

# CSE4015 – HUMAN COMPUTER INTERACTION

## DA 1 and DA 3 – Daily Expense System

### Front End & Back End

**SANJIL K C**  
**20BCE1855**

#### **Abstract**

We are developing a Web application named as “Daily Expense Tracker System” and this application is used to manage the application user ‘s daily expenses in a more efficient and manageable way. By using this application, we can reduce the manual calculations for their daily expenses and keep the track of the expenditure. In this application, user can provide his/her expense to calculate his/her total expenses per day and these results will be stored for unique user.

#### **Existing System**

In existing, we need to maintain the Excel sheets, CSV etc. files for the user daily and monthly expenses. In existing, there is no as such complete solution to keep a track of its daily expenditure easily. To do so a person as to keep a log in a diary or in a computer, also all the calculations need to be done by the user which may sometimes results in errors leading to losses.

#### **Modules**

- In Daily Expense Tracker System, we use PHP and MySQL database. This is the project which keeps records of daily expenses.
- User

#### **User Module**

1. **Dashboard:** In this section, user can briefly view expenses on a daily basis, monthly basis, and yearly basis.
2. **Expenses:** In this section user can manage the expenses (add/delete).
3. **Expense Report:** In this section, user can view expenses on day-wise basis, month-wise basis, year-wise basis, and category-wise according to periods of time.
4. **Profile:** In this section, user can update his/her profile.
5. **Change Password:** In this section, user can change his/her passwords
6. **Logout:** Through this button, user can log out.

## Requirement Specification

### ➤ Hardware Configuration:

#### Client Side

RAM	512 MB
Hard Disk	10GB
Processor	1.0 GHz

#### Server Side

RAM	1 GB
Hard Disk	20GB
Processor	2.0GHz

### ➤ Software Requirement:

#### Client Side

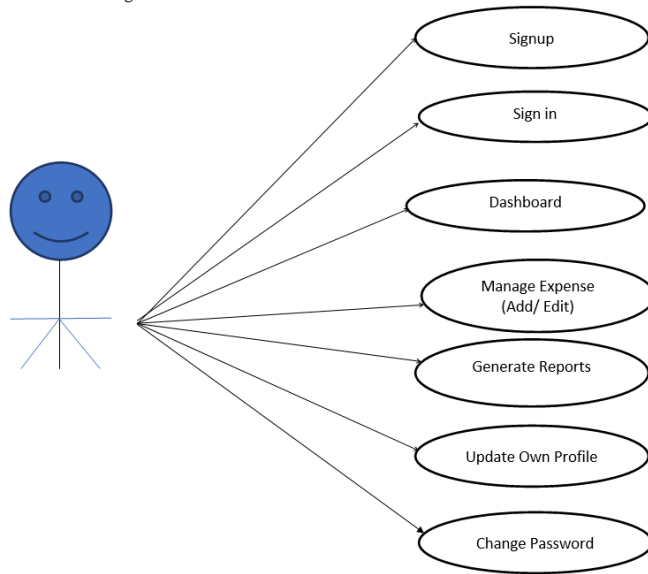
Web Browser	Google Chrome or any compatible browser
Operating System	Windows or any equivalent OS

#### Server Side

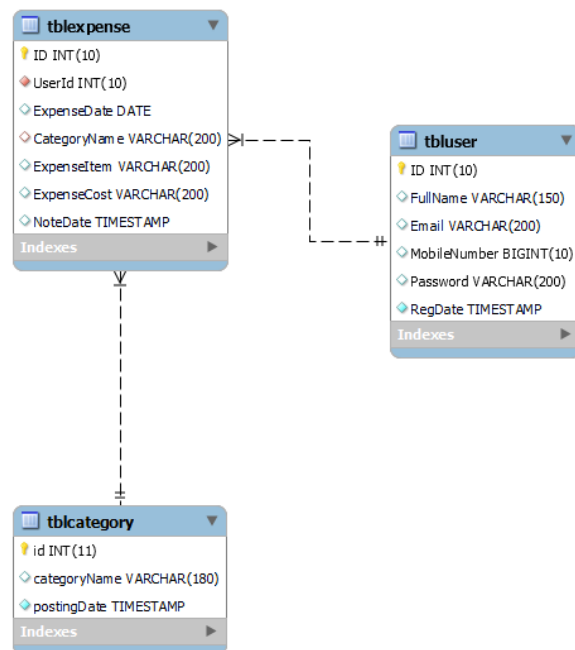
Web Server	APACHE
Server side Language	PHP5.6 or above version
Database Server	MySQL
Web Browser	Google Chrome or any compatible browser
Operating System	Windows or any equivalent OS

