Faculty of Computing

SECV 2223: Web Programming

Section 09

Students' Industrial Training Management System

Prepared by: <Group 3>

Name	Matric No
Gan Heng Lai	A21EC0176
Ng Kai Zheng	A21EC0101
Lew Chin Hong	A21EC0044
Yeo Chun Teck	A21EC0148

1. File and Database

1.1 File Name

File	Name
Login / Landing Page	Index.php
Configuration File	config.php

1.2 Database Name

The name of the database for the system is “projectdatabase”

2. Login Details for all user level

Username	Password	User Level
admin	0000	1
abuadmad	2222	2
ahassan	2222	2
aibrahim	2222	2
akrah	2222	2
fhassan	2222	2
ykhan	2222	2
gohsu	2222	2
hahmeh	2222	2
jacklim	2222	2
mali	2222	2
haidah	1111	3
ganlai	1111	3
kaizheng	1111	3
lewhong	1111	3

yeoteck	1111	3
aida	1111	3

3. Added Criteria

3.1 The Usage Of Session

3.1.1 The Importance of Usage of Session

A Session refers to a period of interaction between a user and a website. A session begins when a user accesses a website and ends when they leave or remain inactive for a certain period. In the developed system, web sessions are often used to store user-specific information, such as login credentials, allowing the website to maintain its state and provide a personalised experience

The Web sessions are implemented in multiple different parts of the system to enable websites to remember and identify individual users. Starting from the landing page, we used the web session to store the information related to the users including the username, the password, the user level and the login status. This information has its use for different purposes.

Through the implementation of the session, the user data could be passed on from one page to another. With this, we could ensure that the user could navigate between different pages to access the functionality that is corresponding to their user level. In the developed system, some functionalities have demands on the authority of the user. Some users at higher levels could achieve some functionalities which are unreachable for users at lower levels. Based on this concept, the pages of the system are designed to provide features according to the user level. Even though all of the users are on the same page, they could access their features without having conflict with their authorities. This is built from the existence of the user-level information that is stored in the session array. The developed system is using this information to determine the authority of the user on any page. This provides variability to the functionalities of the system. With this. any page in the system could support a wide range of users and unnecessary resource expenses could be avoided because the duplication of the page with the same purpose is removed from the development.

The usage of sessions in the system also ensures the continuity of the system across different features. Every functionality needs a bunch of data that is related to the user for

processing in consecutive tasks. However, transferring massive data between different functionalities would cause excessive load on the system. And it is also irrational to be implemented in the system because this could increase the complexity of the system. Furthermore, movement between different functionality may have a possibility to encounter loss of data in transmission. Hence, in this instance, the usage of sessions would be crucial by reducing the amount of required data. The system could confirm the users ‘ identity with the information stored in the session array. The information stored in the session array is temporary and shared across all of the functionality in the system. Rather than the transmission of data across pages, the system could directly access the data from the session array and use this as an identifier to access the information from the database. Storing information like usernames could be said that it is sufficient to support most of the tasks of processing data because it has considerable usage in accessing the information from the database.

Besides that, the verification of account access in the system could be facilitated with the implementation of the session. The information of status in any process like login could be stored in the session array. With this, we could skip the steps of verification when we access any functionality because the login status states the accessibility of the user to the system. By referring to the login status, the system could take action to open access or block the access of the user to the system.

The usage of web sessions is crucial for building interactive and personalised web applications or systems. They allow websites to main user context, enable features like user authentication and authorization and provide a seamless and customised user experience across multiple pages or interactions

3.1.2 The Preparation of Usage of Session

Before implementing the session in the developed system, there was preparation we had done so that the session array data could be shared across all the pages in the system. We start from the beginning of our system which is index.php, storing some important data in a session array for later use in the system

```php
// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // username and password sent from form
    $myusername = $_POST["username"];
    $mypassword = $_POST["password"];

    // Escape special characters in the username and password
    $myusername = mysqli_real_escape_string($conn, $myusername);
    $mypassword = mysqli_real_escape_string($conn, $mypassword);

    $sql = "SELECT * FROM user WHERE username='$myusername' AND password='$mypassword'";

    $result = mysqli_query($conn, $sql);

    $count = mysqli_num_rows($result);

    if ($count == 1) {
        $rows = mysqli_fetch_assoc($result);
        $user_name = $rows["username"];
        $user_id = $rows["password"];
        $user_level = $rows["level"];

        // Add user information to the session (global session variables)
        $_SESSION["Login"] = "YES";
        $_SESSION["USER"] = $user_name;
        $_SESSION["ID"] = $user_id;
        $_SESSION["LEVEL"] = $user_level;
```

The diagram shows the initialization of the session array element by assigning the "POST" method data from the login.php to them

3.1.3 The Motive of Implementation of Session

We have two motives to implement sessions in our system:

1. To check the accessibility of the user to the pages of the system.

```php
<?php
session_start();
if ($_SESSION["Login"] != "YES") {
    header("Location: login.php");
}
?>
```

- If the Login status in the session array is not YES then the user would be moved to the login.php to log in to their account so that they could get access to the system

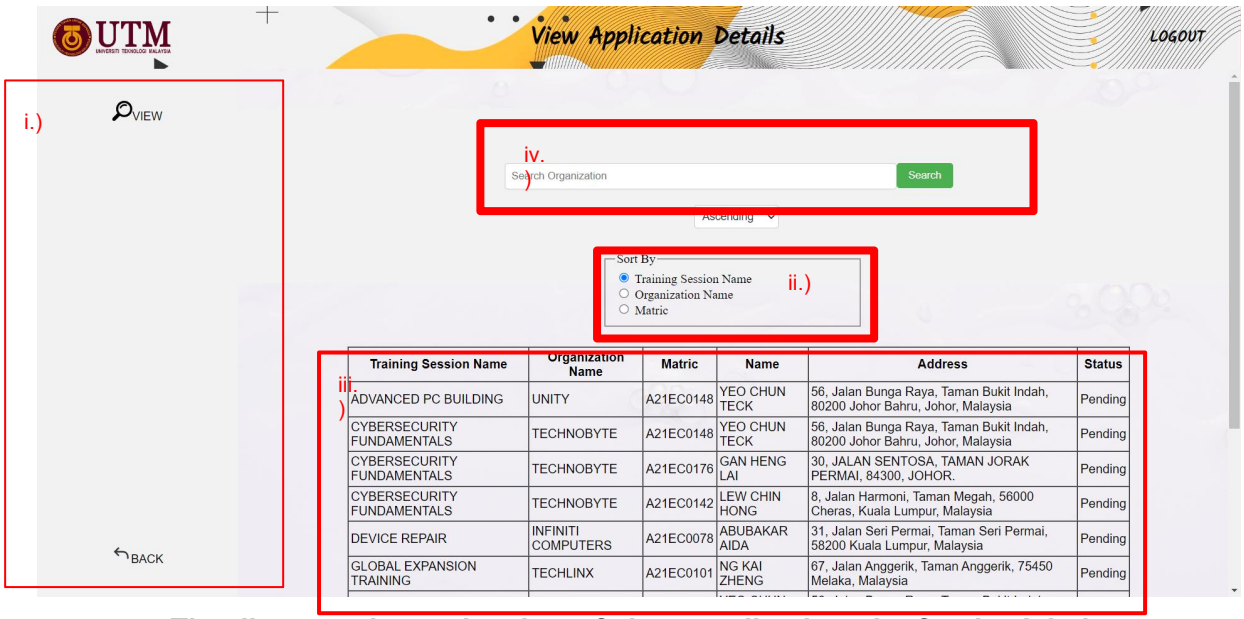
2. The display of different views of a page according to the user's level.
    - In this case, view_application.php would be the example to explain how sessions are implemented.
    - To view the functionality according to the user level, the if-condition syntax has been used.
    - The system uses user identity data($_SESSION["LEVEL"]) in session array data to determine which view to be displayed

a. Initialization of session array data to page variable

```php
<?php
session_start(); // Start up your PHP Session
require_once("config.php"); //read up on PHP includes https://www.w3schools.com/php/php_includes.asp
$username=$_SESSION['USER'];
if ($_SESSION["Login"] != "YES") {
    header("Location: login.php");
}
?>
```

- At the start of the page, we have initialised the username in the session array ("$_SESSION['USER']") to the $username variable that would be used for accessing data in the database

b. The Level 1 User - Admin



The diagram shows the view of view_application.php for the Admin

i. Sidebox Layout

```
if ($_SESSION["LEVEL"] == 3) { ?>
<div>
    <button class="borderless-button left-column-button"
        onclick="window.location.href='view_application.php'"><img src="image/icon/view.svg"
            alt="studentlogo" width="25px" height="25px" />VIEW</button>
    <button class="borderless-button left-column-button"
        onclick="window.location.href='application_form.php'"><img src="image/icon/insert.svg"
            alt="studentlogo" width="25px" height="25px" />APPLY</button>
</div>
<div>
    <button class="borderless-button left-column-button" onclick="window.location.href='mainPage.php'"><img
            src="image/icon/back.svg" alt="studentlogo" width="25px" height="25px" />BACK</button>
</div>
<?php
} else { ?>
<div>
    <button class="borderless-button left-column-button"
        onclick="window.location.href='view_application.php'"><img src="image/icon/view.svg"
            alt="studentlogo" width="25px" height="25px" />VIEW</button>
</div>
<div>
    <button class="borderless-button left-column-button" onclick="window.location.href='mainPage.php'"><img
            src="image/icon/back.svg" alt="studentlogo" width="25px" height="25px" />BACK</button>
</div>
<?php
} ?>
</div>
```

