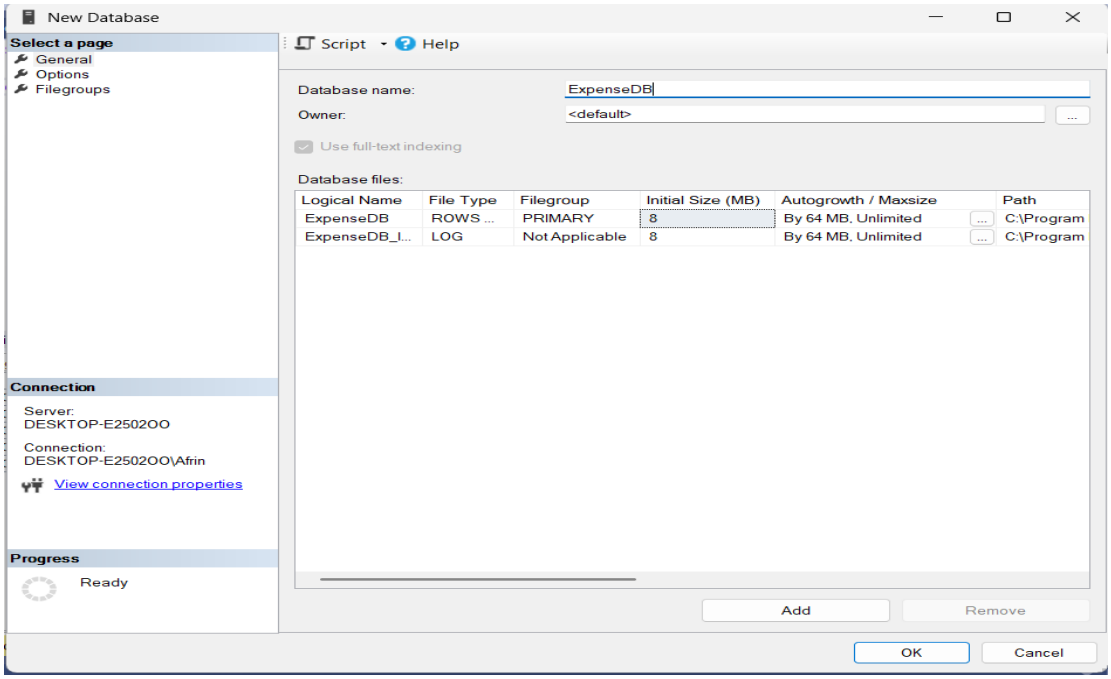
Expense Tracker Project

React & Asp.net Project

Create a database in SQL server Management Studio:



Our database name is Database Name -> ExpenseDB

Install Visual Studio

ASP.NET Core Web API Create a New ASP.NET Core Web API Project

1. Open Visual Studio and select Create a new project.

2. Choose ASP.NET Core Web API and click Next.

3. Enter the project name and location, then click Create.

4. Select the latest .NET version and ensure Use controllers is checked.

5. Click Create to generate the project structure.

Install Required

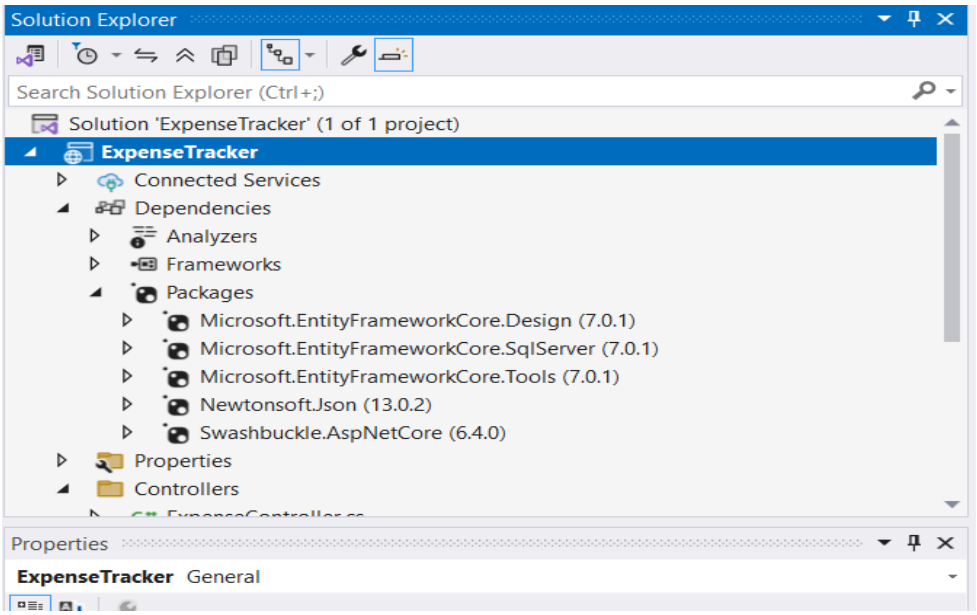
Packages Open Manage NuGet Packages by right-clicking your project Install the packages below