## Use Case Diagram

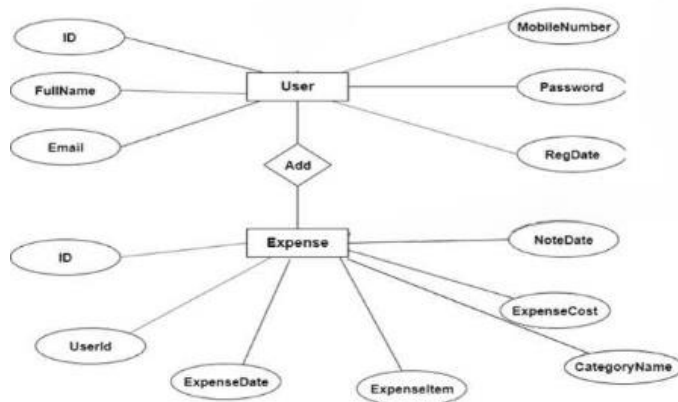
Use Case Diagrams User



## Class / Schema Diagram



## ER Diagram



## Implementation and System Testing

- After all phase have been perfectly done, the system will be implemented to the server and the system can be used.

### System Testing

- The goal of the system testing process was to determine all faults in our project. The program was subjected to a set of test inputs and many explanations were made and based on these explanations it will be decided whether the program behaves as expected or not. Our Project went through two levels of testing

- ☐ Unit testing
- ☐ Integration testing

## **DESCRIPTION OF PROJECT**

The Daily Expense System is a web-based application built using PHP and MySQL to help users track and manage their daily expenses. It provides a convenient platform for individuals or households to keep a record of their spending habits, which can be crucial for budgeting and financial planning.

### ***Key Features:***

#### **1. User Authentication:**

- Users can create accounts with a username and password.
- Authentication ensures that only registered users can access and manage their expenses.

#### **2. Expense Tracking:**

- Users can log their daily expenses by providing details such as category (e.g., food, transportation, entertainment), amount spent, and a short description.
- The system records the date and time of each expense entry.

#### **3. Expense Categories:**

- Users can categorize their expenses to provide a structured view of their spending habits. Categories can be predefined (e.g., Food, Transportation, Entertainment) or customizable.

#### **4. Expense Listing:**

- The system displays a list of all expenses, showing details like category, amount, description, and date/time.
- Users can sort and filter expenses based on criteria like date range or category.

#### **5. Expense Reports:**

- Users can generate reports summarizing their expenses over specific time periods (e.g., daily, weekly, monthly).
- Reports can include graphs or charts to visualize spending patterns.

#### **6. Budget Setting:**

- Users can set monthly or weekly spending limits for different expense categories.
- The system can provide notifications or warnings when users approach or exceed their budget.

#### **7. Search Functionality:**

- Users can search for specific expenses based on keywords, categories, or date ranges.

#### **8. User Profile Management:**

- Users can update their personal information, change passwords, and manage their account settings.

#### **9. Data Backup and Restore:**

- Users may have the option to back up their expense data and restore it in case of system failures or data loss.

### ***Technical Stack:***

- PHP: Used for server-side scripting to handle user requests, process data, and interact with the database.
- MySQL: A relational database management system used to store and manage expense data.
- HTML/CSS: Used for creating the user interface and styling the web pages.

### ***Database Schema:***

The MySQL database will consist of at least two tables:

1. Users table:

- Columns: user\_id (Primary Key), username, password, email, etc.

2. Expenses table:

- Columns: expense\_id (Primary Key), user\_id (Foreign Key referencing Users table), category, amount, description, date\_time, etc.

### ***Security Considerations:***

- Use hashed passwords and salts to securely store user credentials.
- Implement input validation and sanitization to prevent SQL injection and cross-site scripting (XSS) attacks.
- Apply proper access controls to ensure users can only access their own data.

### ***Deployment:***

The system can be deployed on a web server with PHP and MySQL support. It's important to secure the server and database, and use HTTPS for secure communication.

This project provides a practical solution for individuals or households to manage their daily expenses effectively, aiding them in making informed financial decisions.

# 1. Register

## Sample Code:

```
<?php
require('config.php');
if (isset($_REQUEST['firstname'])) {
    if ($_REQUEST['password'] == $_REQUEST['confirm_password']) {
        $firstname = stripslashes($_REQUEST['firstname']);
        $firstname = mysqli_real_escape_string($con, $firstname);
        $lastname = stripslashes($_REQUEST['lastname']);
        $lastname = mysqli_real_escape_string($con, $lastname);

        $email = stripslashes($_REQUEST['email']);
        $email = mysqli_real_escape_string($con, $email);

        $password = stripslashes($_REQUEST['password']);
        $password = mysqli_real_escape_string($con, $password);

        $trn_date = date("Y-m-d H:i:s");

        $query = "INSERT into `users` (firstname, lastname, password, email,
trn_date) VALUES ('$firstname','$lastname', '' . md5($password) . '',
'$email', '$trn_date')";
        $result = mysqli_query($con, $query);
        if ($result) {
            header("Location: login.php");
        }
    } else {
        echo ("ERROR: Please Check Your Password & Confirmation password");
    }
}
?>
<!DOCTYPE html>
<html lang="en">
```

## Description of the Code:

This PHP code is responsible for handling the registration page of the Daily Expense System. Let's break down the key components and functionality:

### **1. File Inclusions:**

```
```php
require('config.php');
```
```

- This line includes an external file named `config.php`, which likely contains database connection information and other configuration settings.