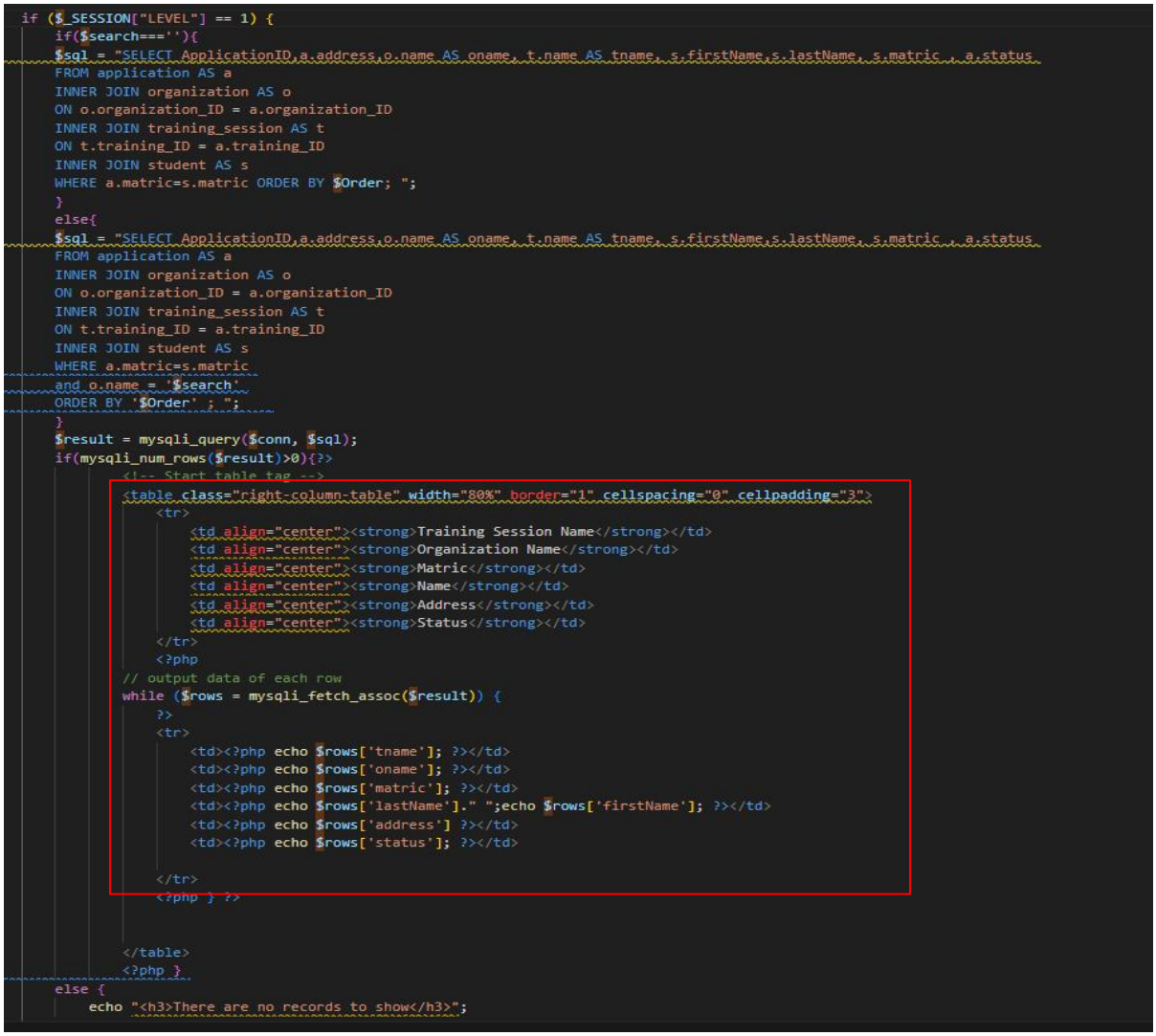
- There are only two buttons, which are “View the application” and “Back to Main Menu” to click.

ii. SortBy Field

```
<div class="filterselect">
    <fieldset style="width: 300px; margin: -50px auto 25px auto;">
        <legend>Sort By</legend>
        <div>
            <input type="radio" id="Training_Session" name="SortBy" value="tname"
                <?php if(!isset($_POST['SortBy'])||$_POST['SortBy']==='tname') echo 'checked' ?> />
            <label for="Training_Session">Training Session Name</label>
        </div>
        <?php if($_SESSION['LEVEL']==1 ||$_SESSION['LEVEL']==3){ ?>
        <div>
            <input type="radio" id="Organization_Name" name="SortBy" value="oname"
                <?php if(isset($_POST['SortBy'])&&$_POST['SortBy']==='oname') echo 'checked' ?> />

            <label for="Organization_Name">Organization Name</label>
        </div>
        <?php }
if($_SESSION['LEVEL']==1 ||$_SESSION['LEVEL']==2){ ?>
        <div>

            <input type="radio" id="Matric" name="SortBy" value="s.matric"
                <?php if(isset($_POST['SortBy'])&&$_POST['SortBy']==='s.matric') echo 'checked' ?> />
            <label for="Matric">Matric</label>
        </div>
        <?php
    }
    ?>
    </fieldset>
    </form>

</div>
```

- The Admin can sort the application data by “Training Session Name”, “Organization Name”, Matric

iii. Application Table



```php
if ($_SESSION["LEVEL"] == 1) {
    if($search===''){
    $sql = "SELECT ApplicationID,a.address,o.name AS oname, t.name AS tname, s.firstName,s.lastName, s.matric , a.status
    FROM application AS a
    INNER JOIN organization AS o
    ON o.organization_ID = a.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric ORDER BY $Order; ";
    }
    else{
    $sql = "SELECT ApplicationID,a.address,o.name AS oname, t.name AS tname, s.firstName,s.lastName, s.matric , a.status
    FROM application AS a
    INNER JOIN organization AS o
    ON o.organization_ID = a.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric
    and o.name = '$search'
    ORDER BY '$Order' ; ";
    }
    $result = mysqli_query($conn, $sql);
    if(mysqli_num_rows($result)>0){?>
        <!-- Start table tag -->
        <table class="right-column-table" width="80%" border="1" cellspacing="0" cellpadding="3">
            <tr>
                <td align="center"><strong>Training Session Name</strong></td>
                <td align="center"><strong>Organization Name</strong></td>
                <td align="center"><strong>Matric</strong></td>
                <td align="center"><strong>Name</strong></td>
                <td align="center"><strong>Address</strong></td>
                <td align="center"><strong>Status</strong></td>
            </tr>
            <?php
        // output data of each row
        while ($rows = mysqli_fetch_assoc($result)) {
            ?>
            <tr>
                <td><?php echo $rows['tname']; ?></td>
                <td><?php echo $rows['oname']; ?></td>
                <td><?php echo $rows['matric']; ?></td>
                <td><?php echo $rows['lastName']." ";echo $rows['firstName']; ?></td>
                <td><?php echo $rows['address'] ?></td>
                <td><?php echo $rows['status']; ?></td>

            </tr>
            <?php } ?>


        </table>
        <?php }
    else {
        echo "<h3>There are no records to show</h3>";
```

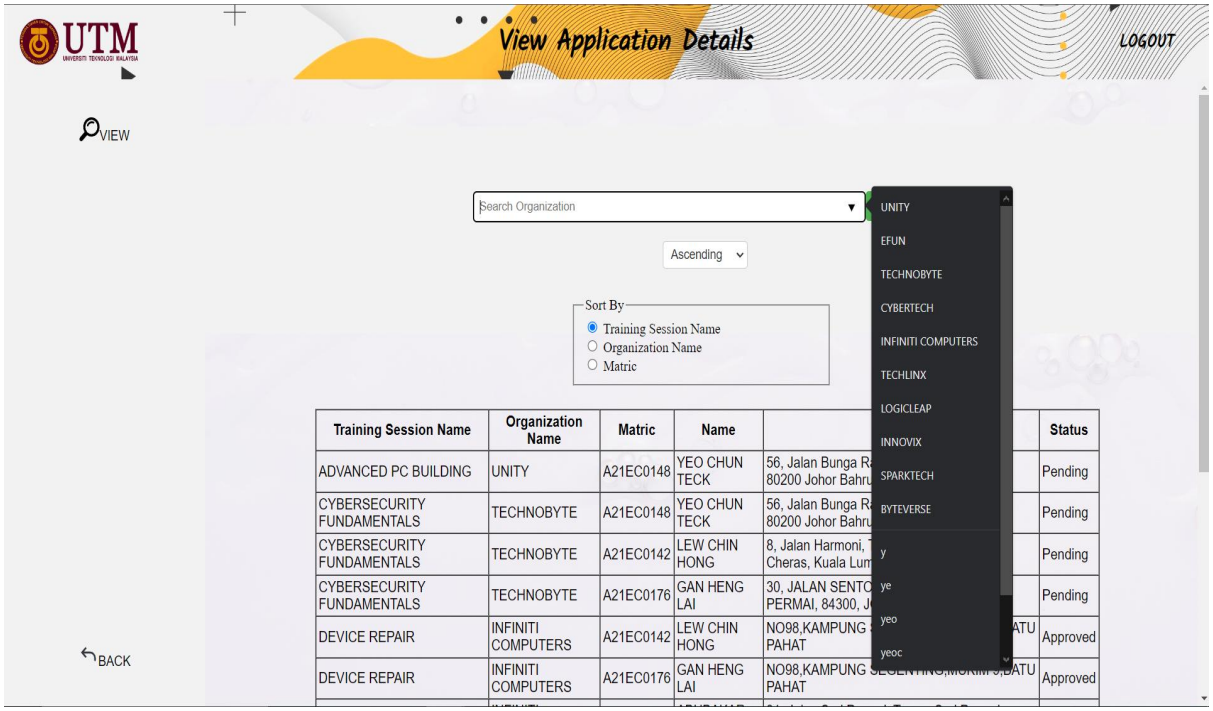
- The Admin has the authority to view all of the applications from all of the companies that have registered into the system
- The information he could view includes:
    - Training Session Name
    - Organisation Name
    - Matric
    - Name
    - Address
    - Status

iv. Search Bar

- For Admin, he could search all of the organisation

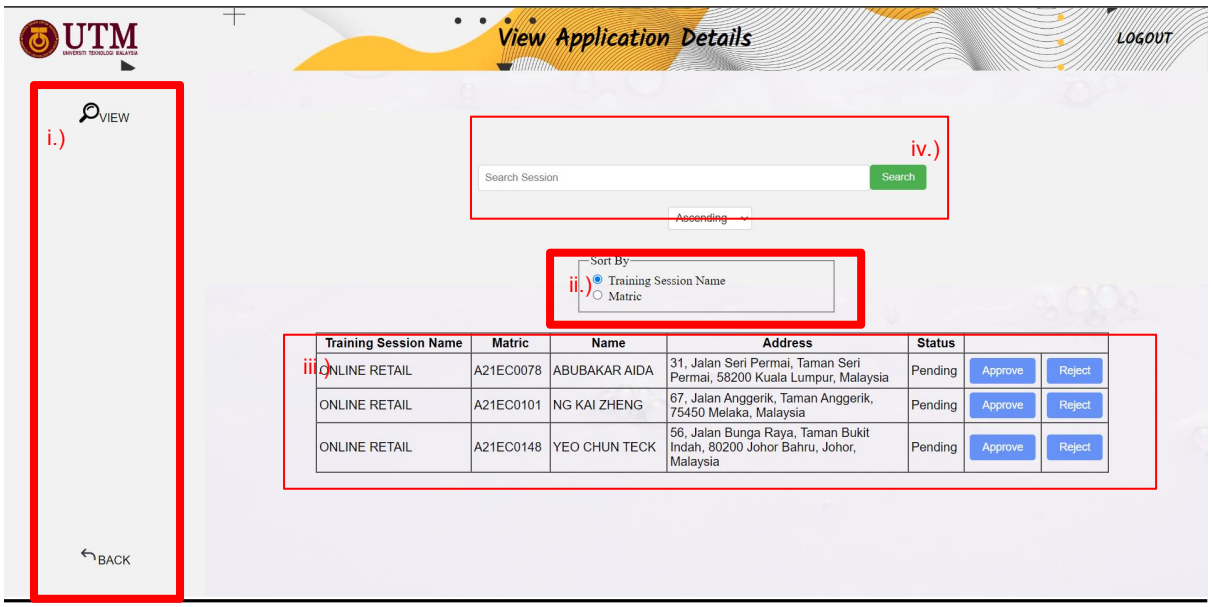
```php
if ($_SESSION["LEVEL"] == 1) {
    if($search===''){
    $sql = "SELECT ApplicationID,a.address,o.name AS oname, t.name AS tname, s.firstName,s.lastName, s.matric , a.status
    FROM application AS a
    INNER JOIN organization AS o
    ON o.organization_ID = a.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric ORDER BY $Order; ";
    }
    else{
    $sql = "SELECT ApplicationID,a.address,o.name AS oname, t.name AS tname, s.firstName,s.lastName, s.matric , a.status
    FROM application AS a
    INNER JOIN organization AS o
    ON o.organization_ID = a.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric
    and o.name = '$search'
    ORDER BY '$Order' ; ";
    }
    $result = mysqli_query($conn, $sql);
    if(mysqli_num_rows($result)>0){?>
        <!-- Start table tag -->
        <table class="right-column-table" width="80%" border="1" cellspacing="0" cellpadding="3">
            <tr>
                <td align="center"><strong>Training Session Name</strong></td>
                <td align="center"><strong>Organization Name</strong></td>
                <td align="center"><strong>Matric</strong></td>
                <td align="center"><strong>Name</strong></td>
                <td align="center"><strong>Address</strong></td>
                <td align="center"><strong>Status</strong></td>
            </tr>
            <?php
        // output data of each row
        while ($rows = mysqli_fetch_assoc($result)) {
            ?>
            <tr>
                <td><?php echo $rows['tname']; ?></td>
                <td><?php echo $rows['oname']; ?></td>
                <td><?php echo $rows['matric']; ?></td>
                <td><?php echo $rows['lastName']." ";echo $rows['firstName']; ?></td>
                <td><?php echo $rows['address'] ?></td>
                <td><?php echo $rows['status']; ?></td>

            </tr>
            <?php } ?>


        </table>
        <?php }
    else {
        echo "<h3>There are no records to show</h3>";
```

