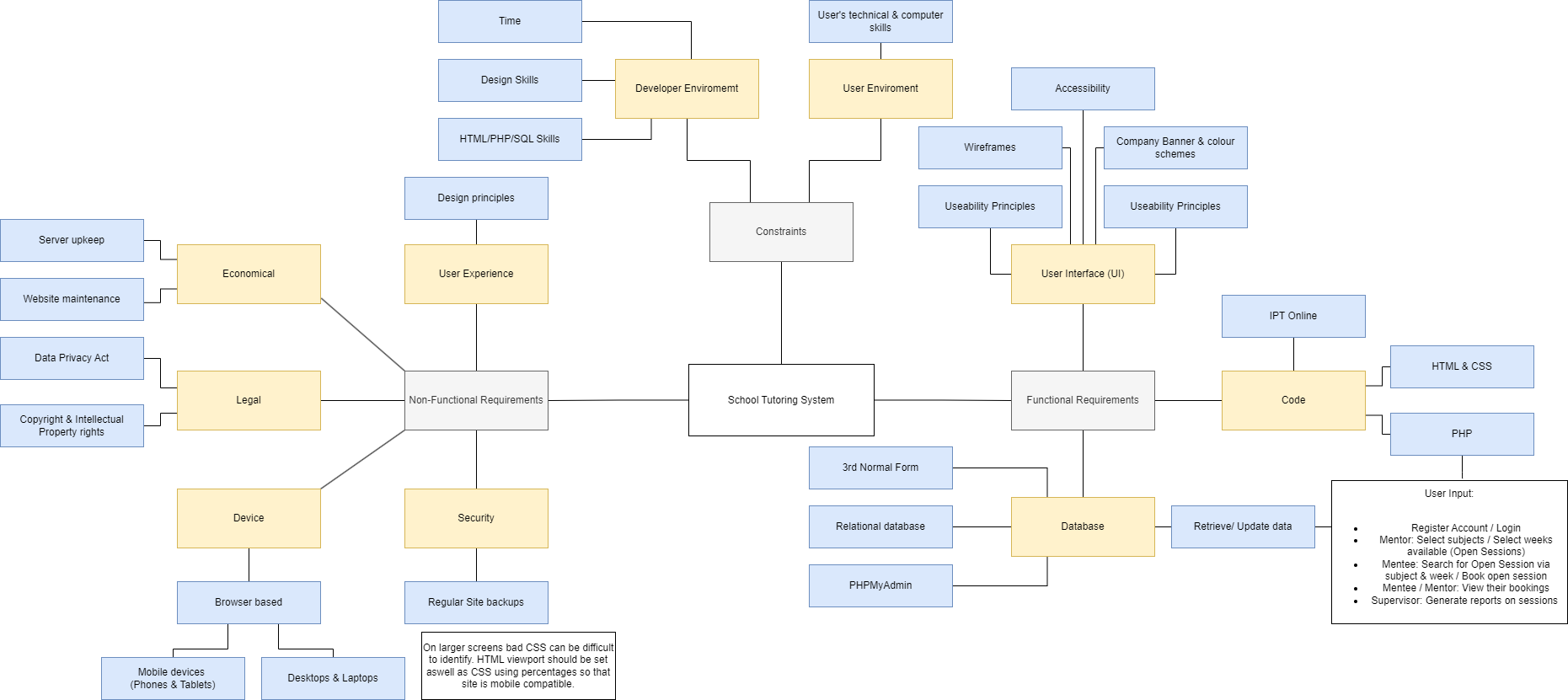
# EXPLORE

## Problem Identification:

A school requires an interactive web application to integrate a vertical academic tutoring process. This will centralise and simplify the tutoring process to one website, granting younger students (Mentees) greater opportunity to find and book sessions with older students (Mentors) for the subjects they require assistance with. Mentors need to be able to easily select the weeks they are available each term and the subjects they are able to competently tutor as well as be able to view all of their open sessions that have been booked and to view feedback provided by mentees. Mentees need to be able to search for open sessions for the subjects they need assistance with and then be able to view their future sessions as well as be able to provide feedback to their mentors. Supervisors need to be able to see all feedback and generate reports on the grade/ subject session ratios. This web application will be developed using HTML and CSS whilst using PHP with MySQL to communicate with the relational database built in PHPMyAdmin. This application has a login system, allowing students to manage and view their sessions and bookings from home as well as at school.

## Assumptions:

* The platform will be developed through IPTOnline using MySQL and PHPMyAdmin.
* There is one tutoring session each week (Monday Period 6 in the Peer Support Lesson).
* Students are at an age where they are competent in using web applications/ the internet/ technology and understand what they are doing. (Or have a respective guardian that can assist.)
* All users have a school email.
* Each tutoring session has only one mentor and one mentee.
* Students know/ remember which school week and term it is (EG: Term 4 Week 3).
* Both the mentors and mentees remember when and where the tutoring sessions are (and attend them).

## Criteria:

**Prescribed:**

By the due date, 8:30 am Monday 2nd October 2023:

* Designed and implemented an effective normalised relational database to store data and user inputs.
* Students can register as a mentee or mentor. (Supervisor accounts are preset to protect system safety and data integrity)
* Mentees can find tutoring sessions for their subject then book it and provide feedback.
* Mentors can select subjects to tutor and open sessions when they are available, and view provided feedback.
* Both Mentees and Mentors can view their future sessions
* Supervisors can view all open and booked sessions as well as reports for the number of sessions booked for subject/ year level combinations.

**Self-determined:**

By before the due date, on Friday 29th September 2023:

* (Ensure all prescribed criteria are met.)
* (Build a prototype login and register system that assists the criteria above.)
* Ensure all code is commented, indented, and appropriately used whitespace on all pages.
* Effectively use HTML and PHP to create a consistent website structure.
* Effectively use CSS to create a consistent website design and user experience UX.
* Appropriately use HTML and CSS to have a consistent and learnable UI and adhere to design principles.
* Use selection and iteration in PHP to produce efficient code.
* Implement measures to protect data integrity.
* Use PHP to interpret and display data retrieved from the SQL database.
* Use consistent variables and file naming conventions.
* Have a learnable user interface (UI) that is effective and accessible while maintaining an enjoyable user experience (UX)
* Personally understand how all of the PHP functions work to best utilise them efficiently

## Database fields:

userID, fName, sName, email, tel (op), grade, password, subjectID, subjectName, sessionID, menteeID, mentorID, year, term, week, booked, lvlID, lvlName, feedbackID, feedback.

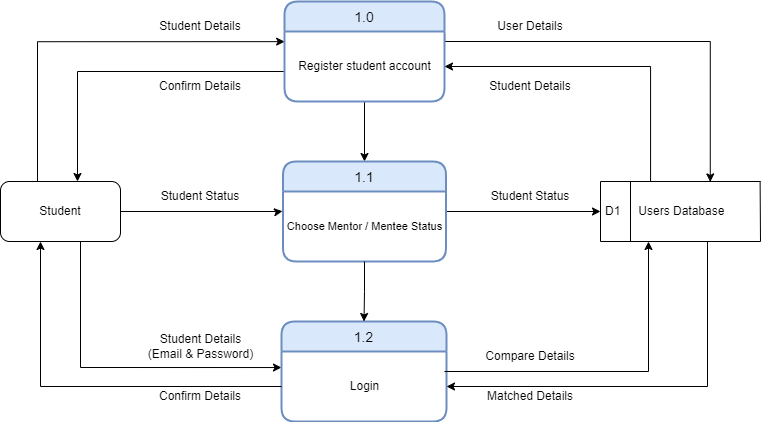
## Audience:

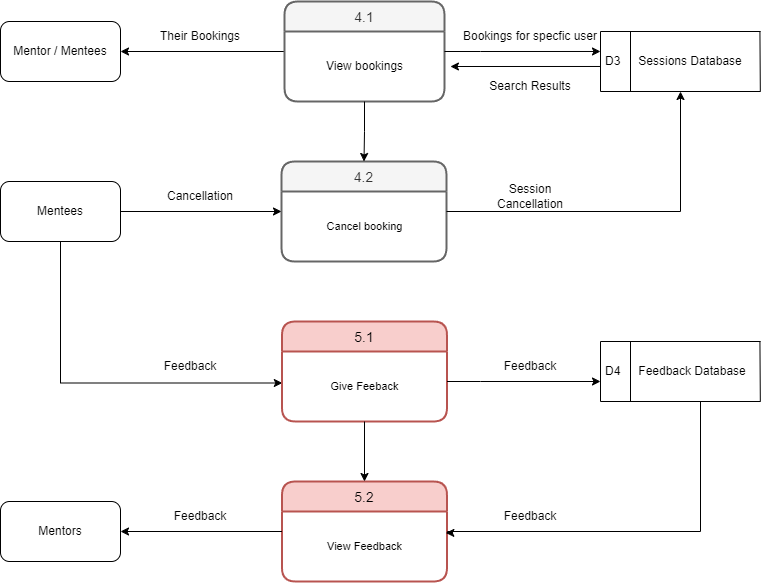
The users of this web application will be students who actively need tutoring and students who are able to tutor. There will also be system administrators (Supervisors) who have higher system privileges. Assistance may be needed for younger students. A simple UI is implemented for easy learnability and to maintain accessibility. Students will register as either a mentee or a mentor. Mentors will be able to select subjects they can tutor, and then open sessions for the weeks they are available. The mentees can then search for open sessions for the subjects they require tutoring for by week, and then book a session with a specific mentor.

