COS 108 – Web Programming and Application

Name : Chaw Thiri Win

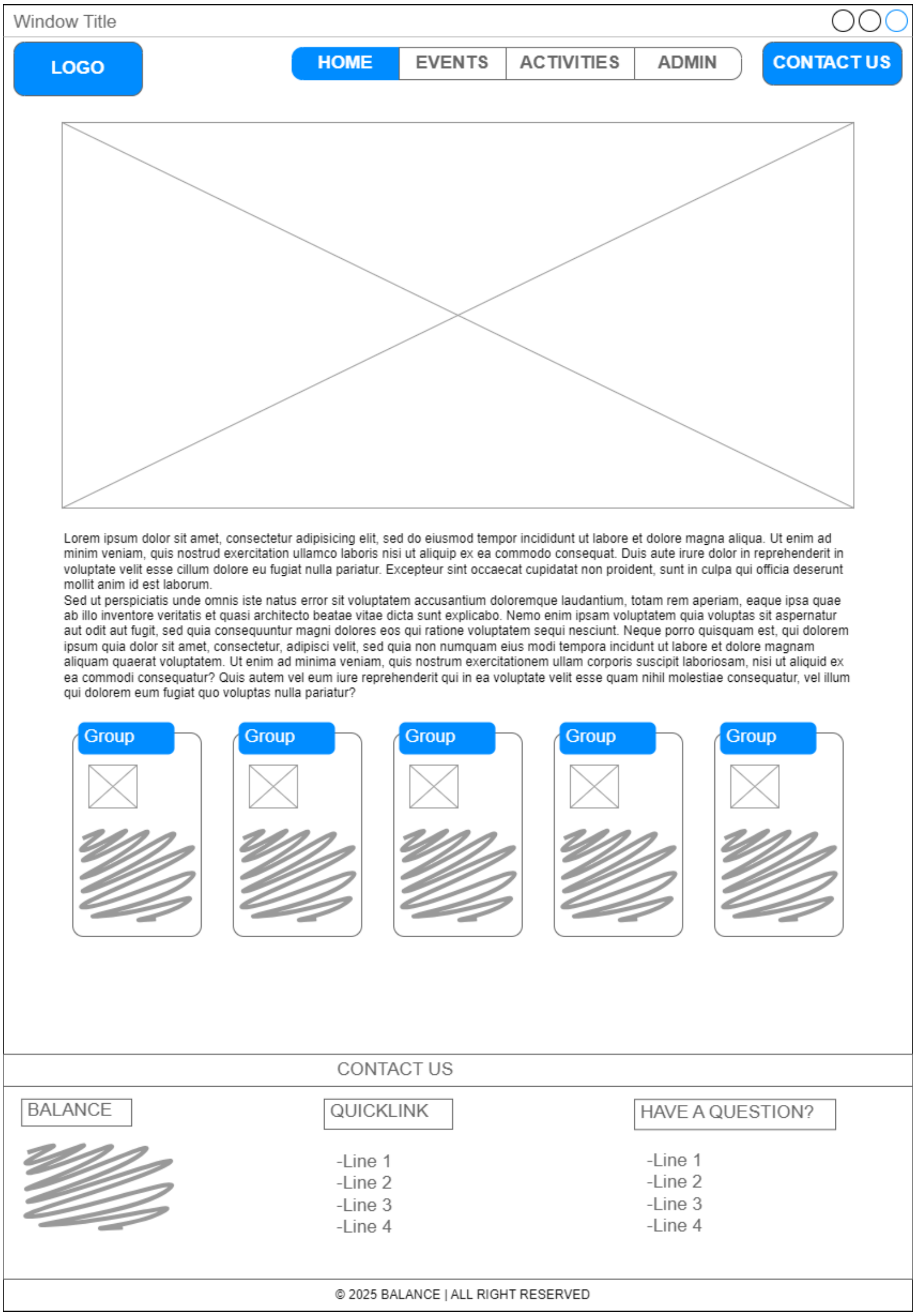
Contents

System Overview

The "Balance" website redesign seeks to establish a vibrant space for health enthusiasts, encouraging work-life balance and healthy lifestyles by disseminating health-related information and making it easier to sign up for weekly events and activities. The website, which will be constructed with HTML, CSS, and JavaScript, will have an intuitive user interface with data validation, interactive forms, and a consistent design throughout at least five pages. In order to ensure a thorough and interesting experience for both the general public and company employees, it will also integrate database connectivity via PHP and MySQL to collect user information for event enrollment and give administrators safe access to manage user data.

1. Storyboard

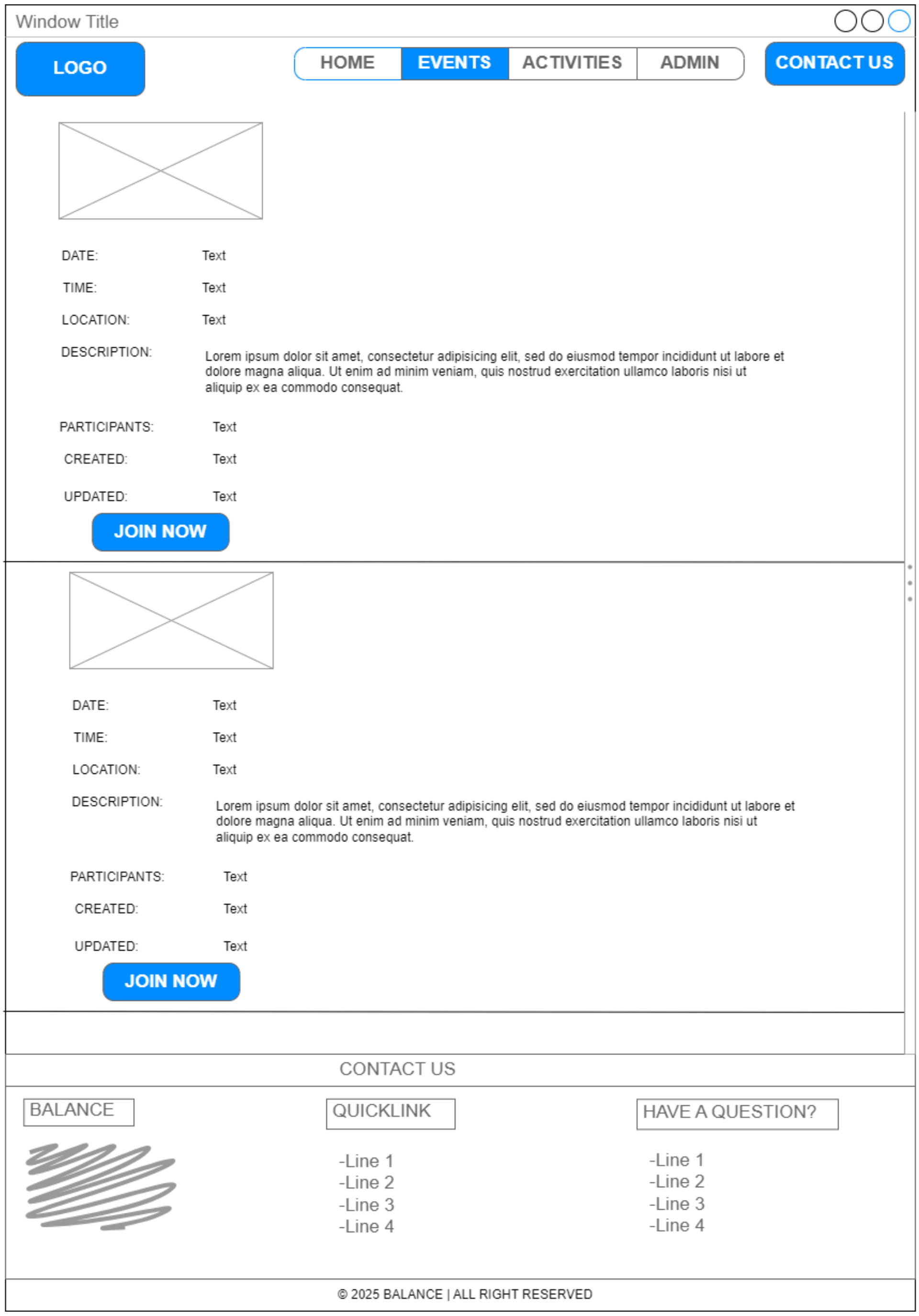
The is the storyboard of the project. Here we will see the Mock-ups design of the website of each pages.



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. Neque porro quisquam est, qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit, sed quia non numquam eius modi tempora incidunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur, vel illum qui dolorem eum fugiat quo voluptas nulla pariatur?

Figure – 1.1 : Home Form



DATE: Text

TIME: Text

LOCATION: Text

DESCRIPTION: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

PARTICIPANTS: Text

CREATED: Text

UPDATED: Text

DATE: Text

TIME: Text

LOCATION: Text

DESCRIPTION: Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

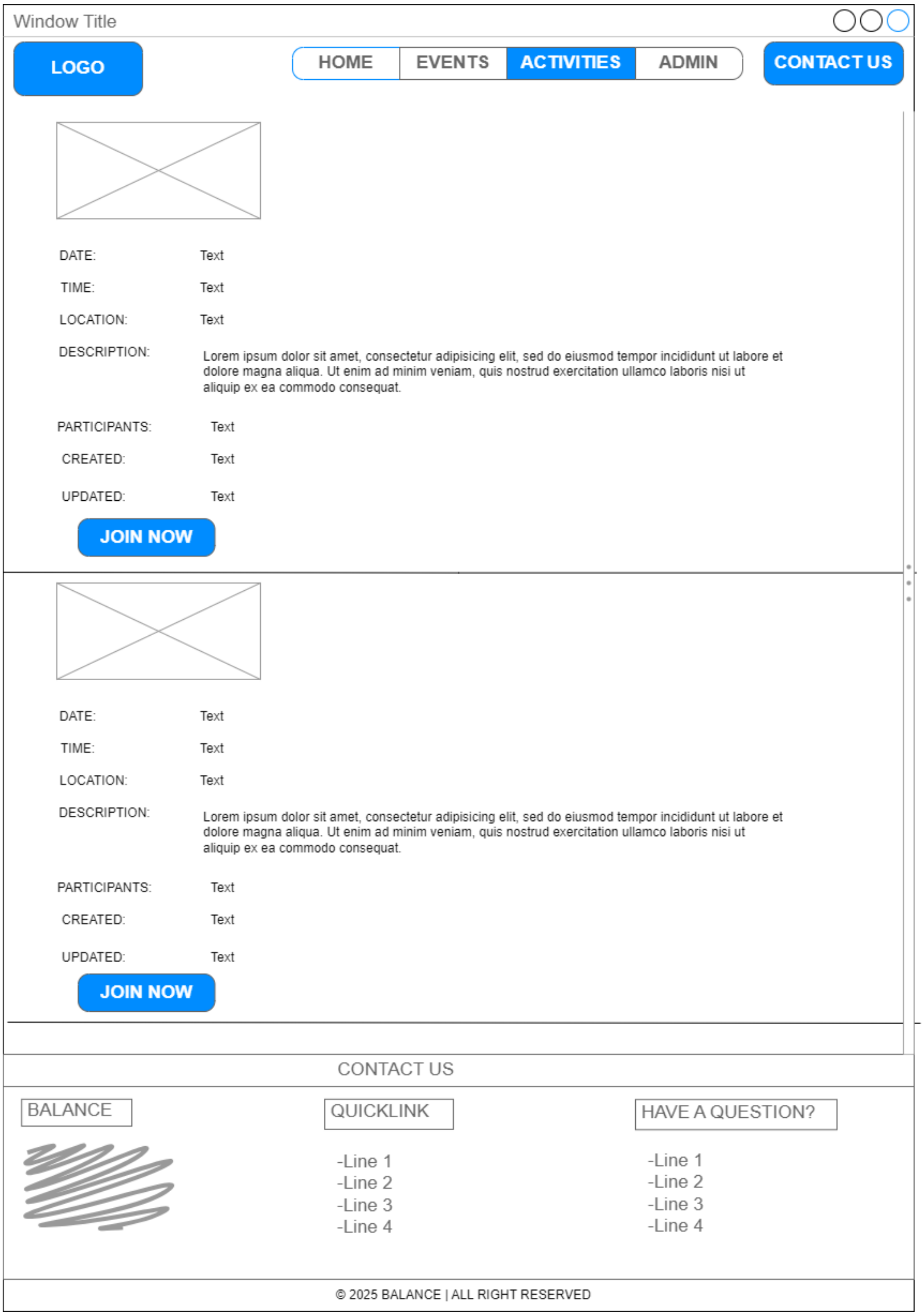
PARTICIPANTS: Text

CREATED: Text

UPDATED: Text

© 2025 BALANCE | ALL RIGHT RESERVED

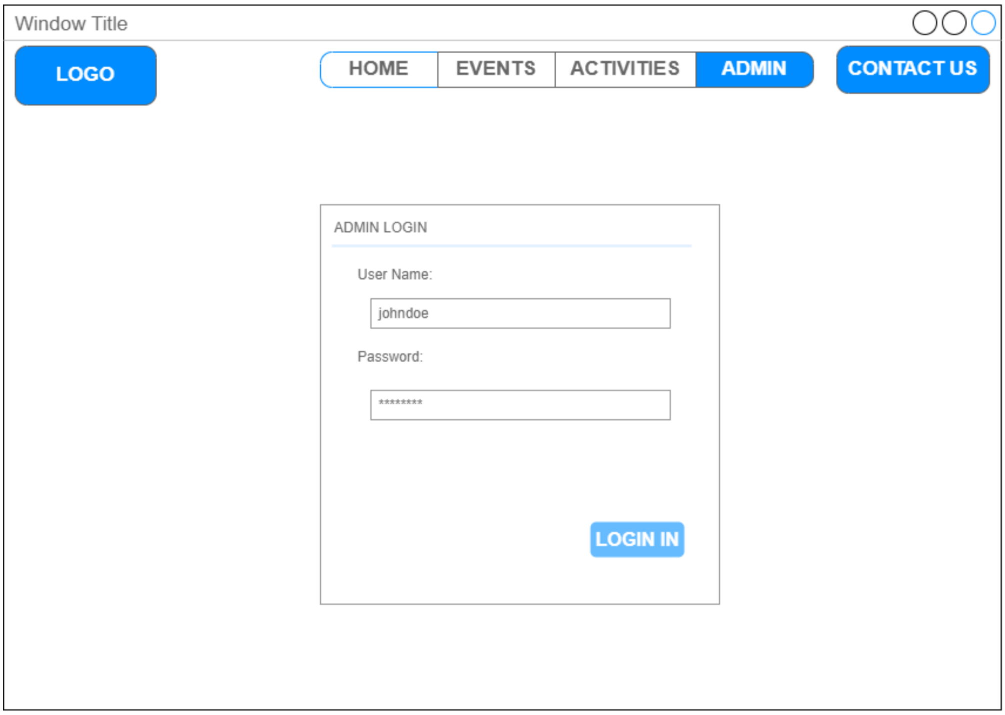
Figure – 1.2 : Events Form



DATE:	Text
TIME:	Text
LOCATION:	Text
DESCRIPTION:	Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
PARTICIPANTS:	Text
CREATED:	Text
UPDATED:	Text

