# CSTP 2104 Project

## Expenses Recorder

ASP.NET Core Web App & Local SQL Server

Jia Xi Lin
000460612

GitHub Repository
https://github.com/hisonlin/CSTP2104-FinalProject

## Overview

It is a simple personal expenses recording web app. You can easily manage your expenses. This project is created by ASP.NET Core Web App and hosted on a local server. All expenses data are stored into a local SQL server.

## Goals

1. Create Web App layout using ASP.NET Core Web App
2. Store all the expenses data into a local SQL server

## Features and Specifications

The personal expenses recording web app is built on ASP.NET Core Web App, providing users with a seamless platform to efficiently manage their finances. The user interface begins with an intuitive 'Home' page, serving as the gateway to the application. Once accessed, users can effortlessly navigate to the 'My Expenses' page, where a comprehensive overview of all recorded expenses is displayed.

Within this section, users have the flexibility to add new expenses, edit existing entries, or remove records as needed.

The entire expense data is securely stored and managed within a local SQL server, ensuring reliability and data integrity. This design offers users a centralized hub to organize their financial activities, facilitating easy monitoring, tracking, and control of personal expenditures.

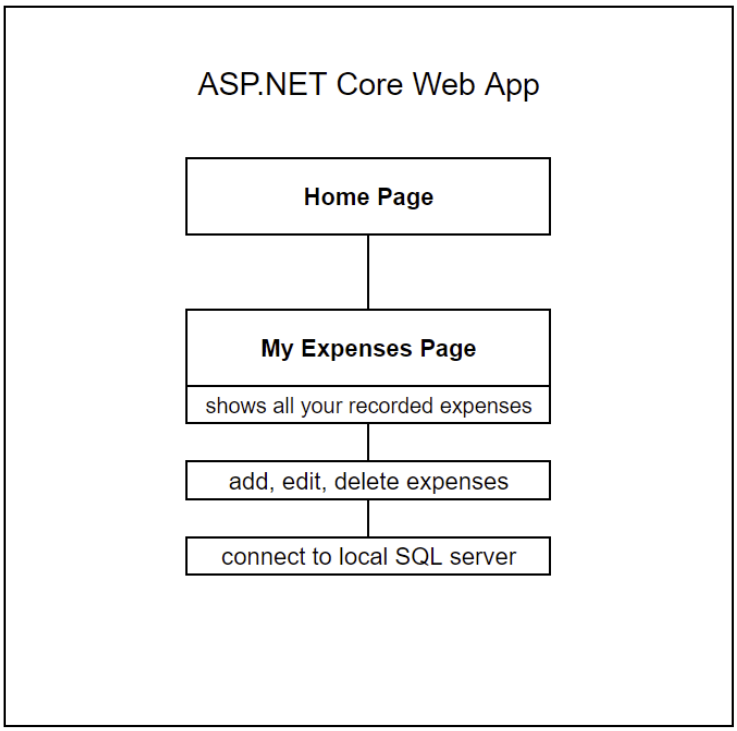
## Project Design

The project design centers around an ASP.NET Core Web App, establishing a user-friendly personal expenses recording system. Commencing with a welcoming 'Home' page, users can seamlessly transition to the 'My Expenses' section, where a comprehensive overview of their financial transactions is presented.

Within this interface, users enjoy the convenience of adding, modifying, or deleting expenses to tailor their financial records.

The entire expense dataset is securely housed in a local SQL server, ensuring robust data storage and retrieval.

This design emphasizes user accessibility and data management, fostering a streamlined approach for users to effectively organize and monitor their financial activities.



