

CS Project

Analysis

The project is a database-backed website for a job-search site. The website will allow users to one of two types of account. A low-level user will create an account and be able to search for job posts. They can also log out and delete their account. A high-level user can create a company account with the same abilities as a low-level user with the addition of being able to add jobs.

The Advanced Higher Concepts are:

- It will have an easy-to-use interface with validated inputs
- User login details and job details are stored in SQL databases
- PHP is used to link the webpages to the external database
- External CSS is used to style the webpage
- CSS Media queries are used to ensure that the webpages are compatible on different devices like mobiles
- Queries are used to input form data into the databases
- Session variables are used to ensure that user login details are carried across to every page they visit
- Results of a database search are displayed on the webpage using HTML table elements like <th> and <tr>

The **scope** of this project will include:

- Analysis of the project and a full design including diagrams and pseudocode
- A working webpage and database which is suitable for the use case of a job website
- A test plan which has been planned out and used to ensure the project works fully as intended
- An evaluation of the project at the end

The **boundaries** of this project include:

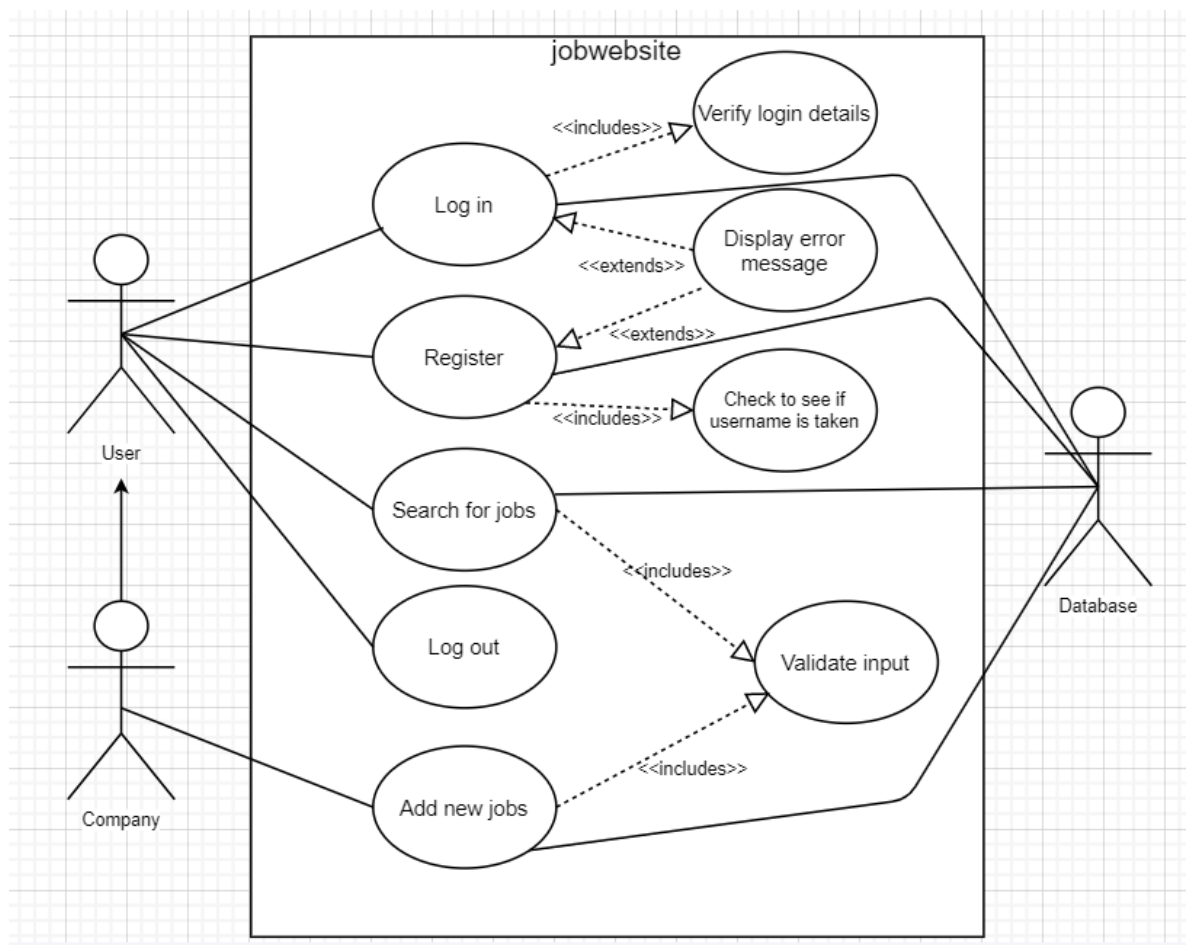
- Every user must have a unique username