### **2. Form Handling:**

```
```php
if (isset($_REQUEST['firstname'])) {
```
```

- This conditional statement checks if the form has been submitted. It does so by checking if the 'firstname' parameter is set in the request.

### **3. Password Confirmation:**

```
```php
if ($_REQUEST['password'] == $_REQUEST['confirm_password']) {
```
```

- This condition checks if the entered password matches the confirmed password.

### **4. Data Sanitization:**

- The following lines are used to sanitize user input to prevent SQL injection attacks:

```
```php
$firstname = stripslashes($_REQUEST['firstname']);
$firstname = mysqli_real_escape_string($con, $firstname);
// (Repeat for 'lastname', 'email', and 'password')
```
```

### **5. Date Generation:**

```
```php
$trn_date = date("Y-m-d H:i:s");
```
```



- This line generates the current date and time in the format `YYYY-MM-DD HH:mm:ss`.

#### **6. Database Query:**

```
```php
$query = "INSERT into `users` (firstname, lastname, password, email, trn_date)
VALUES ('$firstname','$lastname', '' . md5($password) . '', '$email', '$trn_date')";
```
```

- This SQL query inserts the user's registration information (first name, last name, hashed password, email, and registration date) into the 'users' table.

#### **7. Executing the Query:**

```
```php
$result = mysqli_query($con, $query);
```
```

- This line executes the SQL query using the database connection (`\$con`) and stores the result.

#### **8. Redirecting After Successful Registration:**

```
```php
if ($result) {
    header("Location: login.php");
}
```
```

- If the registration is successful (i.e., the query was executed without errors), the user is redirected to the login page.

#### **9. Error Handling:**

```
```php
} else {
    echo ("ERROR: Please Check Your Password & Confirmation password");
}
```
```

- If the password and confirmation password do not match, an error message is echoed.

#### **10. HTML Form:**

The rest of the code is HTML and is responsible for rendering the registration form. It includes fields for first name, last name, email, password, and confirmation password. It also includes a checkbox for accepting terms of use and a submit button.

This code combines PHP for server-side logic (handling form submission, database interaction) with HTML for rendering the user interface. It also demonstrates basic security practices like data sanitization and password hashing using MD5 (though more secure methods like bcrypt are recommended).

 Output:

Register

First Name

Last Name

Email

Password

Confirm Password

☐ I accept the [Terms of Use](#) & [Privacy Policy](#)

Register

Already have an account? [Login Here](#)

## 2. Login page

### Sample Code:

```
<?php
require('config.php');
session_start();
$errormsg = "";
if (isset($_POST['email'])) {

    $email = stripslashes($_REQUEST['email']);
    $email = mysqli_real_escape_string($con, $email);
    $password = stripslashes($_REQUEST['password']);
    $password = mysqli_real_escape_string($con, $password);
    $query = "SELECT * FROM `users` WHERE email='$email'and password='".
md5($password) . "'";
    $result = mysqli_query($con, $query) or die(mysqli_error($con));
    $rows = mysqli_num_rows($result);
    if ($rows == 1) {
        $_SESSION['email'] = $email;
        header("Location: index.php");
    } else {
        $errorMsg = "Wrong";
    }
} else {
}
?>
```

### Description of the Code:

This PHP code is responsible for handling the login page of the Daily Expense System. Let's break down the key components and functionality:

#### 1. File Inclusions:

```
```php
require('config.php');
```
```

- This line includes an external file named `config.php`, which likely contains database connection information and other configuration settings.

#### 2. Session Start:

```
```php
session_start();
```
```

- This initiates a session. Sessions are used to persist data across multiple pages or requests for a particular user.

### 3. Error Message Variable:

```
```php
$errormsg = "";
```
```

- This variable is used to store error messages related to the login process.

### 4. Form Submission Handling:

```
```php
if (isset($_POST['email'])) {
```
```

- This conditional statement checks if the form has been submitted. It does so by checking if the 'email' parameter is set in the POST request.