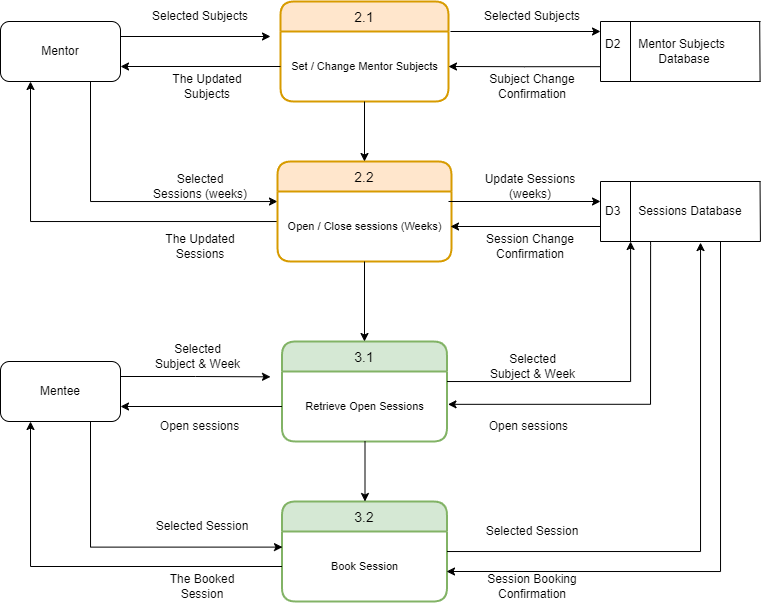
## Mind Map:

# Develop

## Data Flow Diagram (DFD)

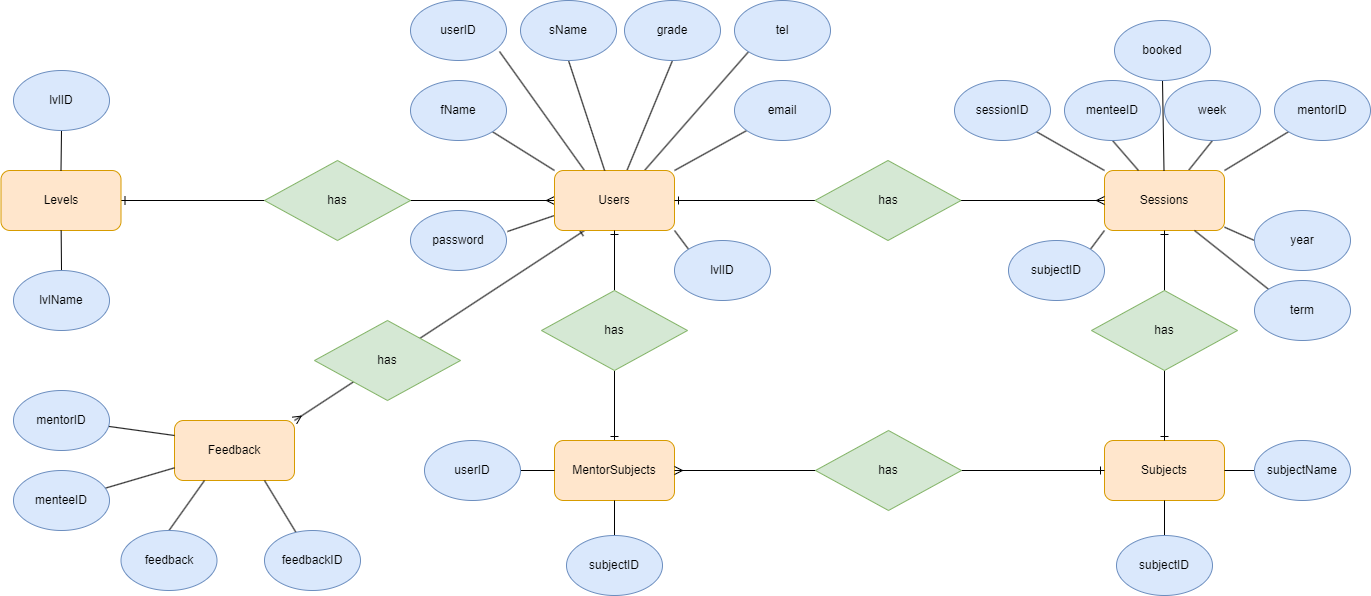






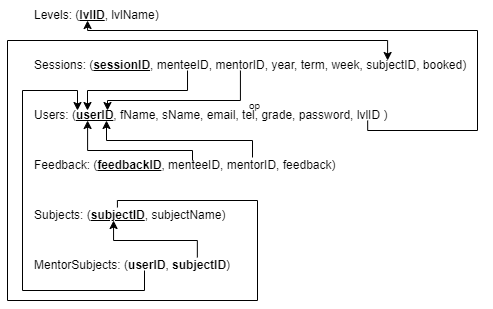
## Entity Relationship Diagram (ERD)

An entity relationship diagram (ERD) is used to display the relationships between the separate entities within the database. ERDs display the logical connections within the structure of the database.



## Relational Schema

A relational schema is used to display the direct foreign key to primary key connections between the columns of each table within the database. Column names that are bolded and underlined are primary keys, whilst the table with multiple bolded columns has a composite key.



## Data Dictionary

Users

|  |  |  |  |
| --- | --- | --- | --- |
| FIELD | DATA TYPE | VALIDATION RULES | EXAMPLE DATA |
| userID | Int(5) | Primary key, AUTO\_INCREMENT | 1 |
| fName | varchar(35) | NOT NULL | Riley |
| sName | varchar(35) | NOT NULL | Hampson |
| email | varchar(50) | NOT NULL | rhamp14@eq.edu.au |
| tel | int(10) |  | 04 1111 2222 |
| grade | int(2) | NOT NULL | 11 |
| password | varchar(35) | NOT NULL | Password |
| lvlID | int(1) | Foreign key,  NOT NULL | 1 |

Levels

|  |  |  |  |
| --- | --- | --- | --- |
| FIELD | DATA TYPE | VALIDATION RULES | EXAMPLE DATA |
| lvlID | int(1) | Primary key,  AUTO\_INCREMENT | 1 |
| lvlName | varchar(25) | NOT NULL | Mentee |

MentorSubjects

|  |  |  |  |
| --- | --- | --- | --- |
| FIELD | DATA TYPE | VALIDATION RULES | EXAMPLE DATA |
| userID | int(5) | Foreign key,  NOT NULL | 1 |
| subjectID | int(2) | Foreign key,  NOT NULL | 1 |

Subjects

|  |  |  |  |
| --- | --- | --- | --- |
| FIELD | DATA TYPE | VALIDATION RULES | EXAMPLE DATA |
| subjectID | int(5) | Foreign key,  NOT NULL | 1 |
| subjectName | int(2) | Foreign key,  NOT NULL | Digital Solutions |

Sessions

|  |  |  |  |
| --- | --- | --- | --- |
| FIELD | DATA TYPE | VALIDATION RULES | EXAMPLE DATA |
| sessionID | int(5) | Primary key,  AUTO\_INCREMENT | 5 |
| menteeID | int(5) | Foreign key | 1 |
| mentorID | int(5) | Foreign key,  NOT NULL | 5 |
| year | int(4) | NOT NULL | 2023 |
| term | int(1) | NOT NULL | 4 |
| week | int(2) | NOT NULL | 2 |
| subjectID | int(2) | Foreign key | 1 |
| booked | varchar(1) | NOT NULL | Y |

|  |  |  |  |
| --- | --- | --- | --- |
| FIELD | DATA TYPE | VALIDATION RULES | EXAMPLE DATA |
| feedbackID | int(5) | Primary key,  AUTO\_INCREMENT | 5 |
| menteeID | int(5) | Foreign key | 1 |
| mentorID | int(5) | Foreign key,  NOT NULL | 5 |
| feedback | varchar(2000) | NOT NULL | You explained the topic great! Please continue doing as you do. |

## Sample Tables

Users

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| userID | fName | sName | email | tel | grade | password | lvlID |
| 1 | Riley | Hampson | [rhamp14@eq.edu.au](mailto:rhamp14@eq.edu.au) | 04 1111 2222 | 11 | Password | 1 |
| 2 | John | Doe | [johndoe@mail.com](mailto:johndoe@mail.com) | 04 1234 5678 | 12 | !pA55w0Rd | 2 |
| 3 | Glen | Nixon | [gniox4@eq.edu.au](mailto:gniox4@eq.edu.au) | 04 8765 4321 | 0 | sir | 3 |

|  |  |
| --- | --- |
| subjectID | subjectName |
| 1 | Digital Solutions |
| 2 | Accounting |
| 3 | Chemistry |

