BCS 350: Web Database Development

JLi Spring 2024

Capstone Project Milestones

Notes: To help you manage your capstone project, I broke down the capstone project into three milestone stages and defined a list of tasks to be completed. I also listed references that can help you in each task and suggested a deadline for each milestone. I highly suggest you follow this list to work on your project. You are not required to submit any milestone report. But the completion of each milestone helps you stay on the right schedule. If you encounter any issue, please seek help as early as possible.

Milestone 1: by 4/5
Milestone 2: by 4/19
Milestone 3: by 5/1

Names	**Tasks**	**Completion of Tasks** **Yes - if the task is completed** **No - if the task is not completed** **Partial – describe what part is not completed**
Milestone 1: Create database/table(s) and populate table(s); Create the main menu page and complete four function modules; Sanitize user input. (40% of the project work)		
1	Run mysql as root and create a new database and a new user with password that are specified below and grant this user the full privilege to the new database: Database name: **`bcs350sp24`** Username: **`usersp24`** Password: **`pwdsp24`**	
2	Document the commands that creates the database and user. Please document commands in the project report file.	
3	Write a php file `setupDB.php` to create your database application table(s) and populate application table(s) with initial values (use “CREATE TABLE IF NOT EXISTS …”). If there is one application table in the database, it should have at least 5 fields. Primary key and/or other indexes must be specified.	
References:	example8-14 Create a table from a command prompt example8-8 Add data from a command prompt example10-7 Create a table using PHP	

	example10-10 Add records using PHP example10-10-2 Add multiple values using PHP (Module 6 Video Lecture 4)	
4	Make a main menu web page that has links to List Records, Add Records, Search for Records, and Delete Records. Each link leads to corresponding web pages to list or manipulate the records in your database.	
5	**Listing records** of database table(s). You can decide what information will be listed on the web page. The data must be displayed in a neat tabular format. A link is provided for the user to return back to the main menu.	
6	**Adding a record** into the database. Provide text fields or other HTML forms to allow user to add a record into the database. A link is provided for the user to return back to the main menu.	
7	**Searching for records** in the database. Use a drop-down menu for user to choose a field to search and a text field for user to enter the information of that field to search for, for example, in our class example, if a user chooses "author" then the text field allows a user to enter the "author" information to look up. A link is provided for the user to return back to the main menu.	
8	**Deleting a record** from the database. You have two ways to do this. The first approach is you can modify book examples to list all records and provide "Delete Record" button for each record. Another approach is you can allow user to search for a record and delete it if the record exists and the user confirms to delete it. A link is provided for the user to return back to the main menu.	
9	All input data should be sanitized to prevent injection attacks. Prepared statements with placeholders must be used for "adding records" task to sanitize the user input. All output data must also be sanitized to prevent injection attacks.	
References	example10-6 Insert record, delete record, and sanitize output data example10-7 Create a table example10-8 Describe a table and output in a table format	

	example10-9 Drop a table example10-10 Add records example10-11 List records in a table example10-12 Update records example10-13 Delete records example10-19 Sanitize input data using prepared statements example10-20 Sanitize input data using PHP sanitizing functions (with database connection)	
Hints:	The files for main menu and 4 function modules should be php files so that session control can be added later.	
Milestone 2: Complete user registration and log in. Implement web authentication and password security. (40% of the project work)		
1	Create a table *users* to store the registered user's information including username, email, and password. The username is the primary key. The users table doesn't have initial values. Add related code to `setupDB.php` file so that the `setupdb.php` creates the application table(s) and the users table and populate the application table(s).	
2	Make a user registration page. The registration form should at least have fields for **email, username, password**, and **confirm password.**	
3	After a user submits data, check if the username is available. If yes, add the user into the *users* table; otherwise, display an error message and provide a link back to the registration page so user can choose another username to complete the registration.	
4	User input must be sanitized before stored into the *users* table.	
5	The password stored in the *users* table must be salted and hashed.	
6	Implement the form-based web authentication for user log in (not basic HTTP authentication). Make a user login page that has a form to allow user to log in with username and password. If the user enters incorrect username or password, display an error message and provide a link back to the login page so user can reenter the username and password. If the user enters correct username and password, session starts and the user enters the main menu page.	

7	Implement session control to the four function modules in the main menu so if the user didn’t log in, the four function modules wouldn’t work.	
8	Add a log out link to the main menu page to log out a user. Session is deleted when a user logs out.	
References	example12-3 Register new user to the users table example12-5 Log in a user and start session. example12-6 Retrieve session variables on other pages after user logs in. example12-8 Log out a user	
Milestone 3: Implementation of client-side validation and server-side validation (15% of the project work)		
1	Implement client-side validation using JavaScript. Use the same criteria as the book example 16-2. Password and Confirm Password must match for the password to be accepted.	
2	Implement server-side validation with PHP. Use the same criteria for JavaScript validation.	
References	Example16-1.html Registration form example16-3.php JavaScript validation and PHP validation	
Hints:	The registration form and JavaScript validation goes to the registration form file. The PHP validation goes to the registration process file.	
Finalize Project: Integrate all modules, test and debug your project. Document project.		