****

**BAHRIA UNIVERSITY, (Karachi Campus)**

*Department of Software Engineering*

**REPORT**

**COURSE: SEL 221**

**Software design and architecture**

**CLASS: BSE – 4A (SPRING - 2024)**

**Expense Tracker**

**Group Name**

|  |  |
| --- | --- |
| **Student Name** | **Enrollment#** |
| Syed Fahad Ali Gillani | 02-131222-083 |

**Submitted to:**

Course Instructor Lab Instructor . Eng. Ramsha Engr. Asma Shaheen

Submission Date**: 11/06/2024**

**Abstract**

The Expense Tracker is a web-based application developed using ASP.NET C# that enables users to track their financial transactions conveniently. With features such as expense categorization, income tracking, and comprehensive reporting, the system empowers users to manage their finances effectively. By providing a user-friendly interface and robust data security measures, the system aims to simplify the process of financial management for individuals and businesses alike.

Table of Contents

[1. INTRODUCTION: 4](#_Toc168859705)

[2. ABSTRACT: 4](#_Toc168859706)

[3. PROJECT SCOPE: 4](#_Toc168859707)

[4. PROJECT FUNCTIONALITIES: 5](#_Toc168859708)

[5. ASSUMPTIONS AND CONSTRAINTS: 5](#_Toc168859709)

[6. UML DIAGRAMS (FLOW DIAGRAM, USE CASE, CLASS DIAGRAM, AND ENTITY RELATIONSHIP DIAGRAM) 6](#_Toc168859710)

[7. MODULE DISTRIBUTION 7](#_Toc168859711)

[8. CODE 8](#_Toc168859712)

[10. REFRENCE 16](#_Toc168859713)

# **INTRODUCTION:**

Our lives are becoming increasingly complex, and with that comes the need for better financial management tools. In line with this, we propose to develop an Expense Tracker using ASP.NET C#. This system aims to provide users with a robust platform to track their expenses and incomes efficiently, facilitating informed decision-making and financial planning.

# **ABSTRACT:**

The Expense Tracker is a web-based application developed using ASP.NET C# that enables users to track their financial transactions conveniently. With features such as expense categorization, income tracking, and comprehensive reporting, the system empowers users to manage their finances effectively. By providing a user-friendly interface and robust data security measures, the system aims to simplify the process of financial management for individuals and businesses alike.

# **PROJECT SCOPE:**

The scope of the project includes the development of a fully functional web application that allows users to:

* Register and log in securely to their accounts.
* Manipulate expenses with customizable categories.
* Track various sources of income.
* View summarized reports of expenses and incomes.
* Implement role-based access control to manage user permissions.
* Ensure data security through encryption and secure authentication mechanisms.

# **PROJECT FUNCTIONALITIES:**

The Expense Tracker will include the following key functionalities:

* User Registration and Authentication: Secure user registration and login system to access the application.
* Expense Tracking: Ability to manipulate expenses with customizable categories and optional attachments.
* Income Tracking: Capability to manipulate different sources of income.
* Dashboard: Provide users with an overview of their financial status through summarized data.

# **ASSUMPTIONS AND CONSTRAINTS:**

**Assumptions:**

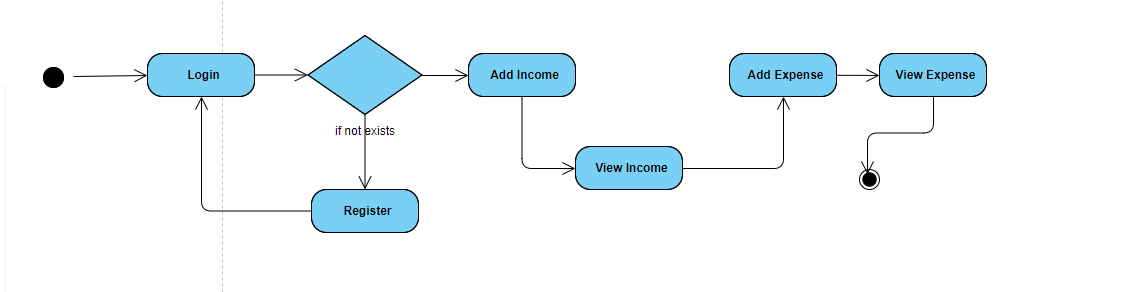
* Users will have a basic understanding of financial management principles.
* The application will be accessed using modern web browsers with stable internet connectivity.
* Users will input accurate and relevant financial data for effective tracking and analysis.

**Constraints:**

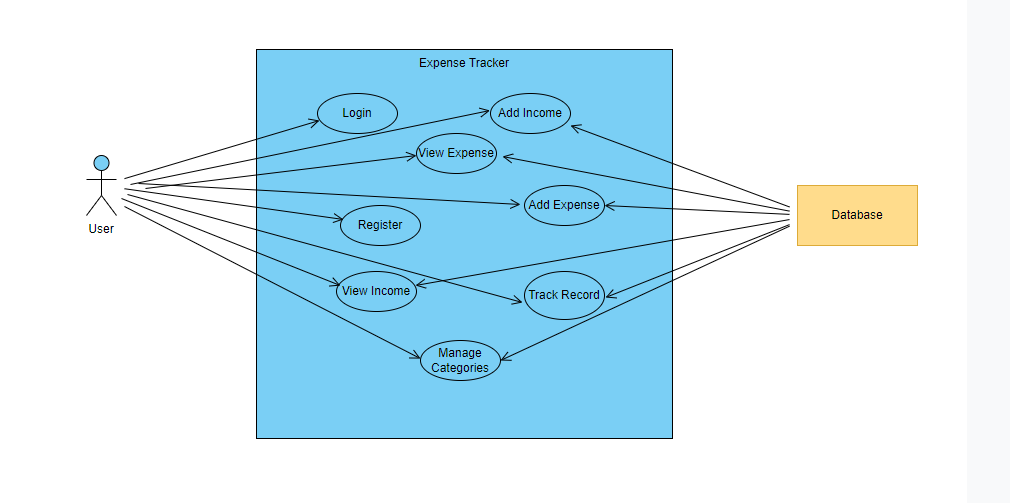
* Limited resources (time and manpower) may affect the project timeline and scope.
* Compatibility issues may arise with older web browsers or mobile devices.
* Data security and privacy concerns need to be addressed to ensure compliance with regulatory requirements.