### 5. Data Sanitization:

- The following lines are used to sanitize user input to prevent SQL injection attacks:

```
```php
$email = stripslashes($_REQUEST['email']);
$email = mysqli_real_escape_string($con, $email);
$password = stripslashes($_REQUEST['password']);
$password = mysqli_real_escape_string($con, $password);
```
```

### 6. Database Query:

```
```php
$query = "SELECT * FROM `users` WHERE email='$email'and password='".
md5($password) . "'";
```
```

- This SQL query checks if there is a user with the provided email and password in the 'users' table. It uses `md5` to hash the password for comparison.

### 7. Executing the Query:

```
```php
$result = mysqli_query($con, $query) or die(mysqli_error($con));
```
```

- This line executes the SQL query using the database connection (`\$con`) and stores the result. If there's an error, it outputs the MySQL error message.

## 8. Checking Query Result:

```
```php
$rows = mysqli_num_rows($result);
if ($rows == 1) {
    $_SESSION['email'] = $email;
    header("Location: index.php");
} else {
    $errmsg = "Wrong";
}
```
```

- If the query returns exactly one row (meaning a matching user was found), it sets the 'email' session variable and redirects the user to the `index.php` page. If no matching user is found, it sets an error message.

## 9. HTML Form:

The rest of the code is HTML and is responsible for rendering the login form. It includes fields for email and password, along with a submit button.

It also provides a checkbox for "Remember me," although the corresponding functionality is not implemented in this code.

## 10. Error Handling:

```
```php
$errmsg = "Wrong";
```
```

- If the login attempt is unsuccessful, this line sets the error message to inform the user.

Overall, this code combines PHP for server-side logic (handling form submission, database interaction) with HTML for rendering the user interface. It also demonstrates basic security practices like data sanitization and password hashing using MD5 (though more secure methods like bcrypt are recommended).

 Output:

# D.E.M.S


Login Panel

Login

☐ Remember me

Don't have an account? [Register Here](#)

### 3. Dashboard

 Sample Code:

```
<?php
    include("session.php");
    $exp_category_dc = mysqli_query($con, "SELECT expensecategory FROM expenses
WHERE user_id = '$userid' GROUP BY expensecategory");
    $exp_amt_dc = mysqli_query($con, "SELECT SUM(expense) FROM expenses WHERE
user_id = '$userid' GROUP BY expensecategory");

    $exp_date_line = mysqli_query($con, "SELECT expensedate FROM expenses WHERE
user_id = '$userid' GROUP BY expensedate");
    $exp_amt_line = mysqli_query($con, "SELECT SUM(expense) FROM expenses WHERE
user_id = '$userid' GROUP BY expensedate");
?>
<!DOCTYPE html>
<html lang="en">

<head>

    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>Expense Manager - Dashboard</title>

    <!-- Bootstrap core CSS -->
    <link href="css/bootstrap.min.css" rel="stylesheet">

    <!-- Custom styles for this template -->
    <link href="css/style.css" rel="stylesheet">

    <!-- Feather JS for Icons -->
    <script src="js/feather.min.js"></script>
```

## Description of the Code:

This PHP code represents the dashboard page of the Daily Expense System. Below is a description of the key components and functionality:

### 1. Session Inclusion:

```
```php
include("session.php");
```
```

- This line includes a file named `session.php`, which likely contains code related to managing user sessions. It's used for authentication and tracking user information across pages.

### 2. Database Queries:

- Several MySQL queries are performed to retrieve data related to expense categories, amounts, and dates. These queries group the data by certain criteria (e.g., category or date) to prepare it for visualization.

```
```php
$exp_category_dc = mysqli_query($con, "SELECT expensecategory FROM expenses
WHERE user_id = '$userid' GROUP BY expensecategory");
$exp_amt_dc = mysqli_query($con, "SELECT SUM(expense) FROM expenses WHERE
user_id = '$userid' GROUP BY expensecategory");

$exp_date_line = mysqli_query($con, "SELECT expensedate FROM expenses WHERE
user_id = '$userid' GROUP BY expensedate");
$exp_amt_line = mysqli_query($con, "SELECT SUM(expense) FROM expenses WHERE
user_id = '$userid' GROUP BY expensedate");
```
```

- These queries retrieve information about expenses, grouped by category and date, specifically for the logged-in user (`\$userid`).

### 3. HTML and Bootstrap Components:

- The code includes HTML elements and classes from Bootstrap to structure and style the dashboard page. This includes the sidebar, navigation bar, cards, and other UI components.

### 4. Dynamic Content Rendering:

- User-specific information, such as the user's profile picture, name, and email, is dynamically loaded into the sidebar.

```
```php

<h5><?php echo $username ?></h5>
```
```



```
<p><?php echo $useremail ?></p>
```

## 5. Navigation Links:

- The sidebar contains links for different functionalities of the application, including Dashboard, Add Expenses, Manage Expenses, and user settings.

## 6. Chart.js Integration:

- The code includes JavaScript code that uses Chart.js, a popular charting library, to generate visual representations of expense data.

