases and users. When connecting to the database, you should use 'webauth' as the username and 'webauth' as the password. You set up this user in the Week 6 Hands-on Project.

Since the business doesn't want just anyone to be able to see a list of jobs, this must be protected with authentication. Additionally, the ability to edit, and delete must also be protected with authentication. Currently, there will be a single person managing everything and therefore, only one user account will be required. The username and password is already in the database. They are:

Username	Password
sam	password

Everyone will initially connect to the 'home.php' page.

The 'home.php' page should check if the user is already logged in. If the user is not logged in, the only link will be to add a new job. If the user is logged in, they should be shown the options to view all jobs or add a new job. When the logged in user chooses the option to view all jobs, they should be taken to another page which displays a list of all jobs. There should be an option to edit or delete a selected job.

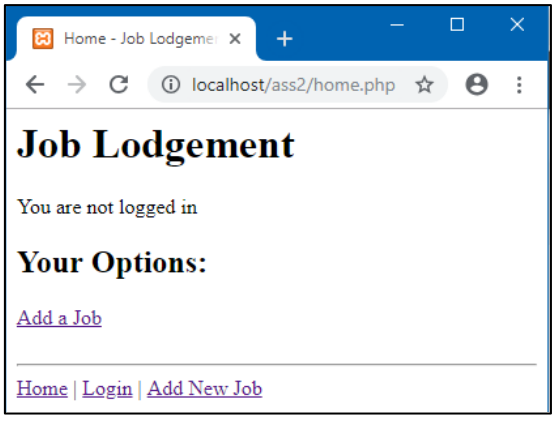

Figure 1 - Home Page (Not Logged In)

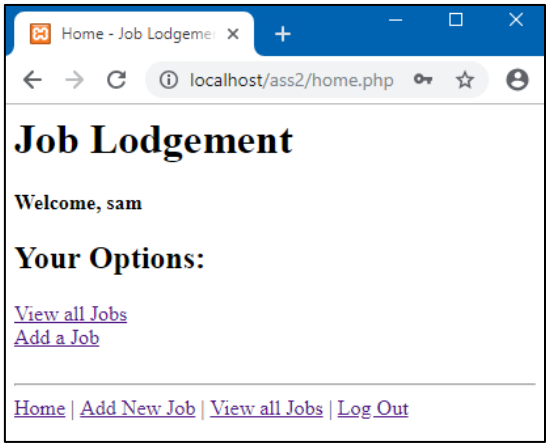

Figure 2 - Home Page (Logged In)

The footer for people who are not logged in should contain three links: a link to home, a link to login, and a link to add a new job.

The footer for people who are logged in should show four links: a home page link, a link to add a new job, a link to view all jobs, and a link to logout.

The login link should take users to the page login.php. The login page should display a form for users to login. If a user accesses login.php when they are already logged in, they should be redirected to home.php. When a user successfully logs in, they should be redirected to home.php.

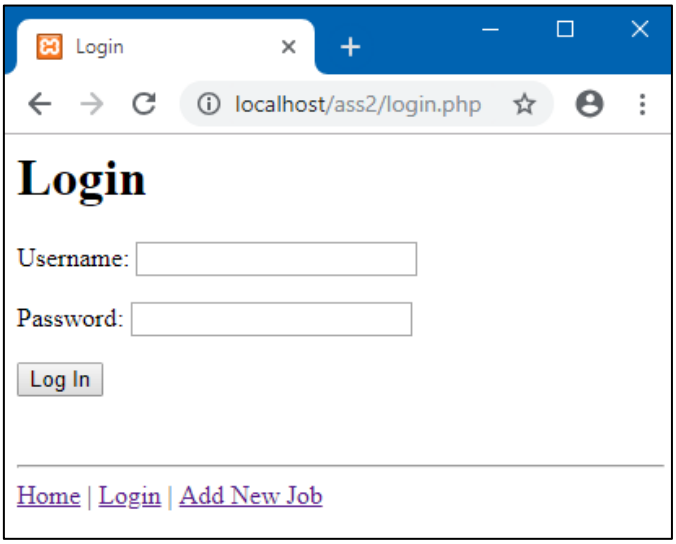

Figure 3 - Login Form

If a user is logged in when they are on the home page, at the top of this page, the message "Welcome, *current_user*" should be shown, with *current_user* replaced with the username of the currently logged in user (this username should not be hardcoded).