# Implementation

1. **Create an ASP.NET Core Web App project**
   Create a new ASP.NET Core Web App project and choose MVC template

2. **Add a data Model (Expense) with properties(Id, Description, Amount, Category and Date)**

   Create a new folder named "**Models**" in your project. Inside this folder, create a class file named **Expense.cs**. Define the Expense model with properties:

```
using System.ComponentModel.DataAnnotations;

namespace ExpenseTracker.Models
{
    6 references
    public class Expense
    {
        11 references
        public int Id { get; set; }
        12 references
        public string Description { get; set; }
        12 references
        public decimal Amount { get; set; }
        12 references
        public string Category { get; set; }

        [DataType(DataType.Date)]
        12 references
        public DateTime Date { get; set; }
    }
}
```

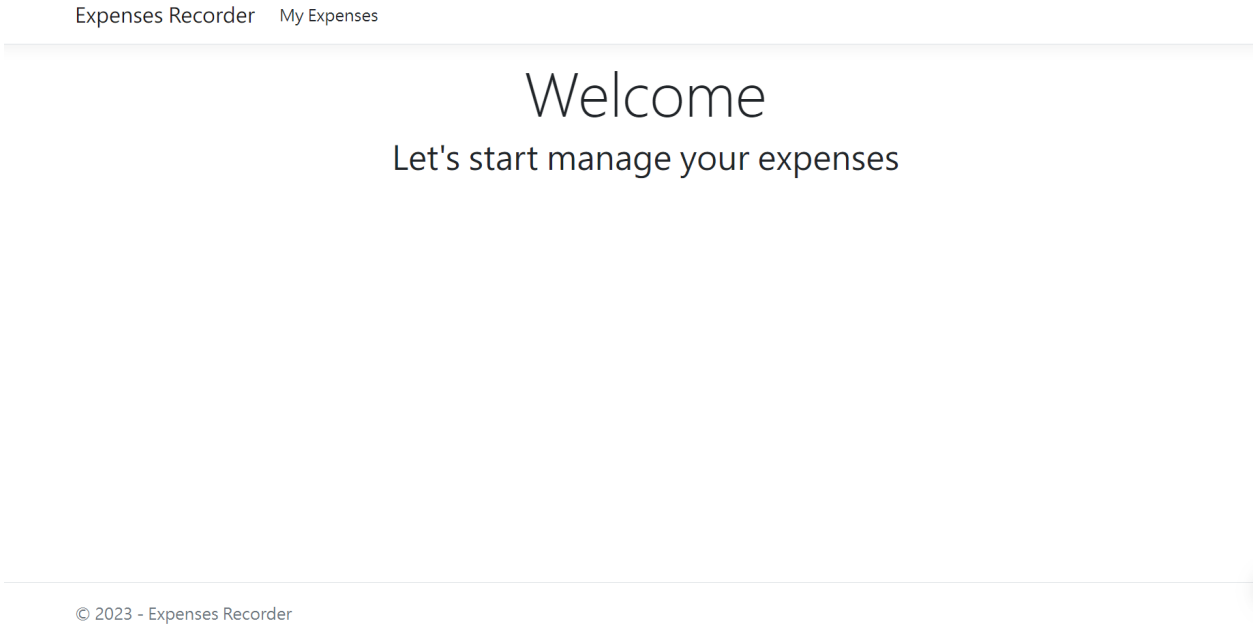
3. **Scaffold the Expense Model - allow Create, Read, Update, and Delete (CRUD) operations**
   The scaffold process creates the following files:
   - Pages/Expenses: Create, Delete, Details, Edit, and Index.
   - Data/ExpenseContext.cs

4. **Create the initial database schema using EF's migration feature**
   After scaffolding, run the following commands in the Package Manager Console to create and apply the initial migration:
   **`Add-Migration InitialCreate`**
   **`Update-Database`**