DATE:	Text
TIME:	Text
LOCATION:	Text
DESCRIPTION:	Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
PARTICIPANTS:	Text
CREATED:	Text
UPDATED:	Text

Figure – 1.3 : Activities Form

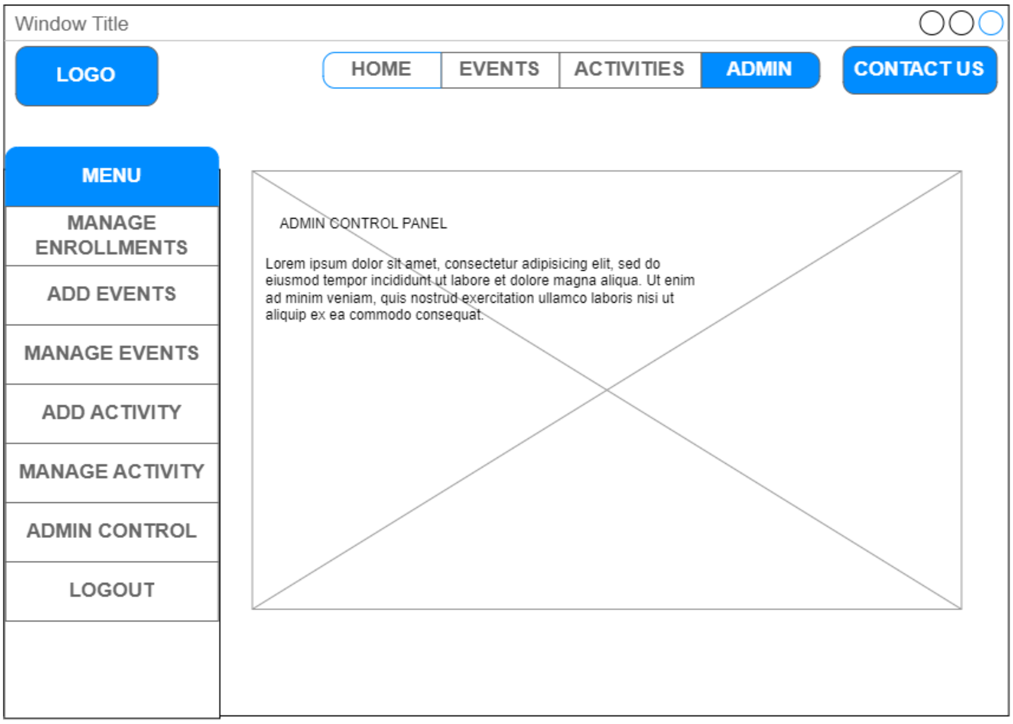


ADMIN LOGIN

User Name: johndoe

Password: ********

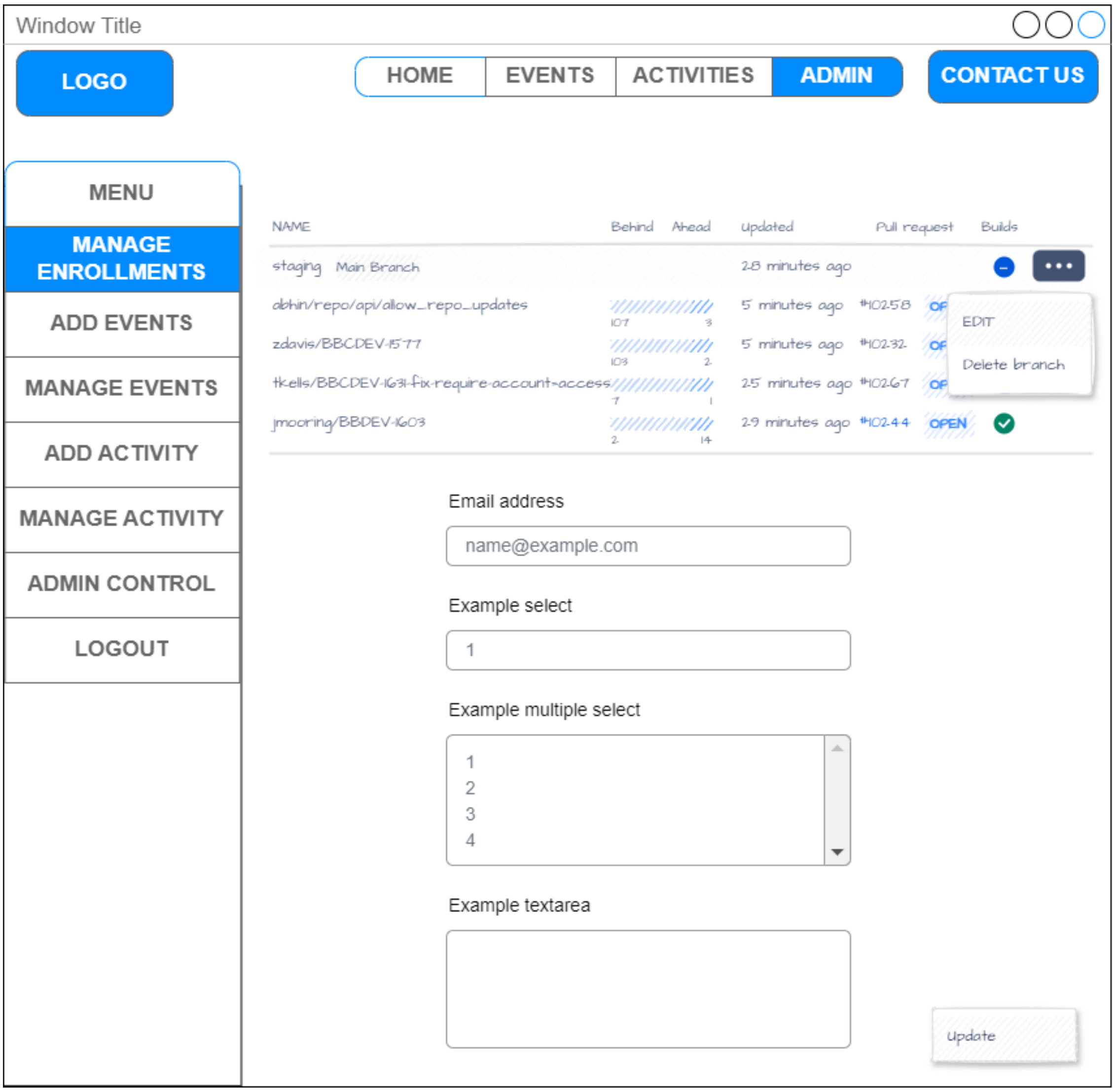
Figure – 1.4 : Admin Login Form



ADMIN CONTROL PANEL

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Figure – 1.5 : Admin Home Form



NAME	Behind	Ahead	Updated	Pull request		Builds
staging Main Branch			28 minutes ago			⊖
abhin/repo/api/allow_repo_updates	107	3	5 minutes ago	#10258	OP	
zdavis/BBCDEV-1577	103	2	5 minutes ago	#10232	OP	
tkells/BBCDEV-1631-fix-require-account-access	7	1	25 minutes ago	#10267	OP	
jmooring/BBDEV-1603	2	14	29 minutes ago	#10244	OPEN	✓

Email address

name@example.com

Example select

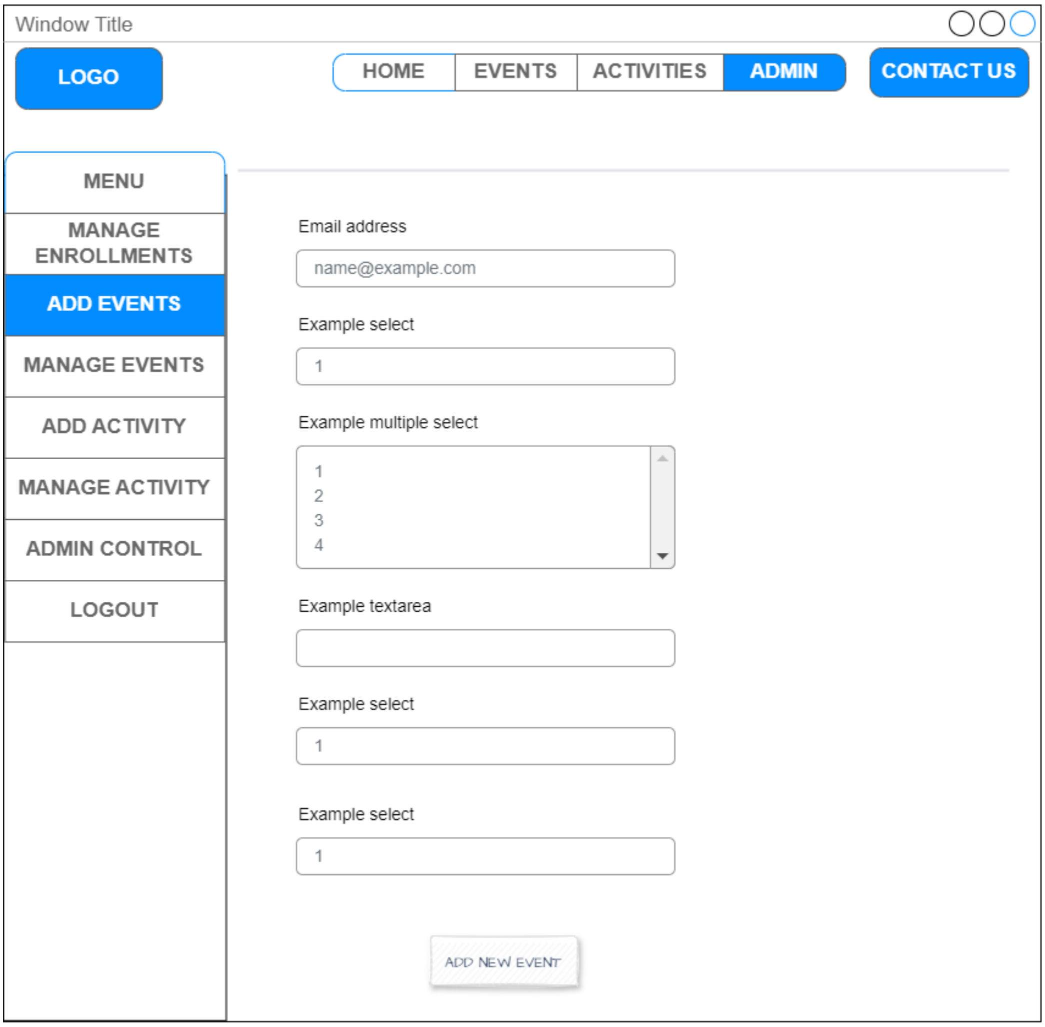
1

Example multiple select

1
2
3
4

Example textarea

Figure – 1.6 : Manage Enrollments Form



Email address

name@example.com

Example select

1

Example multiple select

1
2
3
4

Example textarea

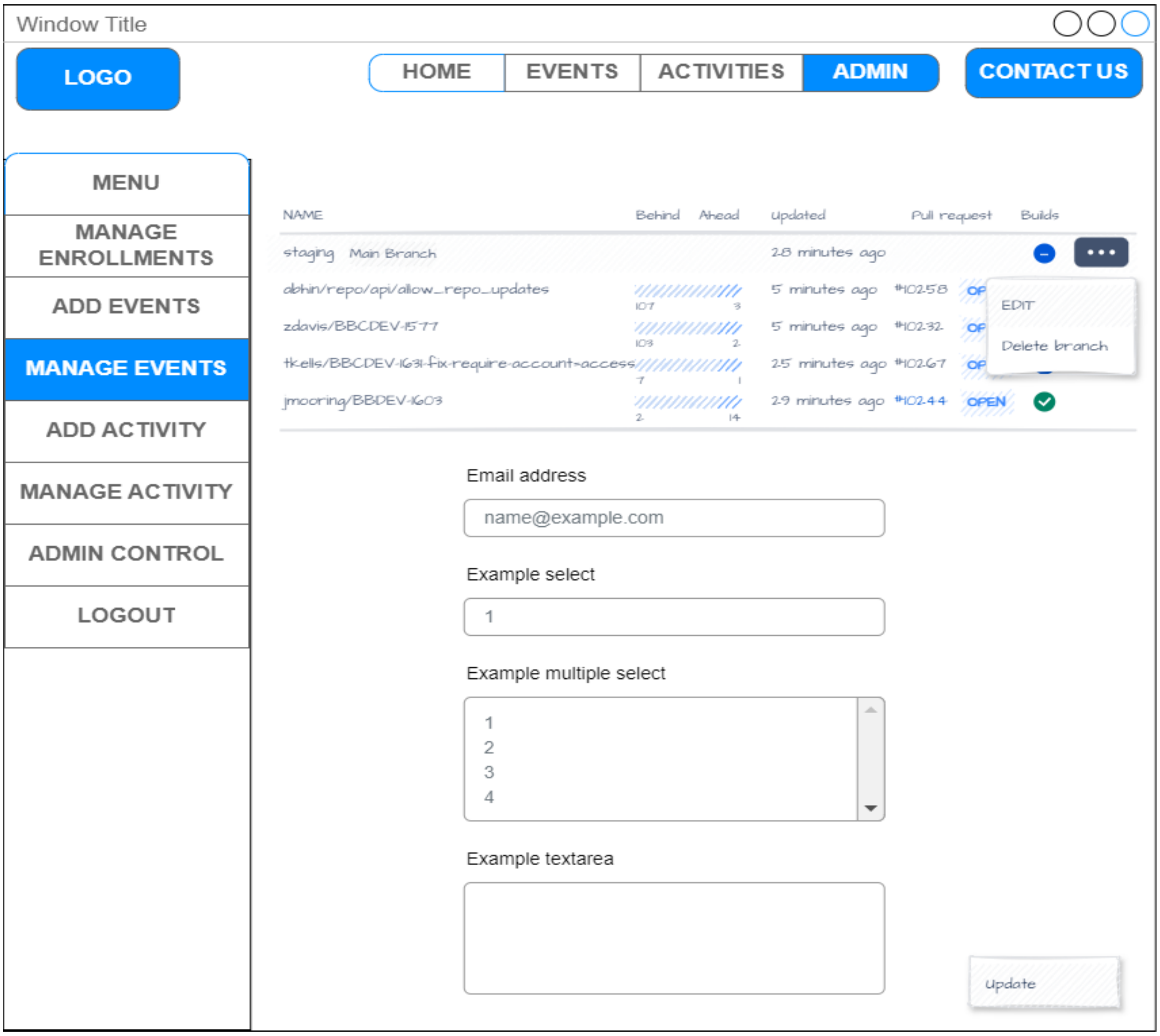
Example select

1

Example select

1

Figure – 1.7 : Add Events Form



NAME	Behind	Ahead	Updated	Pull request		Builds
staging Main Branch			28 minutes ago			−
abhin/repo/api/allow_repo_updates	107	3	5 minutes ago	#10258	OP	
zdavis/BBCDEV-1577	103	2	5 minutes ago	#10232	OP	
tkells/BBCDEV-1631-fix-require-account-access	7	1	25 minutes ago	#10267	OP	
jmooring/BBDEV-1603	2	14	29 minutes ago	#10244	OPEN	✓