On the view all jobs page, each job in the list of jobs should have an edit and delete button. The list should be presented in a table with the columns: Customer Name, Customer Email, Severity,

Description. Two unnamed columns should be included at the end to store the Edit and Delete buttons for each job.

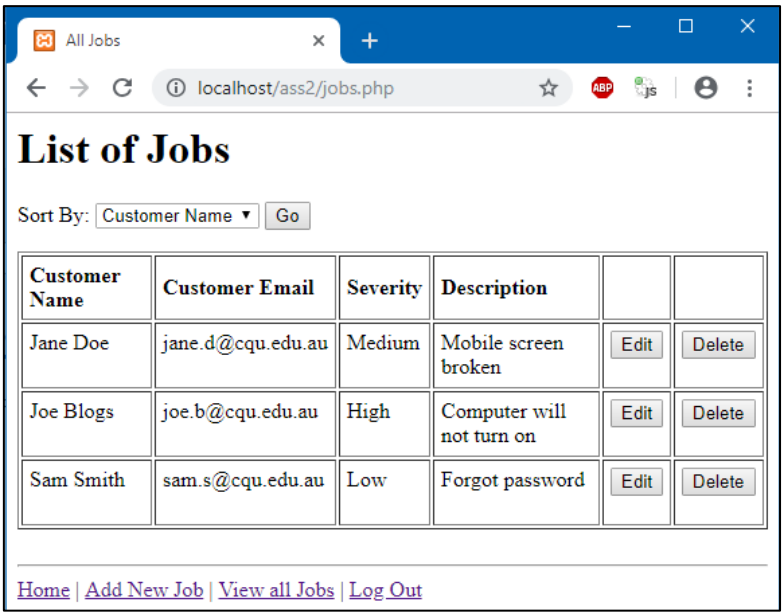

Figure 4 – View all Jobs Page

There should be two sort options a user can select. These are sort by severity (high to low) and sort by customer name (alphabetically A-Z). Default sorting should be by customer name. Although not specifically covered in the Hands-on Projects, you have the requisite knowledge required to achieve this function.

Severity types include High, Medium and Low. However, these are stored as an int value in the database (i.e. 1 maps to High, 2 maps to Medium, and 3 maps to low). Anywhere you display the severity, you should display the text and not the number. Marks will be deducted if the number appears in place of the text.

When a logged in user clicks the edit option for a job, they should be taken to the edit.php page. On this page, there should be a prepopulated form of input elements which a user can use to change details about the selected job. There should be two buttons: Update and Cancel. The Update button will save the new values to the database and display a message informing the user of a successful update. The Cancel button should return the user to the list of all jobs. Problem description should use a textarea for input.