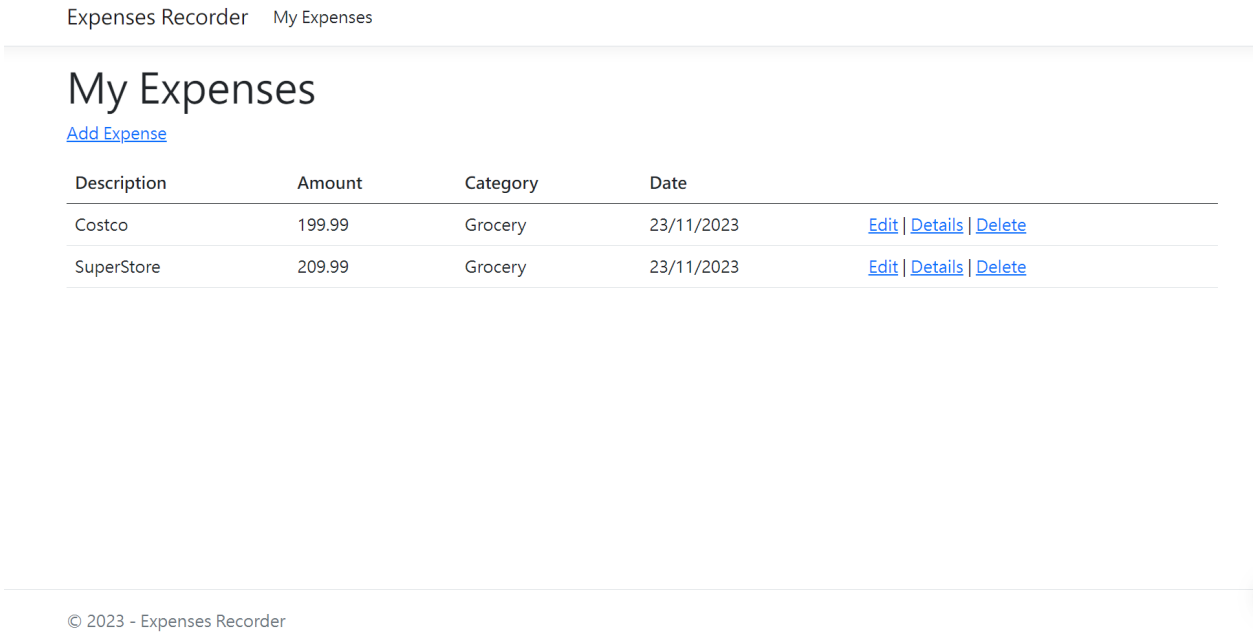
5. **Update the layout of the pages**
   Customize the layout by modifying the Razor views (e.g., Create.cshtml, Index.cshtml, Edit.cshtml, Delete.cshtml) to match desired layout.