- Their password must be at least 3 characters long and they will be given a hash value in the database
- Users can search for any job opening and companies can add as many job openings as they want

The **constraints** of this project are:

- It will be written in PHP and the database queries will be done in MySQL.
- XAMPP is used as the PHP interpreter and the database will be viewed in PHPMyAdmin
- There will be no economic constraints to the project
- It will have to abide by the Data Protection Act and GDPR
- It must be completed by the SQA deadline of March 2020

Use-case Diagram



Requirements Specification

The functional requirements of the project are:

- Users will be able to register for an account and their details will be added to the linked database using PHP to process form input
- It will have a simple interface which makes the website easy to use and navigate
- Users can log in and their input will be compared to the database data using SQL
- CSS will be used to style the pages
- A connection will be made whenever the user communicates with the database
- Users will be able to search the database for jobs and the results will be displayed on the webpage
- Company users will have the ability to add jobs to the database
- Session variables will be used to carry login details across pages
- Media queries will be used to ensure that the pages are compatible on different screen sizes like mobile phones and tablets

End-User Requirements are as follows:

Low-level users of the website will expect to be able to:

- Register an account and have their details stored in the database
- Login to their account whenever they visit the website
- Search for job posts and have their results displayed on the screen

Higher-level users (companies) will expect to be able to:

- Have the same abilities listed above
- Add new job openings to the database which will be able to be viewed by all users when they search

Inputs, Processes and Outputs

Users will input the following when registering an account:

- First name
- Last name
- Phone Number
- Username (input when logging in also)
- Password (input when logging in also)

The website will have the following:

- Processes
 - Validate the user inputs like ensuring fields are not left blank and passwords meet a minimum length requirement
 - Create a connection with the database
 - Execute queries when users prompt it to
 - Check user login data matches database data
 - Use session variables
 - Use CSS media queries for device compatibility
- Outputs
 - Output error messages when for example a condition is not met
 - Search results when a user searches for a job