# **UML DIAGRAMS (FLOW DIAGRAM, USE CASE, CLASS DIAGRAM, AND ENTITY RELATIONSHIP DIAGRAM)**

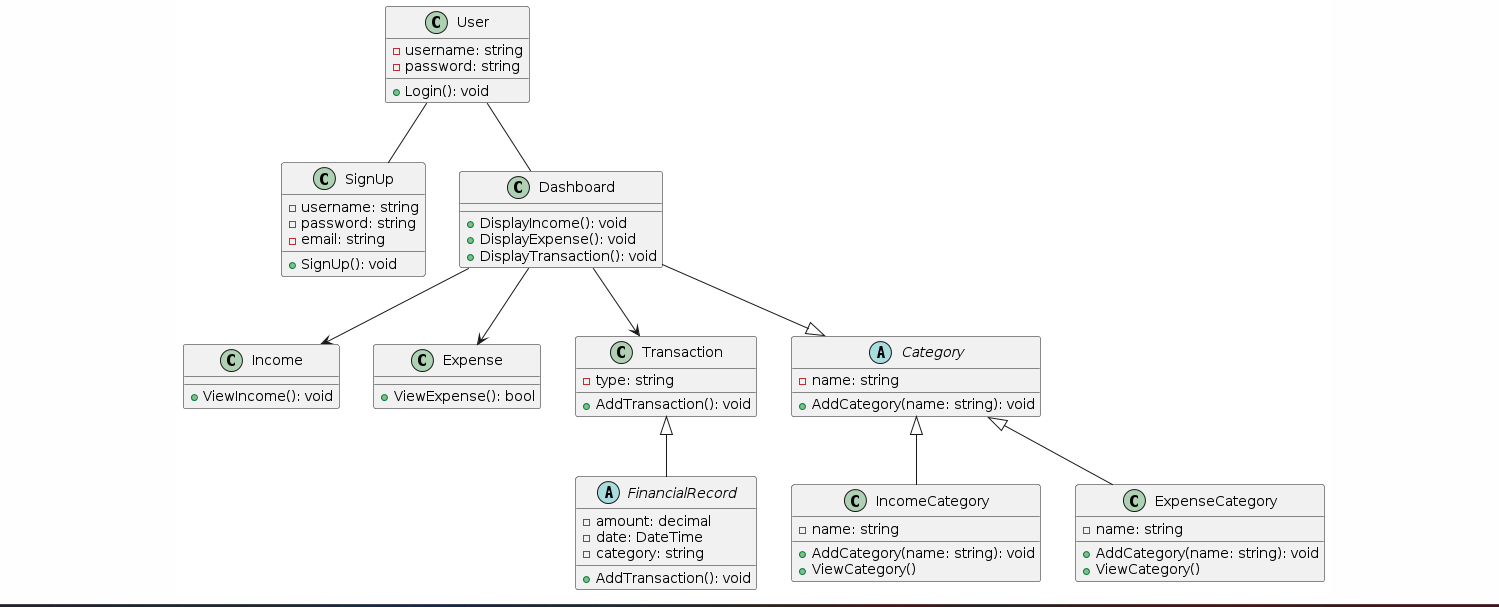
**FLOW DIAGRAM**



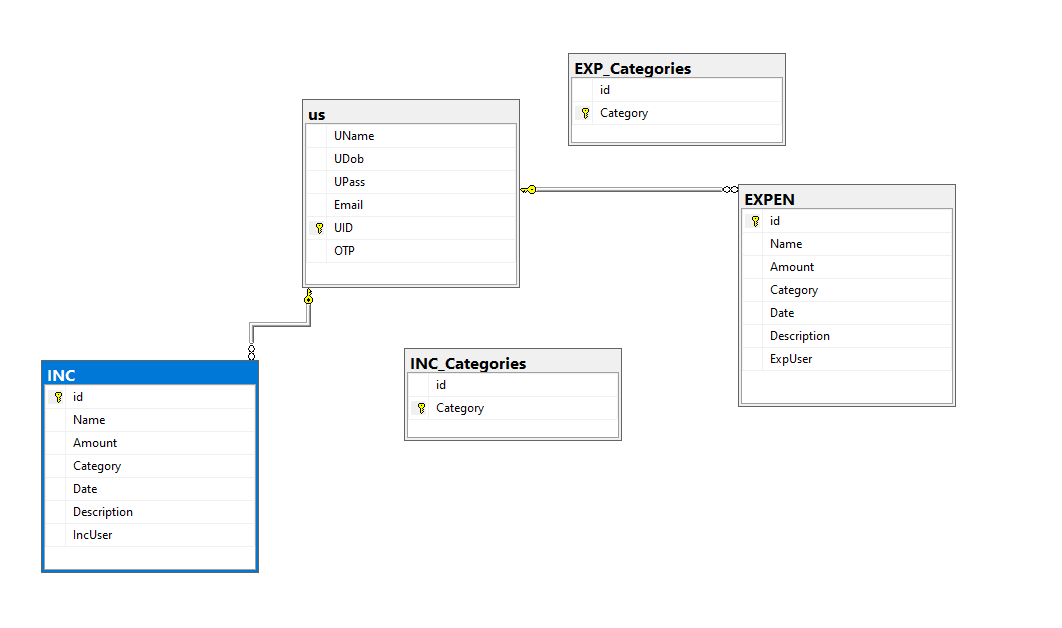
**USE CASE DIAGRAM**



**CLASS DIAGRAM**

****

**ER-DIAGRAM**



# **MODULE DISTRIBUTION**

I began by laying the groundwork for my Expense Tracker project, focusing initially on creating user-friendly Login and Signup pages to ensure seamless access. Transitioning to the core of the application, the Expense Tracking Module, I meticulously organized the database structure to efficiently manage income, expenses, and transaction data. Developing intuitive interfaces for adding, and categorizing expenses, I prioritized simplicity and functionality. Regularly soliciting and incorporating user feedback, I fine-tuned the system to address any usability issues. Finally, I crafted a Dashboard module to offer users a clear snapshot of their spending habits and financial health. My goal throughout was to create an Expense Tracker that simplifies financial management and empowers users to make informed decisions about their finances.

# **CODE**

**Singleton**

public class UserSessionManager

{

private static UserSessionManager \_instance;

private static readonly object \_lock = new object();

private static bool \_isAnyUserLoggedIn = false; // Global state to track if any user is logged in

private Dictionary<string, string> \_activeSessions;

private UserSessionManager()

{

\_activeSessions = new Dictionary<string, string>();

}

public static UserSessionManager Instance

{

get

{

lock (\_lock)

{

if (\_instance == null)

{

\_instance = new UserSessionManager();

}

return \_instance;

}

}

}

public bool IsUserLoggedIn(string userId)

{

return \_activeSessions.ContainsKey(userId);

}

public bool IsAnyUserLoggedIn()

{

return \_isAnyUserLoggedIn;

}

public void LoginUser(string userId, string sessionId)

{

lock (\_lock)

{

if (\_isAnyUserLoggedIn)

{

throw new Exception("Another user is already logged in");

}

if (!IsUserLoggedIn(userId))

{

\_activeSessions[userId] = sessionId;

\_isAnyUserLoggedIn = true; // Set the global state to true

}

else

{

throw new Exception("User already logged in");

}

}

}

public void LogoutUser(string userId)

{

lock (\_lock)

{

if (IsUserLoggedIn(userId))

{

\_activeSessions.Remove(userId);

\_isAnyUserLoggedIn = false; // Set the global state to false

}

}

}

public string GetUserSession(string userId)

{

return \_activeSessions.TryGetValue(userId, out var sessionId) ? sessionId : null;

}

}

**Adapter**

public interface IDataAccessAdapter

{

void Signup(string name, DateTime dob, int pass, string email);

void InsertIncome(string name, int amount, string cat, string incuser, DateTime date, string desc);