- Two types of charts are generated:
  - A bar chart showing expenses by category.
  - A line chart showing expenses by date over the whole year.
- Data for these charts is obtained from the database queries executed earlier.

```
```javascript
var ctx = document.getElementById('expense_category_pie').getContext('2d');
// ...
var line = document.getElementById('expense_line').getContext('2d');
// ...
```
```

## 7. Chart Data Population:

- The data retrieved from the database queries is used to populate the charts. The PHP while loops are used to generate JavaScript code dynamically based on the query results.

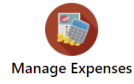
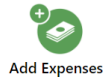
```
```javascript
labels: [<?php while ($a = mysqli_fetch_array($exp_category_dc)) { echo "" .
$a['expensecategory'] . " "; } ?>],
data: [<?php while ($b = mysqli_fetch_array($exp_amt_dc)) { echo "" .
$b['SUM(expense)] . " "; } ?>],
```
```

- These lines create the labels and corresponding data points for the charts.

Overall, this code combines PHP for server-side logic (database queries, session management) with HTML and Bootstrap for the user interface. It also utilizes Chart.js for data visualization, providing users with a graphical representation of their expenses.

 Output:

## Dashboard

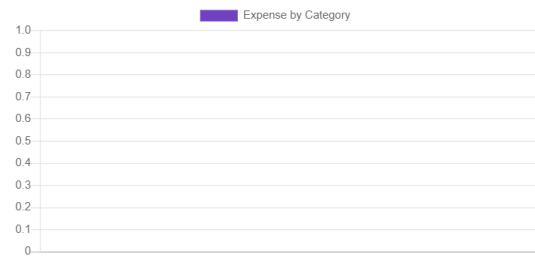


## Full-Expense Report

### Yearly Expenses



### Expense Category



## 4 Add Expense

 *Sample Code:*

```
<?php
include("session.php");
$update = false;
$del = false;
$expenseamount = "";
$expensedate = date("Y-m-d");
$expensecategory = "Entertainment";
if (isset($_POST['add'])) {
    $expenseamount = $_POST['expenseamount'];
    $expensedate = $_POST['expensedate'];
    $expensecategory = $_POST['expensecategory'];

    $expenses = "INSERT INTO expenses (user_id,
expense,expensedate,expensecategory) VALUES ('$userid',
'$expenseamount','$expensedate','$expensecategory')";
    $result = mysqli_query($con, $expenses) or die("Something Went Wrong!");
    header('location: add_expense.php');
}

if (isset($_POST['update'])) {
    $id = $_GET['edit'];
    $expenseamount = $_POST['expenseamount'];
    $expensedate = $_POST['expensedate'];
    $expensecategory = $_POST['expensecategory'];

    $sql = "UPDATE expenses SET expense='$expenseamount',
expensedate='$expensedate', expensecategory='$expensecategory' WHERE
user_id='$userid' AND expense_id='$id'";
    if (mysqli_query($con, $sql)) {
        echo "Records were updated successfully.";
    } else {
        echo "ERROR: Could not able to execute $sql. " . mysqli_error($con);
    }
    header('location: manage_expense.php');
}
```

 *Description of the Code:*

This PHP code represents the "Add Expense" page of the Daily Expense System. Below is a description of the key components and functionality:

### 1. Session Inclusion:

```
```php
include("session.php");
```
```

- This includes a file named `session.php`, which likely contains code related to managing user sessions. It's used for authentication and tracking user information across pages.

### 2. Variables Initialization:

- Several variables are initialized, including `\$update`, `\$del`, `\$expenseamount`, `\$expensedate`, and `\$expensecategory`. These are used to manage the state of the form (whether it's in update mode, delete mode, or adding mode), as well as to hold values for expense details.

### 3. Form Submission Handling:

- The code checks if the form has been submitted using `\$\_POST` to add, update, or delete an expense.

- If the "Add Expense" button is clicked (`\$\_POST['add']`), the entered expense details are captured and inserted into the database.

- If the "Update" button is clicked (`\$\_POST['update']`), the expense details are updated in the database based on the expense ID.

- If the "Delete" button is clicked (`\$\_POST['delete']`), the corresponding expense record is deleted from the database.

- After each operation, the page is redirected to the "Manage Expenses" page (`header('location: manage_expense.php');`).

### 4. Update and Delete Handling:

- There are separate blocks for handling updates and deletes. This involves capturing the expense details and executing the corresponding SQL query.

### 5. Form Pre-filling for Update and Delete:

- If the page is accessed for editing or deleting an expense (`\$\_GET['edit']` or `\$\_GET['delete']` is set), the form fields are pre-filled with the expense details based on the expense ID.

```
```php
if (isset($_GET['edit'])) {
```

```

        // ...
    }

    if (isset($_GET['delete'])) {
        // ...
    }
    ...

```

## 6. HTML and Bootstrap Components:

- The code includes HTML elements and classes from Bootstrap to structure and style the "Add Expense" page. This includes form fields for entering expense details.

## 7. Form Submission Handling for Add Expense:

- If the page is in "Add Expense" mode, the form submission triggers the insertion of a new expense record into the database.