- The search list shows the name of all registered organisation



```
                <datalist id="ice-cream-flavors">
                    <?php
if($_SESSION["LEVEL"] == 1){
$sql = "SELECT name FROM organization;";}
else if($_SESSION["LEVEL"] == 2){
$sql = "SELECT name FROM training_session AS t
        WHERE organization_ID = (
        SELECT organization_ID
        FROM coordinator
        WHERE username='$username');";}
else{
$sql = "SELECT name FROM organization AS o
        WHERE organization_ID IN (
        SELECT organization_ID
        FROM application
        WHERE matric=(
        SELECT matric FROM student
        WHERE username='$username'));";
}
$result = mysqli_query($conn, $sql);
while ($rows = mysqli_fetch_assoc($result)) {
?>
                    <option value="<?php echo $rows['name']; ?>">
                        <?php
}
?>
                </datalist>
```

c. The Level 2 User - Coordinator

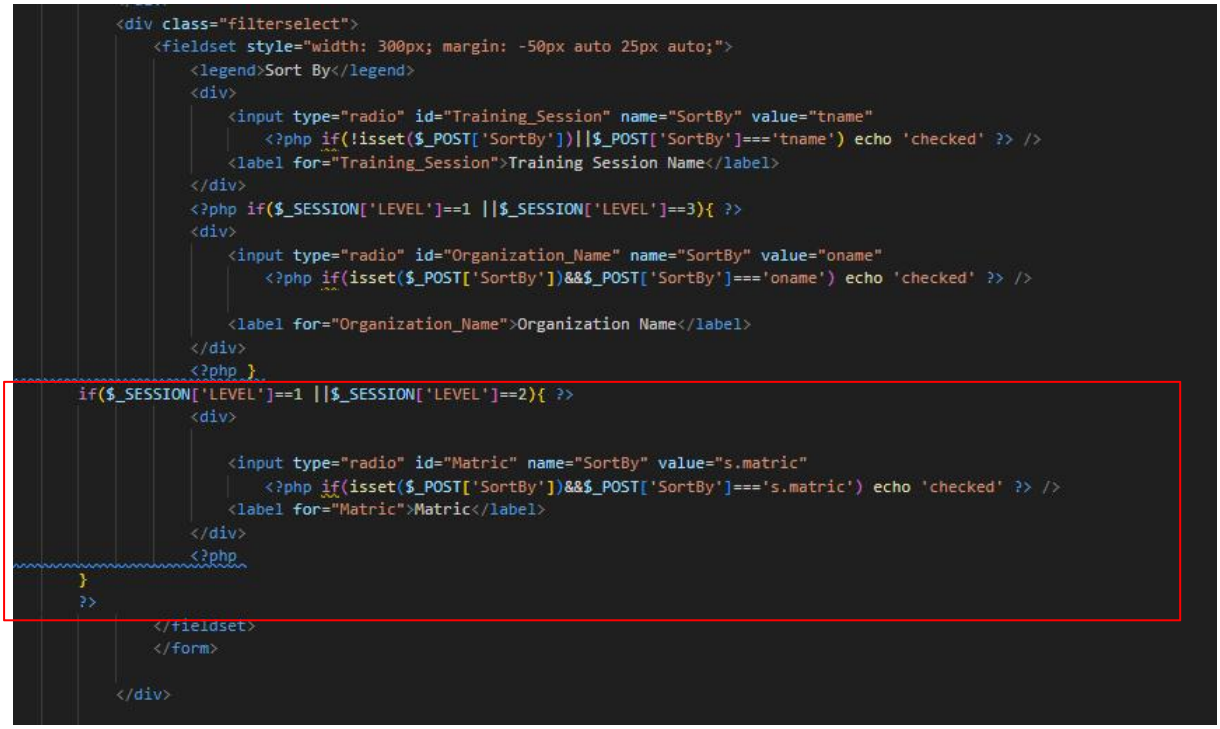


The diagram shows the view of view_application.php for the Admin

i. Sidebox Layout

- The layout and the code are the same as Admin

ii. SortBy Field



- The Admin can sort the application data by “Training Session Name”,” Matric”

iii. Application Table

```
        <?php } else if ($_SESSION['LEVEL'] == 2) {
    $sql = "SELECT coordinate_ID FROM coordinator WHERE username='$username'";
    $result = mysqli_query($conn, $sql);
    $row = mysqli_fetch_assoc($result);
    $coordinator_ID = $row['coordinate_ID'];
    if($search===''){
    $sql = "SELECT ApplicationID,a.address, t.name AS tname, s.firstName,s.lastName, s.matric , a.status, current_amount,required_amount
    FROM application AS a
    INNER JOIN coordinator AS c
    ON a.organization_ID = c.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric and c.coordinate_ID='$coordinator_ID'ORDER BY $Order ;" ;
    }
    else{
    $sql = "SELECT ApplicationID,a.address, t.name AS tname, s.firstName,s.lastName, s.matric , a.status, current_amount,required_amount
    FROM application AS a
    INNER JOIN coordinator AS c
    ON a.organization_ID = c.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric
    and c.coordinate_ID='$coordinator_ID'
    and t.name = '$search'
    ORDER BY $Order ;" ;
    }
    $result = mysqli_query($conn, $sql);

    if(mysqli_num_rows($result)>0){?>
        <!-- Start table tag -->
        <table class="right-column-table" width="80%" border="1" cellspacing="0" cellpadding="3">
            <!-- Print table heading -->
            <tr>
                <td align="center"><strong>Training Session Name</strong></td>
                <td align="center"><strong>Matric</strong></td>
                <td align="center"><strong>Name</strong></td>
                <td align="center"><strong>Address</strong></td>
                <td align="center"><strong>Status</strong></td>
                <td colspan="2" align="left"> </td>
            </tr>
            <?php
        // output data of each row
        while ($rows = mysqli_fetch_assoc($result)) {
            ?>
            <tr>
                <td><?php echo $rows['tname']; ?></td>
                <td><?php echo $rows['matric']; ?></td>
                <td><?php echo $rows['lastName']." ";echo $rows['firstName']; ?></td>
                <td style="width: 300px;"><?php echo $rows['address'] ?></td>
                <td><?php echo $rows['status']; ?></td>
                <td align="center"><a
                        href="update_application.php?id=<?php echo $rows['ApplicationID']."&status=Approved";?>"
                        target="_parent"><button class="Evaluate" id="Approve"
                            <?php if($rows['status']=='Approved'||$rows['current_amount']==$rows['required_amount']) echo 'disabled' ?>>Approve</button></a>
                </td>
                <td align="center"><a
                        href="update_application.php?id=<?php echo $rows['ApplicationID']."&status=Rejected";?>"
                        target="_parent"><button class="Evaluate" id="Reject"
                            <?php if($rows['status']=='Rejected'||$rows['status']=='Stopped') echo 'disabled' ?>>Reject</button></a>
                </td>
            </tr>
            <?php } ?>

        </table>
        <?php }
    else {
        echo "<h3>There are no records to show</h3>";
    }
```

- The coordinator has the authority to view all of the applications from his own company
- The information he could view includes:
    - Training Session Name
    - Matric
    - Name
    - Address
    - Status

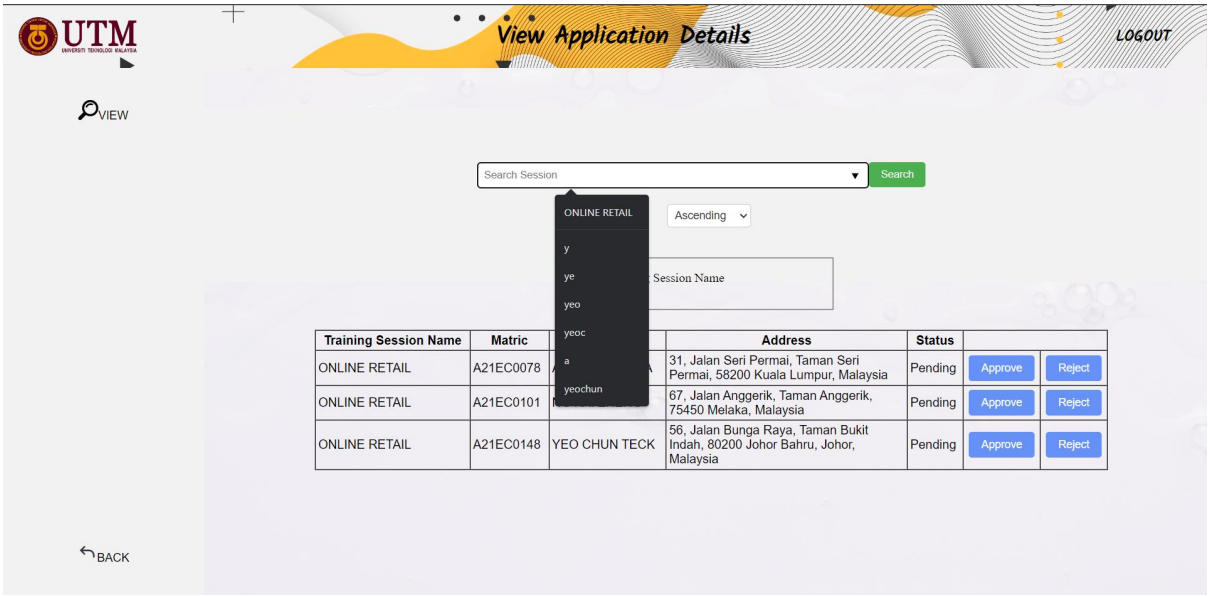
a. There are 2 buttons displayed which are “Approve” and “Reject” for the coordinator to announce the evaluation outcome