void InsertExpense(string name, int amount, string cat, string expuser, DateTime date, string desc);

object ViewExpense(string incuser);

object ViewIncome(string expuser);

void Insert\_INC\_Category(string cat);

void Insert\_EXP\_Category(string cat);

object ViewIncomeCategory();

object ViewExpenseCategory();

void Delete\_INC\_Category(string cat);

void Delete\_EXP\_Category(string cat);

void DeleteExpense(string name);

void DeleteIncome(string name);

}

// Adapter class implementing the IDataAccessAdapter interface

public class DataAccessAdapter : IDataAccessAdapter

{

private readonly DataLayer \_dataLayer;

public DataAccessAdapter()

{

\_dataLayer = new DataLayer();

}

public void Signup(string name, DateTime dob, int pass, string email)

{

\_dataLayer.Signup(name, dob, pass, email);

}

public void InsertIncome(string name, int amount, string cat, string incuser, DateTime date, string desc)

{

\_dataLayer.InsertIncome(name, amount, cat, incuser, date, desc);

}

public void InsertExpense(string name, int amount, string cat, string expuser, DateTime date, string desc)

{

\_dataLayer.InsertExpense(name, amount, cat, expuser, date, desc);

}

public object ViewExpense(string incuser)

{

return \_dataLayer.ViewExpense(incuser);

}

public object ViewIncome(string expuser)

{

return \_dataLayer.ViewIncome(expuser);

}

public void Insert\_INC\_Category(string cat)

{

\_dataLayer.Insert\_INC\_Category(cat);

}

public void Insert\_EXP\_Category(string cat)

{

\_dataLayer.Insert\_EXP\_Category(cat);

}

public object ViewIncomeCategory()

{

return \_dataLayer.ViewIncomeCategory();

}

public object ViewExpenseCategory()

{

return \_dataLayer.ViewExpenseCategory();

}

public void Delete\_INC\_Category(string cat)

{

\_dataLayer.Delete\_INC\_Category(cat);

}

public void Delete\_EXP\_Category(string cat)

{

\_dataLayer.Delete\_EXP\_Category(cat);

}

public void DeleteExpense(string name)

{

\_dataLayer.DeleteExpense(name);

}

public void DeleteIncome(string name)

{

\_dataLayer.DeleteIncome(name);

}

}

public class BLayer

{

private readonly IDataAccessAdapter \_dataAccessAdapter;

public BLayer(IDataAccessAdapter dataAccessAdapter)

{

\_dataAccessAdapter = dataAccessAdapter;

}

public void Signup(string name, DateTime dob, int pass, string email)

{

\_dataAccessAdapter.Signup(name, dob, pass, email);

}

public void InsertIncome(string name, int amount, string cat, string incuser, DateTime date, string desc)

{

try

{

\_dataAccessAdapter.InsertIncome(name, amount, cat, incuser, date, desc);

}

catch (Exception ex)

{

throw new ApplicationException("An error occurred in the business layer: " + ex.Message);

}

}

public void InsertExpense(string name, int amount, string cat, string expuser, DateTime date, string desc)

{

try

{

\_dataAccessAdapter.InsertExpense(name, amount, cat, expuser, date, desc);