Email address

name@example.com

Example select

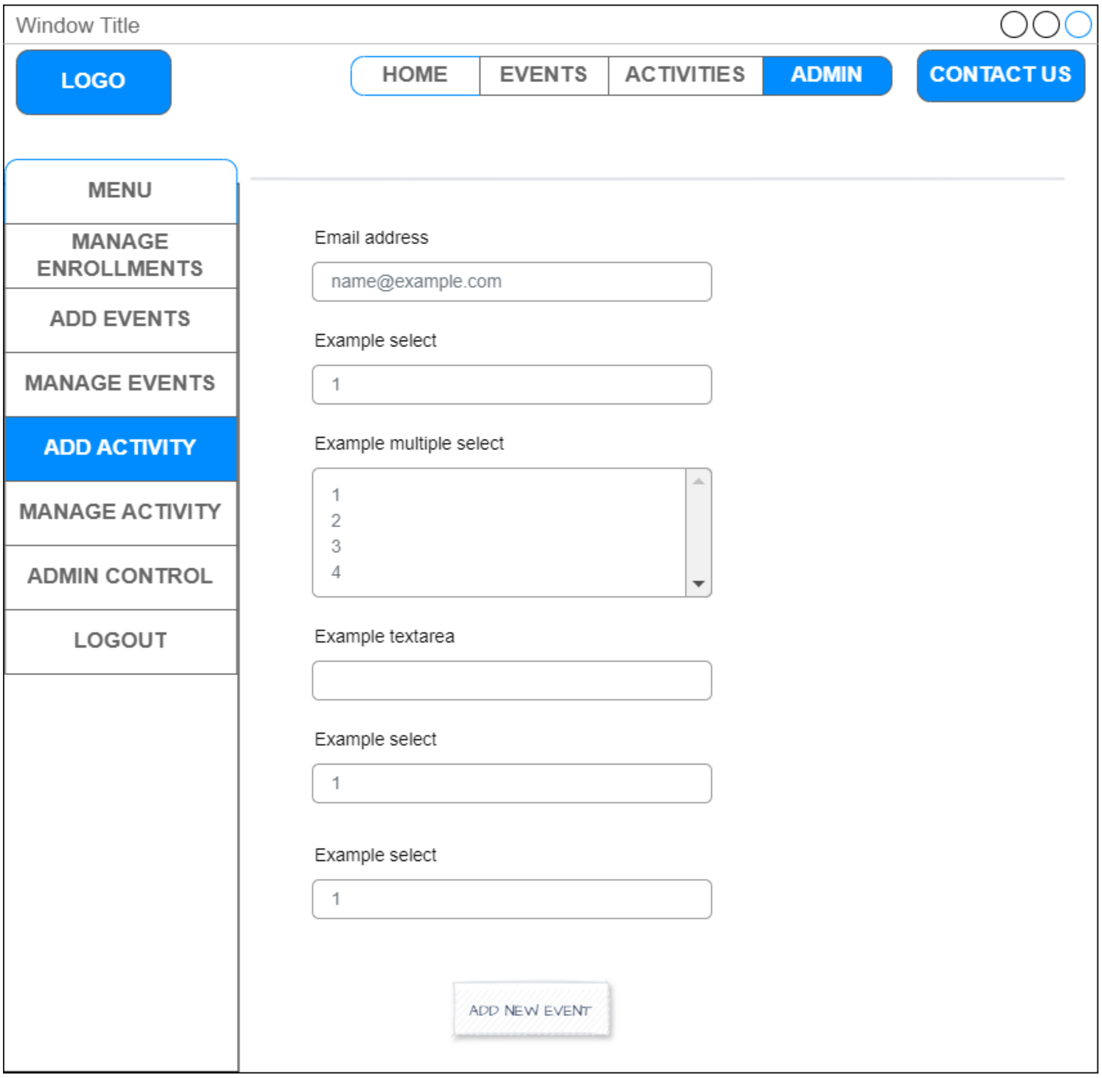
1

Example multiple select

1
2
3
4

Example textarea

Figure – 1.8 : Manage Events Form



Email address

name@example.com

Example select

1

Example multiple select

1
2
3
4

Example textarea

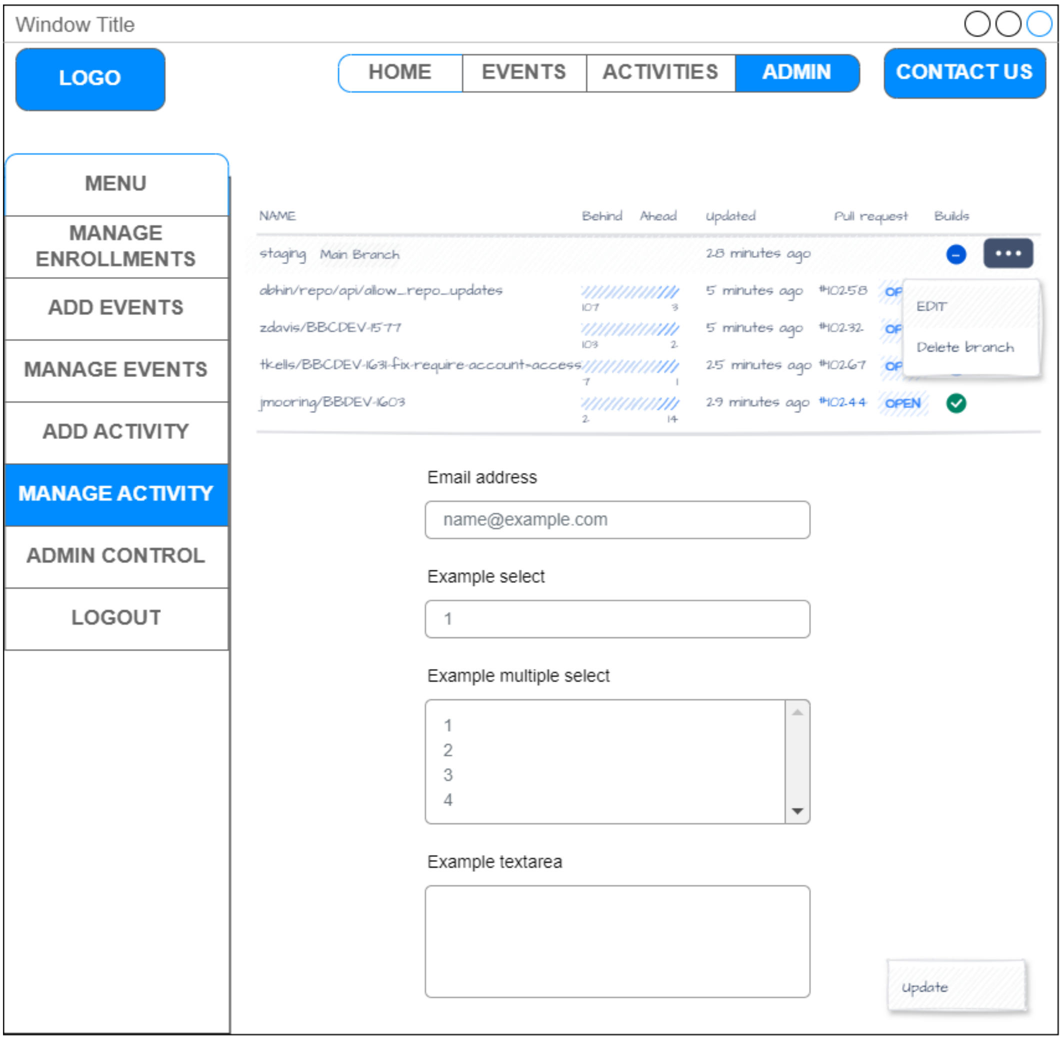
Example select

1

Example select

1

Figure – 1.9 : Add Activity Form



NAME	Behind	Ahead	Updated	Pull request		Builds
staging Main Branch			28 minutes ago			●
abhin/repo/api/allow_repo_updates	107	3	5 minutes ago	#10258	OP	
zdavis/BBCDEV-1577	103	2	5 minutes ago	#10232	OP	
tkells/BBCDEV-1631-fix-require-account-access	7	1	25 minutes ago	#10267	OP	
jmooring/BBDEV-1603	2	14	29 minutes ago	#10244	OPEN	✓