iv. Search Bar

- For Coordinator, he could search all of the training sessions of his company

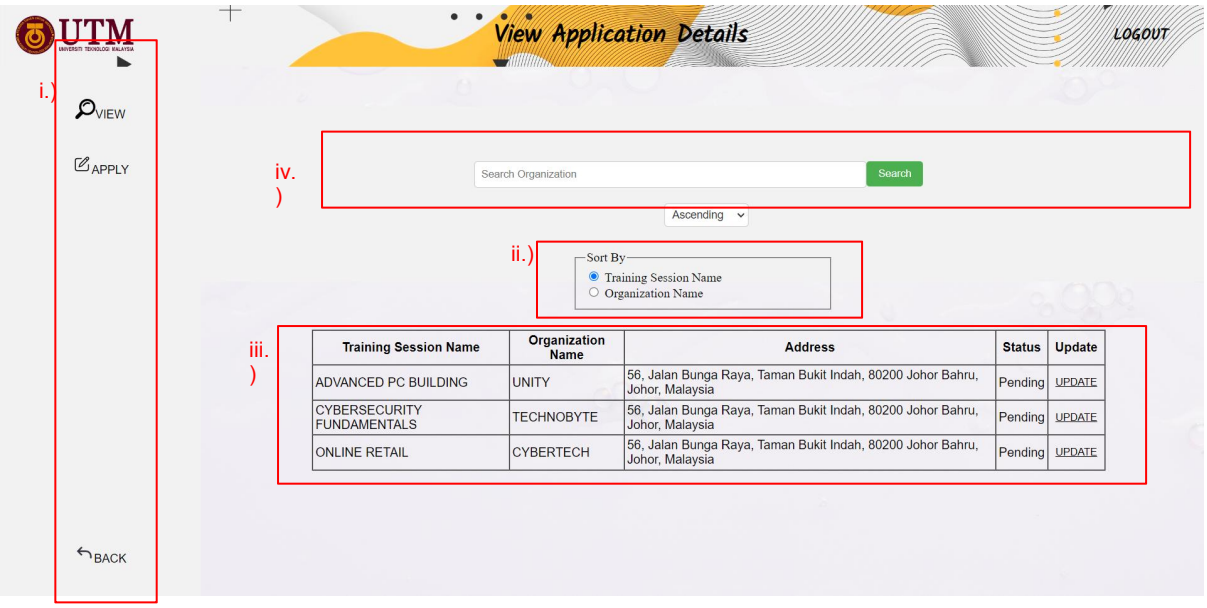
```
        <?php } else if ($_SESSION['LEVEL'] == 2) {
    $sql = "SELECT coordinate_ID FROM coordinator WHERE username='$username'";
    $result = mysqli_query($conn, $sql);
    $row = mysqli_fetch_assoc($result);
    $coordinator_ID = $row['coordinate_ID'];
    if($search===''){
    $sql = "SELECT ApplicationID,a.address, t.name AS tname, s.firstName,s.lastName, s.matric , a.status, current amount,required amount
    FROM application AS a
    INNER JOIN coordinator AS c
    ON a.organization_ID = c.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric and c.coordinate_ID='$coordinator_ID'ORDER BY $Order ;" ;
    }
    else{
    $sql = "SELECT ApplicationID,a.address, t.name AS tname, s.firstName,s.lastName, s.matric , a.status, current amount,required amount
    FROM application AS a
    INNER JOIN coordinator AS c
    ON a.organization_ID = c.organization_ID
    INNER JOIN training_session AS t
    ON t.training_ID = a.training_ID
    INNER JOIN student AS s
    WHERE a.matric=s.matric
    and c.coordinate_ID='$coordinator_ID'
    and t.name = '$search'
    ORDER BY $Order ;" ;
    }
    $result = mysqli_query($conn, $sql);

    if(mysqli_num_rows($result)>0){?>
            <!-- Start table tag -->
            <table class="right-column-table" width="80%" border="1" cellspacing="0" cellpadding="3">
                <!-- Print table heading -->
                <tr>
                    <td align="center"><strong>Training Session Name</strong></td>
                    <td align="center"><strong>Matric</strong></td>
                    <td align="center"><strong>Name</strong></td>
                    <td align="center"><strong>Address</strong></td>
                    <td align="center"><strong>Status</strong></td>
                    <td colspan="2" align="left"> </td>
                </tr>
                <?php
            // output data of each row
            while ($rows = mysqli_fetch_assoc($result)) {
                ?>
                <tr>
                    <td><?php echo $rows['tname']; ?></td>
                    <td><?php echo $rows['matric']; ?></td>
                    <td><?php echo $rows['lastName']." ";echo $rows['firstName']; ?></td>
                    <td style="width: 300px;"><?php echo $rows['address'] ?></td>
                    <td><?php echo $rows['status']; ?></td>
                    <td align="center"><a
                            href="update_application.php?id=<?php echo $rows['ApplicationID']."&status=Approved";?>"
                            target="_parent"><button class="Evaluate" id="Approve"
```

- The search list shows the name of all registered organisation




```
                <datalist id="ice-cream-flavors">
                    <?php
if($_SESSION["LEVEL"] == 1){
$sql = "SELECT name FROM organization;";}
else if($_SESSION["LEVEL"] == 2){
$sql = "SELECT name FROM training_session AS t
        WHERE organization_ID = (
        SELECT organization_ID
        FROM coordinator
        WHERE username='$username');";}
else{
$sql = "SELECT name FROM organization AS o
        WHERE organization_ID IN (
        SELECT organization_ID
        FROM application
        WHERE matric=(
        SELECT matric FROM student
        WHERE username='$username'));";
}
$result = mysqli_query($conn, $sql);
while ($rows = mysqli_fetch_assoc($result)) {
?>
                    <option value="<?php echo $rows['name']; ?>">
                        <?php
}
?>
                </datalist>
```

d. The Level 3 User - Student



The diagram shows the view of view_application.php for the Student

i. Sidebox Layout

```
if ($_SESSION["LEVEL"] == 3) { ?>
<div>
    <button class="borderless-button left-column-button"
        onclick="window.location.href='view_application.php'"><img src="image/icon/view.svg"
            alt="studentlogo" width="25px" height="25px" />VIEW</button>
    <button class="borderless-button left-column-button"
        onclick="window.location.href='application_form.php'"><img src="image/icon/insert.svg"
            alt="studentlogo" width="25px" height="25px" />APPLY</button>
</div>
<div>
    <button class="borderless-button left-column-button" onclick="window.location.href='mainPage.php'"><img
            src="image/icon/back.svg" alt="studentlogo" width="25px" height="25px" />BACK</button>
</div>
<?php
} else { ?>
<div>
    <button class="borderless-button left-column-button"
        onclick="window.location.href='view_application.php'"><img src="image/icon/view.svg"
            alt="studentlogo" width="25px" height="25px" />VIEW</button>
</div>
<div>
    <button class="borderless-button left-column-button" onclick="window.location.href='mainPage.php'"><img
            src="image/icon/back.svg" alt="studentlogo" width="25px" height="25px" />BACK</button>
</div>
<?php
} ?>
</div>
```

iii.)

- There are three buttons, which are “View the application”,” Apply the application” and “Back to Main Menu” to click.

ii. SortBy

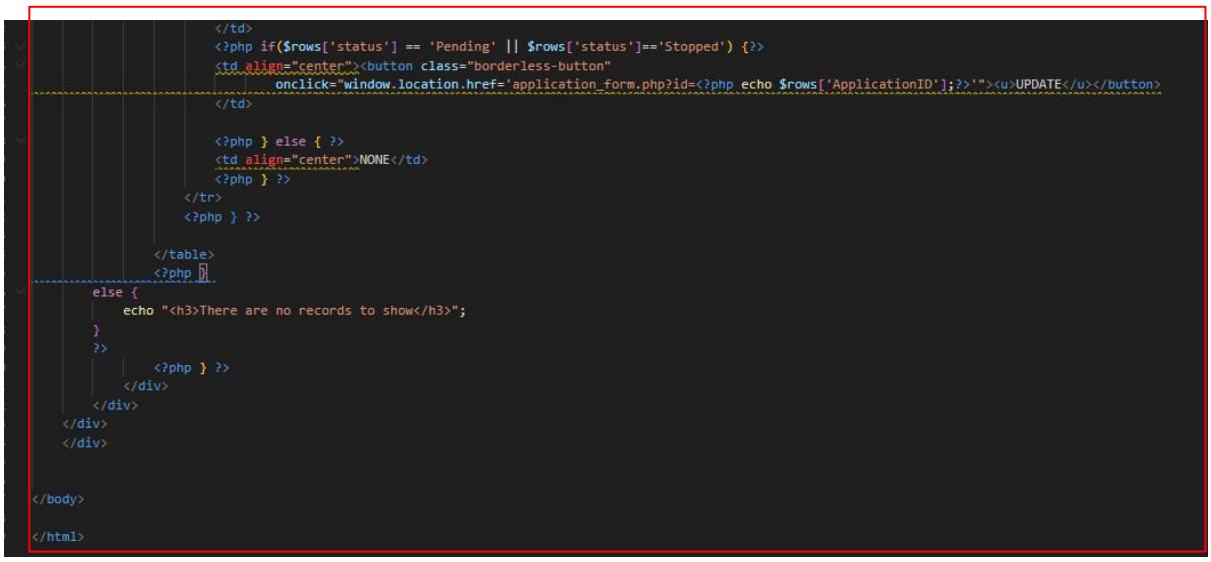
Field

```
<div class="filterselect">
    <fieldset style="width: 300px; margin: -50px auto 25px auto;">
        <legend>Sort By</legend>
        <div>
            <input type="radio" id="Training_Session" name="SortBy" value="tname"
                <?php if(!isset($_POST['SortBy'])||$_POST['SortBy']==='tname') echo 'checked' ?> />
            <label for="Training_Session">Training Session Name</label>
        </div>
        <?php if($_SESSION['LEVEL']==1 ||$_SESSION['LEVEL']==3){ ?>
        <div>
            <input type="radio" id="Organization_Name" name="SortBy" value="oname"
                <?php if(isset($_POST['SortBy'])&&$_POST['SortBy']==='oname') echo 'checked' ?> />

            <label for="Organization_Name">Organization Name</label>
        </div>
        <?php }
if($_SESSION['LEVEL']==1 ||$_SESSION['LEVEL']==2){ ?>
        <div>

            <input type="radio" id="Matric" name="SortBy" value="s.matric"
                <?php if(isset($_POST['SortBy'])&&$_POST['SortBy']==='s.matric') echo 'checked' ?> />
            <label for="Matric">Matric</label>
        </div>
        <?php
    }
    ?>
        </fieldset>
        </form>

    </div>
```