Levels MentorSubjects Subjects

|  |  |
| --- | --- |
| lvlID | lvlName |
| 1 | Mentee |
| 2 | Mentor |
| 3 | Supervisor |

|  |  |
| --- | --- |
| userID | subjectID |
| 2 | 1 |
| 4 | 3 |
| 5 | 1 |

Sessions Feedback

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| sessionID | menteeID | mentorID | year | term | week | subjectID | booked |
| 1 | 1 | 2 | 2023 | 4 | 1 | 2 | Y |
| 2 | 1 | 2 | 2023 | 4 | 2 | 2 | Y |
| 3 | 4 | 5 | 2023 | 4 | 5 | 1 | Y |
| 4 | *NULL* | 2 | 2023 | 4 | 3 | *NULL* | N |

|  |  |  |  |
| --- | --- | --- | --- |
| feedbackID | menteeID | mentorID | feedback |
| 1 | 1 | 2 | Good job! |
| 2 | 6 | 4 | You could try explaining the topics in a different way. |
| 3 | 7 | 5 | Amazing job |

## Normalisation

Normalisation is a decomposition system for databases that eliminates data redundancies and assists when inserting, deleting, and updating anomalies (Richard Peterson, 2019). A database that is normalised will reduce human error thus preserving data integrity across the system.

**First normal form (1NF)**

For this task, 1NF has been met. This is because all data has been verified atomic across all tables. The attribute values are singular and have been separated.

**Second normal form (2NF)**

Before moving to 2NF, 1NF must be compliant. For this task, 2NF has been met. For 2NF all non-key fields must be fully dependant on the designated primary key. Where it is practical, there are primary keys (Users = userID, Levels = lvlID, Sessions = sessionID, feedback= feedbackID, Subjects = subjectID) whilst for the MentorSubjects table there is a composite key because the two columns used will never produce duplicate results, rendering an extra column useless and wasteful.

**Third normal form (3NF)**

Before moving to 3NF, 2NF must be compliant. For this task, 3NF has been met. For 3NF it is further checking that all non-key fields are fully dependent on their respective primary key. It also recognises the practicality such as EG: not splitting the users into three different categories (mentee, mentor, supervisor) or splitting sessions into open and booked. EG: in Users it is easier to keep the value “1” instead of “Mentor” every time there is a mentor.

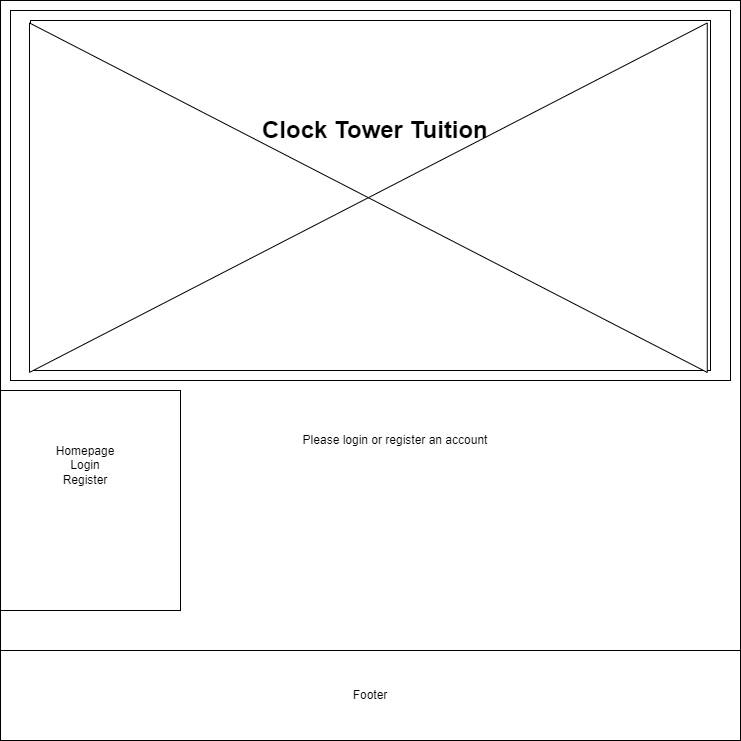
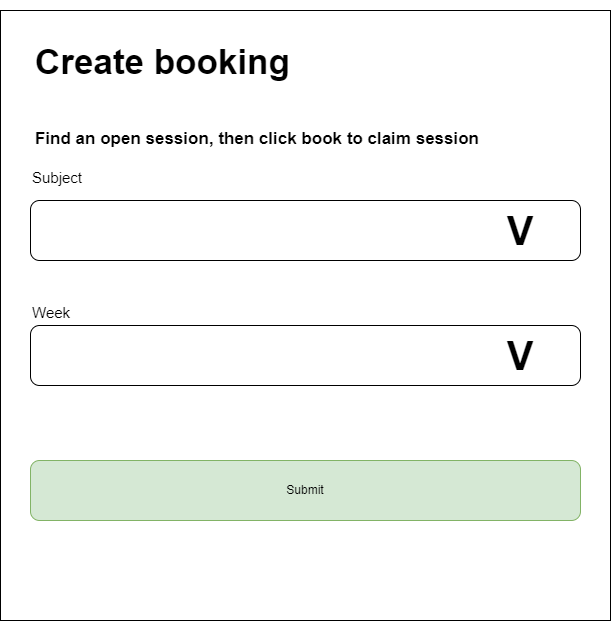
# User Interface

## Wireframes

The main homepage will contain a large prominent banner image containing the school institute along with the school’s name. Consequent form and action pages will have a dark grey banner with white text that matches the dynamic footer instead, whilst the nav section and article spaces will be lighter greys with dark text for contrast, assisting readability through design principles. The grey theme creates an easy formal mood, whilst being visually appealing.

Headings will be large and simple and should be mostly self-explanatory as to the purpose of the page. An appropriate amount of white space will be used to ensure the screen does not feel crowded and messy.

The header, footer and nav files will be separate files that are incorporated into the page using the “include ‘header.php’;” function. This limits repetition and allows for easy editing,



Drop-down menus are used along with php for ease of maintenance, meaning if a new subject is added, it doesn’t have to be manually added to the HTML form, instead it is automatically added due to the SQL. Drop-down menus are fixed, ensuring data integrity.

Placeholders are added as a guide to assist with data accuracy, reliability and again integrity.

Form attributes are also used, such as in the email, phone, and grade form. EG: An email must have a @ symbol, the phone number must be 8-10 digits and grade is limited from 1-12.

The banner will have ALT added for the visually impaired for accessibility.

Both the Nav links and homepage text are dynamic, changing when you login, changing with account types and when you start booking/ have booked sessions.

The footer is also dynamic, sitting at the bottom of the screen if the body section is smaller than the screen, and moves to the bottom of the page if the body section is larger than the screen.

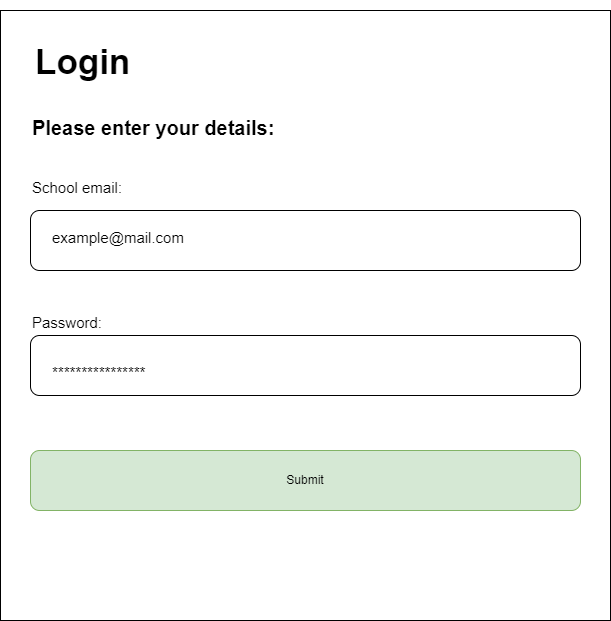
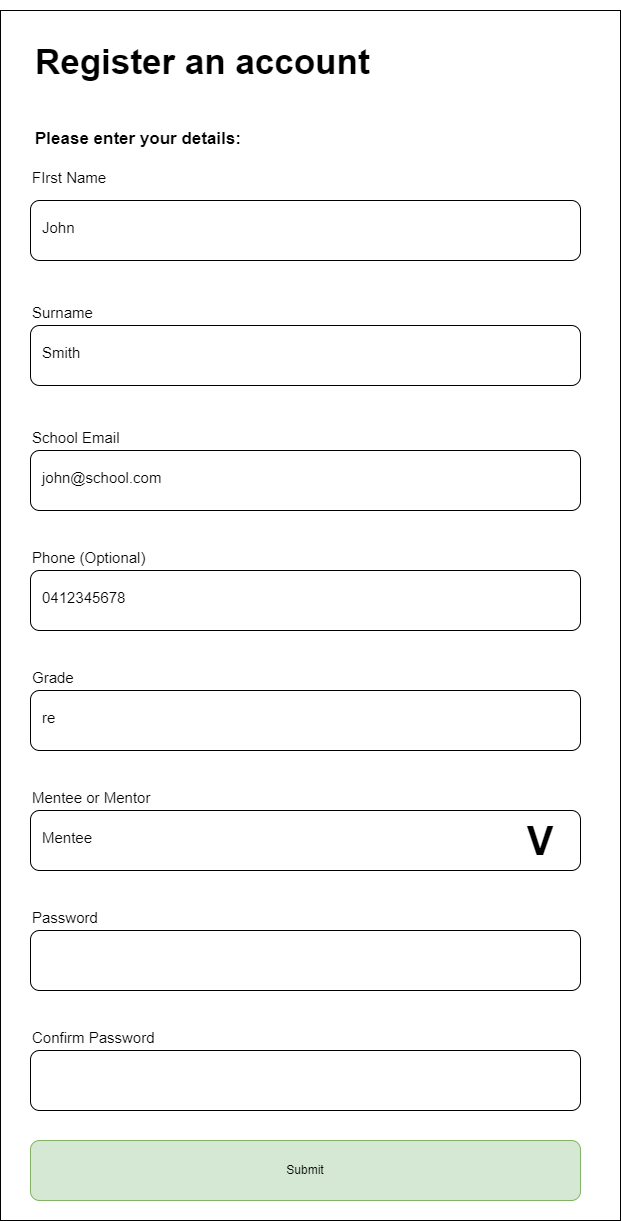
Manipulation of hierarchy to ensure easy learnability. A large banner with large text, a left-position Nav bar and a content page and footer at the bottom. These make it easy to navigate and remember.