Project Plan

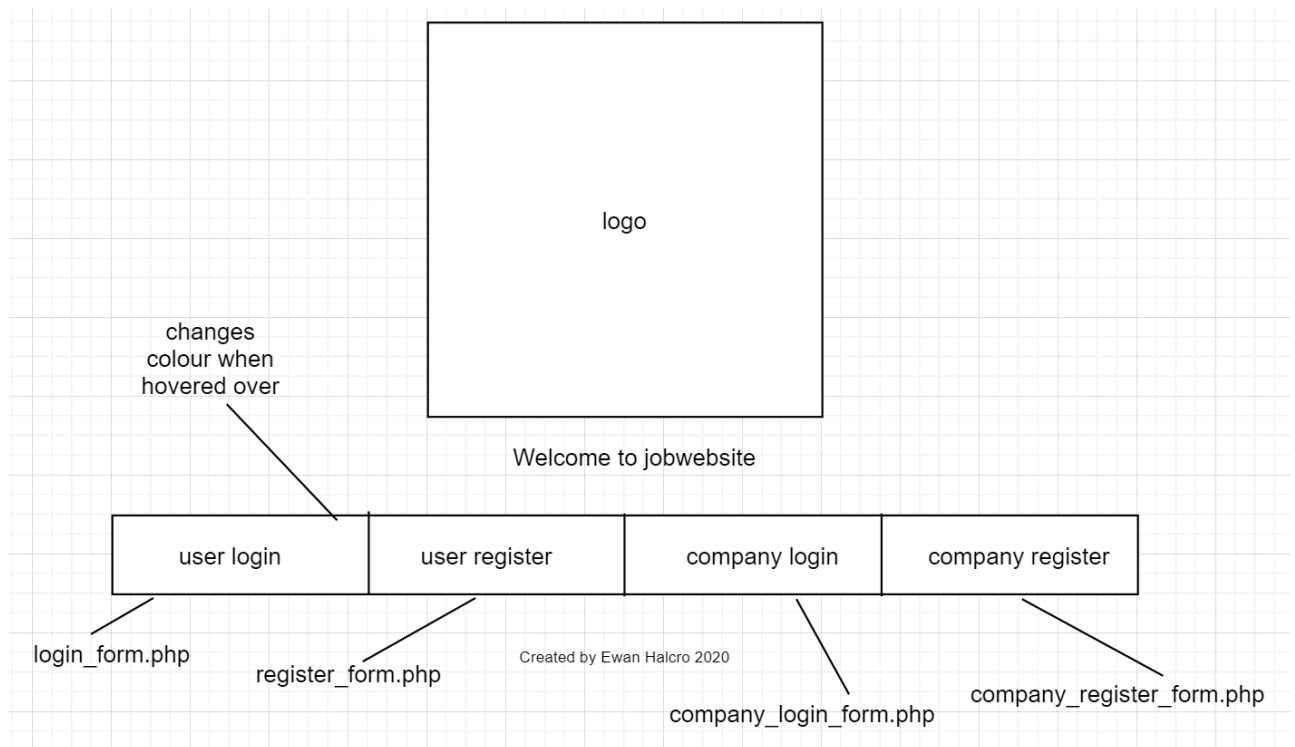
Resources needed

Analysis	Google Docs Google Sheets Google Chrome Draw.io diagram software
Design	Google Docs Google Chrome Draw.io diagram software
Implementation	Google Docs Google Chrome XAMPP PHP Processor Atom Text Editor PHPMyAdmin/MySQL
Testing	Same as implementation
Evaluation	Google Docs Google Chrome Atom Text Editor

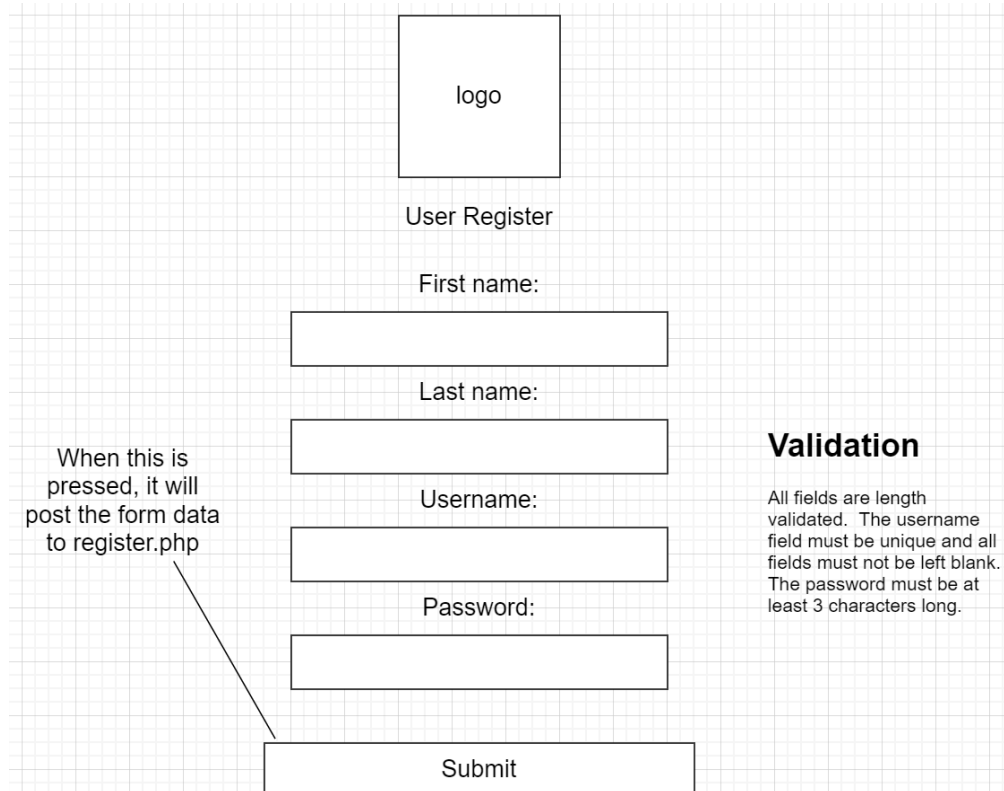
[illegible]