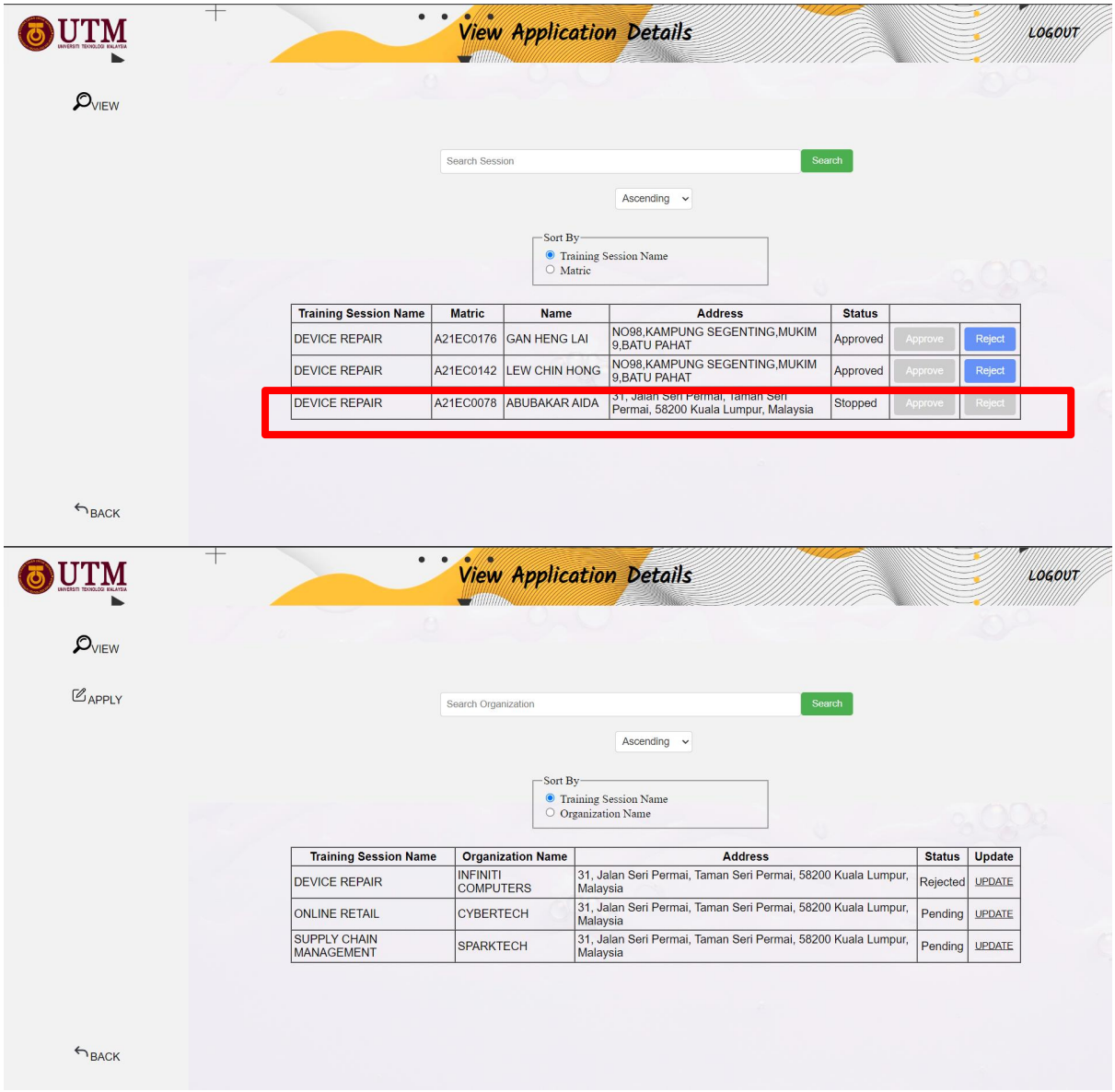
- The Student can sort the application data by “Training Session Name” and ”Organization Name”

iii. Application Form

```
    else {
        echo "<h3>There are no records to show</h3>";
    }
    ?>
        <?php } else {

    $sql = "SELECT matric FROM student WHERE username='$username'";
    $result = mysqli_query($conn, $sql);
    $row = mysqli_fetch_assoc($result);
    $matric = $row['matric'];
    if($search===''){

    $sql = "SELECT ApplicationID,a.address, o.name AS oname, t.name as tname, a.status
            FROM application AS a
            INNER JOIN organization AS o
            ON o.organization_ID = a.organization_ID
            INNER JOIN training_session AS t
            ON t.training_ID = a.training_ID
            WHERE a.matric='$matric'ORDER BY $Order";
    }
    else{
     $sql = "SELECT ApplicationID,a.address, o.name AS oname , t.name as tname, a.status
             FROM application AS a
             INNER JOIN organization AS o
             ON o.organization_ID = a.organization_ID
             INNER JOIN training_session AS t
             ON t.training_ID = a.training_ID
             WHERE a.matric='$matric'
             AND o.name='$search'
             ORDER BY $Order";
    }
    $result = mysqli_query($conn, $sql);

    if (mysqli_num_rows($result) > 0) { ?>
            <!-- Start table tag -->
            <table class="right-column-table" width="80%" border="1" cellspacing="0" cellpadding="3">
                <!-- Print table heading -->
                <tr>
                    <td align="center"><strong>Training Session Name</strong></td>
                    <td align="center"><strong>Organization Name</strong></td>
                    <td align="center"><strong>Address</strong></td>
                    <td align="center"><strong>Status</strong></td>
                    <td align="center"><strong>Update</strong></td>

                </tr>
                <?php
            // output data of each row
            while ($rows = mysqli_fetch_assoc($result)) {
                ?>
                <tr>
                    <td><?php echo $rows['tname']; ?></td>
                    <td><?php echo $rows['oname']; ?></td>
                    <td><?php echo $rows['address']; ?></td>
                    <td align="center">
                        <?php if($rows['status'] === 'Stopped'){ echo "Rejected"; } else { echo $rows['status']; } ?>
```



- The Student has the authority to view all of the applications he applied
- The information he could view includes
    - Training Session Name
    - Organisation Name
    - Address
    - Status
- There is a button displayed which are “Update” for the student to update the application form submitted before it is finished and evaluated
- The status column for the student view is a bit different
- In the developed system, there are 4 statuses to indicate the evaluation outcomes. They are “Pending”, “Approved”, “Rejected” and “Stopped”.
- Stopped” status is not available to be viewed by the student. Oppositely, it would display as “Rejected” in the view of the student



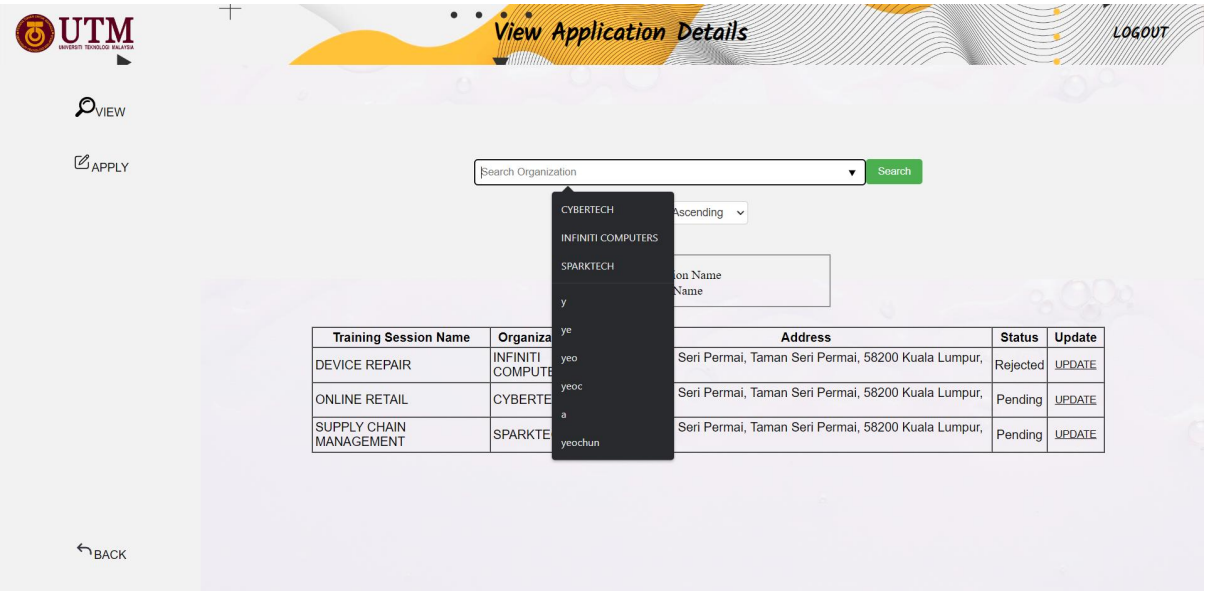
- Like the diagram shown above, for the student who is named Abdubakar Aida, his application to Device Repair Training Session from Infiniti Computers is displayed as rejected but actually, its status is displayed as Stopped which indicates this training session had stopped accepting any application. In other ways, it means that Abdubakar is rejected since the company had hired a sufficient amount of interns for this training session

iv. Search Bar

- The student could search the organisation that he had applied to

```
    else {
        echo "<h3>There are no records to show</h3>";
    }
    ?>
            <?php } else {

    $sql = "SELECT matric FROM student WHERE username='$username'";
    $result = mysqli_query($conn, $sql);
    $row = mysqli_fetch_assoc($result);
    $matric = $row['matric'];
    if($search===''){

    $sql = "SELECT ApplicationID,a.address, o.name AS oname, t.name as tname, a.status
            FROM application AS a
            INNER JOIN organization AS o
            ON o.organization_ID = a.organization_ID
            INNER JOIN training_session AS t
            ON t.training_ID = a.training_ID
            WHERE a.matric='$matric'ORDER BY $Order";
    }
    else{
     $sql = "SELECT ApplicationID,a.address, o.name AS oname , t.name as tname, a.status
            FROM application AS a
            INNER JOIN organization AS o
            ON o.organization_ID = a.organization_ID
            INNER JOIN training_session AS t
            ON t.training_ID = a.training_ID
            WHERE a.matric='$matric'
            AND o.name='$search'
            ORDER BY $Order";
    }
    $result = mysqli_query($conn, $sql);

    if (mysqli_num_rows($result) > 0) { ?>
            <!-- Start table tag -->
            <table class="right-column-table" width="80%" border="1" cellspacing="0" cellpadding="3">
                <!-- Print table heading -->
                <tr>
                    <td align="center"><strong>Training Session Name</strong></td>
                    <td align="center"><strong>Organization Name</strong></td>
                    <td align="center"><strong>Address</strong></td>
                    <td align="center"><strong>Status</strong></td>
                    <td align="center"><strong>Update</strong></td>

                </tr>
                <?php
            // output data of each row
            while ($rows = mysqli_fetch_assoc($result)) {
                ?>
                <tr>
                    <td><?php echo $rows['tname']; ?></td>
                    <td><?php echo $rows['oname']; ?></td>
                    <td><?php echo $rows['address']; ?></td>
                    <td align="center">
                        <?php if($rows['status'] === 'Stopped'){ echo "Rejected"; } else { echo $rows['status']; } ?>
```

- The search list shows the name of all applied organisation




```
                <datalist id="ice-cream-flavors">
                    <?php
if($_SESSION["LEVEL"] == 1){
$sql = "SELECT name FROM organization;";}
else if($_SESSION["LEVEL"] == 2){
$sql = "SELECT name FROM training_session AS t
        WHERE organization_ID = (
        SELECT organization_ID
        FROM coordinator
        WHERE username='$username');";}
else{
$sql = "SELECT name FROM organization AS o
        WHERE organization_ID IN (
        SELECT organization_ID
        FROM application
        WHERE matric=(
        SELECT matric FROM student
        WHERE username='$username'));";
}
$result = mysqli_query($conn, $sql);
while ($rows = mysqli_fetch_assoc($result)) {
?>
                    <option value="<?php echo $rows['name']; ?>">
                        <?php
}
?>
                </datalist>
```

- Other pages in the system are designed in a way like view_application.php. From view_application.php, we could know that there are many differences in the same functionality of different users. All of these differences are supposed to be separated by creating duplicate pages to avoid conflict of these differences to happen. However, by using a web session in the system

development, all of these differences can be combined in a single page successfully without causing any conflict and confusion. The user could access their authority on a single page.