}

catch (Exception ex)

{

throw new ApplicationException("An error occurred in the business layer: " + ex.Message);

}

}

public object ViewExpense(string incuser)

{

return \_dataAccessAdapter.ViewExpense(incuser);

}

public object ViewIncome(string expuser)

{

return \_dataAccessAdapter.ViewIncome(expuser);

}

public void Insert\_INC\_Category(string cat)

{

\_dataAccessAdapter.Insert\_INC\_Category(cat);

}

public void Insert\_EXP\_Category(string cat)

{

\_dataAccessAdapter.Insert\_EXP\_Category(cat);

}

public object ViewIncomeCategory()

{

return \_dataAccessAdapter.ViewIncomeCategory();

}

public object ViewExpenseCategory()

{

return \_dataAccessAdapter.ViewExpenseCategory();

}

public void Delete\_INC\_Category(string cat)

{

\_dataAccessAdapter.Delete\_INC\_Category(cat);

}

public void Delete\_EXP\_Category(string cat)

{

\_dataAccessAdapter.Delete\_EXP\_Category(cat);

}

public void DeleteExpense(string name)

{

\_dataAccessAdapter.DeleteExpense(name);

}

public void DeleteIncome(string name)

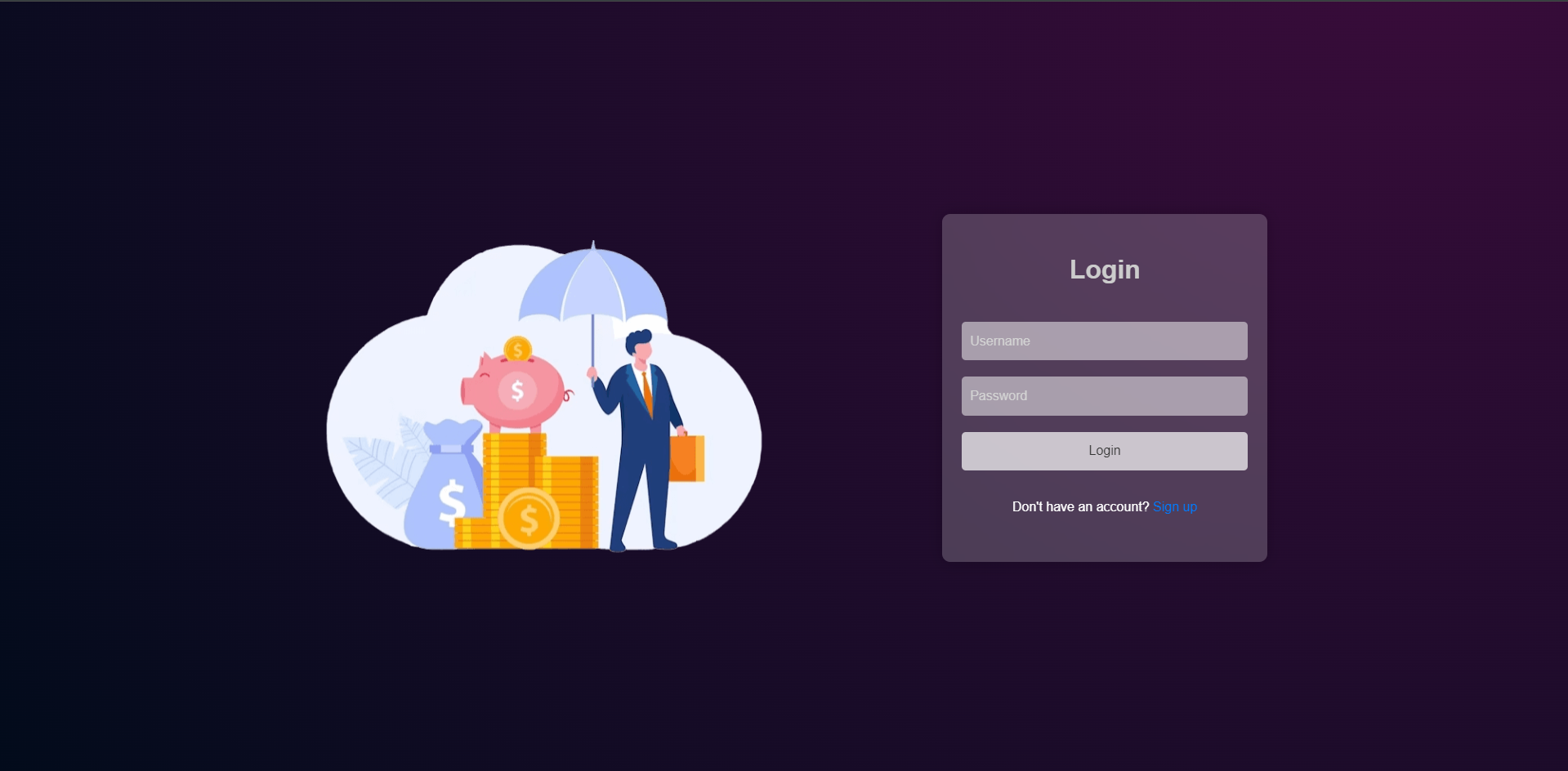
{

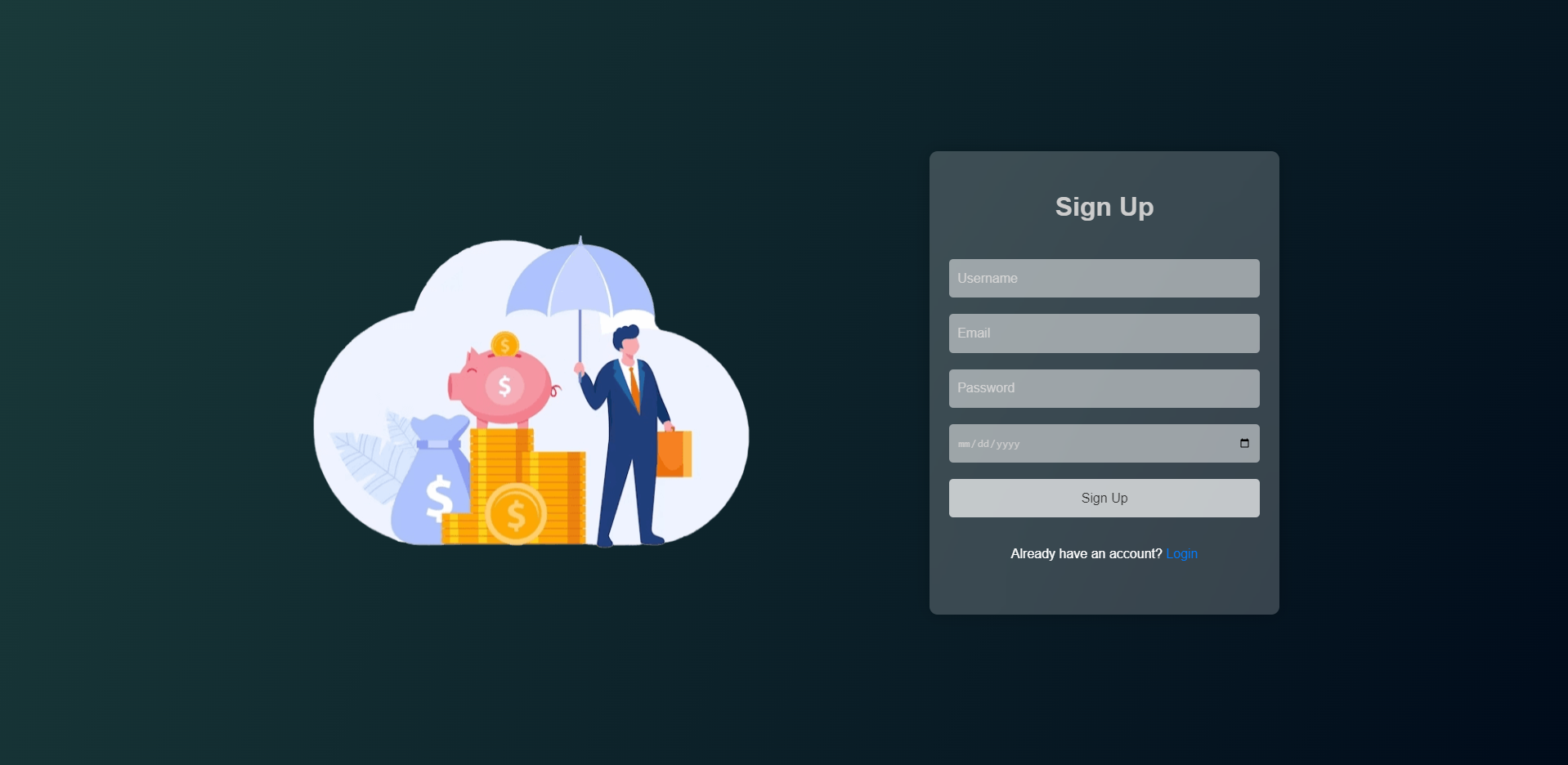
\_dataAccessAdapter.DeleteIncome(name);