Design

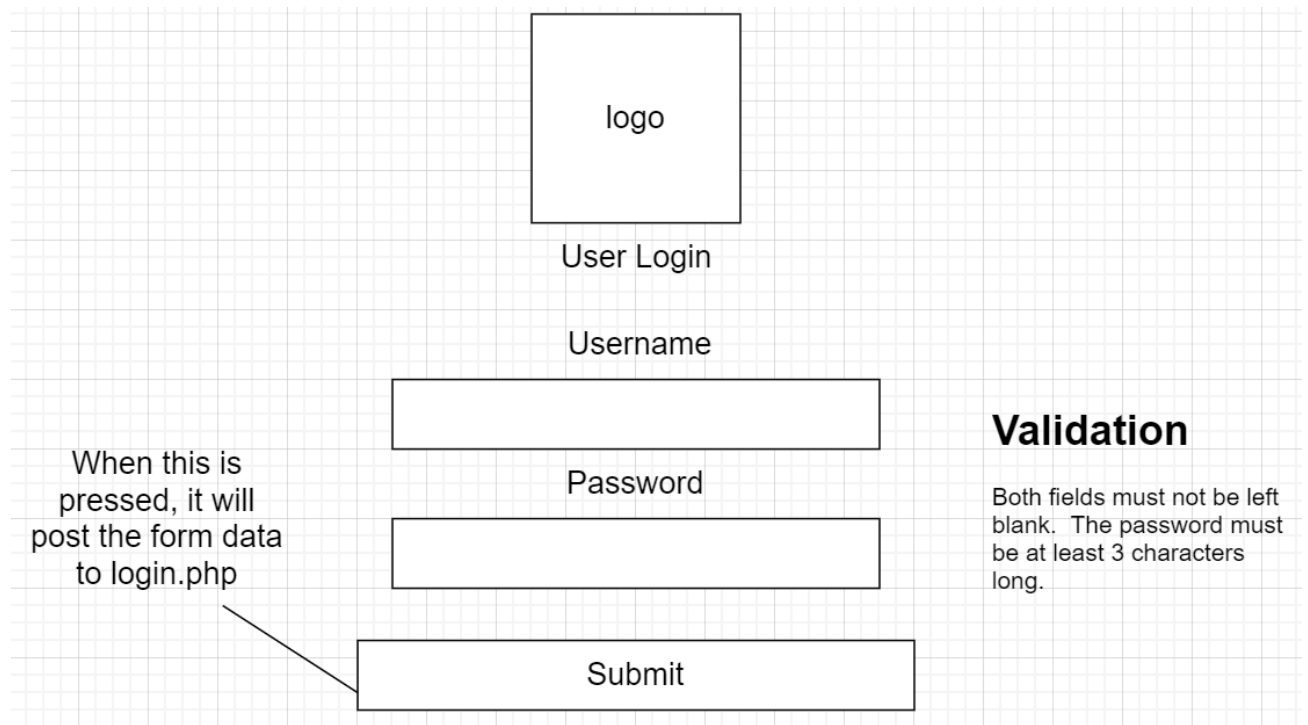
Home Page Prototype



Registration Page



User Login Page



When this is pressed, it will post the form data to login.php

logo

User Login

Username

Password

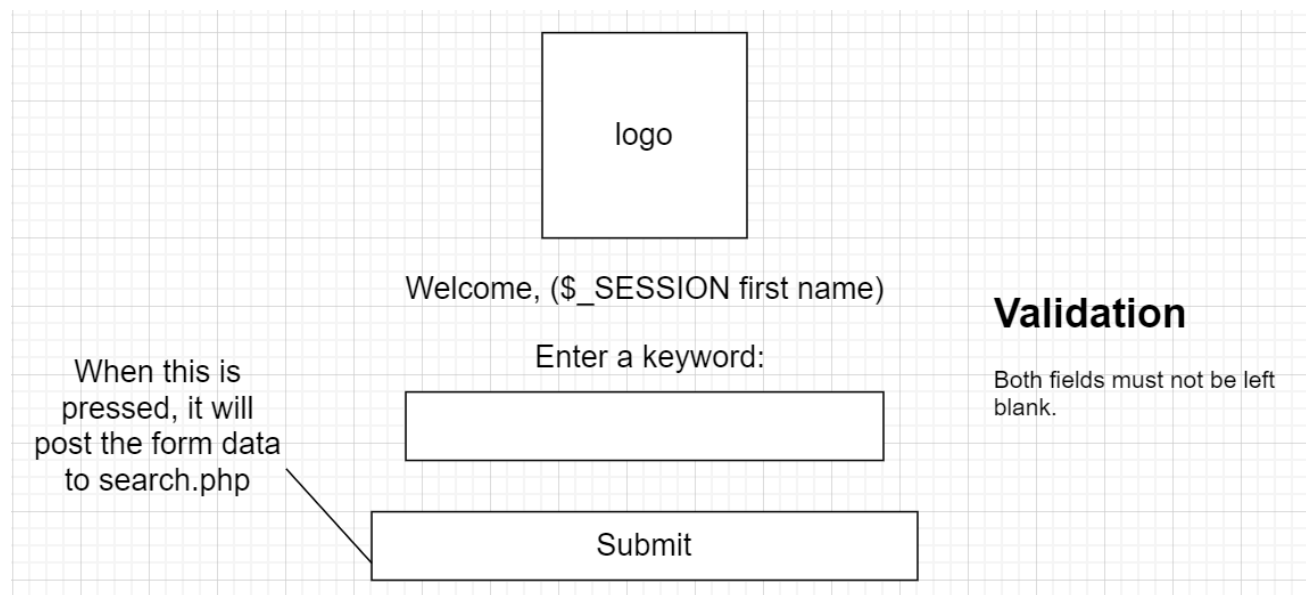
Submit

Validation

Both fields must not be left blank. The password must be at least 3 characters long.

The wireframe for the User Login Page is set against a light gray grid. At the top center is a square box labeled 'logo'. Below it is the text 'User Login'. Further down are two horizontal rectangular input fields, the first labeled 'Username' and the second labeled 'Password'. At the bottom center is a wide rectangular button labeled 'Submit'. A line points from the text 'When this is pressed, it will post the form data to login.php' to the 'Submit' button. To the right of the form fields is a section titled 'Validation' with the text 'Both fields must not be left blank. The password must be at least 3 characters long.'

Search Page



When this is pressed, it will post the form data to search.php

logo

Welcome, (\$_SESSION first name)

Enter a keyword:

Submit

Validation

Both fields must not be left blank.

The wireframe for the Search Page is set against a light gray grid. At the top center is a square box labeled 'logo'. Below it is the text 'Welcome, (\$_SESSION first name)'. Further down is the text 'Enter a keyword:' followed by a horizontal rectangular input field. At the bottom center is a wide rectangular button labeled 'Submit'. A line points from the text 'When this is pressed, it will post the form data to search.php' to the 'Submit' button. To the right of the form fields is a section titled 'Validation' with the text 'Both fields must not be left blank.'

Company Register Page

logo

Company Register

Company name:

Password

Submit

When this is pressed, it will post the form data to company_register.php

Validation

All fields are length validated. The company name field must be unique and both fields must not be left blank. The password must be at least 3 characters long.

Company Login

logo

Company Login

Company name:

Password

Submit

When this is pressed, it will post the form data to company_login.php

Validation

Both fields must not be left blank. The password must be at least 3 characters long.

Add Page

logo

Welcome (\$_SESSION company name)

Company Name

Title

Description

Hours

Submit

When this is pressed, it will post the form data to add.php

Validation
All fields are length validated. All fields must not be left blank. The hours field must be an integer.