3.2 The benefit of using Cookie

In this project, cookies are used to enhance the login process by storing the user's credentials and enabling automatic login for returning users. Here's an explanation of how cookies are used to benefit the login process:

```php
if (isset($_COOKIE["username"]) && isset($_COOKIE["password"]) && isset($_COOKIE["level"])) {
    $myusername = $_COOKIE["username"];
    $mypassword = $_COOKIE["password"];
```

1. Checking for Existing Cookies
    - At the beginning of the code, it checks if the username, password, and level cookies are already set
    - If the cookies exist, it retrieves the stored username and password values

```php
// Escape special characters in the username and password
$myusername = mysqli_real_escape_string($conn, $myusername);
$mypassword = mysqli_real_escape_string($conn, $mypassword);

$sql = "SELECT * FROM user WHERE username='$myusername' AND password='$mypassword'";

$result = mysqli_query($conn, $sql);

$count = mysqli_num_rows($result);

if ($count == 1) {
    $rows = mysqli_fetch_assoc($result);
    $user_name = $rows["username"];
    $user_id = $rows["password"];
    $user_level = $rows["level"];

    // Add user information to the session (global session variables)
    $_SESSION["Login"] = "YES";
    $_SESSION["USER"] = $user_name;
    $_SESSION["ID"] = $user_id;
    $_SESSION["LEVEL"] = $user_level;
```

2. Validating User Credentials
    - It escapes any special characters in the retrieved username and password values to prevent SQL injection attacks

- Then a database query is executed to check if the provided username and password match a record in the user table
- If a match is found, the user's credentials are considered valid

```
// Update the expiration time of the cookies
setcookie("username", $user_name, time() + (86400 * 30), "/");
setcookie("password", $user_id, time() + (86400 * 30), "/");
setcookie("level", $user_level, time() + (86400 * 30), "/");
```

3. If the user's credentials are valid, the code updates the expiration time of the cookies using the `setcookie()` function
    - The cookies are set to expire 30 days from the current time, ensuring that the user remains logged in for an extended period

```
header("Location: mainPage.php");
```

4. Redirecting to the Main Page
    - After updating the cookies, the code redirects the user to the mainPage.php
    - Since the cookies are now set, the user will be automatically logged in on subsequent visits as long as the cookies are still valid

```
// Check if the form is submitted
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // username and password sent from form
    $myusername = $_POST["username"];
    $mypassword = $_POST["password"];

    // Escape special characters in the username and password
    $myusername = mysqli_real_escape_string($conn, $myusername);
    $mypassword = mysqli_real_escape_string($conn, $mypassword);

    $sql = "SELECT * FROM user WHERE username='$myusername' AND password='$mypassword'";

    $result = mysqli_query($conn, $sql);

    $count = mysqli_num_rows($result);

    if ($count == 1) {
        $rows = mysqli_fetch_assoc($result);
        $user_name = $rows["username"];
        $user_id = $rows["password"];
        $user_level = $rows["level"];

        // Add user information to the session (global session variables)
        $_SESSION["Login"] = "YES";
        $_SESSION["USER"] = $user_name;
        $_SESSION["ID"] = $user_id;
        $_SESSION["LEVEL"] = $user_level;
```

```
    // Set the cookies
    setcookie("username", $user_name, time() + (86400 * 30), "/");
    setcookie("password", $user_id, time() + (86400 * 30), "/");
    setcookie("level", $user_level, time() + (86400 * 30), "/");

    header("Location: mainPage.php");
    exit();
} else {
    $_SESSION["Login"] = "NO";
    header("Location: login.php?Message=" . urlencode("Wrong username or password"));
```

5. Handling New Login Attempts
    - If no valid cookies are found, the code checks if the login form has been submitted ($_SERVER["REQUEST_METHOD"] == "POST")
    - If it has, it retrieves the submitted username and password values and performs the same validation process as mentioned earlier
    - If the login credentials are valid, the cookies are set, and the user is redirected to the main page
    - Else, the user is redirected back to the login page with an appropriate error message

```
// Delete the cookies
setcookie("username", "", time() - 3600, "/");
setcookie("password", "", time() - 3600, "/");
setcookie("level", "", time() - 3600, "/");
```

6. Deleting the Cookies
    - Each cookie is set with an empty value, and the expiration time is set to a past time (time - 3600) to immediately invalidate the cookies

By using cookies, the login process becomes more user-friendly and convenient. Returning users don't need to enter their credentials every time they visit the login page. Instead, their credentials are retrieved from the cookies, allowing for a seamless login experience.

3.3 User Input Validation

This project performs user input validation using JavaScript. Each validation function is responsible for validating a specific input field based on certain criteria. Here's an explanation of how the user input validation is performed:

1. Validation Functions
    - Each input field has a corresponding validation function. For example, `validateFirstName()` validates the first name, `validateLastName()` validates the last name, and so on
    - These validation functions are called when the form is submitted or when a specific input field loses focus
2. Retrieving Input Values
    - Each validation function starts by retrieving the value of the corresponding input field using `document.form.<input_name>.value`
    - For example, `var firstName = document.form.firstName.value` retrieves the value of the first name input field
3. Regular Expressions
    - Regular expressions (regex) are used to define the pattern that the input value should match.
    - The regex patterns are used to validate input values against specific criteria
    - For example, `var alphaExp = /^[a-zA-Z\s]+$/` represents the pattern for alphabetic characters and spaces only
4. Error Handling
    - If the input value does not match the expected pattern or fails the validation criteria, an error message is displayed to the user
    - The error messages are displayed by setting the `textContent` property of the corresponding error element, such as `errorElement.textContent = "Please enter text for your first name!"
5. Field Focus
    - If an error occurs, the `focus()` function is called on the corresponding input field, which brings the focus back to that field for correction
6. Clearing Error Message
    - After validating an input field, the error message is cleared by setting the `textContent` property of the error element to an empty string: `errorElement.textContent = ""
7. Overall Validation
    - The form's validation functions, such as `validateStudent()`, `validateCoordinator()`, etc., are responsible for calling multiple validation functions for all the relevant input fields
    - These functions return `true` if all the individual validations pass, indicating that the form is valid and can be submitted

By utilizing these validation functions, the user's input has adhered to the specified criteria. If any input fails validation, an appropriate error message is displayed, guiding the user to correct their input before proceeding.

3.4 Dynamic HTML through JavaScript (DOM)

1. The code checks if the 'Message' parameter exists in the URL's GET parameters using isset(). If it is present, the PHP code dynamically generates JavaScript using echo to display an alert dialogue. The alert dialogue shows the value passed through the 'Message' parameter obtained from the GET parameters.

```php
<?php
    if (isset($_GET['Message'])) {
        echo '<script>alert("' . $_GET['Message'] . '")</script>';
    }
?>
```

2. In the provided code snippet, the DOM (Document Object Model) is utilized to validate an input field representing the data such as firstname, lastname, matric no IC (Identification Card) number, contact no, email, course, GPA, username and password. The code accesses the value entered in the input field using document.form.IC.value, demonstrating how the DOM is used to retrieve form data. It then retrieves an error element from the DOM using getElementById(), allowing for a dynamic error message display. The textContent property of the error element is modified to show an appropriate error message if the input element doesn't match the specified pattern. The code also utilizes the DOM's focus() method to set focus on the element input field when an error occurs, bringing attention to the incorrect input.

```
function validateFirstName() {
  var firstName = document.form.firstName.value;
  var alphaExp = /^[a-zA-Z\s]+$/;
  var errorElement = document.getElementById("firstNameError");

  if (!alphaExp.test(firstName)) {
    errorElement.textContent = "Please enter text for your first name!";
    document.form.firstName.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}

function validateLastName() {
  var lastName = document.form.lastName.value;
  var alphaExp = /^[a-zA-Z\s]+$/;
  var errorElement = document.getElementById("lastNameError");

  if (!alphaExp.test(lastName)) {
    errorElement.textContent = "Please enter text for your last name!";
    document.form.lastName.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}

function validateMatric() {
  var matric = document.form.matric.value;
  var matricExp = /^[a-zA-Z0-9]{9}$/;
  var errorElement = document.getElementById("matricError");

  if (!matric.match(matricExp)) {
    errorElement.textContent =
      "Please provide a valid matric number (9 digits)!";
    document.form.matric.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}
function validateIC() {
  var IC = document.form.IC.value;
  var ICExp = /^[0-9]{12}$/;
  var errorElement = document.getElementById("ICError");

  if (!IC.match(ICExp)) {
    errorElement.textContent = "Please provide a valid IC number (12 digits)!";
    document.form.IC.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}

function validateContactNo() {
  var contactNo = document.form.contactNo.value;
  var contactNoExp = /^[0-9]{10,11}$/;
  var errorElement = document.getElementById("contactNoError");

  if (!contactNo.match(contactNoExp)) {
    errorElement.textContent =
      "Please provide a valid contact number (10 or 11 digits)!";
    document.form.contactNo.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}
```

```
function validateEmail() {
  var emailID = document.form.email.value;
  var errorElement = document.getElementById("emailError");

  if (emailID != "") {
    var atPos = emailID.indexOf("@");
    var dotPos = emailID.lastIndexOf(".");
    var emailErrorText;

    if (dotPos != -1 && dotPos - atPos < 2) {
      emailErrorText = "@ and . must be at least 1 character apart";
    }
    if (atPos == 0) {
      emailErrorText = "Email address must not start with @";
    }
    if (atPos == -1) {
      emailErrorText = "Email address must contain @";
    }
    if (dotPos == -1) {
      emailErrorText = "Email address must contain '.'";
    }

    if (emailErrorText != null) {
      errorElement.textContent = emailErrorText;
      document.form.email.focus();
      return false;
    }
  }

  errorElement.textContent = ""; // Clear the error message
  return true;
}

function validateCourse() {
  var course = document.form.course.value;
  var alphaExp = /^[A-Za-z\s]+$/;
  var errorElement = document.getElementById("courseError");

  if (!alphaExp.test(course)) {
    errorElement.textContent =
      "Please enter a valid course name (alphabetic characters only)!";
    document.form.course.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}
```

```
function validateGPA() {
  var gpa = document.form.gpa.value;
  var errorElement = document.getElementById("GPAError");

  if (isNaN(gpa) || gpa < 0 || gpa > 4) {
    errorElement.textContent = "Please provide a valid GPA between 0 and 4.";
    document.form.gpa.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}

function validateUsername() {
  var username = document.form.username.value;
  var usernameExp = /^.{4,}$/;
  var errorElement = document.getElementById("usernameError");

  if (!username.match(usernameExp)) {
    errorElement.textContent =
      "Please provide a username with at least 4 characters!";
    document.form.username.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}

function validatePassword() {
  var password = document.form.password.value;
  var passwordExp = /^.{4,}$/;
  var errorElement = document.getElementById("passwordError");

  if (!password.match(passwordExp)) {
    errorElement.textContent =
      "Please provide a password with at least 4 characters!";
    document.form.password.focus();
    return false;
  }
  errorElement.textContent = ""; // Clear the error message
  return true;
}
```

3.5 Clean, Consistent And Attractive Presentation

In this project, there are two external CSS files, namely "main.css" and "login.css". Each CSS file contains styles specific to its intended HTML page. Separating the CSS into individual files allows for better organization and reusability of styles across multiple HTML pages. This modular approach helps maintain consistency and simplifies the process of updating or modifying styles.