# Test

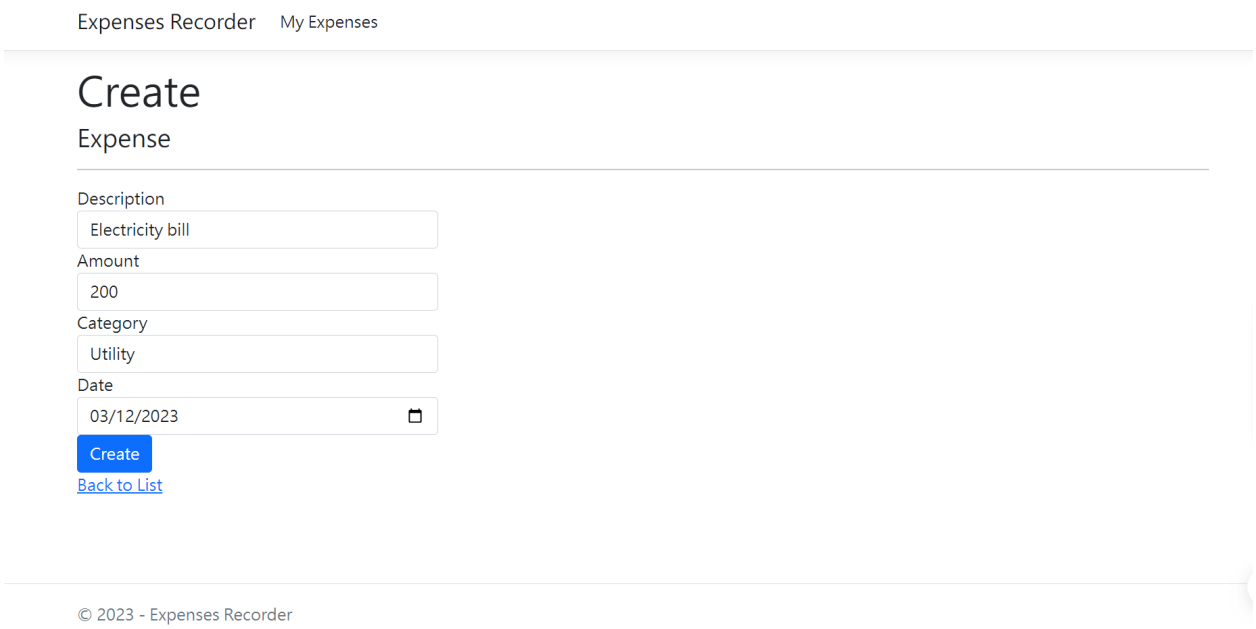
## Home Page



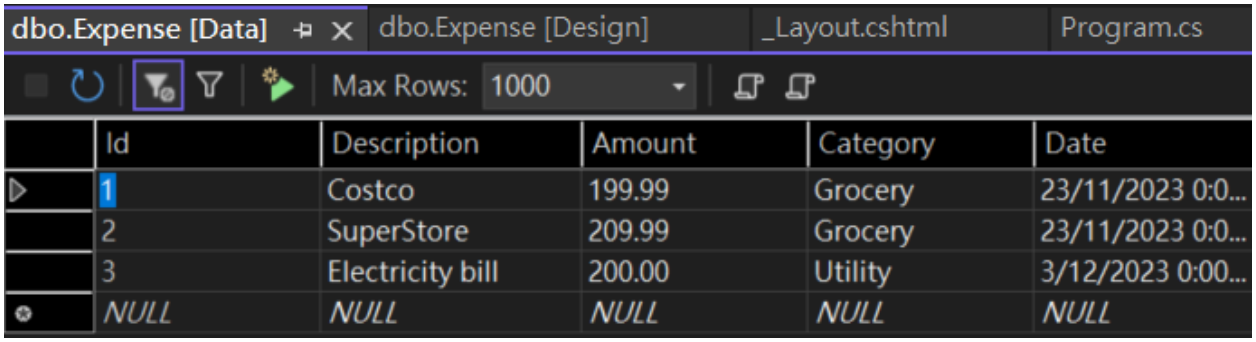

## My Expenses Page



# Add An Expense

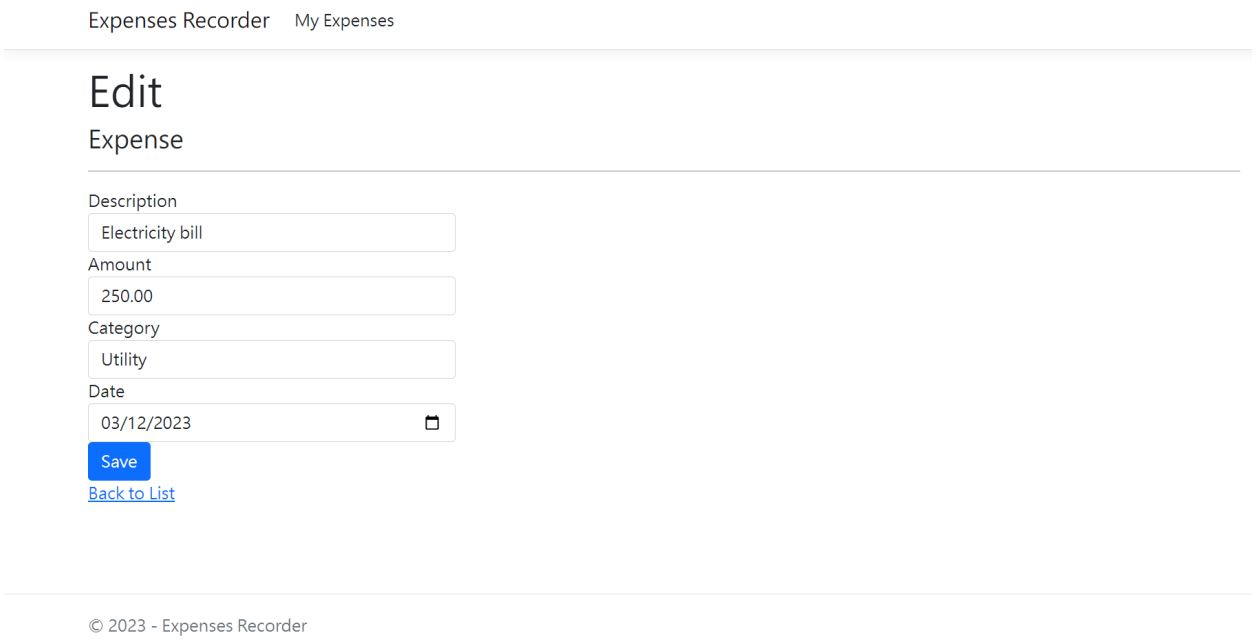



The newly created expense has been stored into the SQL server.

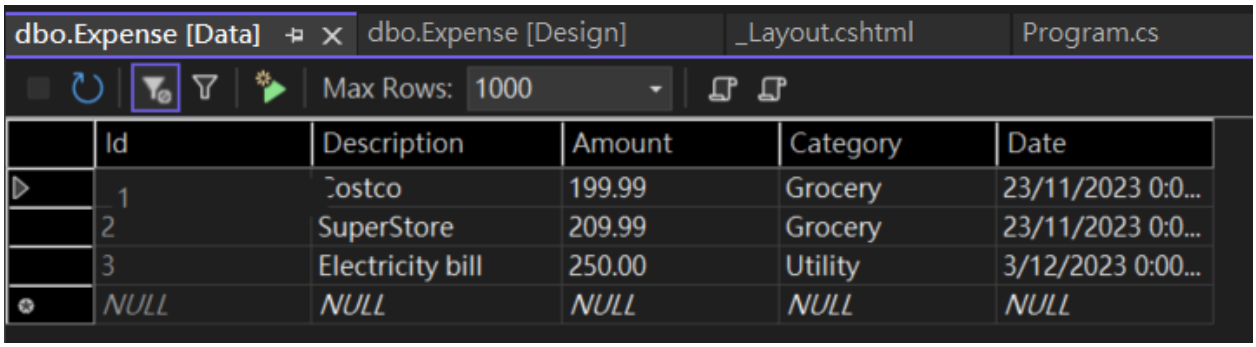



| Id | Description | Amount | Category | Date |
|---|---|---|---|---|
| 1 | Costco | 199.99 | Grocery | 23/11/2023 0:0... |
| 2 | SuperStore | 209.99 | Grocery | 23/11/2023 0:0... |
| 3 | Electricity bill | 200.00 | Utility | 3/12/2023 0:00... |
| NULL | NULL | NULL | NULL | NULL |