Success Message

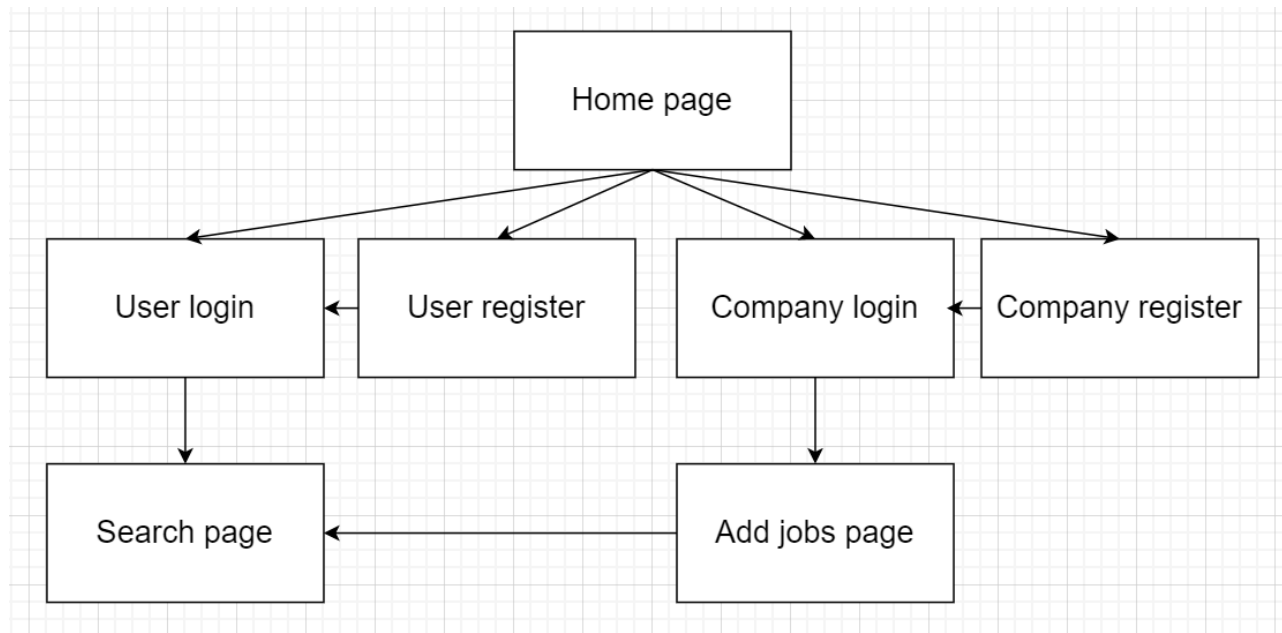
logo

Success for error message

[Link to preceeding page or previous page](#)

Users will be directed to a page like this when they submit a form

Page Structure



PHP Pseudocode

Connect to Database Pseudocode

1. Declare server name PHP variable
2. Declare database username PHP variable
3. Declare database password PHP variable
4. Declare database name PHP variable
5. Connect to the database
6. If this fails
7. Display error message
8. Kill the connection

Registration Pseudocode

1. Start session
2. Call connection PHP file
3. If the first name field on the form is not left empty
4. Assign this value to PHP first name variable using `$_POST`
5. Else
6. Display an error message
7. If the last name field on the form is not left empty

8. Assign this value to PHP last name variable using \$_POST
9. Else
10. Display an error message
11. If the username field on the form is not left empty
12. Assign this value to PHP username variable using \$_POST
13. Else
14. Display an error message
15. If the password field on the form is not left empty
16. Assign this value to PHP password variable using \$_POST
17. Else
18. Display an error message
19. Declare query to check if the username is taken
20. Execute query
21. If it returns results
22. Display a message informing the user
23. Else
24. Hash password
25. Declare query to add details to the database
26. Execute the query
27. Display a success message
28. Close the connection

Login Pseudocode

1. Start session
2. Call the connection file
3. If the username field on the form is not empty
4. Assign this value to the PHP username variable using \$_POST
5. Else
6. Display error message
7. If the password field on the form is not empty
8. Assign this value to the PHP password variable using \$_POST
9. Else
10. Display error message
11. Declare query to check if entered details match database record

12. Execute the query
13. If there is 1 result
14. Display a success message
15. Else
16. Display an error message
17. Declare array
18. If the query returned a result
19. While the result matches the fetched array result
20. Put the result into the array
21. Loop through the array
22. Assign the first name element to a session variable
23. Assign the last name element to a session variable
24. Assign the username element to a session variable
25. Assign the password element to a session variable
26. Close the connection

Search PHP Pseudocode

1. Start session
2. Call the connection file
3. If the input field in the form is not empty
4. Assign this value to the PHP input variable using \$_POST
5. Else
6. Display error message
7. Declare query to search for records which match the user input using wildcards
8. Execute the query
9. Display a message saying how many results there were
10. If there are 1 or more results
11. Display the results in a table form
12. Close connection

Logout Pseudocode

1. Start the session

2. Destroy the session
3. Return to home page

Displaying session variables Pseudocode

1. Start the session
2. If the first name session variable is empty
3. Display the username session variable
4. Else
5. Display the first name session variable

The pseudocode for company register, login and adding jobs will be very similar to some of the above code. Company register and add jobs will be the same as the registration code with different variables and queries. Company login will be the same as user login with different variables and queries.

Query Design

Registration

SELECT	user_ID
FROM	users
WHERE	Username = \$username

INSERT	First_name, last_name, username, password
INTO	users
VALUES	\$first_name, \$last_name, \$username, \$password

Login

SELECT	* (first_name, last_name, username, password, user_ID)
FROM	users

WHERE	Username = \$username and password = \$password
-------	---

Search

SELECT	* (company_name, title, description, hours)
FROM	jobs
WHERE	Any field is like user input using wildcards

Company Register

SELECT	company_ID
FROM	companies
WHERE	company_name = \$company_name

INSERT	Company_name, password
INTO	companies
VALUES	\$company_name, \$password

Company Login

SELECT	* (company_name, password)
FROM	companies
WHERE	Company_name = \$company_name and password = \$password