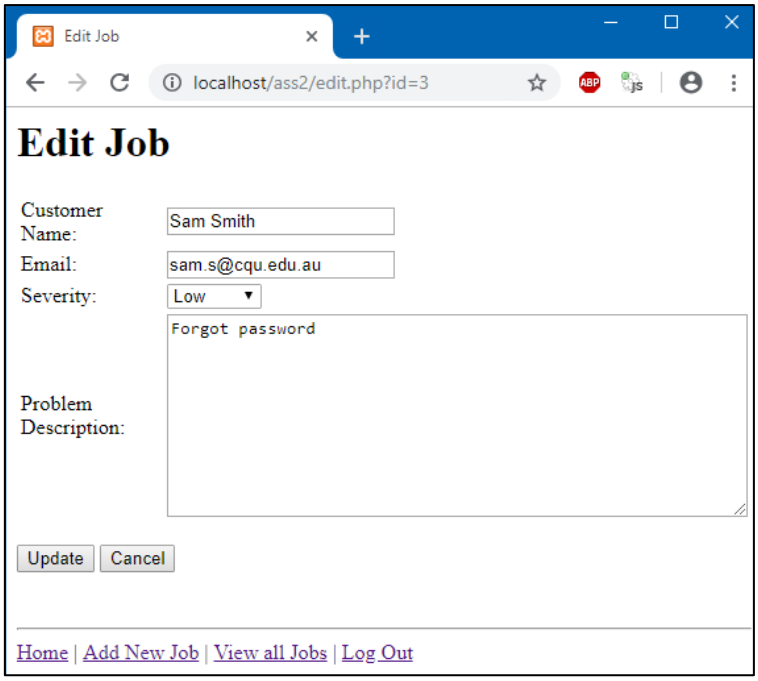

Figure 5 - Edit Page

When a user clicks the delete option for a job, they should be taken to the delete.php page. On this page, there should be a listing of the information about the selected job. There should be two buttons: Delete and Cancel. Delete will remove the job from the database and inform the user of a successful deletion. The Cancel button should return the user to the list of all jobs.

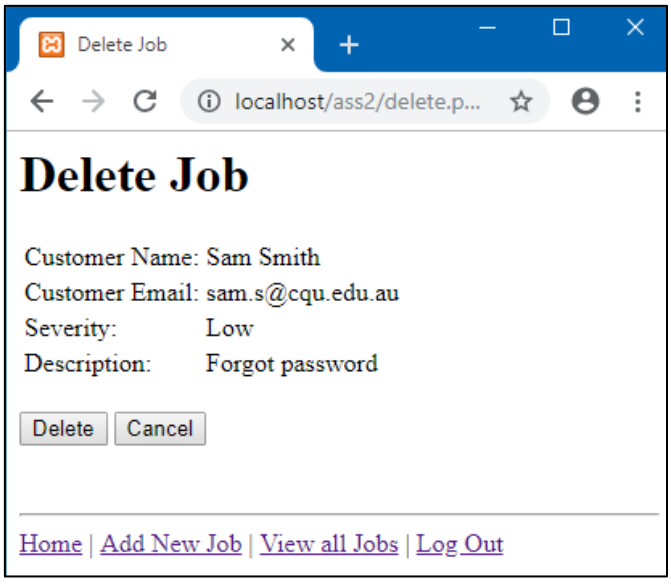

Figure 6 - Delete Page

When a user clicks the Add New Job link, they should be taken to the add.php page. This page should have several input elements corresponding to the database fields for job information. There should be two buttons at the bottom: Add and Cancel. The Add button will save the information about the job to a new record in the database. The Cancel button should return the user to the home page.

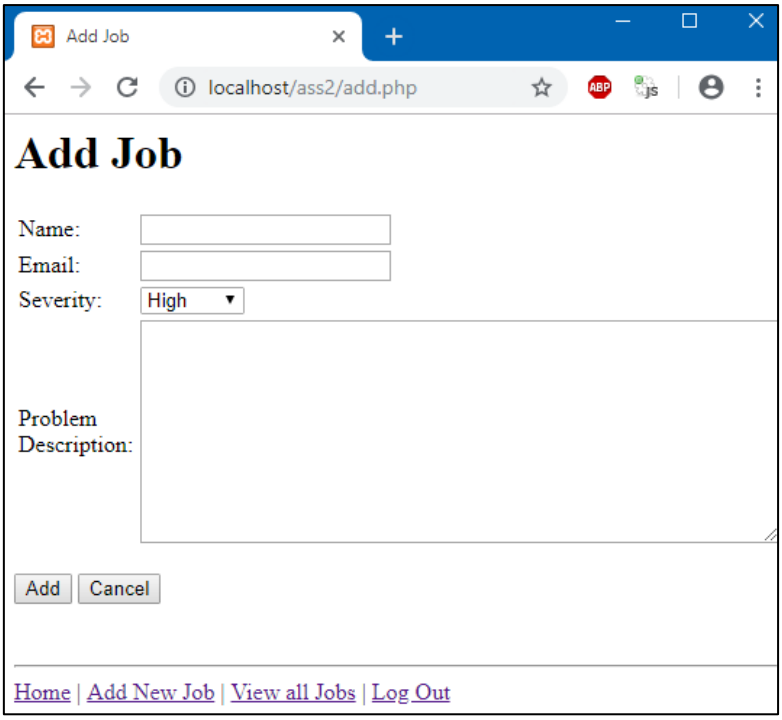

Figure 7 - Add Page

There should be a logout.php page which will allow a user to logout.