Microsoft.EntityFrameworkCore.Design

Microsoft.EntityFrameworkCore.SqlServer

Microsoft.EntityFrameworkCore.Tools

Newtonsoft.Json



Check the package requirements

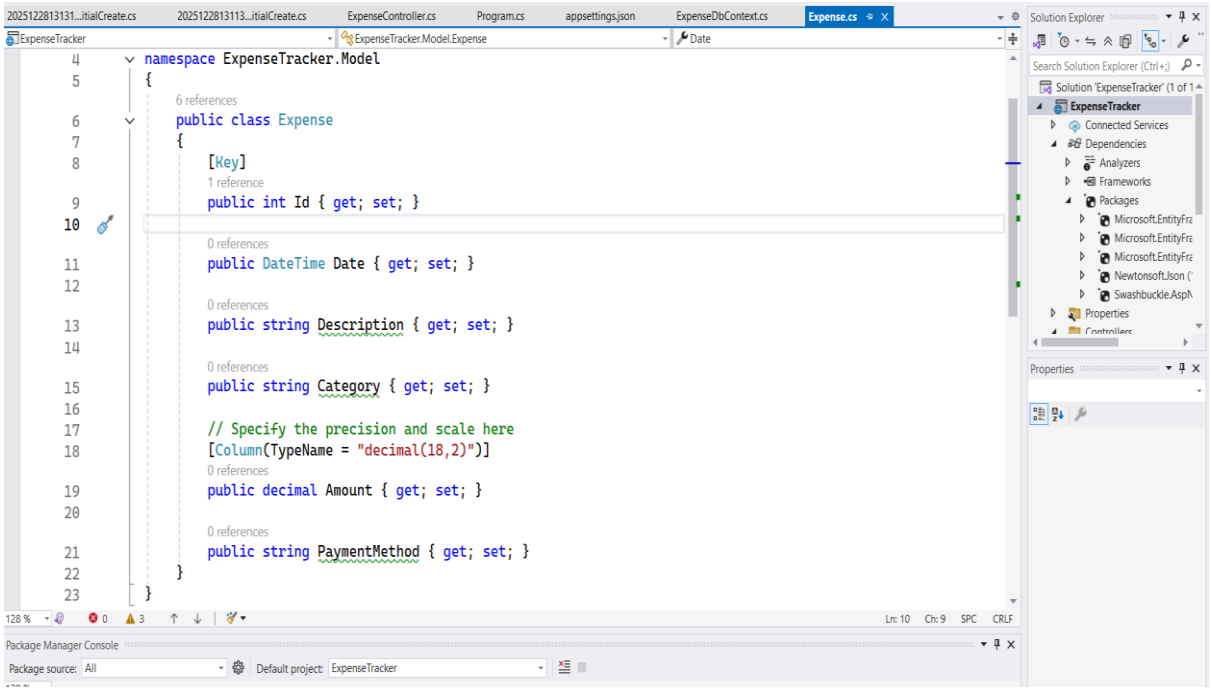
Create model folder. give name Models

First Step Select the Model Folder Right Click and Create the Class **Expense.cs**

