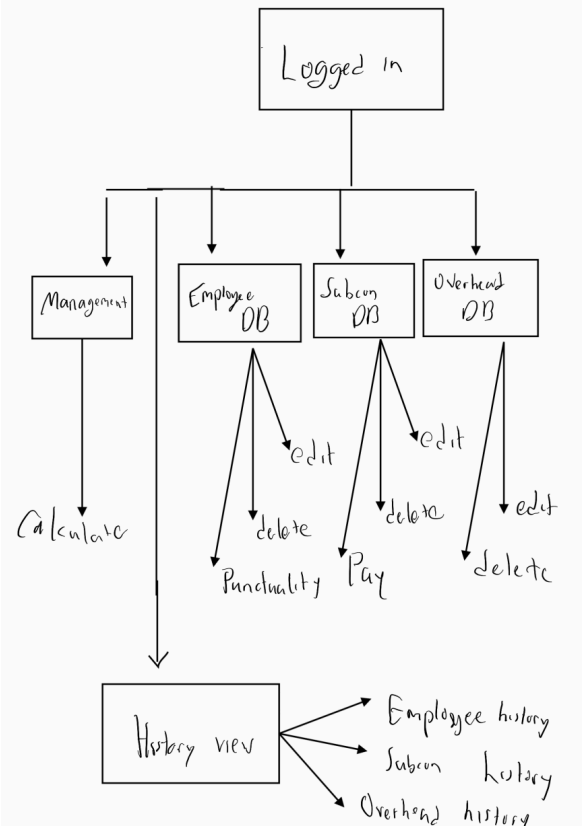
CRITERION B: Design

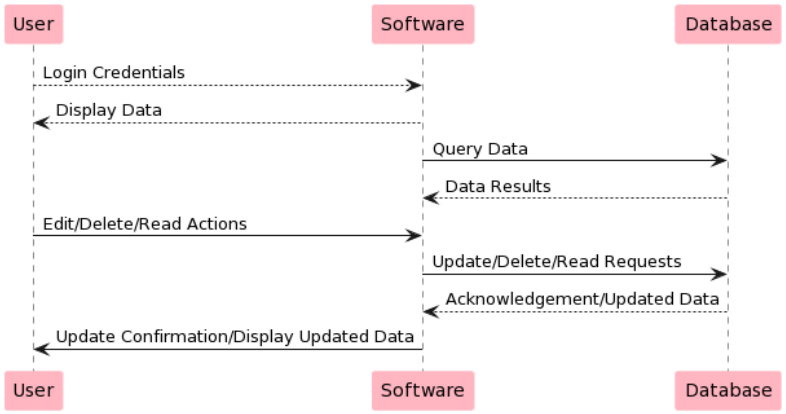
Hierarchal Chart

The figure below shows the different subpages and interfaces connected to the website.



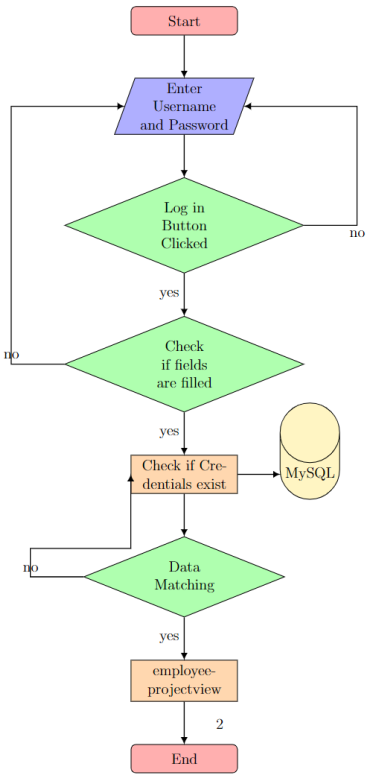
Data Flow diagram

The DFD shows the different components of the program and its interactions, such as how the user can request an edit query from the SQL database using the software.