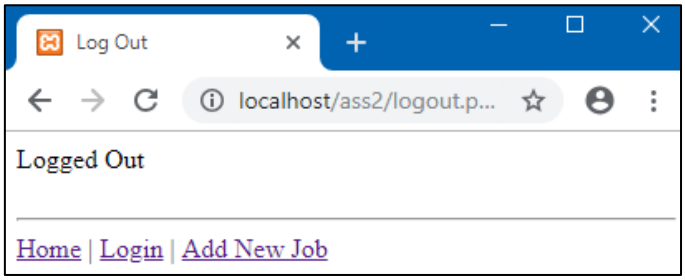

Figure 8 - Logout Page

Users should not be permitted access to the edit, and delete pages unless they are logged in. Users should not be able to view a list of all jobs unless they are logged in. Sessions should be implemented so users only need to log in once to access all the pages. Your code should be free from error, and be appropriately indented and spaced.

The first step in starting your assignment is to login to phpMyAdmin and run the supplied CreateDatabase.sql script. This will create the 'assignment2' database and add the two tables: 'jobs' and 'authroized_users'. It also grants the user webauth SELECT, UPDATE, INSERT, and DELETE privileges.

You will require a number of SQL statements in your PHP coding. The required SQL statement to check for a correct username password combination is given in RequiredStatements.txt, however you will need to create the remaining statements yourself through inspection of the database structure. The given statement has not been formatted for PHP code; this is left for you to complete.

Your site should function without error. If PHP error messages or warnings are displayed, you will lose marks.

What you must submit

You are required to submit 7 core PHP files.

1. login.php
2. home.php
3. jobs.php
4. add.php
5. edit.php
6. delete.php
7. logout.php

If you use other files in the completion of your assignment, you must supply these as well.

You are required to submit 7 screenshots showing each of the above .php files (excluding logout.php) running in your web browser. Paste each screenshot into a single Word document.

Your database does not need to be submitted. You should ensure your completed PHP pages function correctly using the supplied database without any edits to the database data. If your supplied solution does not function correctly with the database, you may lose marks.

You must submit all your files in a single .zip archive named 'ass2.zip'.

Marking Criteria

Item	Marks (out of 100)
Authentication and Session Control *Users should be able to login and logout. Appropriate pages should be protected. Session control should be used. Username should be displayed in welcome message. Other miscellaneous requirements required to meet specification.*	20
Query and display data *Data should be displayed from the database in the table of records. Data should be sorted according to specification. Filtering should be implemented according to specification. Edit and Delete should function for each record. Columns shown should match specification. Other miscellaneous requirements required to meet specification.*	25
Update records in the database *Record details should be saved on update. Record details should be preloaded into form. Update and cancel buttons should function. Edit page should be protected. Form should match specification. Other miscellaneous requirements required to meet specification.*	10
Delete data from the database *Record details are successfully removed from the database on deletion. Record details should not be editable but shown for review before deletion. Delete and cancel buttons function correctly. Delete page should be protected. Page should match specification. Other miscellaneous requirements required to meet specification.*	10
Add records to the database *New record details are successfully saved to the database. Add and Cancel buttons should function correctly. Form should match example provided. Other miscellaneous requirements required to meet specification.*	10

Secure Code
• Secure coding techniques used
  ○ Secure coding techniques include the use of prepared statements where appropriate, limiting input text areas to a maximum length, and separating database connection logic into its own file.
  ○ No marks will be awarded if no secure coding techniques are used.
  ○ Half marks will be awarded if some secure coding techniques are used on some pages, or secure coding techniques have been used incorrectly.
  ○ Full marks will be awarded if all secure coding techniques are used correctly on all pages.
	10

Presentation • Indentation and spacing used in code (5 marks) • PHP variables and functions follow camel case (2.5 marks) • HTML ids and names follow kebab case (2.5 marks) • Code duplication is minimised (5 marks)	15