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
namespace ExpenseTracker.Model
{
public class Expense
{
[Key]
public int Id { get; set; }
public DateTime Date { get; set; }
public string Description { get; set; }
public string Category { get; set; }
[Column(TypeName = "decimal(18,2)")]
public decimal Amount { get; set; }
public string PaymentMethod { get; set; }
```

}

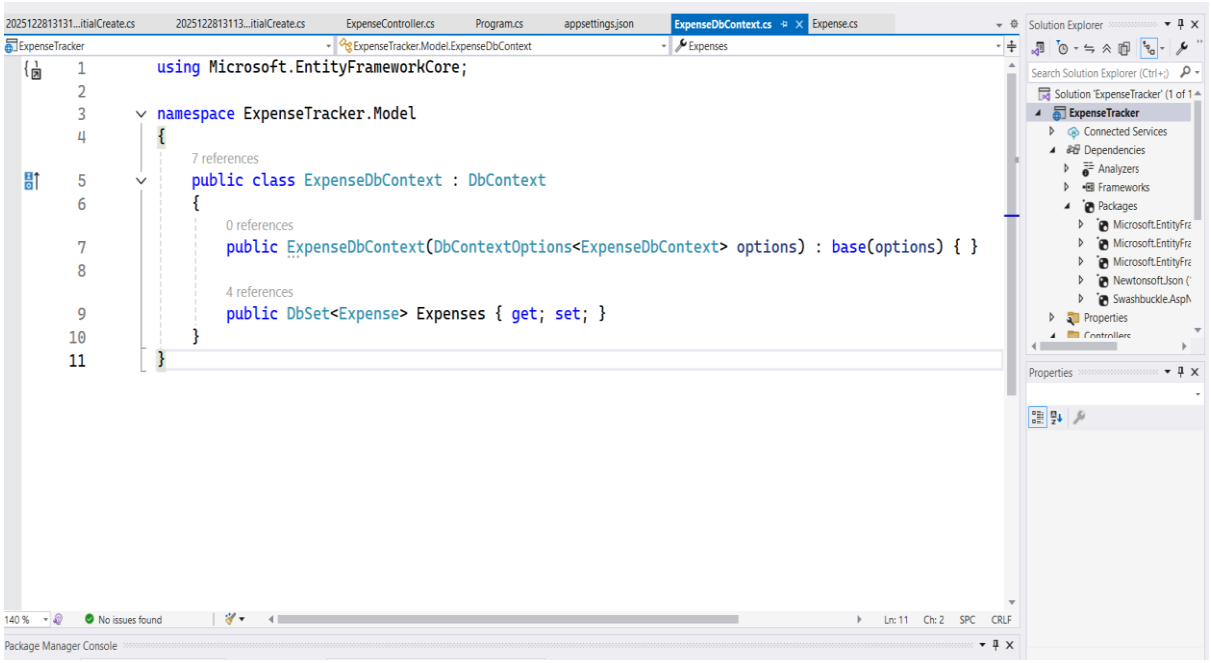
}



ExpenseDbContext Class:

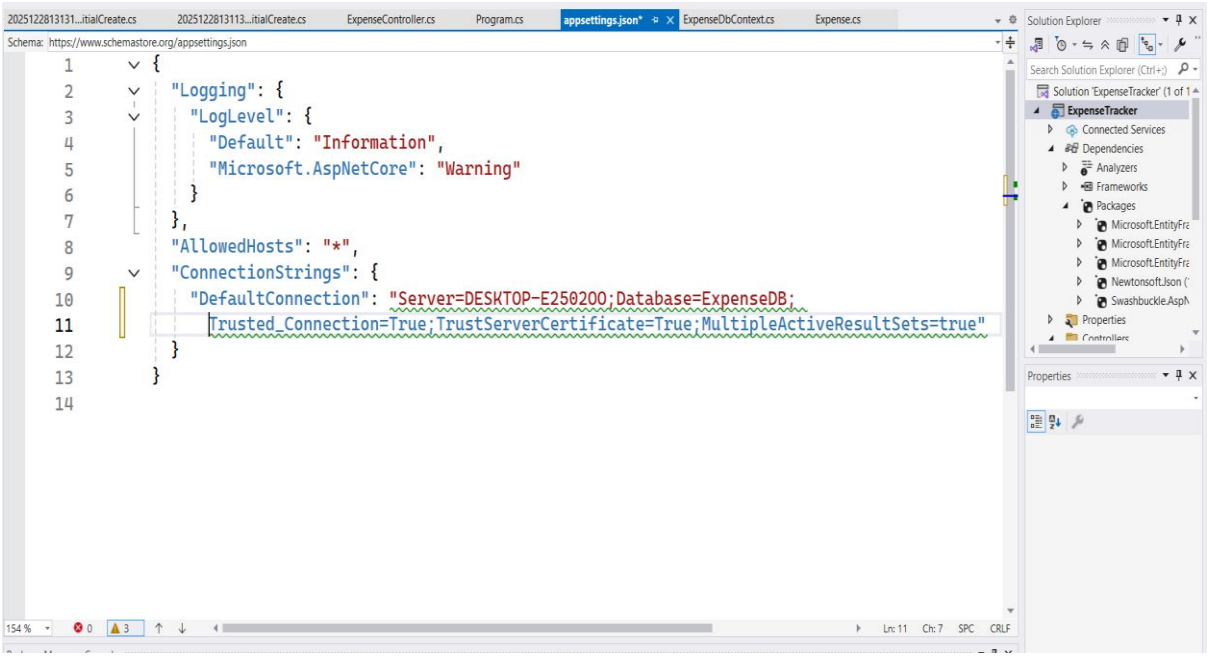
Create another model class – **ExpenseDbContext.cs**