```

```php
<button type="submit" name="add" class="btn btn-lg btn-block btn-success"
style="border-radius: 0%;">Add Expense</button>
```

```

## 8. Form Submission Handling for Update and Delete:

- If the page is in "Update" mode, the form submission triggers an update of the expense record in the database.

- If the page is in "Delete" mode, the form submission triggers the deletion of the expense record from the database.

```

```php
<?php if ($update == true) : ?>
    <button class="btn btn-lg btn-block btn-warning" style="border-radius: 0%;"
type="submit" name="update">Update</button>
<?php elseif ($del == true) : ?>
    <button class="btn btn-lg btn-block btn-danger" style="border-radius: 0%;"
type="submit" name="delete">Delete</button>
<?php else : ?>
    <button type="submit" name="add" class="btn btn-lg btn-block btn-success"
style="border-radius: 0%;">Add Expense</button>
<?php endif ?>
```

```

## 9. JavaScript and Menu Toggle Script:

- The code includes JavaScript for toggling the sidebar.


```
````javascript
$("#menu-toggle").click(function(e) {
    e.preventDefault();
    $("#wrapper").toggleClass("toggled");
});
````
```

## 10. Feather Icons:

- Feather icons are used for displaying icons on the page.

```
````html
<script src="js/feather.min.js"></script>
````
```

Overall, this code combines PHP for server-side logic (handling form submissions, interacting with the database) with HTML and Bootstrap for the user interface. It provides functionality for users to add, update, and delete expense records.

 Output:

### Add Your Daily Expenses

Enter Amount(\$)

Date

21-09-2023




Category

- ☐ Medicine
- ☐ Food
- ☐ Bills and Recharges
- ☒ Entertainment
- ☐ Clothings
- ☐ Rent
- ☐ Household Items
- ☐ Others

Add Expense

## 5 Manage Expense :

 Sample Code:

```
<div class="container-fluid">
  <h3 class="mt-4 text-center">Manage Expenses</h3>
  <hr>
  <div class="row justify-content-center">

    <div class="col-md-6">
      <table class="table table-hover table-bordered">
        <thead>
          <tr class="text-center">
            <th>#</th>
            <th>Date</th>
            <th>Amount</th>
            <th>Expense Category</th>
            <th colspan="2">Action</th>
          </tr>
        </thead>

        <?php $count=1; while ($row =
mysqli_fetch_array($exp_fetched)) { ?>
          <tr>
            <td><?php echo $count;?></td>
            <td><?php echo $row['expensedate'];
?></td>
            <td><?php echo '$'.$row['expense'];
?></td>
            <td><?php echo $row['expensecategory'];
?></td>
            <td class="text-center">
              <a href="add_expense.php?edit=<?php
echo $row['expense_id']; ?>" class="btn btn-primary btn-sm" style="border-
radius:0%;">Edit</a>
            </td>
            <td class="text-center">
              <a href="add_expense.php?delete=<?php
echo $row['expense_id']; ?>" class="btn btn-danger btn-sm" style="border-
radius:0%;">Delete</a>
            </td>
          </tr>
          <?php $count++; } ?>
        </table>
      </div>
    </div>
  </div>
```

```
</div>
</div>
```

### Description of the Code:

This PHP code represents the "Manage Expenses" page of the Daily Expense System. Below is a description of the key components and functionality:

#### 1. Session Inclusion:

```
```php
include("session.php");
```
```

- This includes a file named `session.php`, which likely contains code related to managing user sessions. It's used for authentication and tracking user information across pages.

#### 2. Expense Query:

```
```php
$exp_fetched = mysqli_query($con, "SELECT * FROM expenses WHERE user_id =
'$userid'");
```
```

- This query fetches all expenses associated with the current user.

#### 3. HTML and Bootstrap Components:

- The code includes HTML elements and classes from Bootstrap to structure and style the "Manage Expenses" page. It displays a table to list expenses.

#### 4. Feather Icons:

- Feather icons are used for displaying icons on the page.

```
```html
<script src="js/feather.min.js"></script>
```
```

#### 5. Expense Table:

- The expenses fetched from the database are displayed in a table.

```
```php
<?php $count=1; while ($row = mysqli_fetch_array($exp_fetched)) { ?>
    <!-- Display each expense record -->
    <?php $count++; } ?>
```
```



- The table displays information like expense ID, date, amount, category, and provides options to edit or delete each record.