## Form layouts

Forms will be comprised of textboxes and dropdown menus where appropriate and needed.

The buttons will consistently be green “Submit” buttons, placed below the form.

Labels and headings will have consistent alignment, sizing, and spacing.



## Algorithm – Input, Process, Output Charts (IPO)

register.php viewBookings.php

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| fName, sName, email, tel (op), grade, Mentee or Mentor, password | Algorithm to register account  **IF** submit has been pressed  **IF** password1 = password2  **SET** DbRequest TO  **INSERT INTO** Users  **EXECUTE** DbRequest storing results as Users  **FOR EACH** field of new owner details  **IF** field is valid  **SAVE** field in Users  **ELSE**  **OUTPUT** Error  **ENDIF**  **ENDFOR** | Confirmation or error message |

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| userID | Algorithm to view booking for that user  **IF** var count > 0  **SET** DbRequest TO  **SELECT** s.sessionID, u.fName, u.sName, u.grade, u.email, su.subjectName, s.year, s.term, s.week  **FROM** Sessions s  **JOIN** Users u ON s.mentorID = u.userID  **JOIN** Subjectssu ON s.subjectID = su.subjectID  **WHERE s**.menteeID = '$userID'  **EXECUTE** DbRequest  **IF** inputs are valid  **OUTPUT** table of booked sessions  **ELSE**  **OUTPUT** “Go to create bookings to create bookings to claim a sesson!” | Table of booked sessions for that specific user |

Booking2.php

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| subjectID, week | Algorithm to display free sessions so mentee can book them  **IF** post “Submit” isset  **SET** DbRequest TO  **SELECT** s.sessionID, u.fName, u.sName, u.email, subj.subjectName  **FROM** Sessions s  **INNER JOIN** Users u ON s.mentorID = u.userID  **INNER JOIN** MentorSubjects ms ON u.userID = ms.userID  **INNER JOIN** Subjects subj ON ms.subjectID = subj.subjectID  **WHERE** s.booked = 'N' AND subj.subjectID = Inputted SubjectID AND s.week = inputted week  **EXECUTE** DbRequest  **IF** inputs are valid  **OUTPUT** table of open sessions  **ELSE**  **OUTPUT** “No open Session Message” | Table of open sessions |

# Generate

## SQL Create Tables

CREATE TABLE Levels (

lvlID int(1) AUTO\_INCREMENT,

lvlName varchar(25) NOT NULL,

PRIMARY KEY (lvlID)

);

CREATE TABLE Users (

userID int(5) AUTO\_INCREMENT,

fName varchar(35) NOT NULL,

sName varchar(35) NOT NULL,

email varchar(50) NOT NULL,

tel int(10),

grade int(2) NOT NULL,

password varchar(35) NOT NULL,

lvlID int(1) NOT NULL,

PRIMARY KEY (userID),

FOREIGN KEY(lvlID) REFERENCES Levels(lvlID)

);

CREATE TABLE Subjects(

subjectID int(2) AUTO\_INCREMENT,

subjectName varchar(25) NOT NULL,

PRIMARY KEY (subjectID)

);

CREATE TABLE Sessions(

sessionID int(5) AUTO\_INCREMENT,

menteeID int(5),

mentorID int(5) NOT NULL,

year int(4) NOT NULL,

term int(1) NOT NULL,

week int(2) NOT NULL,

subjectID int(2),

booked varchar(1) NOT NULL,

PRIMARY KEY (sessionID),

FOREIGN KEY(menteeID) REFERENCES Users(userID),

FOREIGN KEY(mentorID) REFERENCES Users(userID),

FOREIGN KEY(subjectID) REFERENCES Subjects(subjectID)

);

CREATE TABLE MentorSubjects(

userID int(5) NOT NULL,

subjectID int(2) NOT NULL,

PRIMARY KEY (userID, subjectID),

FOREIGN KEY(userID) REFERENCES Users(userID),

FOREIGN KEY(subjectID) REFERENCES Subjects(subjectID)

);

CREATE TABLE Feedback(

feedbackID int(5) AUTO\_INCREMENT,

mentorID int(5) NOT NULL,

menteeID int(5) NOT NULL,

feedback varchar(2000) NOT NULL,

PRIMARY KEY (feedbackID),

FOREIGN KEY(mentorID) REFERENCES Sessions(mentorID),

FOREIGN KEY(menteeID) REFERENCES Sessions(menteeID)

);

## SQL Inserting data

INSERT INTO Users(fName, sName, email, grade, password, lvlID)

VALUES ('Riley', 'Hampson', 'rhamp14@eq.edu.au', 11, 'pass', 1);

INSERT INTO Users(fName, sName, email, grade, password, lvlID)

VALUES ('Kaden', 'Cameron', kcame84@eq.edu.au', 11, 'pass', 2);

INSERT INTO Users(fName, sName, email, grade, password, lvlID)

VALUES ('Glen', 'Nixon', gniox4@eq.edu.au', 0, 'pass', 3);

INSERT INTO Subjects(subjectName)

VALUES ('Digital Solutions');

INSERT INTO Subjects(subjectName)

VALUES ('Accounting');

INSERT INTO Subjects(subjectName)

VALUES ('Maths');

INSERT INTO Levels(lvlName)

VALUES ('Mentee');

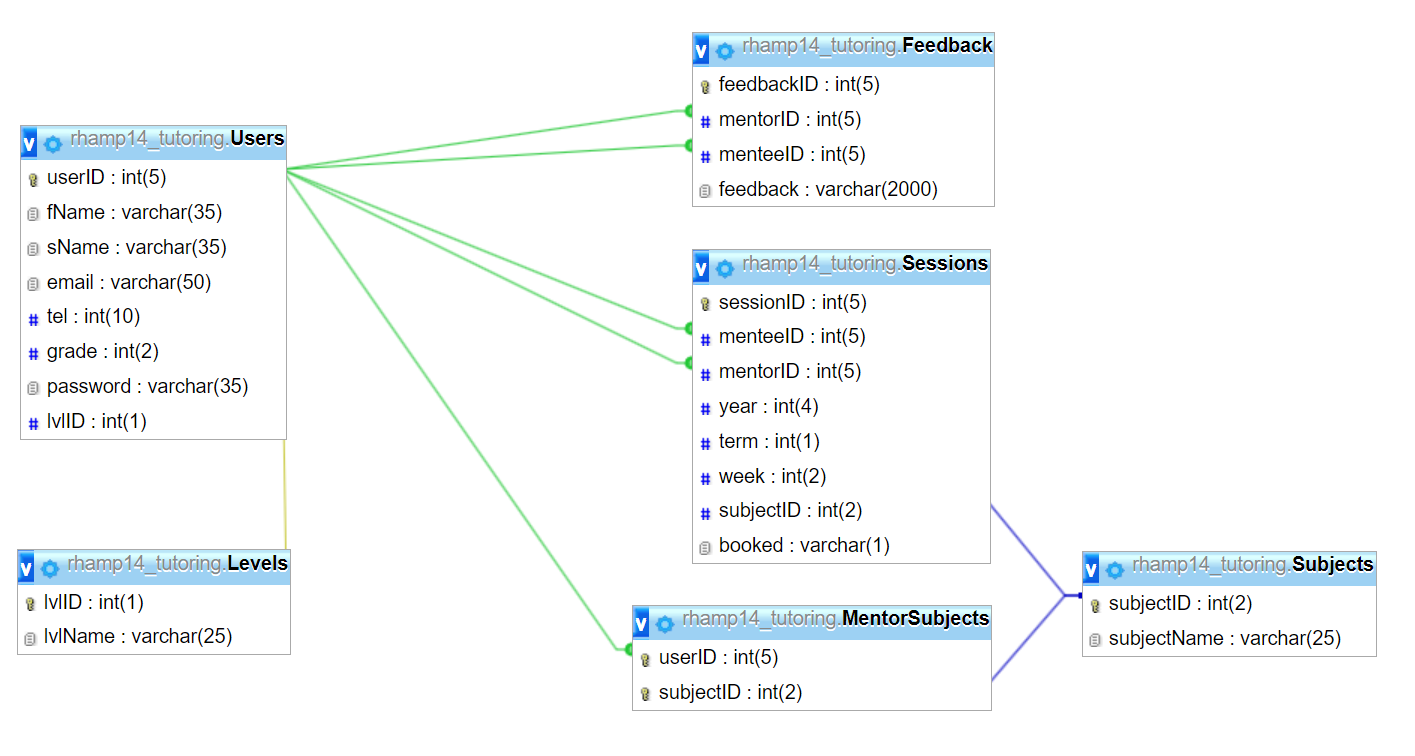
INSERT INTO Levels(lvlName)