When implementing these CSS files, each HTML page that requires specific styles will include a <link> tag in its <head> section. The <link> tag specifies the external CSS file's location using the href attribute and the file's relationship to the HTML file using the rel attribute. For example:

```
<link rel="stylesheet" href="login.css">
<link rel="stylesheet" href="main.css">
```

For the login.css

It has been implemented in:

i. login.php

Below are some styling examples set in the login.css:

```
.center {
  display: flex;
  align-items: center;
  justify-content: center;
}
```

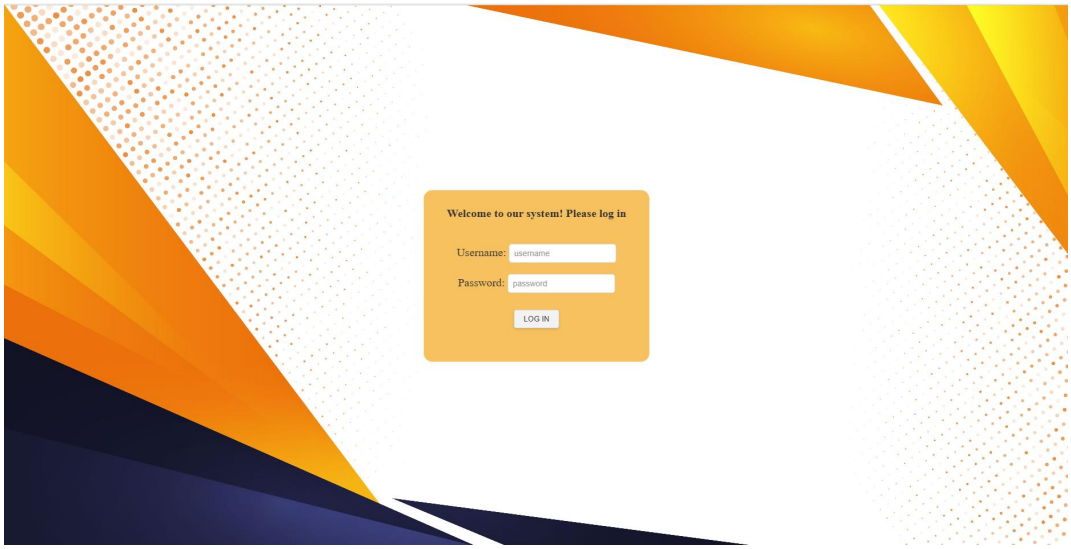
It is used to define a flex container to horizontally and vertically centre its child elements and aligns items (child elements) in the centre both horizontally and vertically

```
.login {
  border-radius: 15px;
  opacity: 0.8;
  background-color: rgb(245, 178, 55);
  text-align: center;
  padding: 30px 40px;
  font-size: 20px;
}
```

It is used to style the login element.

```css
body {
  background-image: url("login-bg.jpg");
  background-size: cover;
  background-repeat: no-repeat;
  background-position: center center;
}
```

It is used to set the background image of the body to "login-bg.jpg", set the background-size to cover, which scales the image to cover the entire background, set the background-repeat to no-repeat, so the image is not repeated and set the background position to the centre of the body.



This is the view of the login.php by using the login.css

For the main.css:

It has been implemented in:

i. mainPage.php
ii. application_form.php
iii. coordinator_form.php
iv. organization_form.php
v. student_form.php
vi. training_session_form.php
vii. update_coordinator_form.php
viii. update_organization_form.php
ix. update_student_form.php
x. update_training_session_form.php
xi. view_application.php

xii. view_coordinator.php

xiii. view_organization.php

xiv. view_student.php

xv. view_training_session.php

In main.css, the CSS styling sets are designed based on the view of different parts in the pages of the system. Hence, the styling sets have been categorised into different groups for producing personalised and customised visual effects on certain parts of the pages.
Below is the combination of styling sets that are categorised into different parts on the pages:

1.)Layout of the page

```css
.grid-container {
  display: grid;
  grid-template-rows: 80px 1fr; /* Adjusted height of the first row */
  grid-template-columns: 1fr 5fr; /* Adjusted width of columns */
  height: 100%;
  width: 100%;
}
```

This CSS code snippet creates a grid container with two rows and two columns. The first row has a fixed height of 80 pixels, while the second row occupies the remaining vertical space.

2.) Viewing of the logo class of the page

```css
.logo {
  grid-row: 1 / 2;
  grid-column: 1 / 3;
  background: url("bg.jpg") bottom center/cover no-repeat;
  font-size: 16px;
  font-family: "Kalam", "Instrument Serif", serif;
  display: flex;
  align-items: center;
  justify-content: space-between;
  padding: 20px;
}
```

It sets the logo class to span from the first row to the second row and from the first column to the third column in the grid display.

This is how the logo class and its child elements look with the styling of the main.css

3.) Viewing of the left column of the page

```
.borderless-button {
  border: none;
  background-color: transparent;
}
```

With these properties, elements with class '. borderless-button' will have no border and a transparent background, giving them a borderless and transparent appearance. The borderless visual appearance could be also seen in the "Update" button and "Delete button" in **8.) Viewing of the "Update and Delete" Button**

```
.left-column-button {
  margin-top: 20px;
  margin-bottom: 20px;
  margin-left: auto;
  margin-right: auto;
  font-size: 15px;
  align-items: center;
  cursor: pointer;
  height: 30px;
  width: 100%;

  /* Animation properties */
  animation-name: slideIn;
  animation-duration: 0.5s;
  animation-timing-function: ease-in-out;
  animation-fill-mode: forwards;
  opacity: 0;
}
```

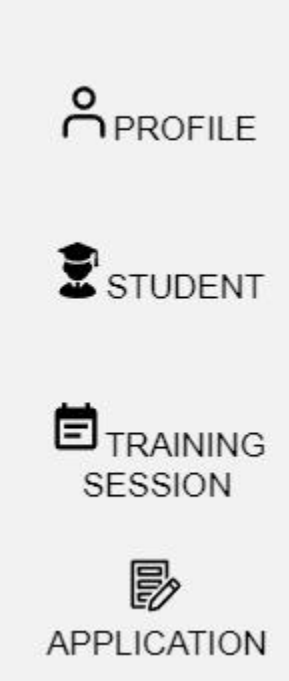
left -column-button class styles the button to have centred alignment, a specific size, a margin for positioning, and an animation to slide in with a fade effect

```
@keyframes slideIn {
  0% {
    opacity: 0;
    transform: translateX(-10%);
  }
  100% {
    opacity: 1;
    transform: translateX(0);
  }
}
```

This animation creates a slide-in effect for the button, starting from the left and gradually becoming fully visible as it moves back to its original position

```css
.left-column1,
.left-column2 {
  grid-row: 2/3;
  grid-column: 1/2;
  background: rgba(242, 242, 242, 255);
  display: flex;
  padding: 20px;
  flex-direction: column;
  text-align: center;
}
```

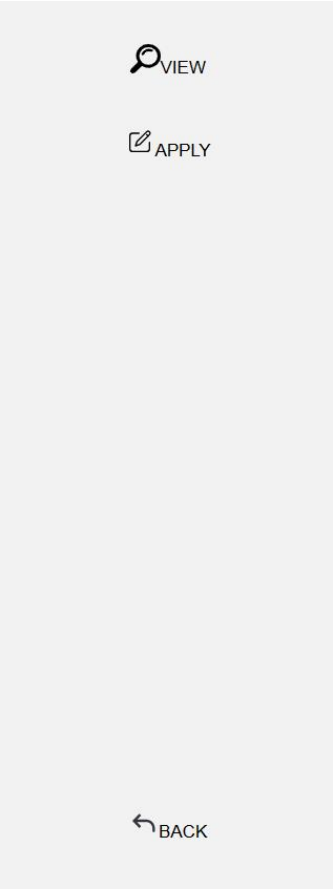
It sets the left-column2 and left-column1 classes to span from the second row to the third row and from the first column to the second column.



This is how the left column in mainPage.php look with the styling of the main.css

```css
.left-column2 {
  justify-content: space-between;
}
```

It will create a layout where the item inside ‘left-column2’ are evenly spaced along the main axis, with the first item aligned to the start and the last item aligned to the end

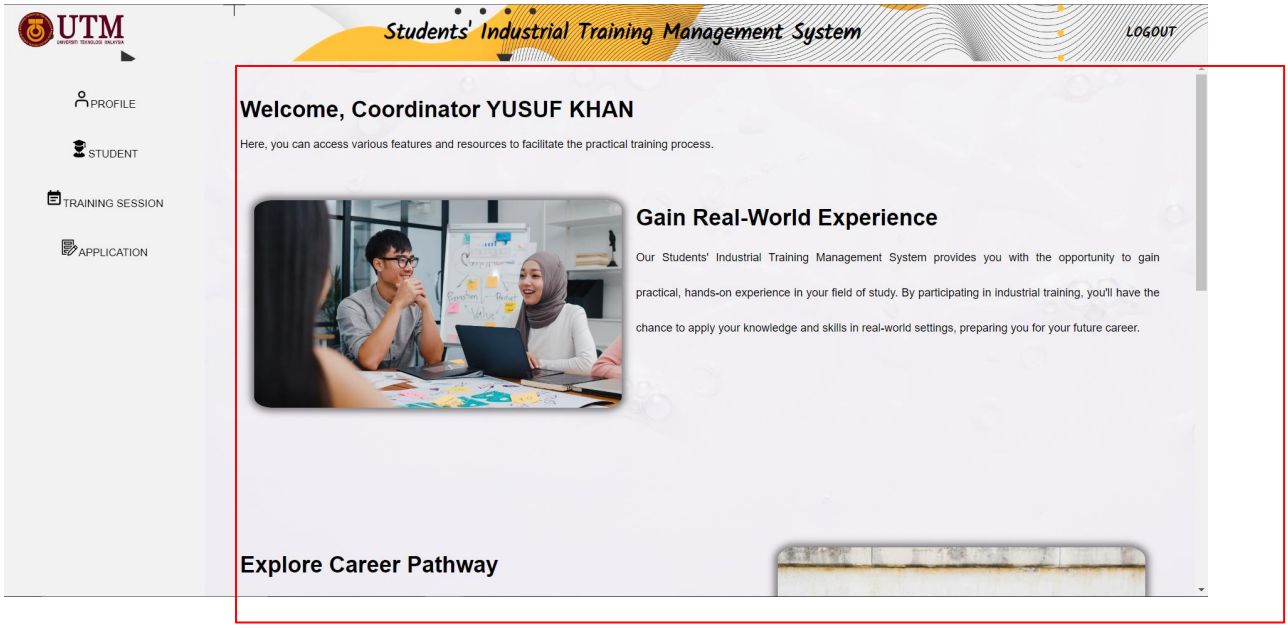


This is the view of the left-column2 after the above styling.

4.) Viewing of right-column of the page

```
.right-column {
  overflow: auto;
  grid-row: 2/3;
  grid-column: 2/3;
  background: url("bg3.jpg") center center/cover no-repeat;
}
```

It specifies how the content of the right-column class that exceeds the available space within the "right-column" element should be handled. When the content exceeds the space, it will create a scrollbar to allow scrolling within the element.