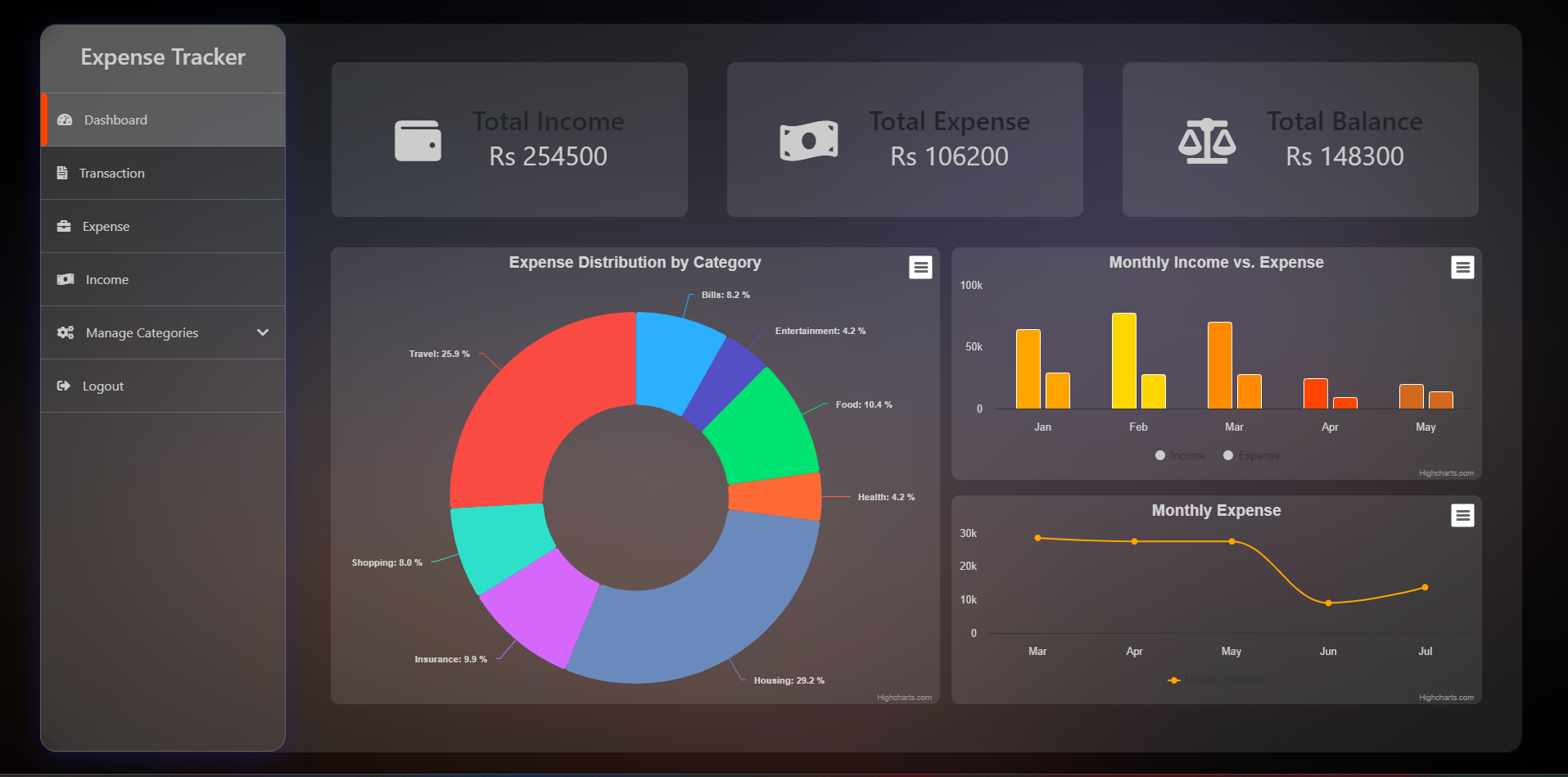
}

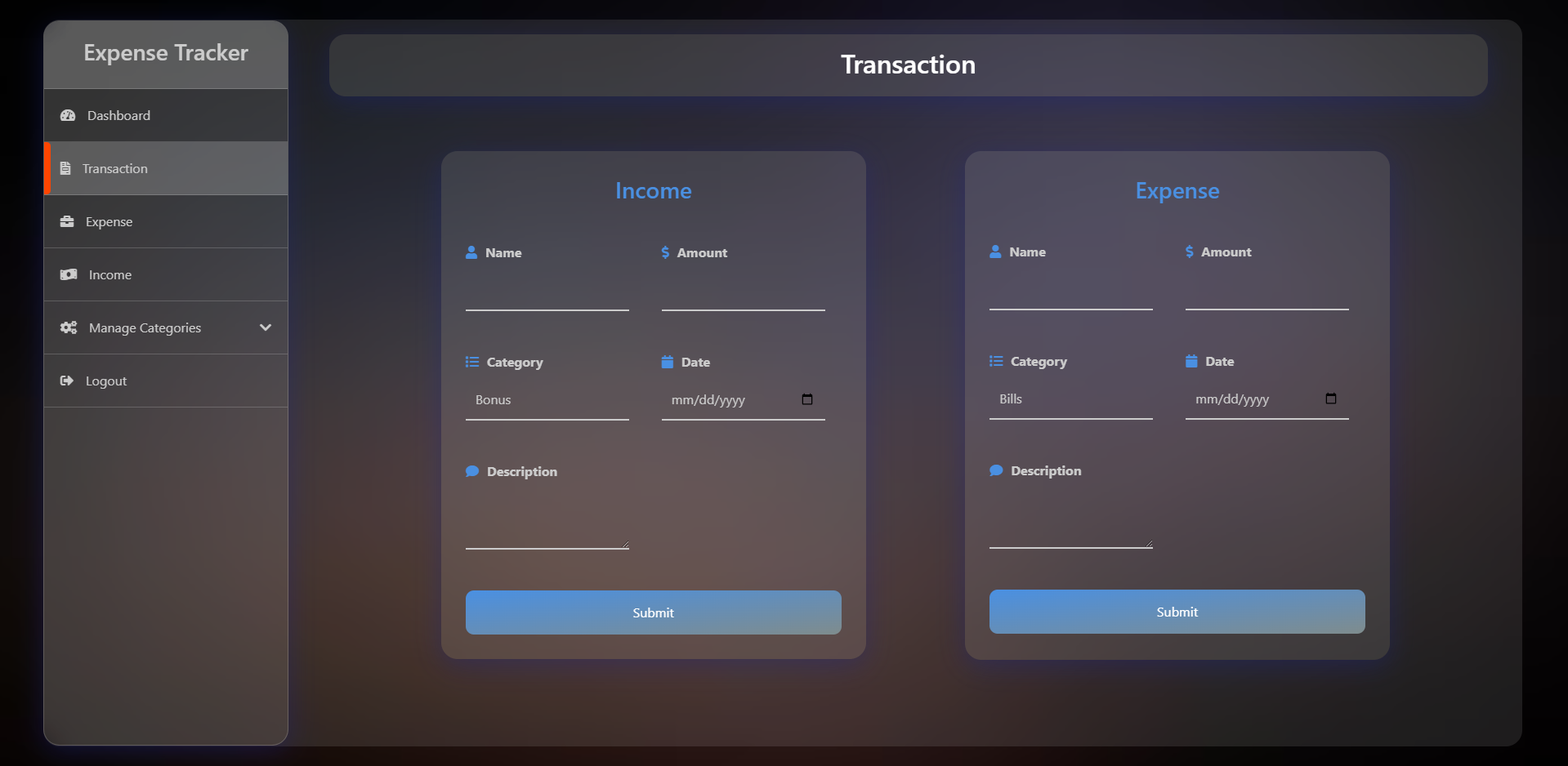
}

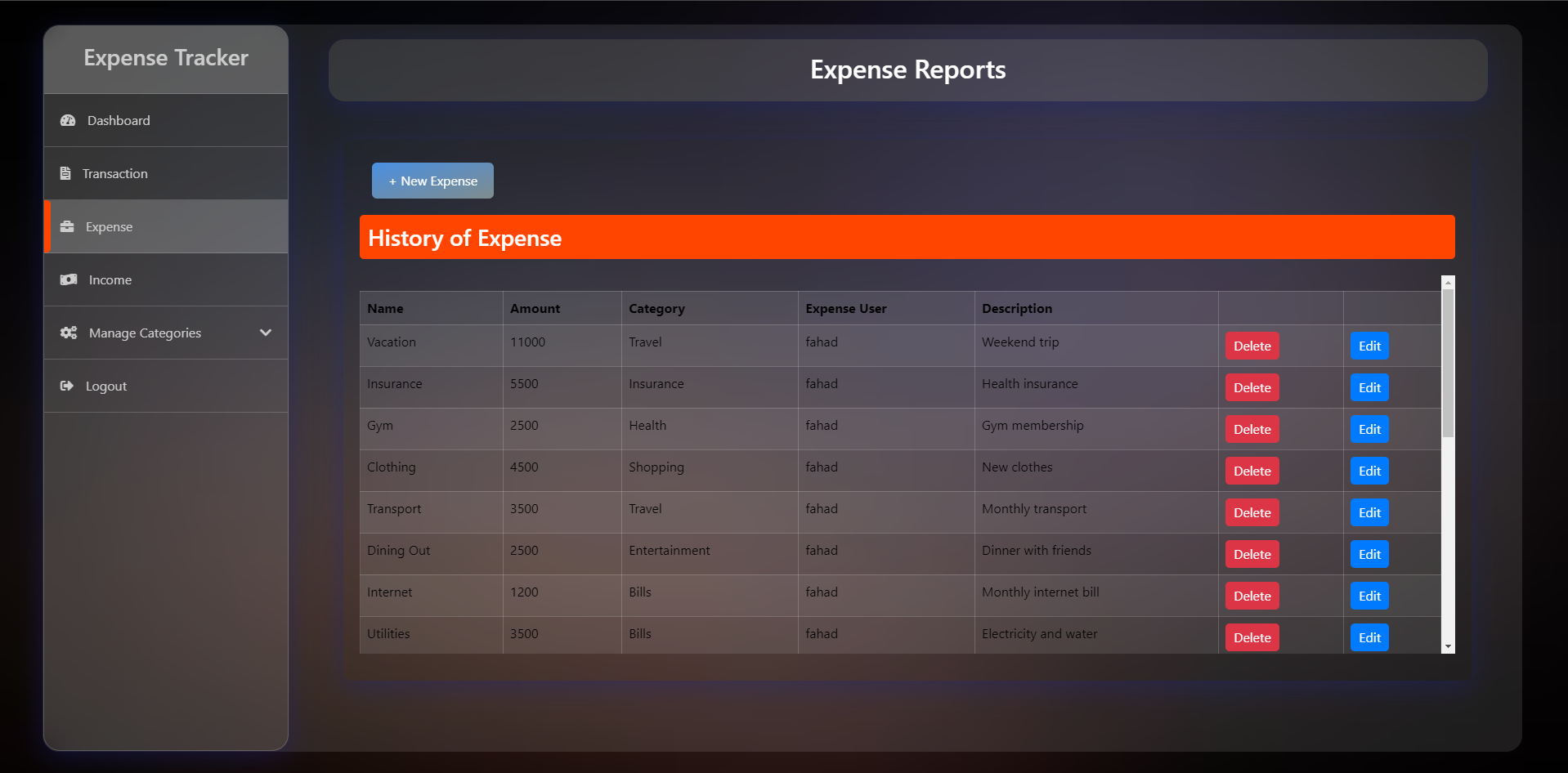
1. **INTERFACE**

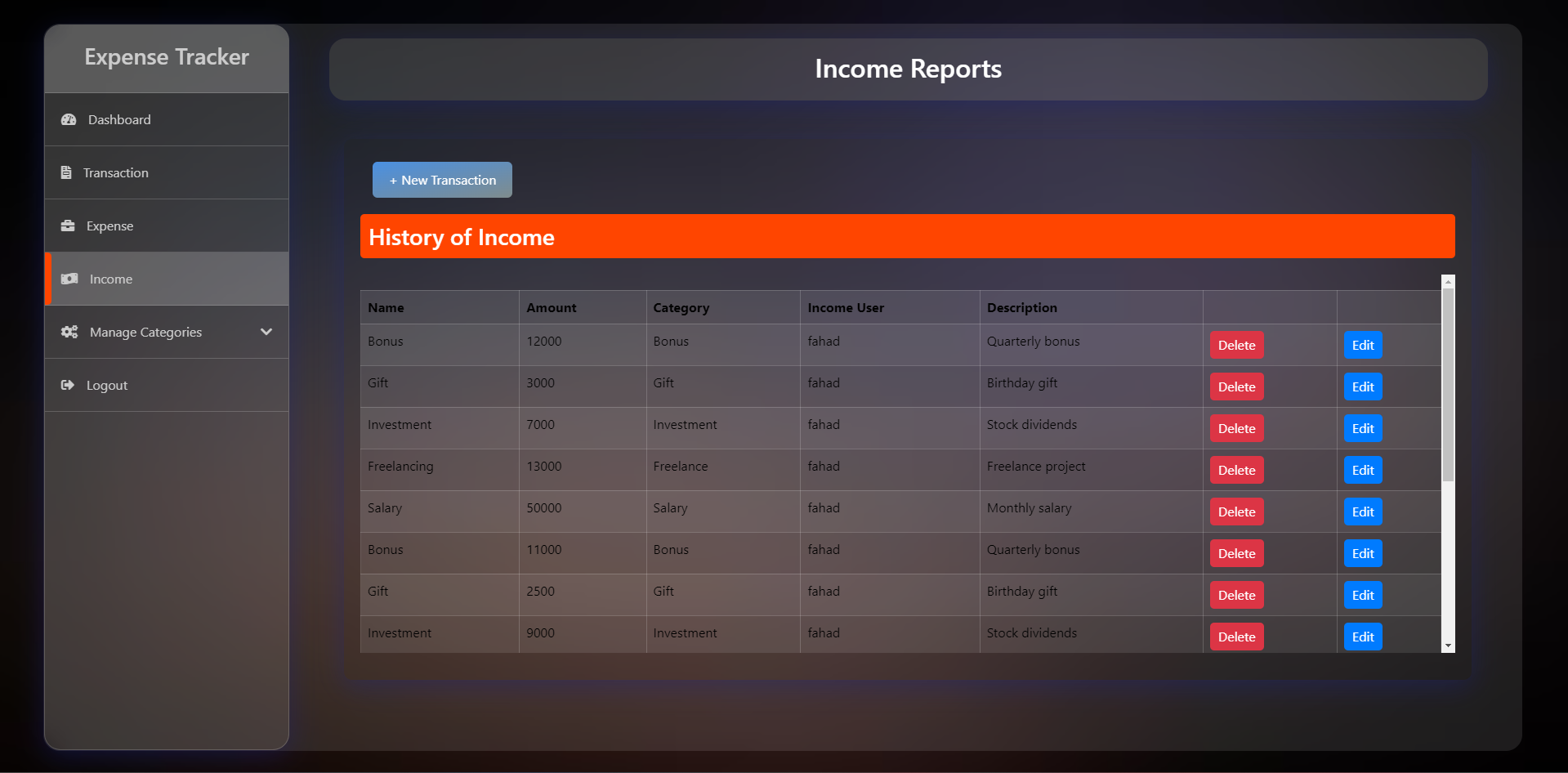