Add jobs

INSERT	Company_name, title, description, hours
INTO	jobs
VALUES	\$company_name, \$title, \$description, \$hours

Data Dictionary

Users Table

Field	Type	Length	Required	Key
first_name	varchar	15	yes	
last_name	varchar	20	yes	
username	varchar	10	yes	
password	varchar	25	yes	
user_ID	int		yes	Primary

Companies Table

Field	Type	Length	Required	Key
company_name	varchar	25	yes	
password	varchar	25	yes	
company_ID	int		yes	Primary

Jobs Table

Field	Type	Length	Required	Key
company_name	varchar	25	yes	
title	varchar	30	yes	
description	varchar	100	yes	
hours	int		yes	
job_ID	int		yes	Primary

Implementation

All screenshots are in a separate file. Code to create the tables is below.

They appear in this order:

- 1. Home.php code**
- 2. Home.php web screenshot**
- 3. Login_form code**
- 4. Login_form web screenshot**
- 5. Login.php code (1)**
- 6. Login.php code (2)**
- 7. Login.php web screenshot (1)**
- 8. Login.php web screenshot (2)**
- 9. Register_form web screenshot**
- 10. Register.php web screenshot**
- 11. Search_form code**
- 12. Search_form web screenshot**
- 13. Search.php code (1)**
- 14. Search.php code (2)**
- 15. Search.php web screenshot**
- 16. Company_login_form code**
- 17. Company_login_form web screenshot**
- 18. Company_login.php code (1)**
- 19. Company_login.php code (2)**
- 20. Company_login.php web screenshot (1)**
- 21. Company_login.php web screenshot (2)**
- 22. Company_register_form code**
- 23. Company_register_form web screenshot**
- 24. Company_register.php code (1)**
- 25. Company_register.php code (2)**
- 26. Company_register.php web screenshot**

27. **Add_form code**
28. **Add_form web screenshot**
29. **Add.php web screenshot**
30. **Add.php code (1)**
31. **Add.php code (2)**
32. **Logout.php code**
33. **Conn.php code**
34. **Register_form code**
35. **Register.php code (1)**
36. **Register.php code (2)**
37. **Users database**
38. **Companies database**
39. **Jobs database**
40. **Style.css (1)**
41. **Style.css (2)**
42. **Style.css (3)**
43. **Style.css (4)**
44. **Style.css (5)**

```
CREATE TABLE users(  
    first_name varchar(15) NOT NULL,  
    last_name varchar(20) NOT NULL,  
    username varchar(10) NOT NULL UNIQUE,  
    pass varchar(25) NOT NULL,  
    user_ID int PRIMARY KEY AUTO_INCREMENT  
);
```

```
CREATE TABLE companies(  
    company_name varchar(25) NOT NULL,  
    pass varchar(25) NOT NULL,  
    company_ID int PRIMARY KEY AUTO_INCREMENT  
);
```

```
CREATE TABLE jobs(  
    company_name varchar(25) NOT NULL,
```

```
title varchar(30) NOT NULL,  
description varchar(100) NOT NULL,  
hours int NOT NULL,  
job_ID int PRIMARY KEY AUTO_INCREMENT  
);
```

New Skills Learned

For input validation, I used the empty() function. This is used to see if a variable has a value for not. It is a form of server-side validation which will kick in if the client-side validation in the form fails.

To improve data security, I wanted to give the passwords a hash value in the database so that database admin would not be able to see the passwords. I did some research and used the password_hash function to secure people's passwords.

To store session variables, I had to create an array with the fetched results from a query. I researched and used a foreach loop which iterates through an array, separating each element to assign them to a \$_SESSION variable.

The mysqli_real_escape_string() function was used to escape special characters of a string to then use in an SQL query. It took the user input on the search page and validated it, making it usable in the query.

Testing

Plan

To test the solution, I will use sample data in the various forms to check that it definitely works as intended.

On the register and add pages I will input regular data, a password below three characters, an existing username and data which exceeds the length validation.

On the login pages, I will enter correct input and input which does not exist in the database.

On the search page, I will try leaving the field blank and make sure that the user's first name appears on the form using session variables.

An example of a persona for this system is a job-seeker. This persona must be able to register an account, login with these details, have their name appear on the search form and be able to search for jobs.

An example of a use-case is a company adding a job opening onto the system. They must be able to register an account, login with these details, have their name displayed on the add jobs page and add new jobs to the database.