This is the view of the right-column after the above styling

5.) Viewing of Pictures in the Right Column of the Main Page

```css
.right-pic-left {
  border-radius: 5%;
  box-shadow: 0 0 10px 5px rgba(0, 0, 0, 0.5);
  margin: 0px 20px;
  clear: both;
  float: left;
}
```

```css
.right-pic-right {
  border-radius: 5%;
  box-shadow: 0 0 10px 5px rgba(0, 0, 0, 0.5);
  margin: 0px 20px;
  clear: both;
  float: right;
}
```

```css
.right-pic-left,
.right-pic-right {
  transform: scale(1);
  transition: transform 0.5s ease-in-out;
}
```

```css
.right-pic-left:hover,
.right-pic-right:hover {
  transform: scale(1.1);
}
```

```css
.right-content-1,
.right-content-2,
.right-content-3 {
  margin: 0px 0px 250px 0px;
  line-height: 50px;
  clear: both;
}
```

```css
.right-content-3 {
  margin: 0px 0px 100px 0px;
}
```

```css
.right-content-theme {
  font-weight: bold;
  font-size: 32px;
  /* margin-bottom: 20px; *
}
```

This is the view of the right-column after the above styling

6.) Viewing of the Logout Button

```
.logout {
  margin-right: 20px;
  font-size: 20px;
  font-weight: bold;
  font-family: "Kalam", "Instrument Serif", serif;
}
```

7.) Viewing of the “Updating and Deleting” Table

```css
/* CSS for View Table */

.right-column-table {
  border-collapse: collapse;
  width: 85%;
  margin-bottom: 1rem;
}

.right-column-table th,
.right-column-table td {
  border: 1px solid #ddd;
  padding: 0.5rem;
  text-align: center;
}

.right-column-table th {
  background-image: linear-gradient(to bottom, #ff9800, #ffb700);
  font-weight: bold;
}

.right-column-table tr:nth-of-type(even) {
  background-color: #f9f9f9;
}

table.right-column-table tr:not(:first-child):hover {
  background-color: rgb(200, 200, 200);
}
```

Training Session Name	Matric	Name	Address	Status
ONLINE RETAIL	A21EC0078	ABUBAKAR AIDA	31, Jalan Seri Permai, Taman Seri Permai, 58200 Kuala Lumpur, Malaysia	Pending
ONLINE RETAIL	A21EC0101	NG KAI ZHENG	67, Jalan Anggerik, Taman Anggerik, 75450 Melaka, Malaysia	Pending
ONLINE RETAIL	A21EC0148	YEO CHUN TECK	56, Jalan Bunga Raya, Taman Bukit Indah, 80200 Johor Bahru, Johor, Malaysia	Pending

This is the view of the table after the above styling.

8.) Viewing of the “Update and Delete” Button

```css
/* Update and delete Button CSS */

.right-column-table button {
  padding: 5px 10px;
  text-align: center;
  border-radius: 5px;
  transition: background-color 0.3s ease, transform 0.3s ease;
}

.right-column-table button:hover {
  background-color: rgba(243, 165, 49, 0.8);
  transform: scale(1.15);
}

.right-column-table button u {
  text-decoration: none;
  color: white;
}

.borderless-button.update {
  background-color: #007bff;
}

.borderless-button.delete {
  background-color: #dc3545;
}
```

This is the view of the “Update” button and “Delete” buttons after the above styling.

9.) Viewing of Menu button, Logout button, and Table button(Except of Application Table Button) when hovering

```
.logout:hover,
.left-column-button:hover,
.right-column-table button:hover {
  cursor: pointer;
  color: #ffb700;
}
```

These styles customise the appearance of the elements when they are hovered over, making them more interactive and visually appealing

Before hovering	After hovering
	
	
	

These are the view of the Menu button, Logout Button and “Table” button after the above styling.

10.) Viewing of Radio button

```css
input[type="radio"] {
  width: 20px;
  height: 20px;
  position: relative;
  top: 3px;
}
```

```css
input[type="radio"]:checked {
  accent-color: rgba(255, 195, 58, 255);
}
```

This is the view of the radio input after the above styling

11.) Viewing of Submit button or Submit input

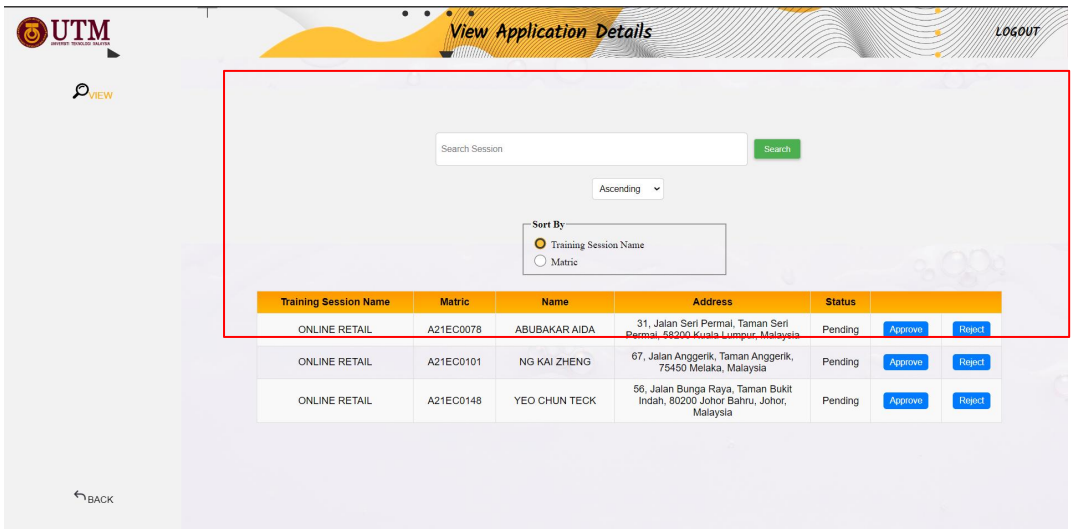
```css
button[type="submit"]:hover,
input[type="submit"]:hover {
  cursor: pointer;
  background-image: radial-gradient(circle at center, #ffcc3a, #ff7f50);
}

input[type="submit"],
button[type="submit"] {
  background-color: #4caf50;
  color: white;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
  border: 1px solid #ccc;
  border-radius: 4px;
  padding: 8px 16px;
  margin: 10px;
  transition: background-color 0.3s ease;
}
```

Before Hovering over	After Hovering over
	

This is the view of the submit input when hovering over after the above styling

12.) Viewing of "Searching and Sorting" Panel

```css
.panel {
  background-color: #f2f2f2;
  padding: 10px;
  border-radius: 4px;
  margin-bottom: 10px;
  margin-top: 50px;
  text-align: center;
  display: flex; /* Add flex display */
  justify-content: center; /* Align items horizontally to the center */
  align-items: center; /* Align items vertically to the center */
}

.panel .searchbox {
  margin-top: 10px;
  margin-right: 10px; /* Add margin to separate elements */
  display: flex;
}

.panel .searchbox input[type="search"] {
  width: 500px;
}

.panel .submit input[type="submit"] {
  background-color: #4caf50;
  color: #fff;
}

.panel .sortfield {
  margin-left: 10px; /* Add margin to separate elements */
  margin-top: 10px;
}

.panel .sortfield {
  display: inline-block;
  position: relative;
}

.searchbox,
.sortfield {
  margin-bottom: 10px;
}
```



This is the view of the “Update” button and “Delete” button after the above styling.

<u>13.) Viewing of “Approve” button and “Reject” button</u>

```css
.Evaluate {
  background-color: #007bff; /* Blue color */
  color: white;
  padding: 8px 16px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
  border-radius: 4px;
  border: none;
  cursor: pointer;
  font-size: 14px;
  transition: background-color 0.3s ease;
}

.Evaluate:disabled {
  background-color: #cccccc;
  cursor: not-allowed;
}

#Approve:hover {
  background-color: #4cc86d; /* Darker blue color */
}

#Reject:hover {
  background-color: #c8614c; /* Darker red color */
}

.error-message {
  color: red;
}
```

After the above styling,

This is the view of the “Approve” and “Reject” buttons when not hovering over

This is the view of the “Approve” and “Reject” buttons when hovering over

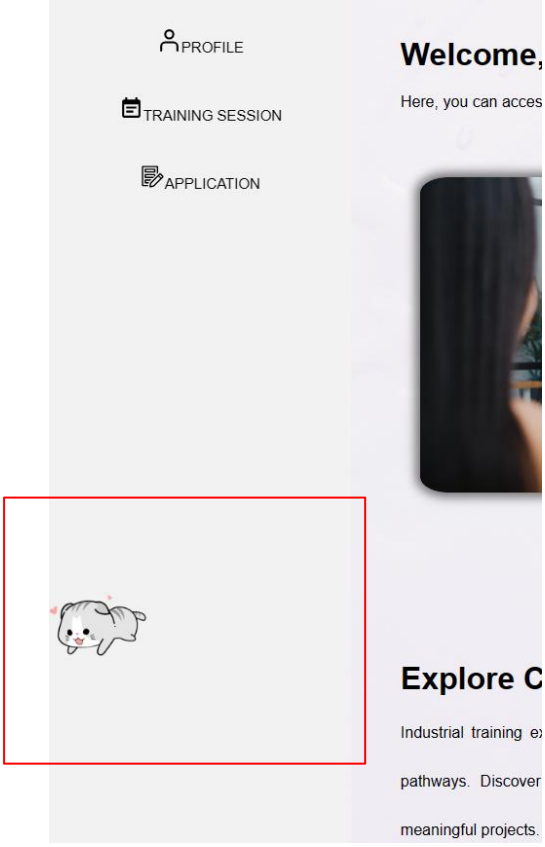
This is the view of “Approve” or “Reject” button when it is disabled

By using external CSS files and properly linking them to the respective HTML pages, the project maintains a well-organised and modular structure, simplifying the management and customization of styles throughout the website.

14.) Cat

```css
/* Addition css for cat */
.cat {
  cursor: pointer;
  width: 100px;
  height: 100px;
  position: absolute;
  top: 70%;
  left: 0;
  z-index: 1;
}
```

```php
<?php if ($_SESSION["LEVEL"] == 3) {?>
<img class="cat" src="image/addition/cat.png" alt="Cat" />
<script>
var minX = window.innerWidth * 0.11;
var minY = window.innerHeight * 0.50;
var maxY = window.innerHeight * 0.90;
var audio = new Audio("image/addition/meow.mp3");

function randomPosition() {
    var x = Math.random() * minX;
    var y = Math.random() * (maxY - minY) + minY;
    document.querySelector(".cat").style.top = y + "px";
    document.querySelector(".cat").style.left = x + "px";
    if (audio.paused) {
        audio.play();
    } else {
        audio.currentTime = 0;
    }
}

document.querySelector(".cat").onclick = randomPosition;
</script>
<?php }?>
```



This part combines with javascript to create a clickable cat, when clicked then the cat will randomly move to a settings area and play the soundtrack “meow”. This cat is only for the student to make the website more attractive.

Conclusion

The project uses sessions, cookies, user input validation, and dynamic HTML through JavaScript to enhance the user experience and functionality of the system.

- Sessions are used to store user-specific information, such as login credentials, allowing the website to maintain its state and provide a personalised experience.
- Cookies are used to enhance the login process by storing the user's credentials and enabling automatic login for returning users.
- User input validation is used to ensure that the data entered by users is valid and meets the required criteria. This helps to prevent errors and ensures that the data is accurate.
- Dynamic HTML through JavaScript is used to dynamically generate HTML content, which can be used to update the user interface without having to reload the entire page.

The use of these technologies in the project contributes to a user-friendly and secure system. Sessions and cookies help to make the login process more convenient for returning users, while user input validation helps to prevent errors and ensure that the data is accurate. Dynamic HTML through JavaScript can be used to update the user interface without having to reload the entire page, which can improve the user experience.

Overall, the project is well-designed and implemented. The use of sessions, cookies, user input validation, and dynamic HTML through JavaScript all contribute to a user-friendly and secure system.