Email address

name@example.com

Example select

1

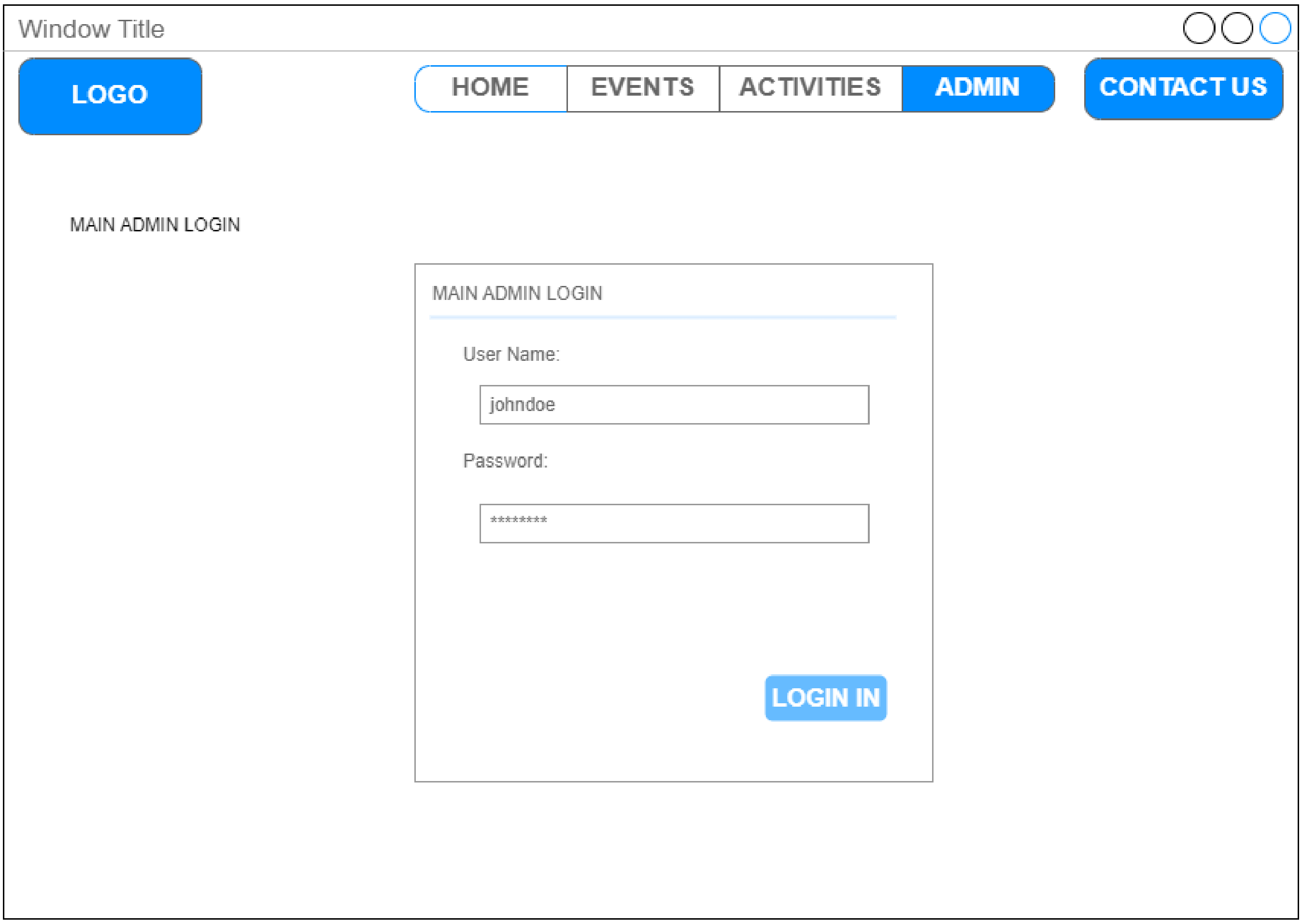
Example multiple select

1
2
3
4

Example textarea

,

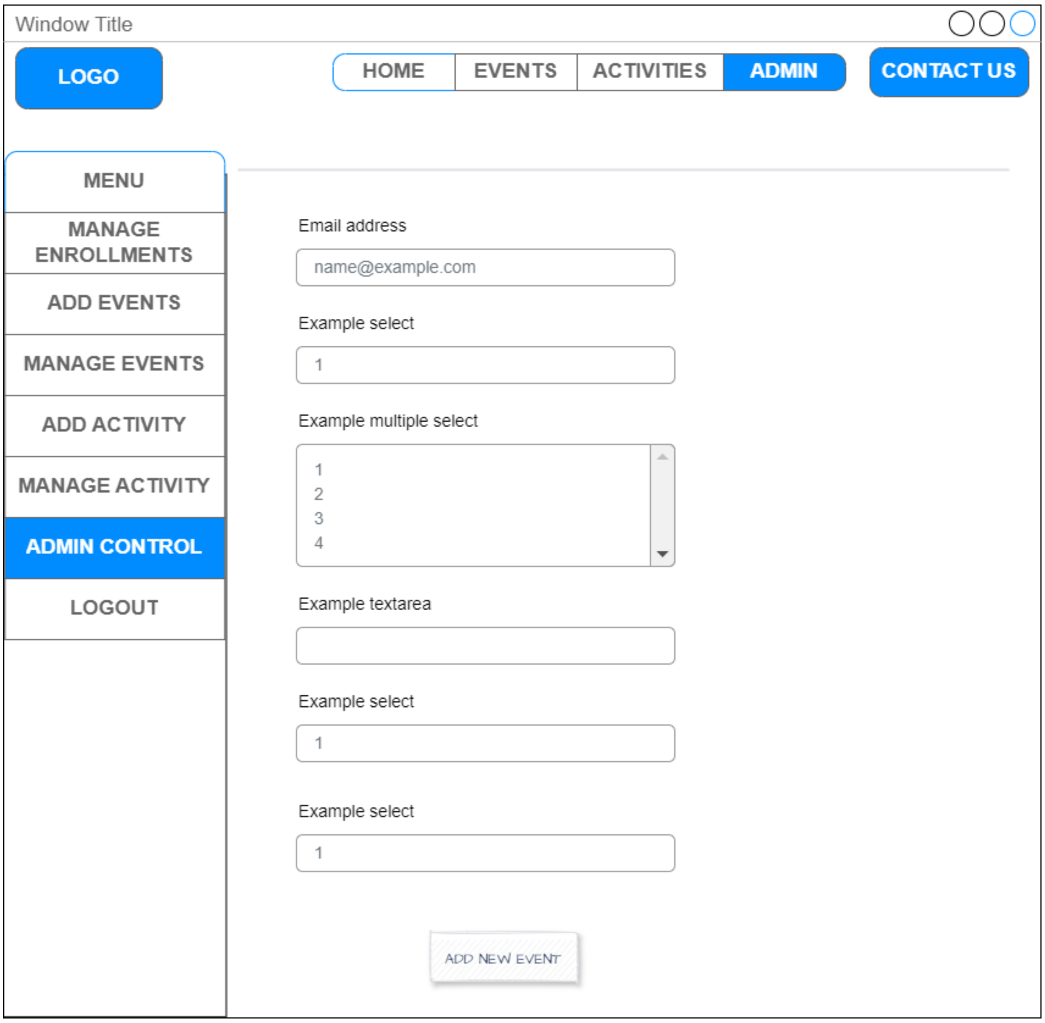
Figure – 1.10 : Manage Activity Form



User Name: johndoe

Password: ********

Figure – 1.11 : Main Admin Login Form



Email address
name@example.com

Example select
1

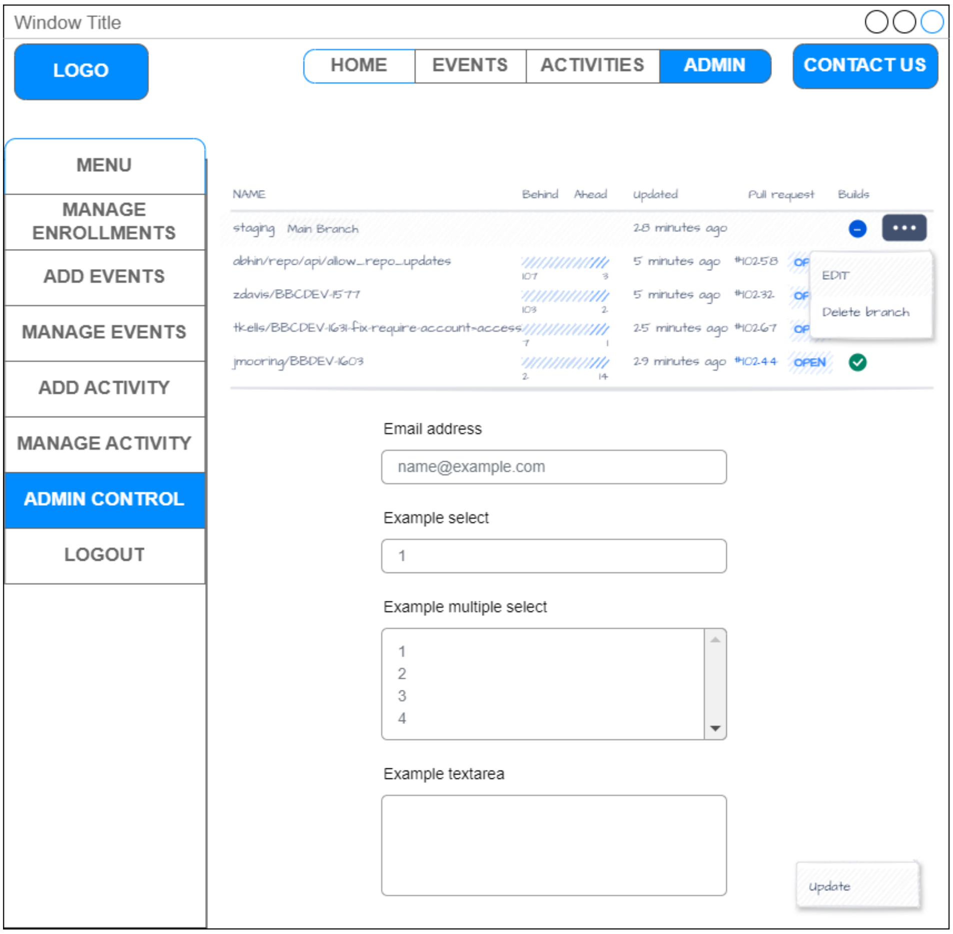
Example multiple select
1
2
3
4

Example textarea

Example select
1

Example select
1

Figure – 1.12 : Add Admin Form



NAME	Behind	Ahead	Updated	Pull request	Builds
staging Main Branch			28 minutes ago		−
abhin/repo/api/allow_repo_updates	107	3	5 minutes ago	#10258 OP	
zdavis/BBCDEV-1577	103	2	5 minutes ago	#10232 OP	
tkells/BBCDEV-1631-fix-require-account-access	7	1	25 minutes ago	#10267 OP	
jmooring/BBDEV-1603	2	14	29 minutes ago	#10244 OPEN	✓

Email address

name@example.com

Example select

1

Example multiple select

1
2
3
4

Example textarea

Figure – 1.13 : Manage Admin Form

2. Detailed Sitemaps

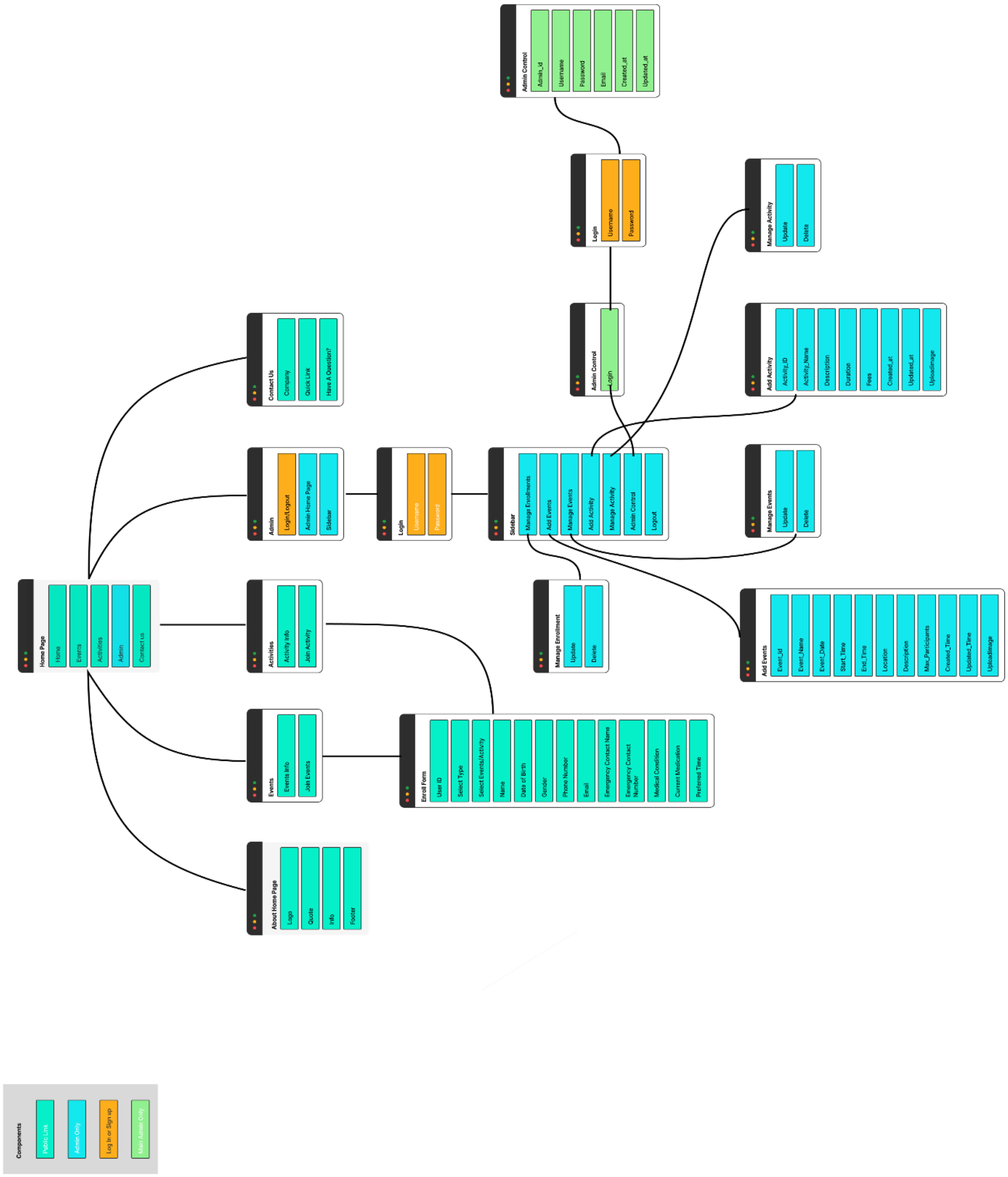
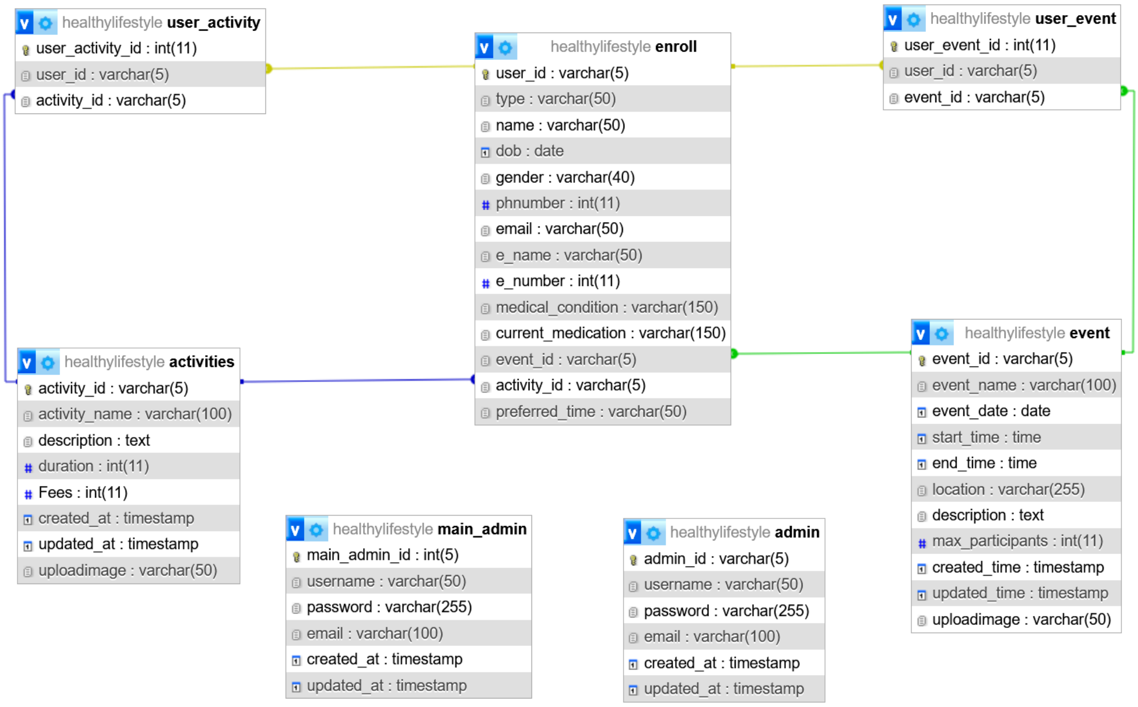


Figure – 2 : SITEMAP OF WORK LIFE BALANCE PROJECT

3. Database Design



healthylifestyle user_activity
user_activity_id : int(11)
user_id : varchar(5)
activity_id : varchar(5)

healthylifestyle enroll
user_id : varchar(5)
type : varchar(50)
name : varchar(50)
dob : date
gender : varchar(40)
phnumber : int(11)
email : varchar(50)
e_name : varchar(50)
e_number : int(11)
medical_condition : varchar(150)
current_medication : varchar(150)
event_id : varchar(5)
activity_id : varchar(5)
preferred_time : varchar(50)

healthylifestyle user_event
user_event_id : int(11)
user_id : varchar(5)
event_id : varchar(5)

healthylifestyle activities
activity_id : varchar(5)
activity_name : varchar(100)
description : text
duration : int(11)
Fees : int(11)
created_at : timestamp
updated_at : timestamp
uploadimage : varchar(50)

healthylifestyle event
event_id : varchar(5)
event_name : varchar(100)
event_date : date
start_time : time
end_time : time
location : varchar(255)
description : text
max_participants : int(11)
created_time : timestamp
updated_time : timestamp
uploadimage : varchar(50)

healthylifestyle main_admin
main_admin_id : int(5)
username : varchar(50)
password : varchar(255)
email : varchar(100)
created_at : timestamp
updated_at : timestamp

healthylifestyle admin
admin_id : varchar(5)
username : varchar(50)
password : varchar(255)
email : varchar(100)
created_at : timestamp
updated_at : timestamp

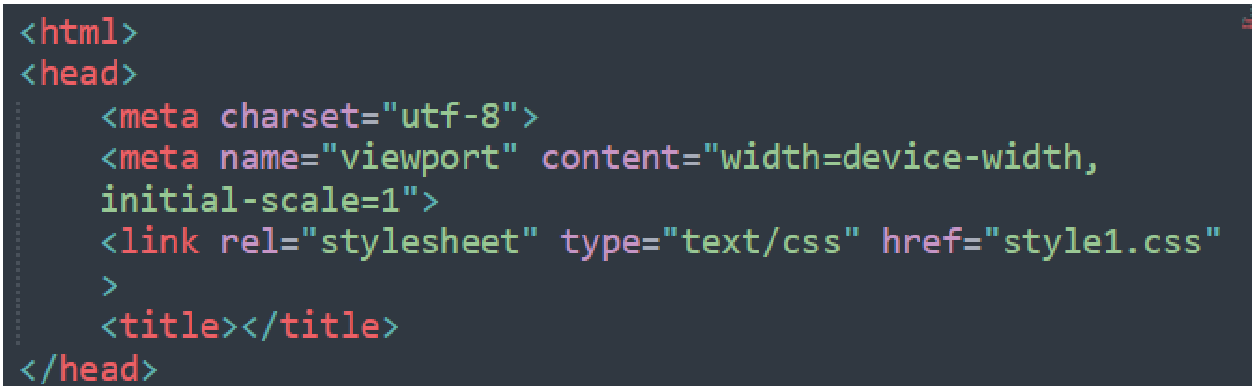
Figure – 3 : Database Design

Activities, events, and user enrollment are the main focuses of the database design. Important tables that contain information about names, descriptions, dates, and participation fees are `activities` and `event`. The `enroll` table contains foreign keys referencing `activities` and `event` and contains user data gathered during enrollment. The `user_activity` and `user_event` tables establish many-to-many connections. The `admin` and `main_admin` tables are used to manage administrative tasks, such as access control and user authentication. Because all of the tables are integrated, the "Balance" platform can manage actions, events, user data, and administrative access in a complete manner.