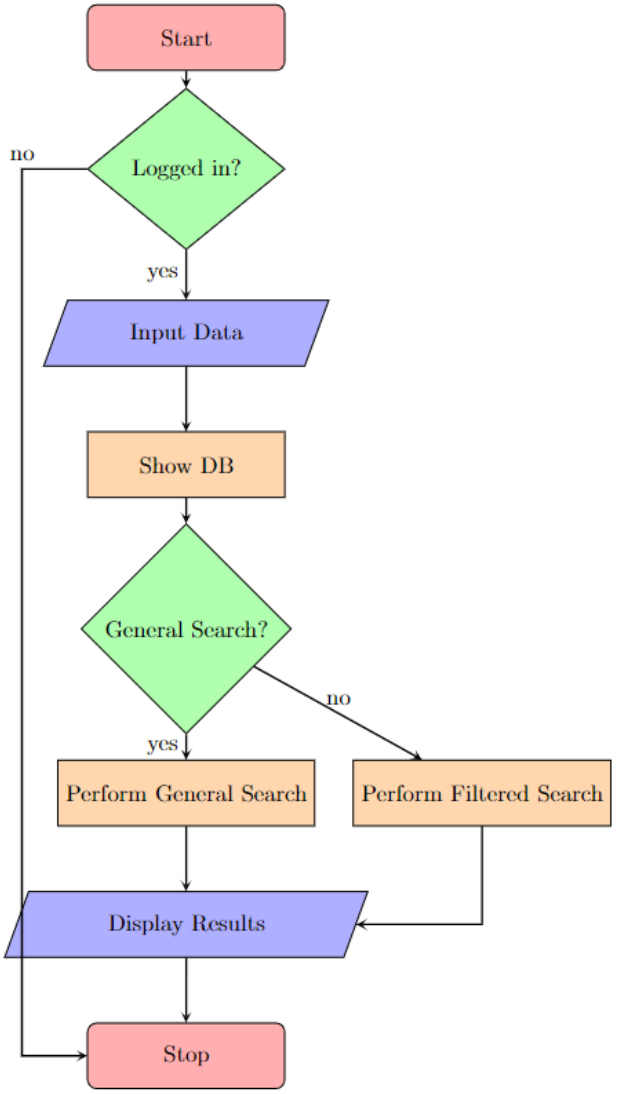
Login flowchart

This flowchart shows the initial login needed to be completed by the user. It cross-checks with an SQL database to check if the data inputted is present in the database.