```
```html
<td><?php echo $count;?></td>
<td>$<?php echo $row['expensedate']; ?></td>
<td><?php echo '$'.$row['expense']; ?></td>
<td><?php echo $row['expensecategory']; ?></td>
<td class="text-center">
    <a href="add_expense.php?edit=<?php echo $row['expense_id']; ?>" class="btn btn-
primary btn-sm" style="border-radius:0%;">Edit</a>
</td>
<td class="text-center">
    <a href="add_expense.php?delete=<?php echo $row['expense_id']; ?>" class="btn btn-
danger btn-sm" style="border-radius:0%;">Delete</a>
</td>
```
```

## 6. JavaScript and Menu Toggle Script:

- The code includes JavaScript for toggling the sidebar.

```
```javascript
$("#menu-toggle").click(function(e) {
    e.preventDefault();
    $("#wrapper").toggleClass("toggled");
});
```
```

## 7. Feather Icons Replacement:

- This script replaces any `` elements with the `data-feather` attribute with the corresponding Feather icon.

```
```javascript
feather.replace()
```
```

Overall, this code combines PHP for server-side logic (fetching expenses from the database) with HTML and Bootstrap for the user interface. It provides functionality for users to view, edit, and delete their expense records.

 Output:

## Manage Expenses

| # | Date | Amount | Expense Category | Action |
|---|------|--------|------------------|--------|
|---|------|--------|------------------|--------|

### 6 Database (Backend):

 Sample Code:

```
-- phpMyAdmin SQL Dump
-- version 4.6.5.2
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Jun 23, 2021 at 07:16 PM
-- Server version: 5.6.21
-- PHP Version: 5.6.3

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `dailyexpense`
--

--
-- Table structure for table `expenses`
--

CREATE TABLE `expenses` (
  `expense_id` int(20) NOT NULL,
  `user_id` varchar(15) NOT NULL,
  `expense` int(20) NOT NULL,
  `expensedate` varchar(15) NOT NULL,
  `expensecategory` varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```

--
-- Table structure for table `users`
--

CREATE TABLE `users` (
  `user_id` int(11) NOT NULL,
  `firstname` varchar(50) NOT NULL,
  `lastname` varchar(25) NOT NULL,
  `email` varchar(50) NOT NULL,
  `profile_path` varchar(50) NOT NULL DEFAULT 'default_profile.png',
  `password` varchar(50) NOT NULL,
  `trn_date` datetime NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

--
-- Indexes for dumped tables
--

--
-- Indexes for table `expenses`
--
ALTER TABLE `expenses`
  ADD PRIMARY KEY (`expense_id`);


--
-- Indexes for table `users`
--
ALTER TABLE `users`
  ADD PRIMARY KEY (`user_id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `expenses`
--
ALTER TABLE `expenses`
  MODIFY `expense_id` int(20) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=77;

--
-- AUTO_INCREMENT for table `users`
--
ALTER TABLE `users`
  MODIFY `user_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```



## Description of the Code:

This code is a SQL dump generated by phpMyAdmin, a popular web-based administration tool for managing MySQL databases. The dump includes the structure and data for two tables in a database named `dailyexpense`. The tables are named `expenses` and `users`.

### 1. Database Information:

- Database Name: `dailyexpense`
- Host: `127.0.0.1` (localhost)
- MySQL Server Version: `5.6.21`
- PHP Version: `5.6.3`

### 2. SQL Modes and Time Zone Settings:

- `SET SQL\_MODE = "NO\_AUTO\_VALUE\_ON\_ZERO";`: This setting ensures that an explicit value must be specified for the `AUTO\_INCREMENT` column to prevent automatically generating zero.
- `SET time\_zone = "+00:00";`: Sets the default time zone to UTC.

### 3. Table Definitions:

- There are two tables: `expenses` and `users`.
- The `expenses` table has columns: `expense\_id`, `user\_id`, `expense`, `expensedate`, and `expensecategory`. It uses the InnoDB storage engine with a default character set of `latin1`.
- The `users` table has columns: `user\_id`, `firstname`, `lastname`, `email`, `profile\_path`, `password`, and `trn\_date`. It also uses the InnoDB storage engine with the same character set.

### 4. Indexes:

- Both tables have primary key indexes defined on their respective `id` columns (`expense\_id` for `expenses` and `user\_id` for `users`).

### 5. Auto-Increment Settings:

- Auto-increment is set for both tables to generate unique identifiers for the `id` columns (`expense\_id` and `user\_id`).
- The initial auto-increment values are set to 77 for `expenses` and 7 for `users`.

### 6. Character Set and Collation Settings:

- The character set for the connection is set to `utf8mb4`.
- There are several statements at the end of the script to revert to the old character set, collation, and character set results settings.

Overall, This code is useful for creating the structure of a MySQL database and its tables, including primary keys, indexes, and auto-increment settings. It also includes some specific configurations related to character sets and collations.