4. Task – 1 Key Features

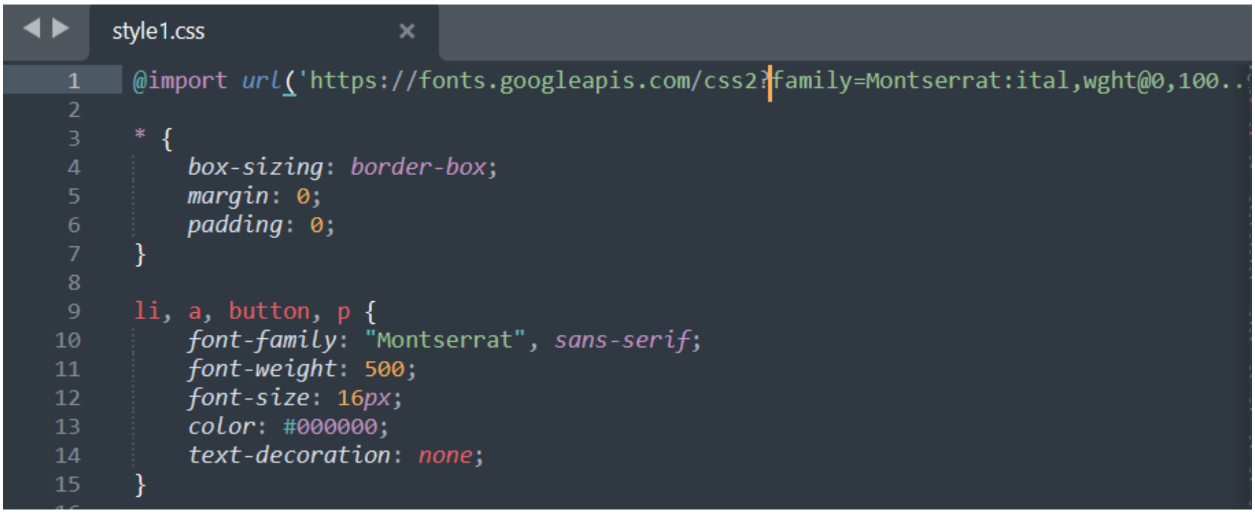
4.1. External Cascading Style

By attaching a distinct `.css` file to your HTML document, we can use external Cascading Style Sheets (CSS) to keep our code structure neat and orderly. Because changes to the external stylesheet are automatically reflected on all connected pages, this method makes updates efficient and enables us to apply uniform styles across several web pages. External CSS further speeds up load times for repeat visitors by allowing browser caching.



```
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width,
    initial-scale=1">
    <link rel="stylesheet" type="text/css" href="style1.css"
    >
    <title></title>
</head>
```

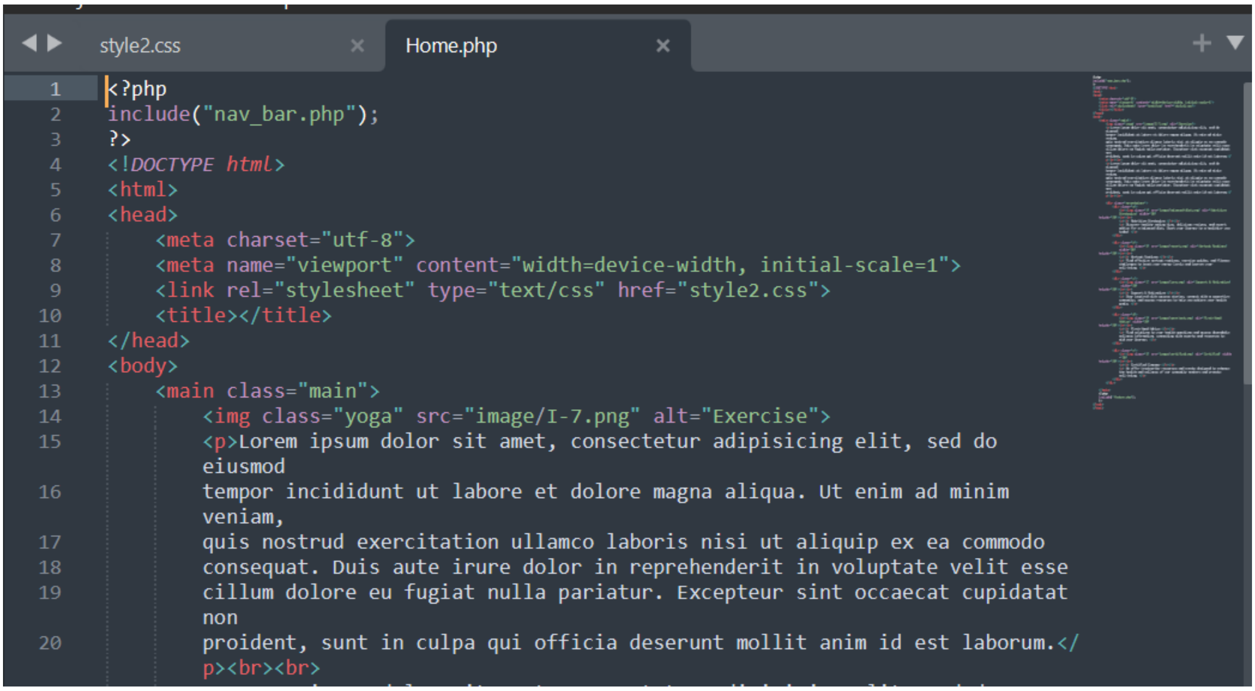
Figure – 4.1.1 : Linking Style in Main Form



style1.css

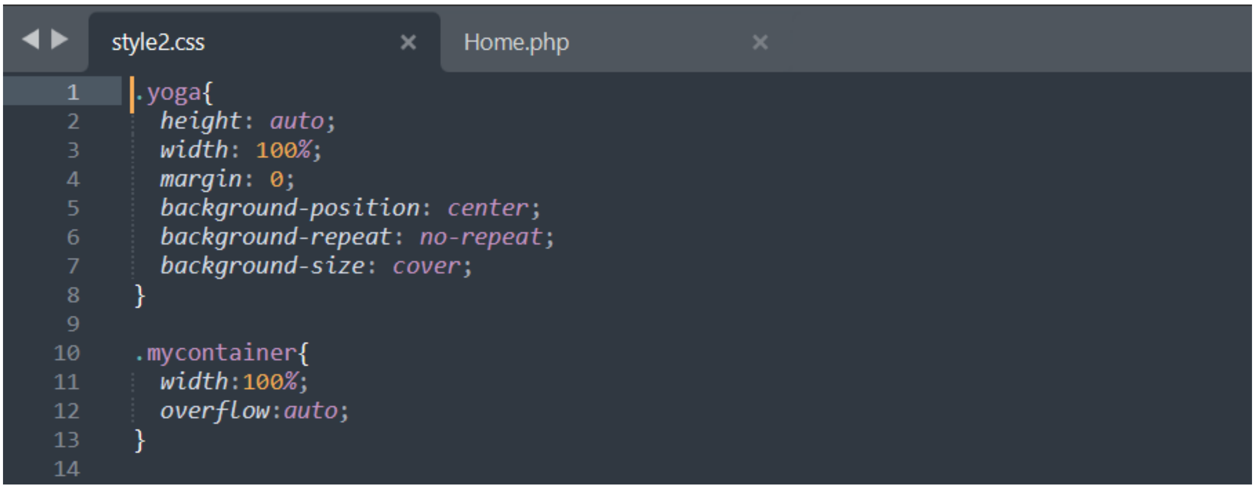
```
@import url('https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0,100..

* {
    box-sizing: border-box;
    margin: 0;
    padding: 0;
}

li, a, button, p {
    font-family: "Montserrat", sans-serif;
    font-weight: 500;
    font-size: 16px;
    color: #000000;
    text-decoration: none;
}
```

Figure – 4.1.2 : External Style Sheet (1)



```
<?php
include("nav_bar.php");
?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" type="text/css" href="style2.css">
    <title></title>
</head>
<body>
    <main class="main">
        <img class="yoga" src="image/I-7.png" alt="Exercise">
        <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do
        eiusmod
        tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim
        veniam,
        quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
        consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse
        cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat
        non
        proident, sunt in culpa qui officia deserunt mollit anim id est laborum.</
        p><br><br>
```

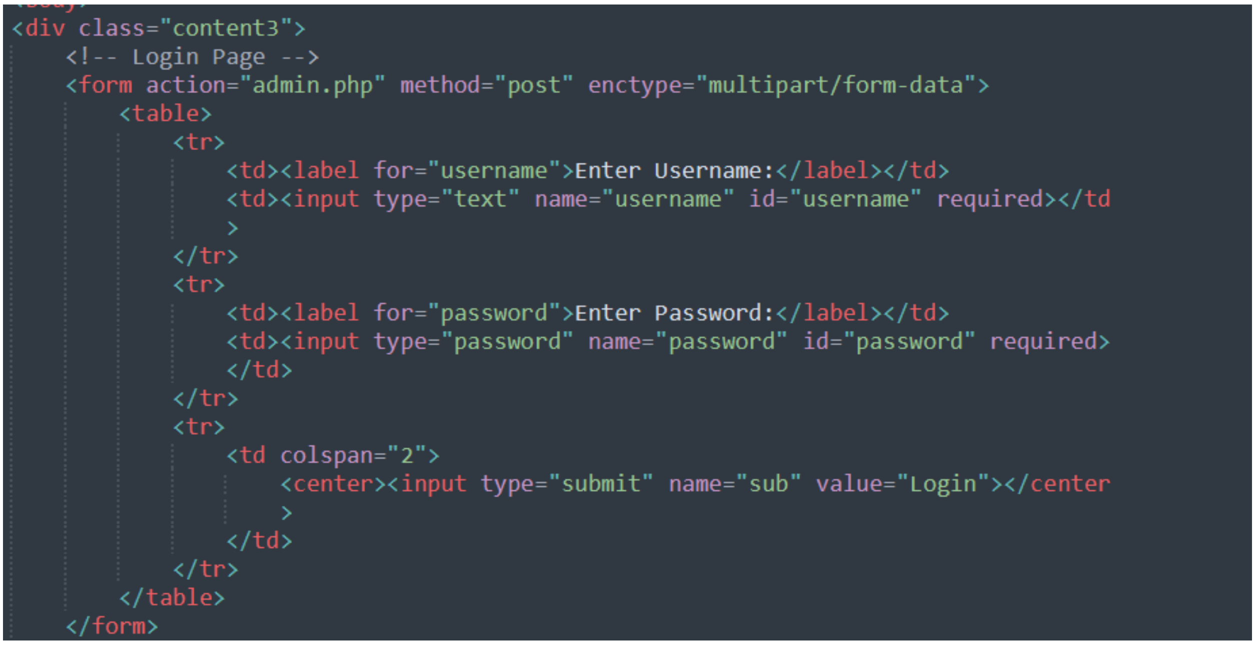
Figure – 4.1.3 : Linking CSS Style in Main Form



```
.yoga{
  height: auto;
  width: 100%;
  margin: 0;
  background-position: center;
  background-repeat: no-repeat;
  background-size: cover;
}

.mycontainer{
  width:100%;
  overflow:auto;
}
```

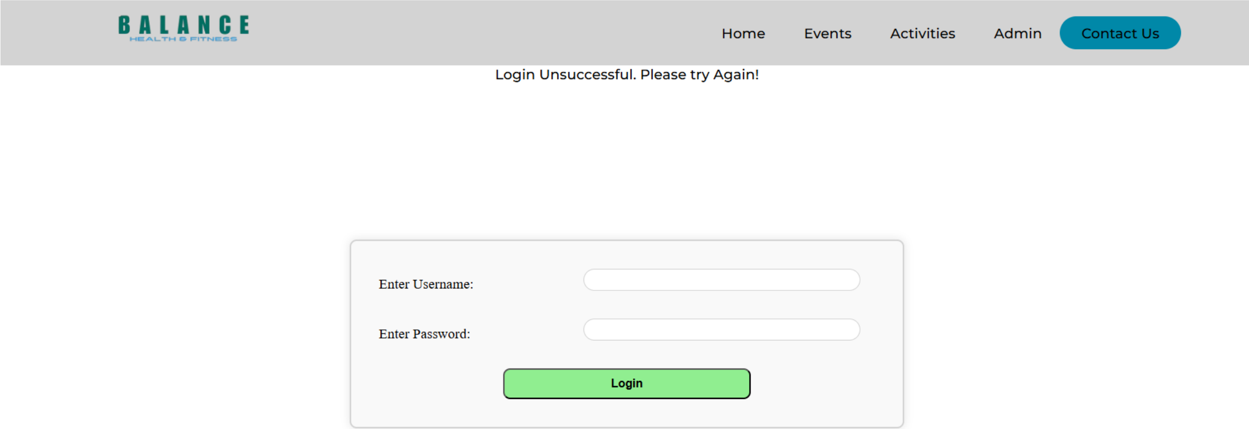
Figure – 4.1.4 : External Style Sheet (2)