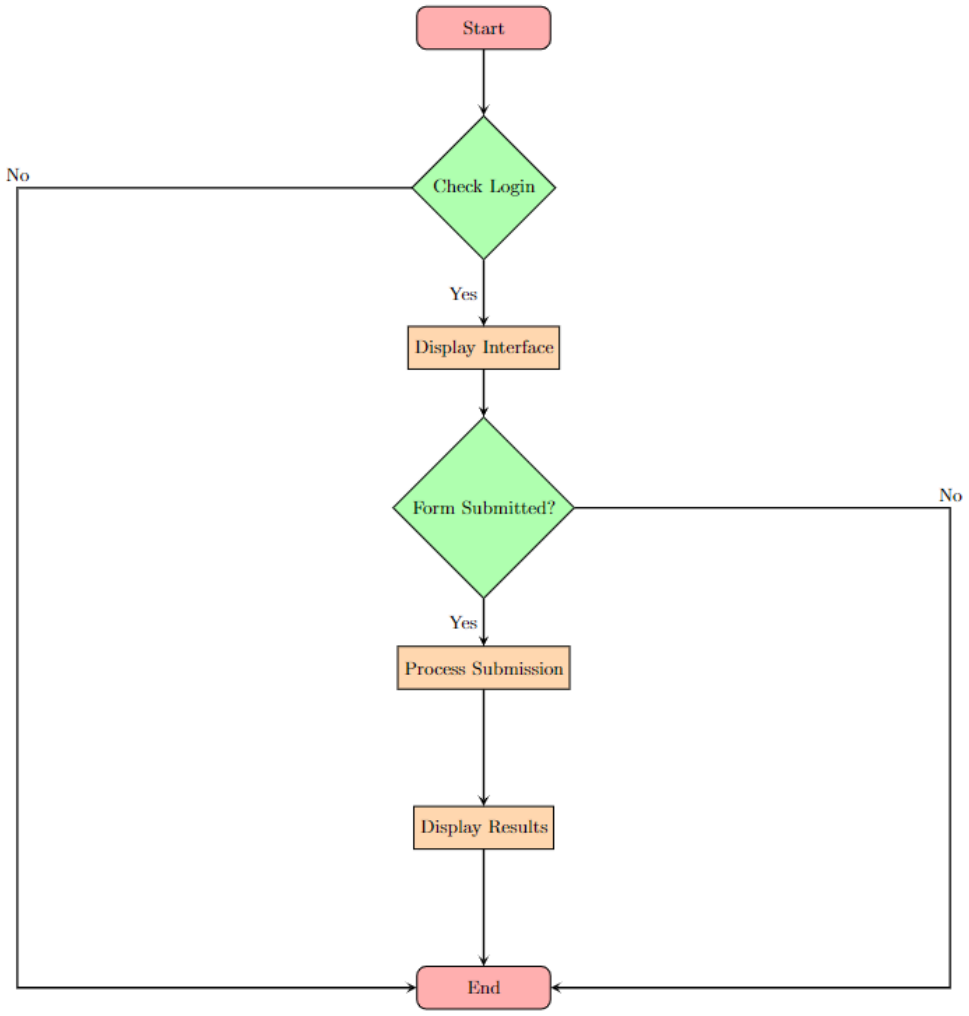
Employeeprojectview

This flowchart shows the employee database, along with the possible functionalities present within the database. All other databases are pretty similar so for the sake of brevity only this one is shown.



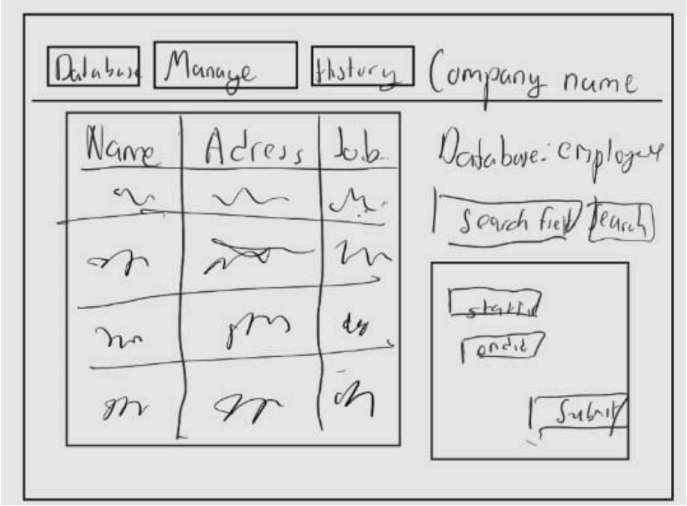
Management

This flowchart shows how the estimation costs page works. The process submission would be the process of calculating the summation of the values of the selected parameters.