🚦 All Outputs with Backend:

## 6.1 Register

Register

SANJIL

K C

sanjil.kc2020@vitstudent.ac.in

.....

.....

☒ I accept the [Terms of Use & Privacy Policy](#)

Register

Already have an account? [Login Here](#)

## 6.2 Login page

D.E.M.S

Login Panel

sanjil.kc2020@vitstudent.ac.in

.....

Login

☒ Remember me

Don't have an account? [Register Here](#)

## Update Profile

Change Photo

Browse

Upload Picture

First name

SANJIL

Last name

K C

Email

sanjil.kc2020@vitstudent.ac.in

Save Changes

## 6.3 Add Expense

### Add Your Daily Expenses

Enter Amount(\$)

100

Date

21-11-2023

Category

- ☒ Medicine
- ☐ Food
- ☐ Bills and Recharges
- ☐ Entertainment
- ☐ Clothings
- ☐ Rent
- ☐ Household Items
- ☐ Others

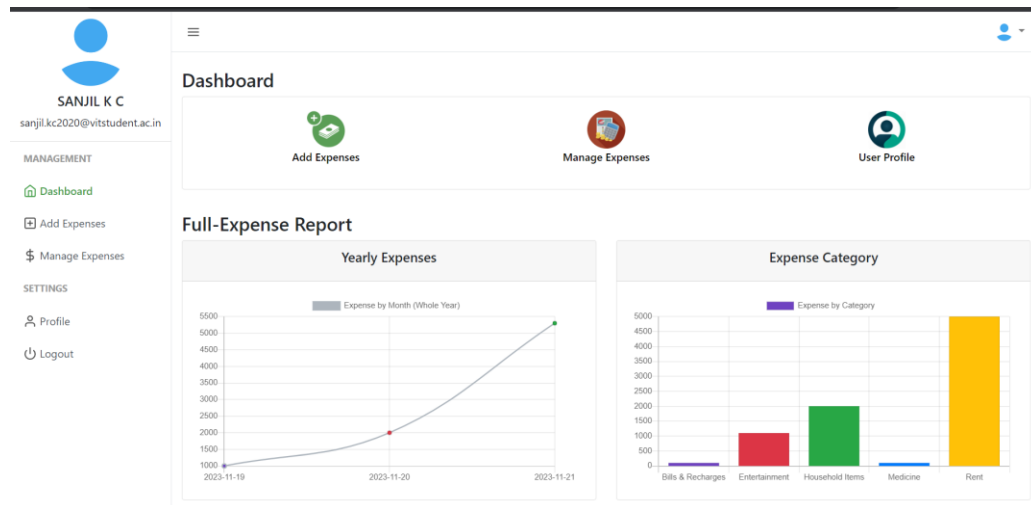
Add Expense

## 6.4 Manage Expense

### Manage Expenses

| # | Date         | Amount | Expense Category  | Action |        |
|---|--------------|--------|-------------------|--------|--------|
| 1 | \$2023-11-21 | \$100  | Medicine          | Edit   | Delete |
| 2 | \$2023-11-21 | \$100  | Entertainment     | Edit   | Delete |
| 3 | \$2023-11-21 | \$5000 | Rent              | Edit   | Delete |
| 4 | \$2023-11-21 | \$100  | Bills & Recharges | Edit   | Delete |
| 5 | \$2023-11-20 | \$2000 | Household Items   | Edit   | Delete |
| 6 | \$2023-11-19 | \$1000 | Entertainment     | Edit   | Delete |

## 6.5 Dashboard



## 7 Conclusion:

In conclusion, the Daily Expense System serves as an effective solution for individuals and households looking to gain better control over their finances. By leveraging the web-based application built with PHP and MySQL, users can easily track and manage their daily expenses, fostering a more informed approach to budgeting and financial planning.

This system's user-friendly interface makes it accessible to a wide range of users, ensuring that everyone, regardless of technical proficiency, can benefit from its features. The ability to record and categorize expenses provides valuable insights into spending habits, allowing users to identify areas for potential savings and make informed financial decisions.

Furthermore, the utilization of MySQL as the database management system ensures data integrity and security, providing users with a reliable platform to store their financial information. The web-based nature of the application also offers the convenience of accessibility from any device with internet connectivity, enhancing its usability and practicality for users on the go.

In essence, the Daily Expense System is not just a tracking tool but a comprehensive aid in cultivating responsible financial habits. By encouraging regular expense monitoring, this application empowers users to make conscious choices about their expenditures, ultimately contributing to their long-term financial well-being. As a testament to the power of technology in personal finance management, this project stands as a valuable resource for those seeking to achieve financial stability and success.

### Code GitHub Link

<https://github.com/sanjil2/DAILY-EXPENSE-SYSTEM>