4.2. Form



```
<div class="content3">
    <!-- Login Page -->
    <form action="admin.php" method="post" enctype="multipart/form-data">
        <table>
            <tr>
                <td><label for="username">Enter Username:</label></td>
                <td><input type="text" name="username" id="username" required></td
                >
            </tr>
            <tr>
                <td><label for="password">Enter Password:</label></td>
                <td><input type="password" name="password" id="password" required>
                </td>
            </tr>
            <tr>
                <td colspan="2">
                    <center><input type="submit" name="sub" value="Login"></center
                    >
                </td>
            </tr>
        </table>
    </form>
```

Figure – 4.2.1 : HTML Login using Form and Table

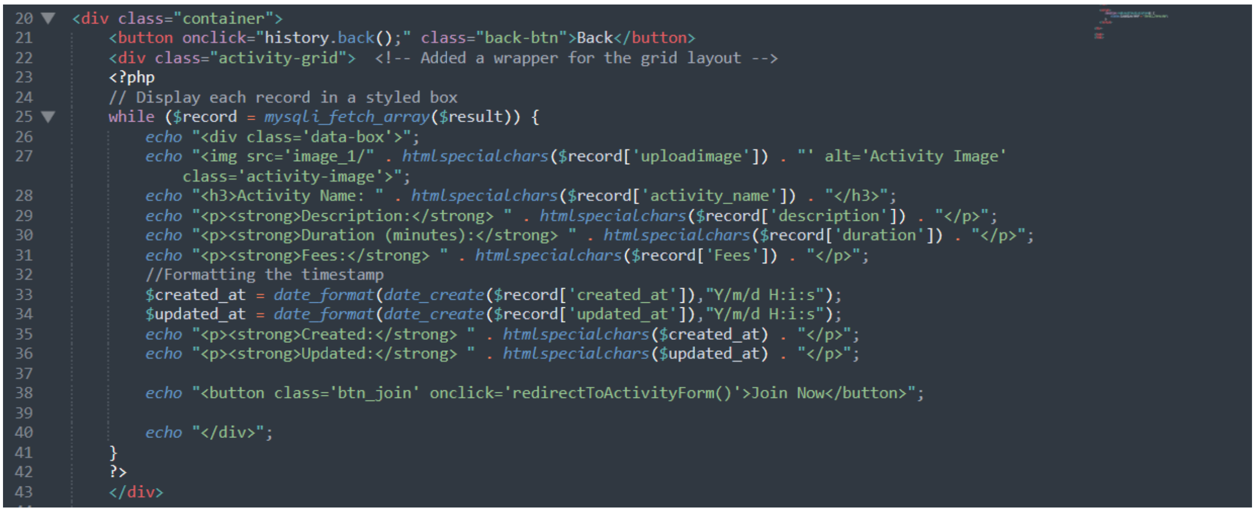


Login Unsuccessful. Please try Again!

Enter Username:

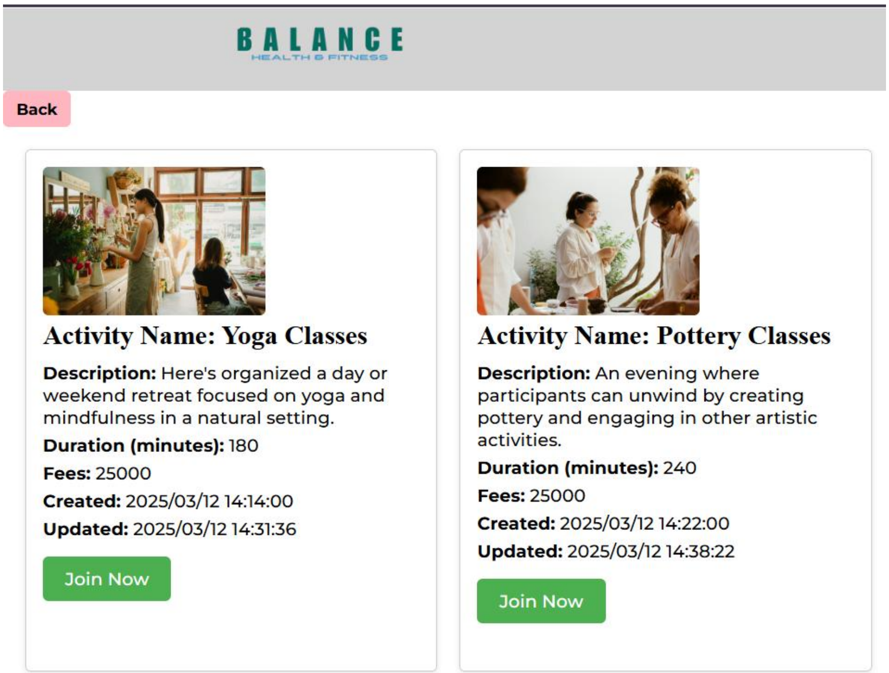
Enter Password:

Figure – 4.2.2 : Output of HTML Login



```
<div class="container">
    <button onclick="history.back();" class="back-btn">Back</button>
    <div class="activity-grid">  <!-- Added a wrapper for the grid layout -->
    <?php
    // Display each record in a styled box
    while ($record = mysqli_fetch_array($result)) {
        echo "<div class='data-box'>";
        echo "<img src='image_1/" . htmlspecialchars($record['uploadimage']) . "' alt='Activity Image'
            class='activity-image'>";
        echo "<h3>Activity Name: " . htmlspecialchars($record['activity_name']) . "</h3>";
        echo "<p><strong>Description:</strong> " . htmlspecialchars($record['description']) . "</p>";
        echo "<p><strong>Duration (minutes):</strong> " . htmlspecialchars($record['duration']) . "</p>";
        echo "<p><strong>Fees:</strong> " . htmlspecialchars($record['Fees']) . "</p>";
        //Formatting the timestamp
        $created_at = date_format(date_create($record['created_at']),"Y/m/d H:i:s");
        $updated_at = date_format(date_create($record['updated_at']),"Y/m/d H:i:s");
        echo "<p><strong>Created:</strong> " . htmlspecialchars($created_at) . "</p>";
        echo "<p><strong>Updated:</strong> " . htmlspecialchars($updated_at) . "</p>";

        echo "<button class='btn_join' onclick='redirectToActivityForm()'>Join Now</button>";

        echo "</div>";
    }
    ?>
    </div>
```

Figure – 4.2.3 : Using PHP and HTML to create Information Table



Activity Name: Yoga Classes

Description: Here's organized a day or weekend retreat focused on yoga and mindfulness in a natural setting.

Duration (minutes): 180

Fees: 25000

Created: 2025/03/12 14:14:00

Updated: 2025/03/12 14:31:36

Activity Name: Pottery Classes

Description: An evening where participants can unwind by creating pottery and engaging in other artistic activities.

Duration (minutes): 240

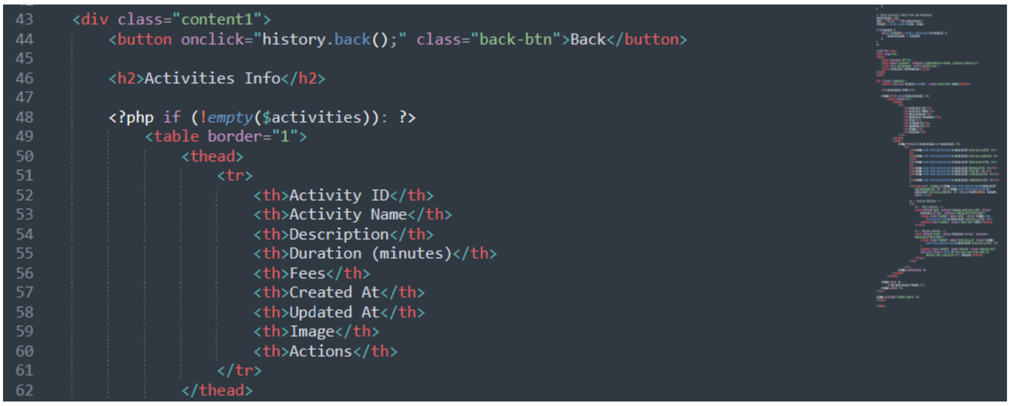
Fees: 25000

Created: 2025/03/12 14:22:00

Updated: 2025/03/12 14:38:22

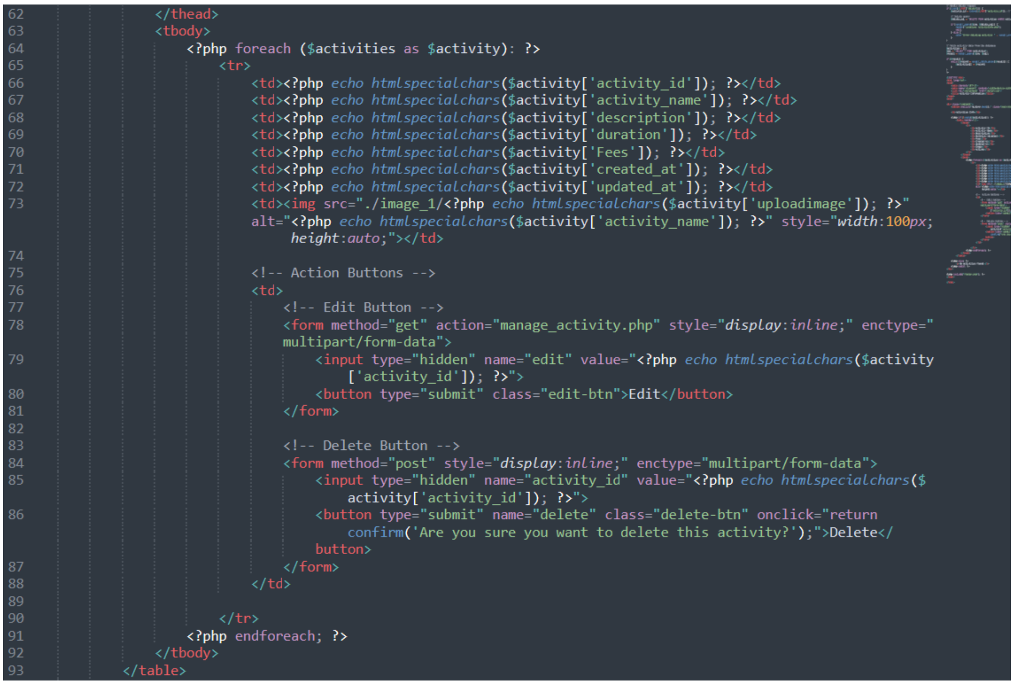
Figure – 4.2.4 : Output of PHP and HTML table

4.3. Use of Table



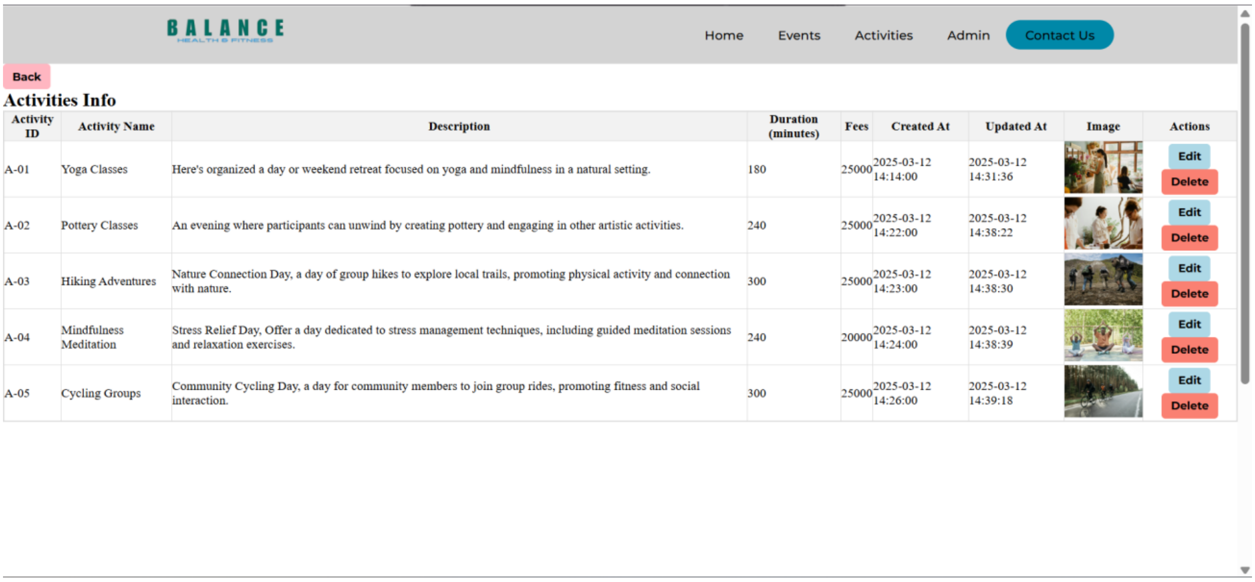
```
<div class="content1">
    <button onclick="history.back();" class="back-btn">Back</button>

    <h2>Activities Info</h2>

    <?php if (!empty($activities)): ?>
        <table border="1">
            <thead>
                <tr>
                    <th>Activity ID</th>
                    <th>Activity Name</th>
                    <th>Description</th>
                    <th>Duration (minutes)</th>
                    <th>Fees</th>
                    <th>Created At</th>
                    <th>Updated At</th>
                    <th>Image</th>
                    <th>Actions</th>
                </tr>
            </thead>
```