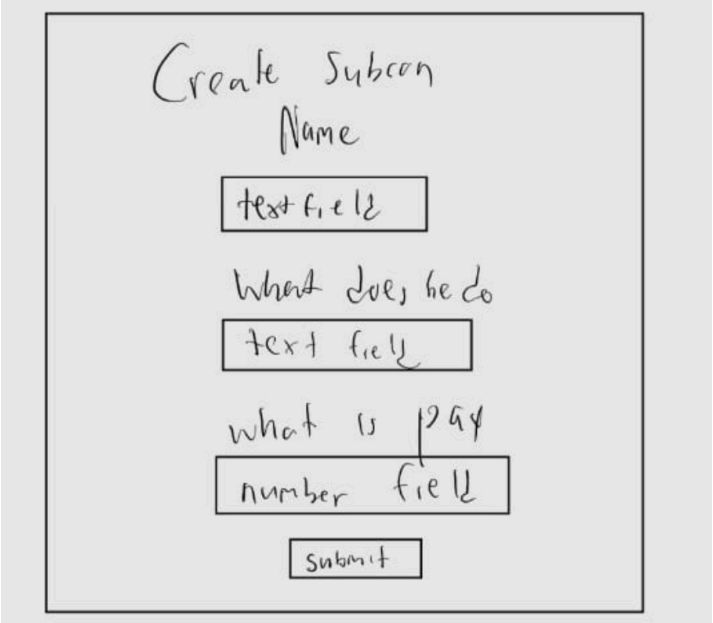


Design prototypes

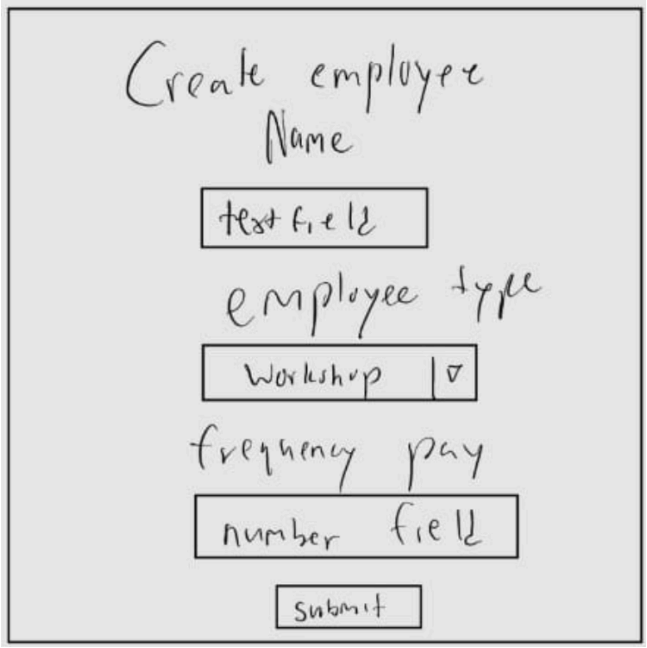
1. Employee database view



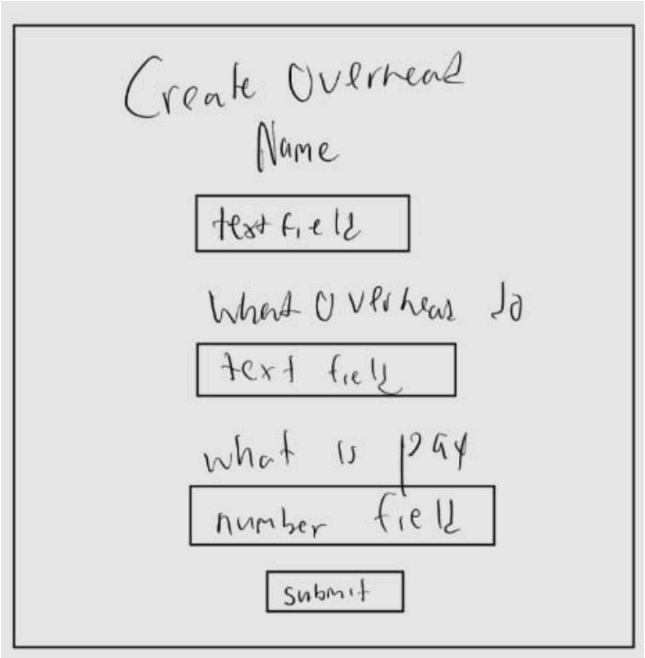
2. Create subcontractor page



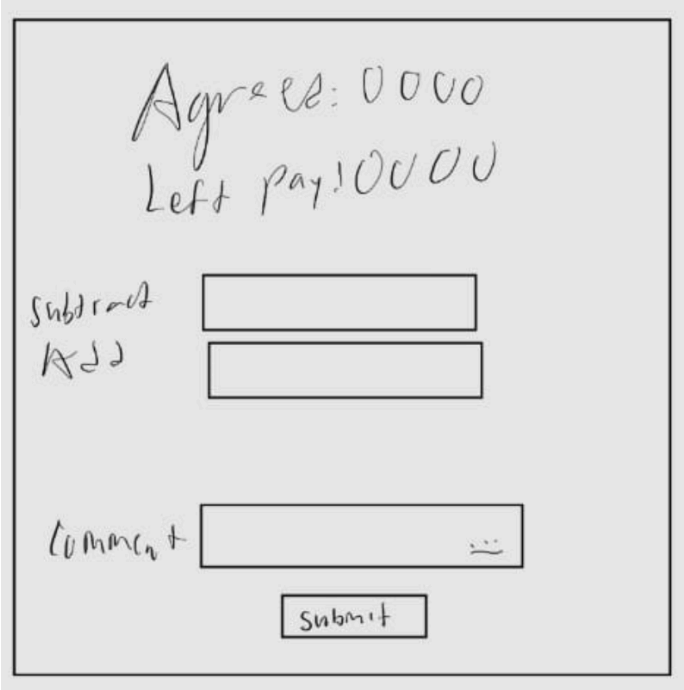
3. Create employee page



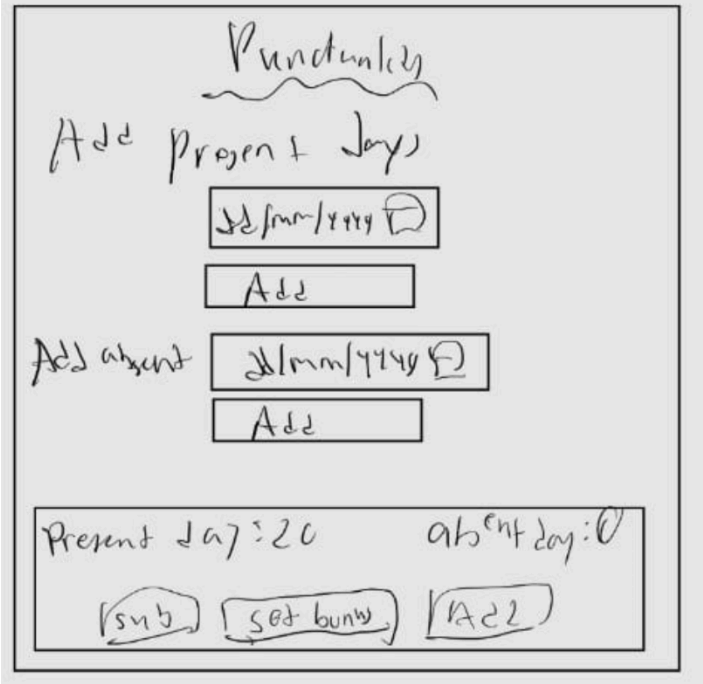
4. Create overhead page



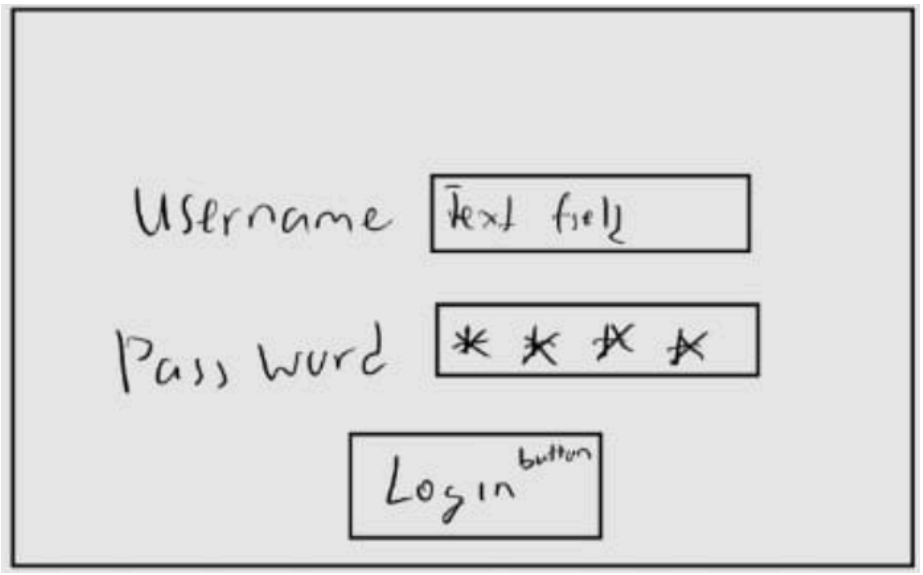
5. Edit pay page



6. Punctuality page



7. Login page



List of data members in SQL Database

1. Login table

Name	datatype	purpose	Special feature
id	int	Stores a unique ID for each instance, ordering them for access and identification	AI(auto_ increment)
username	varchar	Stores the username of Client	-
password	varchar	Stores the password of Client	-