VALUES ('Mentor');

INSERT INTO Levels(lvlName)

VALUES ('Supervisor');

## Database Design



For ID fields auto-increment is used to generate a primary, this ensures each row of data is unique, complying with the normalisation standards.

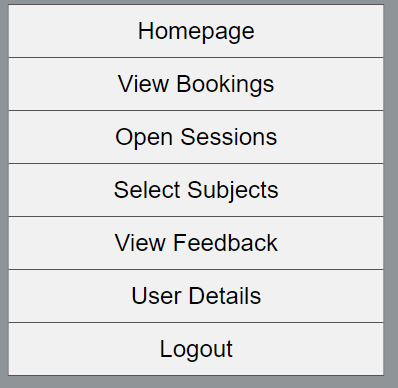
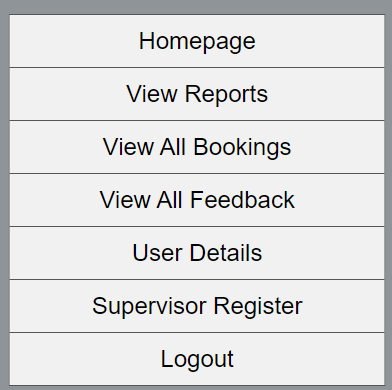
Otherwise, a composite key is used.

The tables must be executed in this order due to foreign key restraints. You can’t create a table that references another column if it doesn’t exist yet.

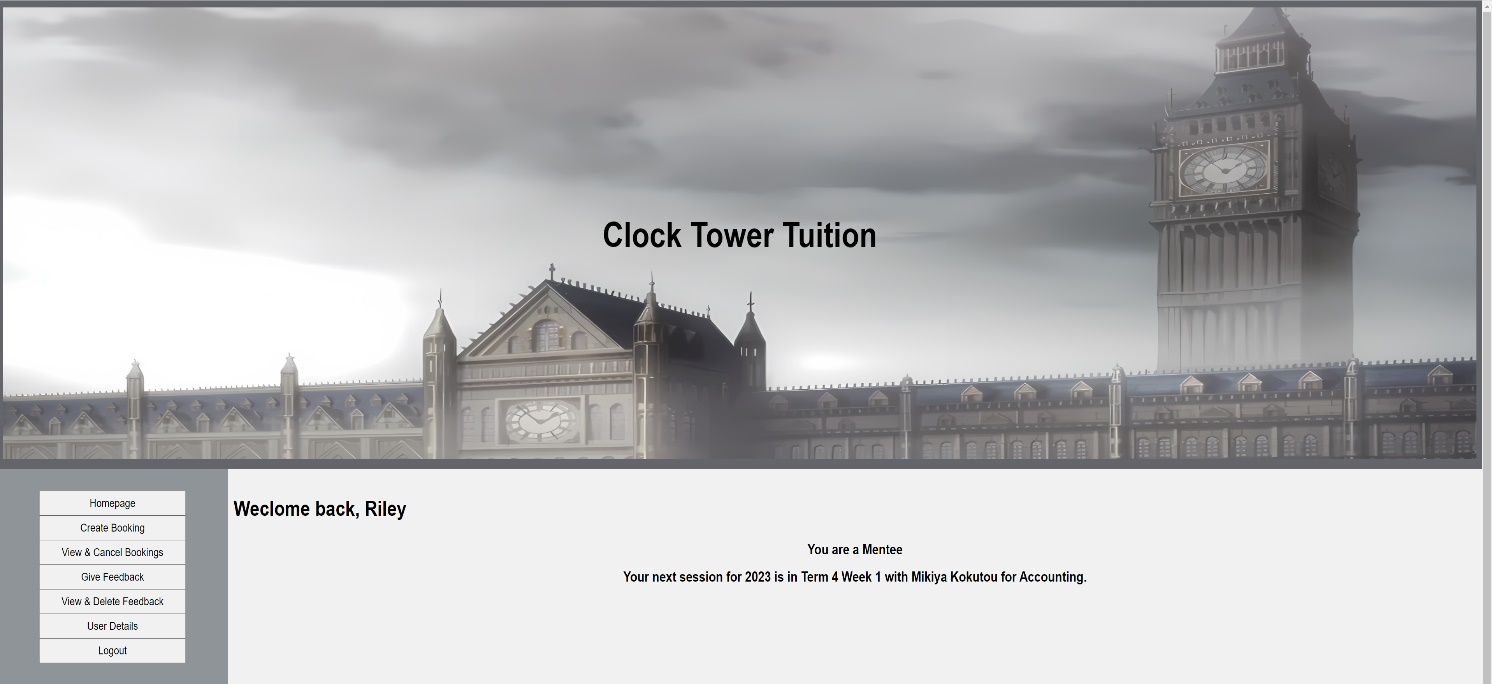
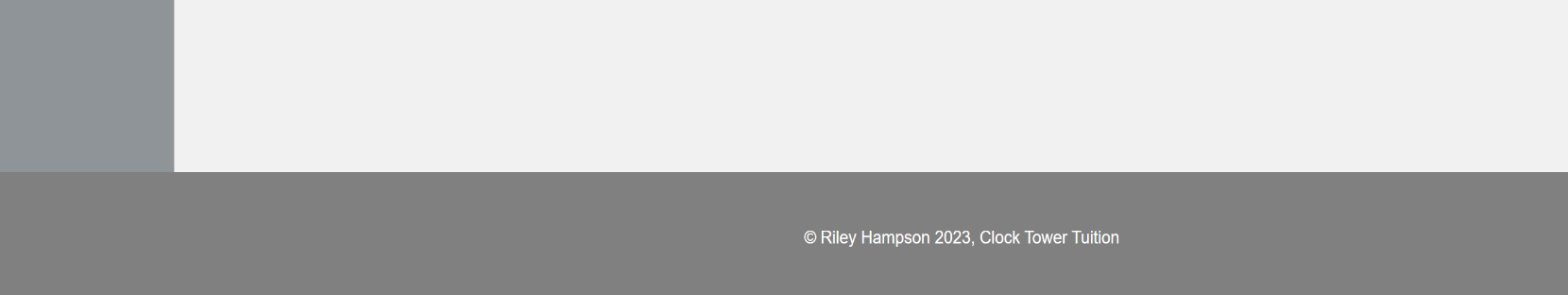
## A screenshot of a computer Description automatically generatedUser interface designs

A building with a clock tower

Description automatically generated

A screenshot of a menu

Description automatically generatedA screenshot of a computer

Description automatically generated

Dynamic nav section depending on user level (Mentee, Mentor, Supervisor). Use of hierarchy on the Nav bar to ensure the most used pages are at the top, and related groups are next to each other. This makes it easier for the user to learn and remember.

Repetitive design for improved learnability.

Simple instructions on all forms, placeholders where necessary.

Use of hierarchy for useability. Checkboxes when selecting fixed items for simplicity.

Drop-down menus for data integrity.

Responsive footer, when the content doesn’t take a full page, it is stretched so the footer sits at the bottom. When the content takes up more than the viewport, it sticks to the bottom. Use of contrasting colours that match the theme are also used here.

Dark text on light background for contrast, easier to read. Simple and familiar layout for easy learnability.

Dynamic homepage message and Menu. Use of hierarchy for easy learnability.

Menu is in Nav section, ensuring its always easy to access on any device.