Figure – 4.3.1 : Creating Table by separating thead



```
            </thead>
            <tbody>
                <?php foreach ($activities as $activity): ?>
                    <tr>
                        <td><?php echo htmlspecialchars($activity['activity_id']); ?></td>
                        <td><?php echo htmlspecialchars($activity['activity_name']); ?></td>
                        <td><?php echo htmlspecialchars($activity['description']); ?></td>
                        <td><?php echo htmlspecialchars($activity['duration']); ?></td>
                        <td><?php echo htmlspecialchars($activity['Fees']); ?></td>
                        <td><?php echo htmlspecialchars($activity['created_at']); ?></td>
                        <td><?php echo htmlspecialchars($activity['updated_at']); ?></td>
                        <td><img src="./image_1/<?php echo htmlspecialchars($activity['uploadimage']); ?>" alt="<?php echo htmlspecialchars($activity['activity_name']); ?>" style="width:100px; height:auto;"></td>

                        <!-- Action Buttons -->
                        <td>
                            <!-- Edit Button -->
                            <form method="get" action="manage_activity.php" style="display:inline;" enctype="multipart/form-data">
                                <input type="hidden" name="edit" value="<?php echo htmlspecialchars($activity['activity_id']); ?>">
                                <button type="submit" class="edit-btn">Edit</button>
                            </form>

                            <!-- Delete Button -->
                            <form method="post" style="display:inline;" enctype="multipart/form-data">
                                <input type="hidden" name="activity_id" value="<?php echo htmlspecialchars($activity['activity_id']); ?>">
                                <button type="submit" name="delete" class="delete-btn" onclick="return confirm('Are you sure you want to delete this activity?');">Delete</button>
                            </form>
                        </td>

                    </tr>
                <?php endforeach; ?>
            </tbody>
        </table>
```

Figure – 4.3.2 : Creating Table by separating tbody

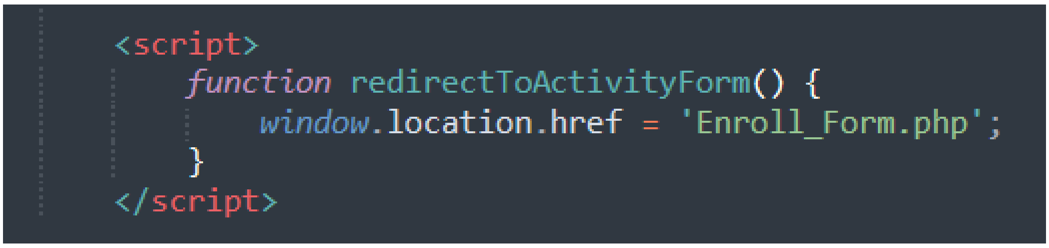


Activities Info

Activity ID	Activity Name	Description	Duration (minutes)	Fees	Created At	Updated At	Image	Actions
A-01	Yoga Classes	Here's organized a day or weekend retreat focused on yoga and mindfulness in a natural setting.	180	25000	2025-03-12 14:14:00	2025-03-12 14:31:36		Edit Delete
A-02	Pottery Classes	An evening where participants can unwind by creating pottery and engaging in other artistic activities.	240	25000	2025-03-12 14:22:00	2025-03-12 14:38:22		Edit Delete
A-03	Hiking Adventures	Nature Connection Day, a day of group hikes to explore local trails, promoting physical activity and connection with nature.	300	25000	2025-03-12 14:23:00	2025-03-12 14:38:30		Edit Delete
A-04	Mindfulness Meditation	Stress Relief Day, Offer a day dedicated to stress management techniques, including guided meditation sessions and relaxation exercises.	240	20000	2025-03-12 14:24:00	2025-03-12 14:38:39		Edit Delete
A-05	Cycling Groups	Community Cycling Day, a day for community members to join group rides, promoting fitness and social interaction.	300	25000	2025-03-12 14:26:00	2025-03-12 14:39:18		Edit Delete

Figure – 4.3.3 : Output Using Table

JavaScript Redirection

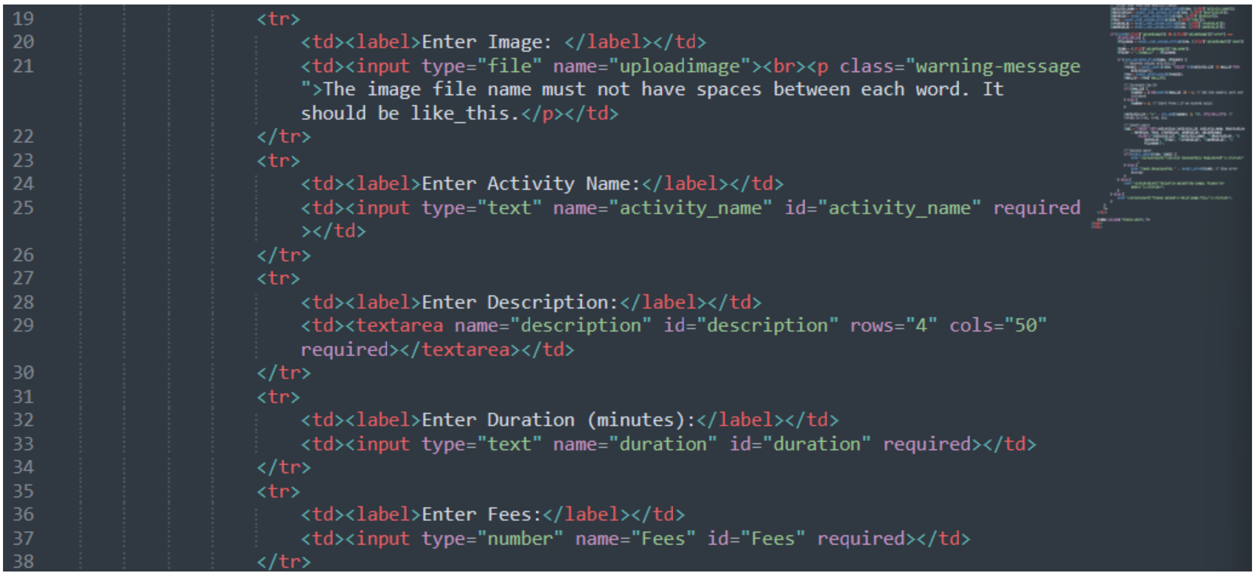


```
<script>
    function redirectToActivityForm() {
        window.location.href = 'Enroll_Form.php';
    }
</script>
```

Figure – 4.3.4 : JavaScript Redirection

5. Task – 2 Key Features

1. HTML Validations

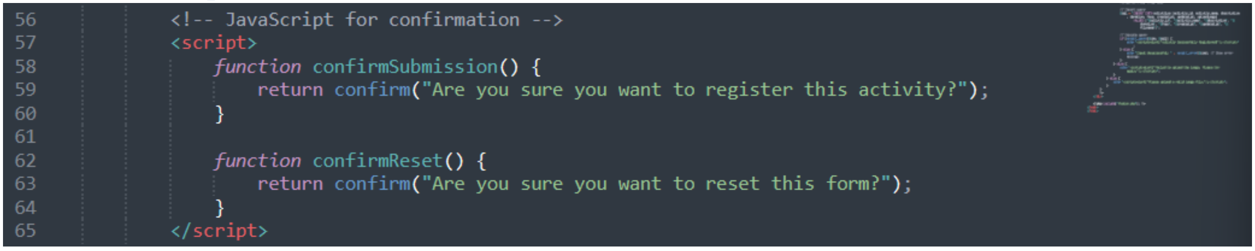


```
<tr>
    <td><label>Enter Image: </label></td>
    <td><input type="file" name="uploadimage"><br><p class="warning-message">The image file name must not have spaces between each word. It should be like_this.</p></td>
</tr>
<tr>
    <td><label>Enter Activity Name:</label></td>
    <td><input type="text" name="activity_name" id="activity_name" required></td>
</tr>
<tr>
    <td><label>Enter Description:</label></td>
    <td><textarea name="description" id="description" rows="4" cols="50" required></textarea></td>
</tr>
<tr>
    <td><label>Enter Duration (minutes):</label></td>
    <td><input type="text" name="duration" id="duration" required></td>
</tr>
<tr>
    <td><label>Enter Fees:</label></td>
    <td><input type="number" name="Fees" id="Fees" required></td>
</tr>
```

Figure – 5.1 : HTML validations

A form field for uploading an image and another for providing the activity name are created by this HTML code snippet. Users can choose a file in the picture input box while a warning indicates that the file name must adhere to a certain naming convention (such as "like_this") and not contain spaces. In order to prevent empty submissions, the activity name input field is marked as `required`, requiring users to complete it before completing the form.

2. JavaScript (Client-Side) Validations



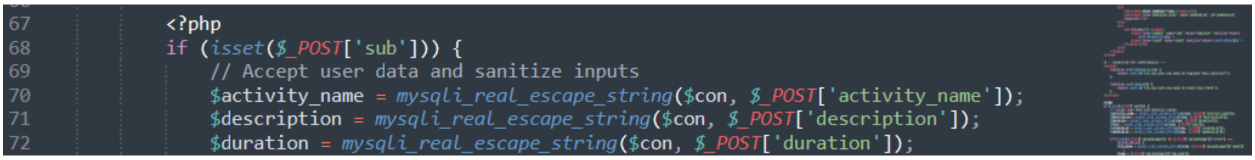
```
<!-- JavaScript for confirmation -->
<script>
    function confirmSubmission() {
        return confirm("Are you sure you want to register this activity?");
    }

    function confirmReset() {
        return confirm("Are you sure you want to reset this form?");
    }
</script>
```

Figure – 5.2 : JavaScript (Client-Side) Validations

The two functions defined by this JavaScript code, `confirmSubmission()` and `confirmReset()`, show a confirmation dialog to the user when they try to submit or reset a form. In order to confirm their action, the user is prompted with the `confirm()` method, which returns `true` if they click "OK" and `false` if they click "Cancel." By requiring user verification and employing these functions as event handlers for form submission and reset operations, the code helps prevent unintentional submissions or resets.

3. PHP (Server-Side) Validations

3.1. Sanitization:

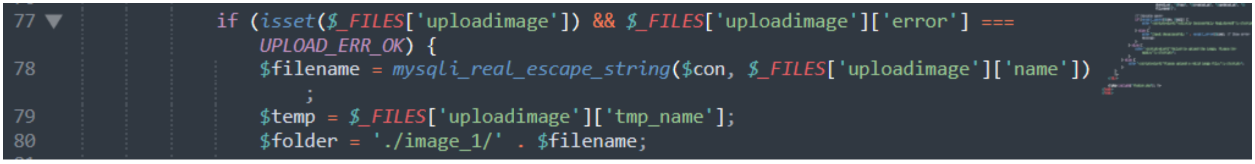


```php
<?php
if (isset($_POST['sub'])) {
    // Accept user data and sanitize inputs
    $activity_name = mysqli_real_escape_string($con, $_POST['activity_name']);
    $description = mysqli_real_escape_string($con, $_POST['description']);
    $duration = mysqli_real_escape_string($con, $_POST['duration']);
```

Figure – 5.3.3.1 : PHP's input sanitization feature

This piece of code illustrates PHP's input sanitization feature, which is an essential security measure that guards against SQL injection attacks. Before using special characters in database queries, it sanitizes user-submitted data for the `activity_name` and `description` fields using the `mysqli_real_escape_string()` function.

3.2. File Type Validation:

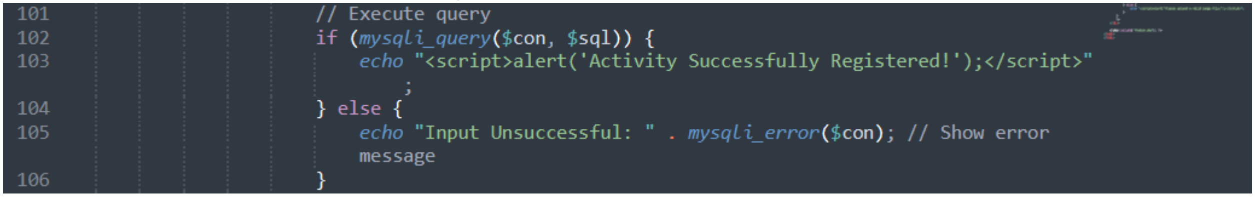


```php
    if (isset($_FILES['uploadimage']) && $_FILES['uploadimage']['error'] === UPLOAD_ERR_OK) {
        $filename = mysqli_real_escape_string($con, $_FILES['uploadimage']['name']);
        $temp = $_FILES['uploadimage']['tmp_name'];
        $folder = './image_1/' . $filename;
```

Figure – 5.3.3.2 : File Type Validation

The `uploadimage` input is used to determine whether a file has been uploaded, and the `UPLOAD_ERR_OK` constant is used to confirm that there were no upload issues. The destination file path where the picture will be placed on the server is created and the uploaded file's name is sanitized to prevent SQL injection if the requirements are satisfied.

3.3. Database Error Handling:

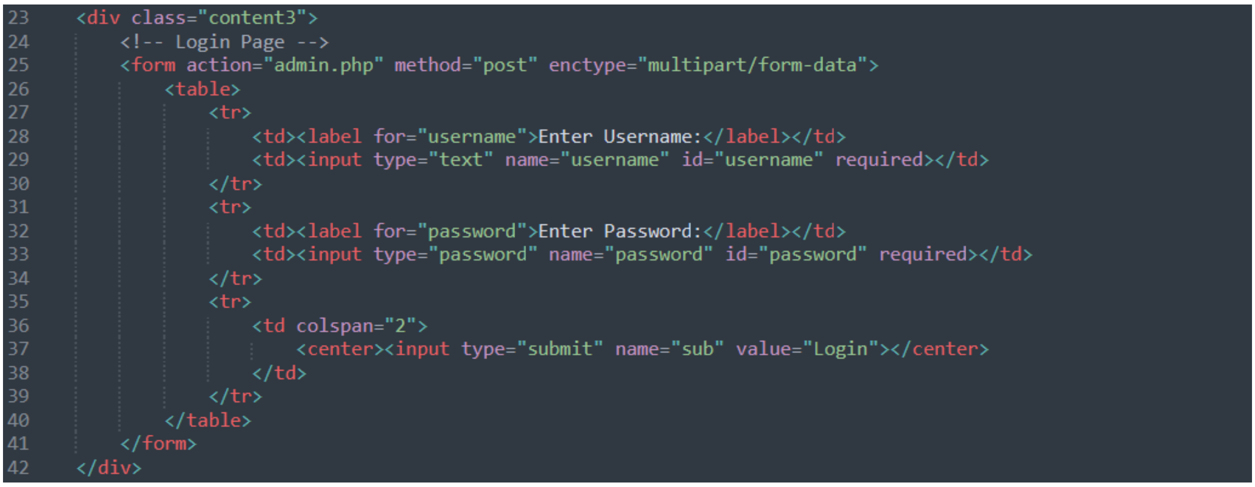


```php
        // Execute query
        if (mysqli_query($con, $sql)) {
            echo "<script>alert('Activity Successfully Registered!');</script>";
        } else {
            echo "Input Unsuccessful: " . mysqli_error($con); // Show error message
        }
```

Figure – 5.3.3.3 : Database Error Handling

In order to enter or update data in the database, this code runs a SQL query using the `mysqli_query()` function. If the query is successful, a JavaScript alert is displayed to the user with a success message. An error message containing the exact MySQL fault is output if the query fails, aiding in the diagnosis of any problems with the query's execution.