2. Subcontractor history

Name	datatype	purpose	Special feature
uniqueid	int	Stores a unique ID for each instance, ordering them for access and identification	AI(auto_ increment)

Historyaction	Varchar	Stores the action being done to any subcon instance, which are creation, edit, and delete	-
historyname	Varchar	Stores the name of the subcontractor before being changed	-
historywork	Varchar	Stores the job of the subcontractor before being changed	-
historypay	int	Stores the pay of the subcontractor before being changed using the edit function.	-
addPay	int	Stores the amount addpay that is to be added to left pay.	-
subPay	int	Stores the amount that is to be added to left pay.	-
leftpay	int	Stores the amount left pay before being changed	-
historyid	int	The id from the subcontractor table that has its changes made to it	-
datechanged 2		The date when changes, creations, or edits are made.	-

3. Overhead history

Name	datatype	purpose	Special feature
uniqueid	int	Stores a unique ID for each instance, ordering them for access and identification	AI(auto_increment)
Historyaction	varchar	Stores the action being done to any overhead iinstance, such as create, edit, and delete	-
historyname	varchar	Stores the name of the	-

		overhead before being changed	
historywork	varchar	Stores the job of the overhead before being changed	-
historypay	int	Stores the pay of the overhead before being changed using the edit function	-
addPay	int	Stores the amount to be added to pay	-
historyvendor	varchar	Stores the vendor associated with the overhead before being changed	-
historycategory	varchar	Stores the category of work or service provided by the overhead before being changed	-
historyfrequency	varchar	Stores the frequency of the overhead’s pay before being changed	-
historydescription	varchar	Stores a description of the overhead’s services before being changed	-
historycost	int	Stores the cost associated with the overhead’’s services before being changed	-
historydate	date	Stores the date of the update to the overhead's record	-
historystatus	varchar	Stores the status of the overhead (e.g., ongoing, completed) before being changed	-
historyinvoiceNo	int	Stores the invoice number associated with the overhead’s services before	-

		being changed	
historycomments	varchar	Stores any comments related to the overhead's services or the history record itself before being changed	-
historyid	int	Id of the instance being changed, created, or deleted	-
datecreated	date	The date when the history record was created	-

4. Employee history

Name	datatype	purpose	Special feature
historyId	int	Stores a unique ID for the employee instance being changed, created, or deleted	AI(auto_increment)
historyNama	varchar	Stores the name of the employee before being changed	-
historyAlamat	varchar	Stores the address of the employee before being changed	-
historyGaji	int	Stores the wage of the employee before being changed	-
historyBonus	int	Stores the bonus amount of the employee before being changed	-
historyPayplan	varchar	Stores the pay plan details of the employee before being changed	-

historyJob	varchar	Stores the job or description of the employee before being changed	-
historyNote	varchar	Stores any notes of information about the employee before being changed	-
historyEmployeetype	varchar	Stores the type of employment (workshop/office) of the employee before being changed	-
historyDate	date	The date when the history record was created or the date of update.	-

5. Employee

Name	datatype	purpose	Special feature
name	varchar(24)	Stores the employee's name	-
alamat	varchar(24)	Stores the employee's address	-
wage	int(11)	Stores the wage of the employee	-
bonus	int(24)	Stores the bonus amount for the employee	-
payplan	varchar(24)	Stores the payment plan or salary.	-

job	varchar(24)	Stores the job or description of the employee	-
note	varchar(100)	Stores comments about the employee	-
employeetype	varchar(24)	Stores the type of employment (office/workshop)	-
id	int(12)	Stores a unique ID for each instance, ordering them for access and identification	AI(auto_increment)

6. Subcontractor

Name	datatype	purpose	Special feature
id	int(12)	Stores a unique ID for each instance, ordering them for access and identification	AI(auto_increment)
name	varchar(24)	Stores the subcontractor's name	-
work	varchar(24)	Stores the type of work	-

		the subcontractor does	
pay	int(24)	Stores the current pay for the subcontractor	-
newPay	int(24)	Stores the pay left to be paid	-