## PHP, HTML, & CSS

### connect.php

<?php

//0 = Not logged in

//1 = Logged in

//Main check data and retrieve data script

if (isset($\_SESSION["LoggedIn"])) {

if ($\_SESSION["LoggedIn"] == 0 ) { //If User isn't logged in

} elseif ($\_SESSION["LoggedIn"] == 1 ) {

if (isset($\_SESSION["fName"])) {

if (!$\_SESSION["fName"] == "" ) {

}

} elseif (!isset($\_SESSION["fName"])) {

$\_SESSION["fName"] = "";

}

}

} elseif (!isset($\_SESSION["LoggedIn"])) { //IF Var isn't set

$\_SESSION["LoggedIn"] = 0;

echo '<script type="text/javascript"> window.location = "index.php" </script>';

}

//DB Connect

$servername = "localhost";

$dBUsername = "rhamp14";

$dBPassword = "Prussia.4";

$dBName = "rhamp14\_tutoring";

$conn = mysqli\_connect($servername, $dBUsername, $dBPassword, $dBName);

if(!$conn) {

die ("connection failed: ".mysqli\_connect\_error());

} ?>

The connect files is used to connect the website to the SQL database. This removes unnecessary redundancy by having the connection file linked to all pages on the site.

### header.php

<!DOCTYPE html>

<html lang="en">

<head>

<title>Clock Tower Tuition</title>

<meta HTTP-EQUIV="Pragma" CONTENT="no-cache">

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet" href="style.css?ver=1.51">

</head>

<body>

<header>

<div class="con1">

<img src="logo.png" alt="Logo">

<div class="center1"><h5>Clock Tower Tuition</h5></div>

</div>

</header>

### nav.php

<!--Section section-->

<section

<!--Nav section-->

<nav>

<ul>

<li><a href="index.php">Homepage</a></li>

<?php

if (isset($\_SESSION["LoggedIn"])) {

if ($\_SESSION["LoggedIn"]==0) {

echo'<li><a href="login.php">Login</a></li>';

echo'<li><a href="register.php">Register</a></li>';

} elseif ($\_SESSION["LoggedIn"]==1) { //Mentee

if ($\_SESSION['lvlID']==1) {

echo'<li><a href="booking.php">Create Booking</a></li>';

echo'<li><a href="viewBookings.php">View & Cancel Bookings</a></li>';

echo'<li><a href="giveFeedback.php">Give Feedback</a></li>';

echo'<li><a href="viewFeedback.php">View & Delete Feedback</a></li>';

echo'<li><a href="userDetails.php">User Details</a></li>';

echo'<li><a href="kill.php">Logout</a></li>';

} elseif ($\_SESSION['lvlID']==2) { //Mentor

echo'<li><a href="viewBookings.php">View Bookings</a></li>';

echo'<li><a href="openSessions.php">Open Sessions</a></li>';

echo'<li><a href="mentorSubjects.php">Select Subjects</a></li>';

echo'<li><a href="viewFeedback.php">View Feedback</a></li>';

echo'<li><a href="userDetails.php">User Details</a></li>';

echo'<li><a href="kill.php">Logout</a></li>';

} elseif ($\_SESSION['lvlID']==3) { //Supervisor

echo'<li><a href="reports.php">View Reports</a></li>';

echo'<li><a href="viewBookings.php">View All Bookings</a></li>';

echo'<li><a href="viewFeedback.php">View All Feedback</a></li>';

echo'<li><a href="userDetails.php">User Details</a></li>';

echo'<li><a href="registerAdmin.php">Supervisor Register</a></li>';

echo'<li><a href="kill.php">Logout</a></li>';

}

}

} else { //Not logged In

echo'<li><h1>Please refresh the page!</h1></li>';

}

?>

</ul>

</nav>

### footer.php

### <!--Footer section-->

### <footer>

### <p>© Riley Hampson <script>document.write(new Date().getFullYear())</script>, Clock Tower Tuition</p>

</footer>

Separating the Connect, header, Nav, and footer page enables easy editing without having to go edit multiple pages.

### index.php

<?php

session\_start(); include 'header.php'; include 'nav.php';

require 'connect.php'; ?>

<!--Article section-->

<article>

<?php

if ($\_SESSION["LoggedIn"]==0) {

echo"<h1>Welcome to the Clock Tower Tuition Website!</h1>";

echo"<h2>Please register an account or login</h2>";

} elseif ($\_SESSION["LoggedIn"]==1) {

echo"<h1>Weclome back, ".$\_SESSION['fName']."</h1>";

if ($\_SESSION['lvlID']==1) {

echo"<h2>You are a Mentee</h2>"; //Statement

$userID=$\_SESSION["userID"]; //Assign Vars

$sql2 = "SELECT sessionID FROM Sessions WHERE menteeID = '$userID'"; //SQL for count

$result2 = mysqli\_query($conn, $sql2);

if($result2){ $count = mysqli\_num\_rows($result2); } else { $count = 0; }

if ($count > 0) {

//This SQL is for the table showing the future bookings

$sql = "SELECT u.fName, u.sName, su.subjectName, s.year, s.term, s.week FROM Sessions s JOIN Users u ON s.mentorID = u.userID JOIN Subjects su ON s.subjectID = su.subjectID

WHERE s.menteeID = '$userID' LIMIT 1";

$result = mysqli\_query($conn, $sql);

while($row = mysqli\_fetch\_assoc($result)){ // Print Message

echo"<h2>Your next session for {$row['year']} is in Term {$row['term']} Week {$row['week']} with {$row['fName']} {$row['sName']} for {$row['subjectName']}.</h2>";

}

} else {

echo"<h2>You have no sessions booked!</h2>";

echo"<h2>Go to 'Create Booking' to book a session!</h2>";

}

} elseif ($\_SESSION['lvlID']==2) {

echo"<h2>You are a Mentor</h2>";

$userID=$\_SESSION["userID"]; //Assign Vars

$sql3 = "SELECT userID FROM MentorSubjects WHERE userID = '$userID'"; //SQL for count

$result3 = mysqli\_query($conn, $sql3);

if($result3){ $count2 = mysqli\_num\_rows($result3); } else { $count2 = 0; }

if ($count2 > 0) {

$sql2 = "SELECT sessionID FROM Sessions WHERE mentorID = '$userID'"; //SQL for count

$result2 = mysqli\_query($conn, $sql2);

if($result2){ $count = mysqli\_num\_rows($result2); } else { $count = 0; }

if ($count > 0) {

//This SQL is for the table showing the future bookings

$sql = "SELECT u.fName, u.sName, su.subjectName, s.year, s.term, s.week FROM Sessions s JOIN Users u ON s.menteeID = u.userID JOIN Subjects su ON s.subjectID = su.subjectID

WHERE s.mentorID = '$userID' LIMIT 1";

$result = mysqli\_query($conn, $sql);

while($row = mysqli\_fetch\_assoc($result)){ // Print Message

echo"<h2>Your next session for {$row['year']} is in Term {$row['term']} Week {$row['week']} with {$row['fName']} {$row['sName']} for {$row['subjectName']}.</h2>";

}

} else {

echo"<h2>You have not opened any sessions!</h2>";

echo"<h2>Go to 'Open Sessions' to open sessions which you can tutor!</h2>";

}

} else {

echo"<h2>You have not selected the subjects you can mentor!</h2>";

echo"<h2>Go to 'Select Subjects' to select the subjects you can tutor!</h2>";

}

} elseif ($\_SESSION['lvlID']==3) {

echo"<h2>You are a Supervisor</h2>";

} else {

echo"<h2>You are a God, apparently, cos something is broken and you have surpassed the highest of admins!</h2>";

}

} else {

echo"<h1>Welcome to the Clock Tower Tuition Website!</h1>";

}

?>

</article></section></div><?php include 'footer.php'; ?> </div></body></html>

The SQL Row Count function is used to check if Mentors have selected the subjects and/ or opened any sessions. This allowes the dynamic message to remind them to select them.