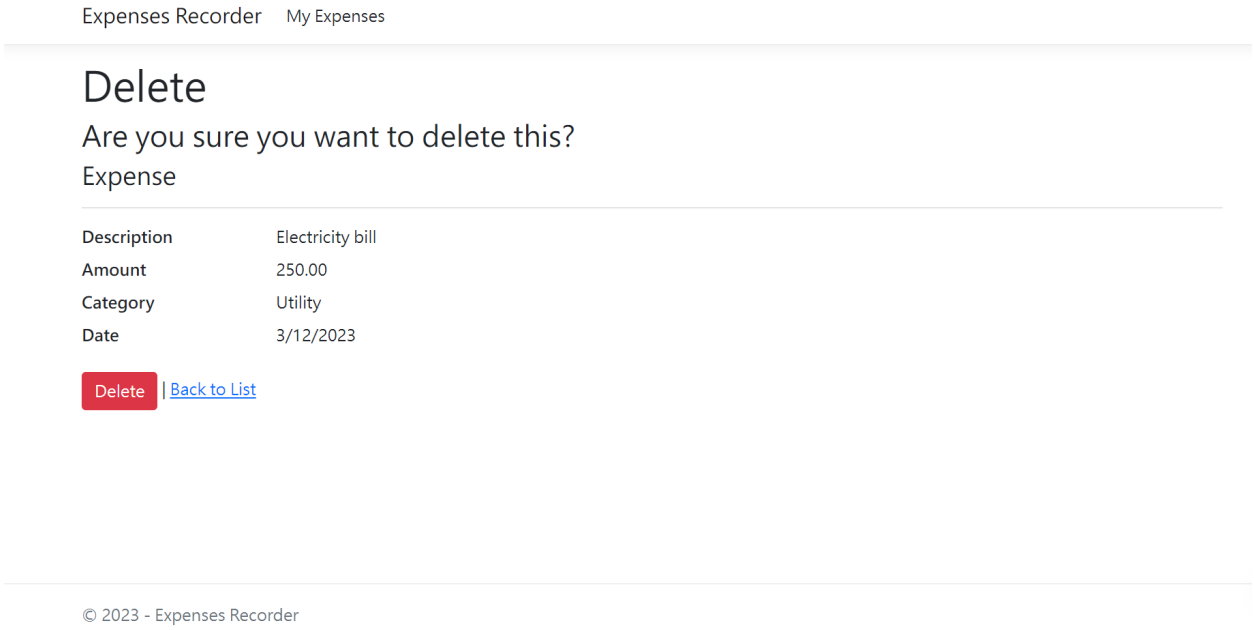
## Edit An Expense



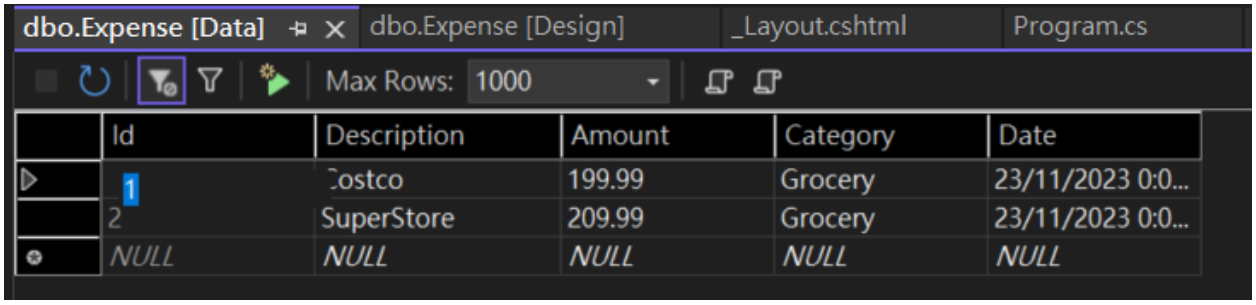

The amount of the Electricity bill has been edited to $250.



# Delete An Expense




The Electricity bill has been deleted from the database.



## Insights Learned

Working on a project to create an ASP.NET Core Web App with CRUD operations for an Expense model provides several insights and learning opportunities. Here are some key takeaways:

1. **ASP.NET Core Fundamentals:**
   - Gain a solid understanding of ASP.NET Core MVC architecture, including models, views, and controllers.
   - Learn how routing works in ASP.NET Core to map URLs to controller actions.

2. **Entity Framework (EF) Integration:**
   - Understand the integration of Entity Framework Core for database operations. Learn how to define data models and use migrations to create and update the database schema.
   - Explore how to use EF Core to perform CRUD operations on the database without writing explicit SQL queries.

3. **Scaffolding for Productivity:**
   - Experience the productivity benefits of scaffolding, which automatically generates boilerplate code for CRUD operations. This can save a significant amount of development time.

4. **Database Migrations:**
   - Learn about database migrations and how they enable versioning and evolution of the database schema over time. Understand the process of applying migrations to create or update the database.

5. **HTML and Razor Syntax:**
   - Work with HTML and Razor syntax in ASP.NET Core views. Understand how to embed C# code within HTML to dynamically generate content based on the model data.