Test evidence

User register (company register results were the same)



User Register

First Name:

Last Name:

Username:

Password:

submit



Successfully registered.

[Return to login page and sign in.](#)

Created by Ewan Halcro 2020

Username:

Password:



Please fill out this field.

a

Password:



Please lengthen this text to 3 characters or more (you are currently using 1 character).

Submit



Sorry, this username is taken.

[Return to register page.](#)

Created by Ewan Halcro 2020

User login (company login is the same)



log in

Username:

Password:

submit

[Don't have an account. Click here to register.](#)

Username:

Password:



Please fill out this field.

a

Password:



Please lengthen this text to 3 characters or more (you are currently using 1 character).

submit



Successfully logged in.

[Proceed to search page.](#)



Login failed. Please return to login page or register.

[Return to login page.](#)

[Register here.](#)

Created by Ewan Halcro 2020

Search page



Welcome, Test

Search for a company, job or hours:

Search

Created by Ewan Halcro 2020



8 result(s) found for ''

Company	Job Title	Description	Hours
BBC	Presenter	Presenting the news	20
Tesco	Nightshift	Stacking shelves	16
Amazon	Delivery Driver	Delivering parcels	25
Microsoft	Software Engineer	Making new apps	40
Google	Internship	Summer Internship in SF	18
Dunder Mifflin	Regional Manager	Not doing any work	37
Torchwood	Team Member	Investigating Alien Stuff	50
NHS	Finance Officer	Spreadsheets	38


[Return to search page.](#)

[Logout.](#)



Welcome, Bob

Search for a company, job or hours:

 Please fill out this field.

Add jobs



Welcome, BBC

Add details below:

Company Name:

Job Title:

Job Description:

Job Hours:



Successfully added.

[Add another.](#)

[Search page.](#)

Created by Ewan Halcro 2020

 		company_name	title	description	hours	job_ID
<input type="checkbox"/>	 Edit  Copy  Delete	BBC	Presenter	Presenting the news	20	1
<input type="checkbox"/>	 Edit  Copy  Delete	Tesco	Nightshift	Stacking shelves	16	2
<input type="checkbox"/>	 Edit  Copy  Delete	Amazon	Delivery Driver	Delivering parcels	25	3
<input type="checkbox"/>	 Edit  Copy  Delete	Microsoft	Software Engineer	Making new apps	40	4
<input type="checkbox"/>	 Edit  Copy  Delete	Google	Internship	Summer Internship in SF	18	5
<input type="checkbox"/>	 Edit  Copy  Delete	Dunder Mifflin	Regional Manager	Not doing any work	37	6
<input type="checkbox"/>	 Edit  Copy  Delete	Torchwood	Team Member	Investigating Alien Stuff	50	7
<input type="checkbox"/>	 Edit  Copy  Delete	NHS	Finance Officer	Spreadsheets	38	8
<input type="checkbox"/>	 Edit  Copy  Delete	Adidas	Athlete	Running	20	9

Evaluation

My project is fit for purpose as I believe it meets all the requirements.

Requirement	Complete
Users will be able to register for an account and their details will be added to the linked database using PHP to process form input with a hashed password	yes
It will have a simple interface which makes the website easy to use and navigate	yes
Users can log in and their input will be compared to the database data using SQL	yes

CSS will be used to style the pages	yes
A connection will be made whenever the user communicates with the database	yes
Users will be able to search the database for jobs and the results will be displayed on the webpage	yes
Company users will have the ability to add jobs to the database	yes
Session variables will be used to carry login details across pages	yes
Media queries will be used to ensure that the pages are compatible on different screen sizes like mobile phones and tablets	yes
Register an account and have their details stored in the database with their password being hashed	yes
Login to their account whenever they visit the website	yes
Search for job posts and have their results displayed on the screen	yes
Add new job openings to the database which will be able to be viewed by all users when they search	yes

In terms of maintainability, my code is fairly maintainable. The PHP is kept simple and contains lots of white space. However, no internal commentary was implemented which would make it much more readable for other people. The use of external CSS means that each page is not complicated with style code and only one line of reference is needed on each page. I have also used meaningful variable names in the PHP code which makes it easier to follow. I also reference the connection to database file with a short require function in every PHP file which removes repeated code and makes it more readable.

In terms of robustness, my code is very robust. Form inputs on every page are validated in the HTML form as well as in the server-side PHP code, meaning it is very difficult to input incorrect or null data into the database. Error messages are shown whenever input is incorrect and the website does not crash when this happens.