****

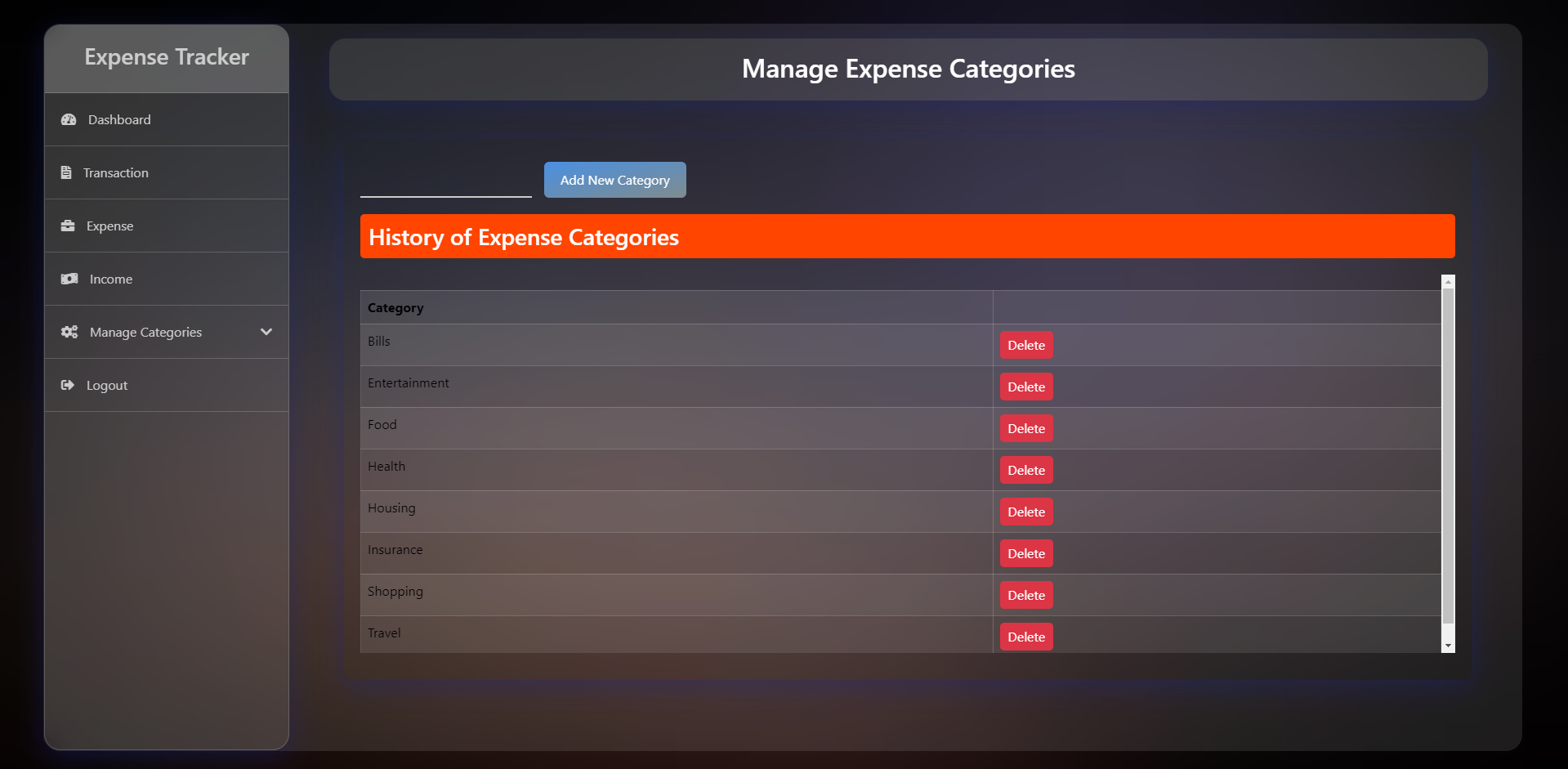
****

****

****

****

****





# **REFRENCE**

<https://drive.google.com/drive/folders/1QcYWAa5g3f2tqWfS4yG41jvz10isMC-y>

**Teacher Signature**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Remarks**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Submission Date**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_