### openSessions.php

<?php session\_start(); include 'header2.php'; include 'nav.php'; require 'connect.php';

//Assign Vars

$userID = $\_SESSION['userID'];

$term = 4; //Term & Year Set in var for when an actual menu to select them is chosen

$year = 2023;

//SQL Query

$sql = "SELECT week, booked FROM Sessions WHERE year = '".$year."' AND term = '".$term."' AND mentorID = '".$userID."' ";

$result = mysqli\_query($conn, $sql);

//Assign to array

$selectedWeeks = array();

$booked = array();

while ($row = mysqli\_fetch\_assoc($result)) {

$selectedWeeks[] = $row['week'];

$booked[$row['week']] = $row['booked'];

}

?>

<!--Article section-->

<article >

<h1>Please select the weeks you can tutor in Term 4 2023!</h1><br><br>

<form name="submit" method="post" action="openSessions.php" class="center"> <!--Main Form-->

<input type="checkbox" id="1" name="weekCB[]" value="1" <?php if (in\_array(1, $selectedWeeks) && isset($booked[1]) && $booked[1] == 'N') echo 'checked'; if (isset($booked[1]) && $booked[1] == 'Y') echo 'disabled'; ?>>

<label for="1">Week 1 <?php if (isset($booked[1]) && $booked[1] == 'Y') echo '(This session has been booked, you CANNOT unselect it.)'; ?> </label><br>

<input type="checkbox" id="2" name="weekCB[]" value="2" <?php if (in\_array(2, $selectedWeeks) && isset($booked[2]) && $booked[2] == 'N') echo 'checked'; if (isset($booked[2]) && $booked[2] == 'Y') echo 'disabled'; ?>>

<label for="2">Week 2 <?php if (isset($booked[2]) && $booked[2] == 'Y') echo '(This session has been booked, you CANNOT unselect it.)'; ?> </label><br>

\*\*Checkbox 3-10 have been deleted to conserve space\*\*

<input name="submit" type="submit" value="Submit" class="btn">

</form>

<?php

if (isset($\_POST['submit'])) {

$weekCB = $\_POST['weekCB'] ;

$userID = $\_SESSION['userID'];

//SQL Query

$sql = "SELECT week FROM Sessions WHERE year = '".$year."' AND term = '".$term."' AND mentorID = '".$userID."'";

$result = mysqli\_query($conn, $sql);

//Assign to array

$selectedWeeks = array();

while ($row = mysqli\_fetch\_assoc($result)) {

$selectedWeeks[] = $row['week'];

}

// Check if a week is selected or unselected

foreach ($selectedWeeks as $selectedWeek) {

if (in\_array($selectedWeek, $weekCB)) { //Subject the same, continue

continue;

} else { // Week is unselected, remove it from the database

$deleteSql = "DELETE FROM Sessions WHERE year = '".$year."' AND term = '".$term."' AND mentorID = '".$userID."' AND week = '".$selectedWeek."'";

mysqli\_query($conn, $deleteSql);

echo '<script type="text/javascript">window.location = "openSessions.php"</script>';

}

}

// Insert newly selected weeks into the database

foreach ($weekCB as $week) {

if (!in\_array($week, $selectedWeeks)) {

$insertSql = "INSERT INTO Sessions (mentorID, year, term, week, booked) VALUES ('".$userID."', '".$year."', '".$term."', '".$week."', 'N')";

mysqli\_query($conn, $insertSql);

echo '<script type="text/javascript">window.location = "openSessions.php"</script>';

}

}

echo "<h2>Available weeks to tutor updated!</h2>";

}

?>

</article></section></div>

<?php include 'footer.php'; ?> </div></body></html>

openSessions.php uses arrays and for each loops to convert checkbox inputs into sql data. This script creates sessions, then can delete them if they HAVEN’T been booked already,

### mentorSubjects.php

<?php session\_start(); include 'header2.php';

include 'nav.php'; require 'connect.php';

//Assign Vars

$userID = $\_SESSION['userID'];

//SQL Query

$sql = "SELECT subjectID FROM MentorSubjects WHERE userID = '".$userID."'";

$result = mysqli\_query($conn, $sql);

//Assign to array

$selectedSubjects = array();

while ($row = mysqli\_fetch\_assoc($result)) {

$selectedSubjects[] = $row['subjectID'];

}

?>

<!--Article section-->

<article >

<h1>Please select the subjects you can tutor!</h1><br><br>

<form name="submit" method="post" action="mentorSubjects.php" class="center"> <!--Main Form-->

<input type="checkbox" id="1" name="subjectCB[]" value="1" <?php if (in\_array(1, $selectedSubjects)) echo 'checked'; ?>>

<label for="1">Digital Solutions</label><br>

\*\*Deleted 1-5 due to code being the same to conserve space \*\*

<input type="checkbox" id="6" name="subjectCB[]" value="6" <?php if (in\_array(6, $selectedSubjects)) echo 'checked'; ?>>

<label for="6">Chemistry</label><br><br><br>

<input name="submit" type="submit" value="Submit" class="btn">

</form>

<?php

if (isset($\_POST['submit'])) {

$subjectCB = $\_POST['subjectCB'] ;

$userID = $\_SESSION['userID'];

//SQL Query

$sql = "SELECT subjectID FROM MentorSubjects WHERE userID = '".$userID."'";

$result = mysqli\_query($conn, $sql);

//Assign to array

$selectedSubjects = array();

while ($row = mysqli\_fetch\_assoc($result)) {

$selectedSubjects[] = $row['subjectID'];

}

// Check if a subject is selected or unselected

foreach ($selectedSubjects as $selectedSubject) {

if (in\_array($selectedSubject, $subjectCB)) { //Subject the same, continue

continue;

} else { // Subject is unselected, remove it from the database

$deleteSql = "DELETE FROM MentorSubjects WHERE userID = '".$userID."' AND subjectID = '".$selectedSubject."'";

mysqli\_query($conn, $deleteSql);

echo '<script type="text/javascript">window.location = "mentorSubjects.php"</script>';

}

}

// Insert newly selected subjects into the database

foreach ($subjectCB as $subjectID) {

if (!in\_array($subjectID, $selectedSubjects)) {

$insertSql = "INSERT INTO MentorSubjects (userID, subjectID) VALUES ('".$userID."', '".$subjectID."')";

mysqli\_query($conn, $insertSql);

echo '<script type="text/javascript">window.location = "mentorSubjects.php"</script>';

}

}

echo "<h2>Subjects Selections Updated</h2>";

} ?>

</article></section></div>

<?php include 'footer.php'; ?> </div></body></html>

mentorSubjects.php allowes mentors to select which subjects they want to tutor. Similar to openSessions.php, this uses arrays to input checkboxes into the sql database.

### booking.php

<?php session\_start(); include 'header2.php';

include 'nav.php'; require 'connect.php'; ?>

<!--Article section-->

<article >

<h1>Please select the Subject and Week (Term 4 2023) </h1><br><br>

<form name="submit" method="post" action="booking2.php">

<?php //Drop Down List

$sql = "SELECT DISTINCT subjectID, subjectName FROM Subjects"; //SQL Query

$result = mysqli\_query($conn, $sql);

echo "Subject: ";

echo "<select name='subjectID'>";

while ($row = mysqli\_fetch\_assoc($result)){

$subjectID = $row['subjectID'];

$subjectName = $row['subjectName'];

echo '<option value="'.$subjectID.'">'.$subjectName.'</option>';

}

echo "</select>";

?>

<br><br>

Week:

<select name="week">

<option value="1"> Week 1</option>

\*\*2-9 Removed for Space\*\*

<option value="10"> Week 10</option>

</select><br><br>

<input name="submit" type="submit" value="Submit"><br/><br/>

</form><br><br>

</article></section></div>

<?php include 'footer.php'; ?> </div></body></html>

### booking2.php

<?php session\_start(); include 'header2.php';

include 'nav.php'; require 'connect.php'; ?>

<!--Article section-->

<article >

<h1>Please select the subject you would like tutoring for! </h1><br><br>

<?php

if (isset($\_POST['submit'])) {

$userID=$\_SESSION["userID"]; //Assign User Vars

$subjectID = $\_POST['subjectID']; //Assign Posts

$week = $\_POST['week'];

//SQL

$sql = "SELECT s.sessionID, u.fName, u.sName, u.email, subj.subjectName

FROM Sessions s

INNER JOIN Users u ON s.mentorID = u.userID

INNER JOIN MentorSubjects ms ON u.userID = ms.userID

INNER JOIN Subjects subj ON ms.subjectID = subj.subjectID

WHERE s.booked = 'N' AND subj.subjectID = '".$subjectID."' AND s.week = '".$week."' ";

$result = mysqli\_query($conn, $sql);

echo"Sub: ".$subjectID." ";

echo"Week: ".$week." ";

//Print Table

echo "<br>";

echo "<table border='1'>";

echo "<tr><th>SessionID</th><th>Mentor Name</th><th>Mentor Email</th><th>Subject</th><th>Button</th></tr>"; //Table titles

while($row = mysqli\_fetch\_assoc($result)){

echo "<tr>

<td>{$row['sessionID']}</td>

<td>{$row['fName']} {$row['sName']}</td>

<td>{$row['email']}</td>

<td>{$row['subjectName']}</td>

<td>

<form method='post' action='bookSession.php'>

<input type='hidden' name='sessionID' value='".$row['sessionID']."'>

<input type='hidden' name='subjectID' value='".$subjectID."'>

<input type='hidden' name='menteeID' value='".$userID."'>

<button type='book' name='book'>Book</button>

</form>

</td>

</tr>";

}

echo "</table>";

} ?>

</article></section></div>

<?php include 'footer.php'; ?></div></body></html>

### bookSesion.php

booking.php inputs which subject and week the mentee is looking for.

booking2.php displays open sessions with mentors that tutor the subject the mentee is after along with a button to book the session.