4. Login Validation:

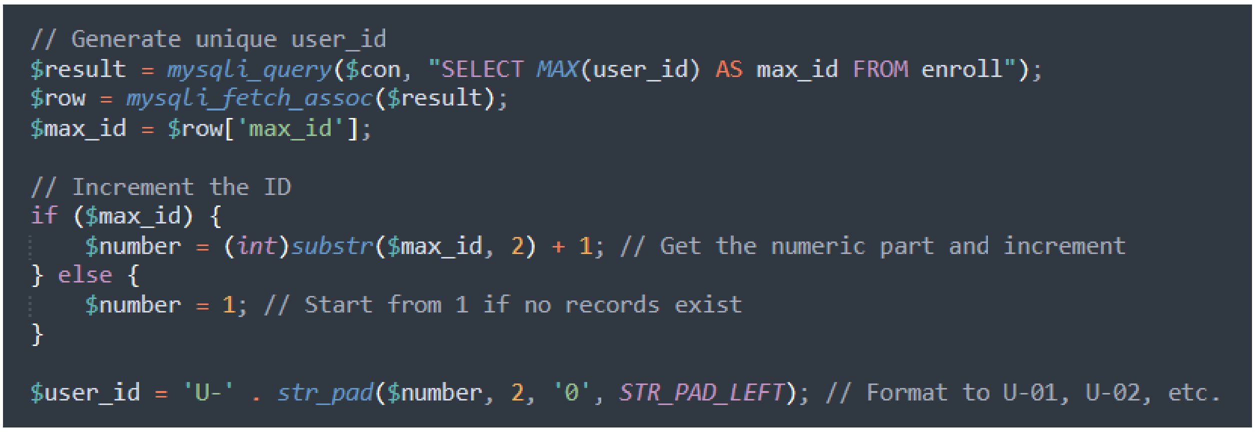


```html
<div class="content3">
    <!-- Login Page -->
    <form action="admin.php" method="post" enctype="multipart/form-data">
        <table>
            <tr>
                <td><label for="username">Enter Username:</label></td>
                <td><input type="text" name="username" id="username" required></td>
            </tr>
            <tr>
                <td><label for="password">Enter Password:</label></td>
                <td><input type="password" name="password" id="password" required></td>
            </tr>
            <tr>
                <td colspan="2">
                    <center><input type="submit" name="sub" value="Login"></center>
                </td>
            </tr>
        </table>
    </form>
</div>
```

Figure – 5.4: Login Validation

Users can enter their username and password in this code snippet, which generates a basic admin page login form. Both input fields are designated as necessary so that the user cannot submit the form without entering any data. The form submits the data to `admin.php` using the POST method. A table is used to organize the layout, and a centered submit button with the label "Login" is included for user interaction.

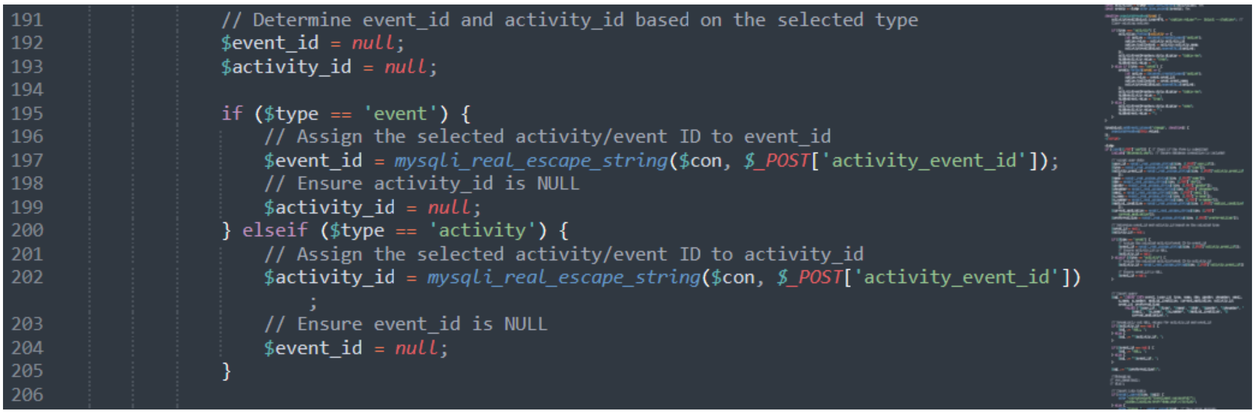
5. Enroll Form:



```php
// Generate unique user_id
$result = mysqli_query($con, "SELECT MAX(user_id) AS max_id FROM enroll");
$row = mysqli_fetch_assoc($result);
$max_id = $row['max_id'];

// Increment the ID
if ($max_id) {
    $number = (int)substr($max_id, 2) + 1; // Get the numeric part and increment
} else {
    $number = 1; // Start from 1 if no records exist
}

$user_id = 'U-' . str_pad($number, 2, '0', STR_PAD_LEFT); // Format to U-01, U-02, etc.
```

Figure – 5.5.1 : Enroll Form

This snippet of code mainly queries the database to determine the maximum existing `user_id` before creating a unique `user_id` for new records in the `enroll` table. The ID's numeric component is then increased; if there are no records, the ID is initialized to 1. The new `user_id` is then formatted as a string containing a zero-padded two-digit number (such as 'U-01' or 'U-02') after the prefix 'U-'.

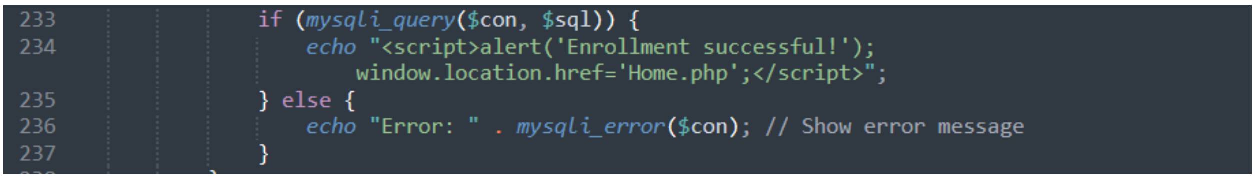


```
// Determine event_id and activity_id based on the selected type
$event_id = null;
$activity_id = null;

if ($type == 'event') {
    // Assign the selected activity/event ID to event_id
    $event_id = mysqli_real_escape_string($con, $_POST['activity_event_id']);
    // Ensure activity_id is NULL
    $activity_id = null;
} elseif ($type == 'activity') {
    // Assign the selected activity/event ID to activity_id
    $activity_id = mysqli_real_escape_string($con, $_POST['activity_event_id']);
    // Ensure event_id is NULL
    $event_id = null;
}
```

Figure – 5.5.2 : Input based on Selected Type

The value of the `$type` variable, which specifies whether the selected type is a `event` or an `activity`, is used by this code snippet to determine the proper assignment of `event_id` and `activity_id`. The `activity_event_id` is retrieved and sanitized from the POST request to assign it to `event_id` if `$type` is 'event'. It makes sure that `activity_id` is set to `null`. On the other hand, if `$type` is 'activity', it changes `event_id` to {null` and gives `activity_id` the same sanitized ID, guaranteeing that only one of the two IDs is populated at a time.

Redirection:

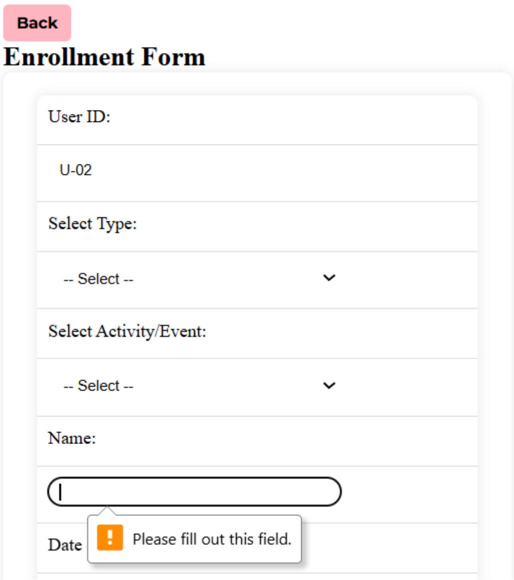


```
if (mysqli_query($con, $sql)) {
    echo "<script>alert('Enrollment successful!'); window.location.href='Home.php';</script>";
} else {
    echo "Error: " . mysqli_error($con); // Show error message
}
```

Figure – 5.5.3 : Redirection

6. Output key features

6.1. Enrollment Form



Enrollment Form

User ID:

U-02

Select Type:

-- Select --

Select Activity/Event:

-- Select --

Name:

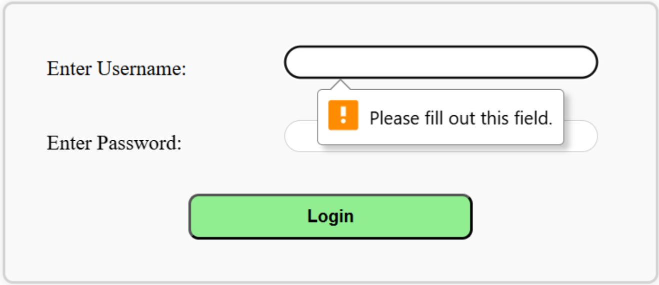
Date

Please fill out this field.

Figure – 6.6.1 : Enrollment Form Validation

As shown in diagram, if user click enroll without filling the form. There validation to fill the form would be appeared.

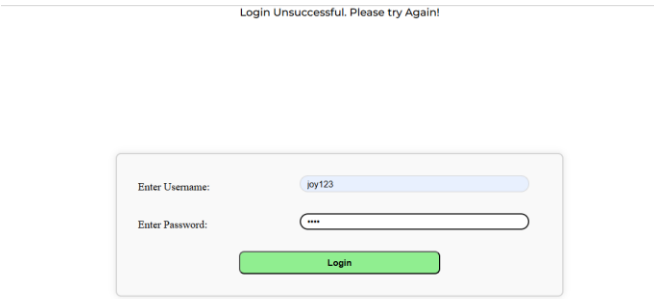
6.2. Admin Validation:


Enter Username:

Please fill out this field.

Enter Password:

Figure – 6.6.2.1 : Admin Validation with null

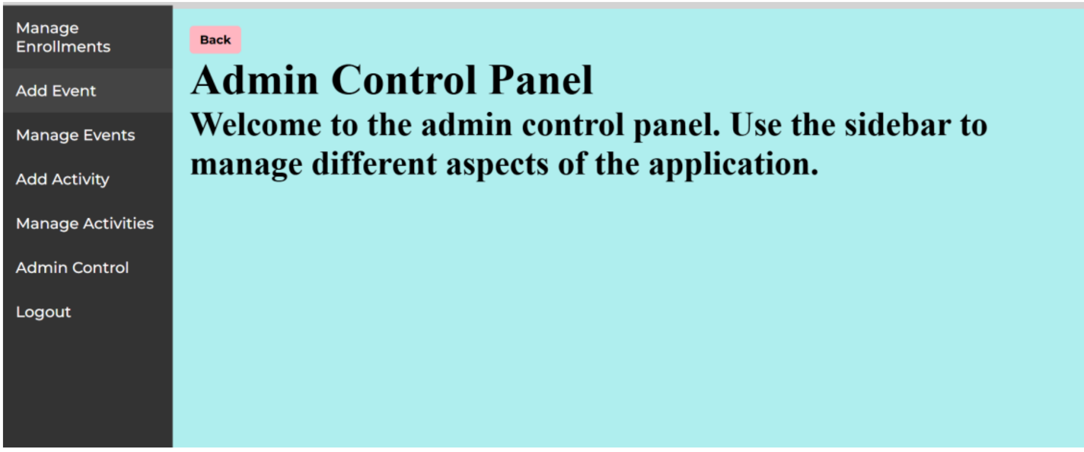

Login Unsuccessful. Please try Again!

Enter Username: joy123

Enter Password: ••••

Figure – 6.6.2.2 : Admin Validation with wrong Username or Password

6.3. Main Admin Validation:

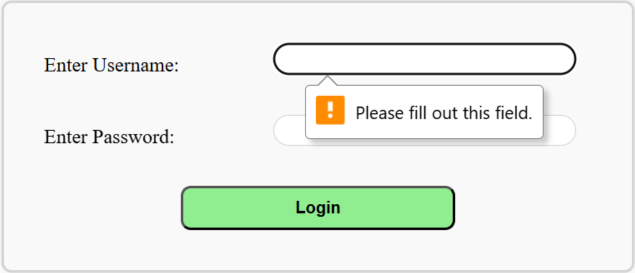


Admin Control Panel

Welcome to the admin control panel. Use the sidebar to manage different aspects of the application.

Figure – 6.3.1 : Main Admin Validation

According to the Image, only Main Admin can go to Admin Control Form, as there is another login form to login.



Enter Username:

Enter Password:

Please fill out this field.

Figure – 6.3.2 : Main Admin Validation with null

Reference list

Aigars Silkalns (2023). *Healthvest*. [online] Colorlib. Available at: https://colorlib.com/wp/template/healthvest/ [Accessed 4 Mar. 2025].

COS-108 Sitemap.pdf and Thiri Win, C. (2025). *COS-108 Sitemap.pdf*. [online] Google Docs. Available at: https://drive.google.com/file/d/1ngmN5XJZAtAKX6jenhBbArsZDFpJAFVD/view?usp=drive_link [Accessed 12 Mar. 2025].

COS-108 Storyboard.drawio and Thiri Win, C. (2025). *COS-108 Storyboard.drawio*. [online] Google Docs. Available at: https://drive.google.com/file/d/1vYKTDYj5TiG-l_ZihpiHVUGF1aEZT14g/view?usp=drive_link [Accessed 12 Mar. 2025].