7. Overhead

Name	datatype	purpose	Special feature
name	varchar(24)	Stores the overhead's name	-
id	int(12)	Stores a unique ID for each instance, ordering them for access and identification	AUTO_INCREMENT
category	varchar(12)	Stores the category of work or service provided by the overhead	-
frequency	varchar(12)	Indicates how often the overhead needs to be paid	-
description	varchar(150)	Provides a detailed description of the overhead	-

cost	int(24)	The amount of money needed for the overhead cost	-
date	varchar(24)	The date when the overhead is due.	-
status	varchar(24)	The current status of the overhead (completed/ ongoing/ pending)	-
invoiceNo	int(24)	The invoice number of the overhead.	-
comments	varchar(150)	Any comments about the overhead	-
vendor	varchar(24)	The vendor or supplier associated with the overhead expense	-

CRITERION B: Test Plan

Test Type	Nature of Test	Example
Client can log in to application	Proper login page that allows access to software after correct input of password.	When inputting admin and admin as password and username respectively, use is redirected to the main employee database
Other people cannot log into the application without proper login credentials	Login information will be cross referenced to data in database, and if it is correct then access is granted, else it is blocked	When inputting admin and admin as password and username respectively, will redirect user to employee database, if something else is inputted it will output an error message.
Prevention of url manipulation to gain access to website	Path traversal and or URL manipulation will result in an inaccessible access and will prompt user to go back to login	When trying to manually path the url, by manually typing /IA/employee…, Entering will result in an error or alert saying to go back to main
Client will be able to create new instances of employee, subcontractor, or overhead	On click, href to a different page where client can add details for a new instance of an employee, subcontractor, or overhead	Clicking button on dropdown menu will transfer client to createemployee.php where a form can be filled out and submitted. Once submitted the database will have a new instance created and made with the same data.
Filter search results for instances of employee, subcontractor, or overhead	Allows client to read certain filtered searches through restrictions made within filter tab	General searching *accounting* will show all fields with *accounting* word within, and specific searching a number for lower bound will only show instances with the lower id bound and beyond.

Filtered search results can have simple arithmetic operations done	Filtered searches will have a corresponding operations tab where some arithmetic operations are automatically done.	General searching accounting will show all fields with accounting word within it, and traversing over to the operations tab will show the summation of the respective instances of calculatable values.
Client can be able to edit instances of employee, subcontractor, or overhead	Clicking EDIT on the options tab will result in a redirect to a page dedicated to editing with autofilled fields. After done submitting, the old data will be sent to the respective history tables and the new field data will update the respective tables.	Clicking editing will href to edit.php where edits can be made to fields. After done submitting, the old data will be sent to the respective history tables and the new field data will update the respective tables.
Client can manage employee instances for their punctuality	Client can click on an instance of an employee's punctuality field where they are redirected.	Clicking punctuality will redirect client to a page where they can add or append absent and present days.
Client can manage employee instances for their bonus	On the same punctuality page, Client can check the present and absent days, and if he is satisfied can add a bonus accordingly	Clicking punctuality will redirect client to a page where they see 5 present dates and comments and 1 absent with comment, client is satisfied with the present days and decides to add 500,000. This is the old field is sent to history and the new bonus is updated by adding the value.
Client can add or subtract the pay from the subcontractor	In the subcontractor database clicking on the pay field will redirect client to a page displaying the	Clicking on pay field for glass maker will redirect to a page where it shows the amount they initially

	pay and left to pay. Client can add or subtract from pay to make left to pay, showing how much he needs to pay.	agreed to pay. After transferring money to his account, he subtracts from the initial pay amount, reducing the amount in the left to pay. This action updates the subcon table whilst saving the last instance in the subcon history table.
Arithmatic operations for estimation of total costs for certain time periods	Client clicking on management in navigation bar redirects him to a page that helps estimate his business cost for a certain set time.	Clicking on management will redirect user to management page. The user can select what to have in the calculations such as including employee bonuses but not subcontractors, and sets the amount of days along with which instances to include for which. Afterwards an estimated sum is given.
Viewable history of all CRUD operations made to instances of employee, subcontractor, or overhead.	Different pages for different histories made to respective databases.	Clicking on the different buttons will result in popups which shows all the changes made to fields.
All changes to database must be done through the application.	No changes need to be made through backend sql insertions	All changes necessary made through website.