<?php session\_start(); include 'header2.php';

include 'nav.php'; require 'connect.php'; ?>

<!--Article section-->

<article >

<h1>Your session has been booked! </h1><br><br>

<h2>Please return to the homepage!</h2><br><br>

<?php

if (isset($\_POST['book'])) {

$userID=$\_SESSION["userID"]; //Assign User Vars

$subjectID = $\_POST['subjectID']; //Assign Posts

$sessionID = $\_POST['sessionID'];

//SQL

$sql = "UPDATE Sessions

SET booked = 'Y', menteeID = '".$userID."', subjectID = '".$subjectID."'

WHERE sessionID ='".$sessionID."'";

$result = mysqli\_query($conn, $sql);

} ?>

</article></section></div> <?php include 'footer.php'; ?></div></body></html>

bookSession.php works on from booking & booking2.php, Updating the booking row, adding in the menteeID and subjectID.

### deleteFeedback.php

<?php session\_start(); include 'header2.php'; include 'nav.php'; require 'connect.php'; ?>

<!--Article section-->

<article

<h1>Your feedback has been deleted! </h1><br><br>

<h2>Please return to the homepage!</h2><br><br>

<?php

if (isset($\_POST['cancel'])) {

$feedbackID = $\_POST['feedbackID']; //Get Session sessionID

//SQL

$sql = "DELETE FROM Feedback

WHERE feedbackID ='".$feedbackID."'";

$result = mysqli\_query($conn, $sql);

} ?> </article></section> </div> <?php include 'footer.php'; ?> </div></body></html>

deleteFeedback.php allowes the mentee to delete the feedback they have provided.

### reports.php

<?php session\_start(); include 'header2.php';

include 'nav.php'; require 'connect.php'; ?>

<!--Article section-->

<article

<h1>Reports</h1>

<?php

$sql = "SELECT sub.subjectName, u.grade, COUNT(\*) AS numSessions

FROM Sessions s

JOIN Users u ON s.menteeID = u.userID

JOIN Subjects sub ON s.subjectID = sub.subjectID

WHERE s.booked = 'Y'

GROUP BY s.subjectID, u.grade

ORDER BY s.subjectID, u.grade;";

$result = mysqli\_query($conn, $sql);

//Display table

echo "<table border='1'>";

echo "<tr><th>Subject Name</th><th>Grade</th><th>Booked Sessions</th></tr>"; //Table titles

while($row = mysqli\_fetch\_assoc($result)){

echo "<tr>

<td>{$row['subjectName']}</td>

<td>{$row['grade']}</td>

<td>{$row['numSessions']}</td>

</tr>";

}

echo "</table>"; ?> </article> </section></div> <?php include 'footer.php'; ?> </div></body> </html>

reports.php uses joins and group by to select the amount in each Grade and subject combination.

# Evaluation

## Criteria

|  |  |  |
| --- | --- | --- |
| **Prescribed:** | Yes/No | Feedback/ Comments |
| By the due date, 8:30 am Monday 2nd October 2023:   * Designed and implemented an effective normalised relational database to store data and user inputs. * Students can register as a mentee or mentor. (Supervisor accounts are preset to protect system safety and data integrity) * Mentees can find tutoring sessions for their subject then book it and provide feedback. * Mentors can select subjects to tutor and open sessions when they are available, and view provided feedback. * Both Mentees and Mentors can view their future sessions * Supervisors can view all open and booked sessions as well as reports for the number of sessions booked for subject/ year level combinations. | Y  Y  Y  Y  Y  Y | Nil |
| **Self-determined:** | Yes/ No |  |
| By before the due date, on Friday 29th September 2023:   * (Ensure all prescribed criteria are met.) * (Build a prototype login and register system that assists the criteria above.) * Ensure all code is commented, indented, and appropriately used whitespace on all pages. * Effectively use HTML and PHP to create a consistent website structure. * Effectively use CSS to create a consistent website design and user experience UX. * Appropriately use HTML and CSS to have a consistent and learnable UI and adhere to design principles. * Use selection and iteration in PHP to produce efficient code. * Implement measures to protect data integrity. * Use PHP to interpret and display data retrieved from the SQL database. * Use consistent variables and file naming conventions. * Have a learnable user interface (UI) that is effective and accessible while maintaining an enjoyable user experience (UX) * Personally understand how all of the PHP functions work to best utilise them efficiently. | Y  Y  Y  Y  Y  Y  Y  Y  Y  Y  Y  Y | Nil  Variables and file Names were either Normal or used camelCase.  Throughout the development of this application, I studied all of the SQL functions (especially the ones with arrays) to ensure I fully understood everything I was working with. |

## Recommendations

This version of the web application is a WIP prototype so there are many features that were not implemented due to time constraints. All of the prescribed criteria have been met, and function correctly.

Currently, the term and year are fixed variables, Admins should have the ability to change these or they could update automatically, therefore automatically changing the titles and webpages accordingly, this feature is essential if this system were to be deployed in actual use. Depending on which school the application the school is deployed in, the grade field on the register page should have its min and max updated accordingly. As the school weeks pass, the completed sessions could have their ‘booked’ field switched from (Y/N) to A (Archived) which can be used to filter out old sessions, ensuring the dynamic message is up to date with the current week and term.

Any tables on the website and the buttons inside of them should receive some CSS to make the application more user-appealing and friendly. A page listing the time and day of the tutoring session each week was supposed to be built first but was skipped in favour of completing the base application.

Mentees should be able to edit any feedback they have provided rather than deleting and resubmitting it. Mentors should be able to report any inappropriate feedback or a moderation system should be implemented. This further extends to registering accounts. All accounts should have to be supervisor-approved before they can book or open sessions on the website, preventing any bad actors from taking advantage of the system. All users should be able to update their User details, with grades being automatically updated each year and older students removed (or archived). Supervisors should be able to edit all user details and delete inappropriate feedback. The feedback section could also use a date/time, allowing the mentor to see when the feedback was given.

It should be discussed who has the final call, in this current system, if a mentee books (claims) a session, the mentor can't unbook it. This causes problems if the mentor is going to be away, but a mentee has already booked it.

Regular backups of the system are required and should be done to a different location or to a local server within each school, protecting all data from any case where the main server goes offline.

**Most importantly** the passwords **aren’t** encrypted and are plain text in the database, this is obviously the first fix if this system were to be deployed.

## Desk Check

|  |  |  |  |
| --- | --- | --- | --- |
| Action | Expected outcome | Actual outcome | Recommendation |
| Login | Assigns variables & moves to the index page | (Worked) | Nil |
| Create booking | Click book & it books the session | (Worked) | Nil |
| View bookings | Displays all sessions for that related user | (Worked) | Nil |
| Give Feedback | Feedback is submitted | (Worked) | Nil |
| View feedback | Displays all feedback for that related user | (Worked) | Nil |
| Open Sessions | Submit available weeks to the database | (Worked) | Nil |
| Select Subjects | Submit tutor subjects to database | (Worked) | Nil |
| View reports | Display subject/ grade combination amounts | (Worked) | Nil |

## Prescribed criteria

For this task, an interactive web application was produced to create a vertical academic tutoring process by going through the problem-solving phases of explore, develop, generate, and evaluate/ refine. During the explore phase a mind map was produced to lay out the problem in a way that is easier to comprehend. During the develop phase multiple dataflow diagrams were created to separate and work out the movement of data throughout the web application. During the design phase, the database was constructed with the assistance of an entity relationship diagram which displayed the logical relationship between the sections of the database. Wireframes were produced to scaffold the design process of the website ensuring a good UI and UX. Algorithms were also produced to decompose difficult sections of code, making them easier to complete. All of these resources mentioned earlier assisted throughout the generate phase, guiding the production of this web application. All of the prescribed criteria have been met or are within arm’s reach due to extensive work on the basis of the application, making further development easy.

## Self-determined criteria

All of the code in the task was or was almost appropriately commented, having correct indentation and correct whitespace. HTML and PHP were used effectively to create a consistent website structure. CSS was effectively used to enhance and ensure a great UX. HTML and CSS were used appropriately to have a learnable UI that adhered to design principles. Measures such as dropdown boxes were implemented to assist in protecting data integrity. Selection and iteration was used in PHP to produce efficient code. I made sure I fully understood all of the PHP and SQL functions involved such as mixing them with arrays and using the SQL Row Function. Lastly, PHP was used to interpret and display data retrieved from the databases.

## Impacts – Personal, Social and Economic

This web application will greatly and positively impact all of these categories. Enabling students’ easier access to tutoring through older students instead of through paid services positively impacts lower-income students who wish to excel in their studies. Connecting more students impacts them socially as they will be able to connect with people, they may have never had the chance to interact with before. The social and economic benefits should have a personal benefit on the students, as they don’t have to pay for tutoring services, and can network with others while personally gaining in their studies, allowing them to succeed more in life.

# Bibliography

ipto.com.au. (n.d.). Digital Solutions Online. [online] Available at: <https://ipto.com.au/>.

W3Schools (2023). W3Schools Online Web Tutorials. [online] W3schools.com. Available at: <https://www.w3schools.com/>.

Richard Peterson (2019). What is Normalization? 1NF, 2NF, 3NF & BCNF with Examples. [online] Guru99.com. Available at: <https://www.guru99.com/database-normalization.html>.