```
using Microsoft.EntityFrameworkCore;


namespace ExpenseTracker.Model
{
   public class ExpenseDbContext : DbContext
   {
      public ExpenseDbContext(DbContextOptions<ExpenseDbContext> options) :
base(options) { }


      public DbSet<Expense> Expenses { get; set; }
   }
}
```



Configure the Database Connection – (appsettings.json)

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection":
"Server=DESKTOPE2502OO;Database=ExpenseDB;Trusted_Connection=True;
TrustServerCertificate=True;MultipleActiveResultSets=true"
  }
}
```



Program.cs

Establish the Database Connection

Add these Context inside the Program.cs file

```
// Database Connection
builder.Services.AddDbContext<ExpenseDbContext>(options =>
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

// Allow React to talk to this API
app.UseCors(policy => policy.AllowAnyHeader()
                    .AllowAnyMethod()
                    .SetIsOriginAllowed(origin => true)
                    .AllowCredentials());
```

Full code structure of program.cs -

```
using Microsoft.EntityFrameworkCore;
using ExpenseTracker.Model;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

// Database Connection
builder.Services.AddDbContext<ExpenseDbContext>(options =>
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

var app = builder.Build();

// Allow React to talk to this API
app.UseCors(policy => policy.AllowAnyHeader()
                        .AllowAnyMethod()
                        .SetIsOriginAllowed(origin => true)
                        .AllowCredentials());

if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.UseHttpsRedirection();
app.UseAuthorization();
app.MapControllers();
app.Run();
```

```
using Microsoft.EntityFrameworkCore;
using ExpenseTracker.Model;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllers();
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

// Database Connection
builder.Services.AddDbContext<ExpenseDbContext>(options =>
    options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));

var app = builder.Build();

// Allow React to talk to this API
app.UseCors(policy => policy.AllowAnyHeader()
                            .AllowAnyMethod()
                            .SetIsOriginAllowed(origin => true)
                            .AllowCredentials());

if (app.Environment.IsDevelopment())
{
    app.UseSwagger();
    app.UseSwaggerUI();
}

app.UseHttpsRedirection();
app.UseAuthorization();
app.MapControllers();
app.Run();
```

Add Controller

Rightclick

Controller -> Add-> new scaffold item -> mvc empty controller-> **ExpenseController.cs**

ExpenseController.cs

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using ExpenseTracker.Model;


namespace ExpenseTracker.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    public class ExpenseController : ControllerBase
```

```
{
    private readonly ExpenseDbContext _context;

    public ExpenseController(ExpenseDbContext context)
    {
        _context = context;
    }

    [HttpGet]
    [Route("GetExpenses")]
    public async Task<IEnumerable<Expense>> GetExpenses()
    {
        return await _context.Expenses.ToListAsync();
    }
    [HttpPost]
    [Route("AddExpense")]
    public async Task<Expense> AddExpense(Expense objExpense)
    {
        _context.Expenses.Add(objExpense);
        await _context.SaveChangesAsync();
        return objExpense;
    }

    [HttpPatch]
    [Route("UpdateExpense/{id}")]
    public async Task<Expense> UpdateExpense([FromRoute] int id, [FromBody] Expense objExpense)
    {
        // Ensure the ID in the object matches the ID in the URL
        objExpense.Id = id;
        _context.Entry(objExpense).State = EntityState.Modified;
        await _context.SaveChangesAsync();
        return objExpense;
    }
```

```
        [HttpDelete]
        [Route("DeleteExpense/{id}")]
        public async Task<bool> DeleteExpense(int id)
        {
            var expense = await _context.Expenses.FindAsync(id);
            if (expense != null)
            {
                _context.Expenses.Remove(expense);
                await _context.SaveChangesAsync();
                return true;
            }
            return false;
        }
    }
}
```

Below the code output:

```
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using ExpenseTracker.Model;

namespace ExpenseTracker.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    1 reference
    public class ExpenseController : ControllerBase
    {
        private readonly ExpenseDbContext _context;

        0 references
        public ExpenseController(ExpenseDbContext context)
        {
            _context = context;
        }

        [HttpGet]
        [Route("GetExpenses")]
        0 references
        public async Task<IEnumerable<Expense>> GetExpenses()
        {
            return await _context.Expenses.ToListAsync();
        }
```

```
[HttpPost]
[Route("AddExpense")]
0 references
public async Task<Expense> AddExpense(Expense objExpense)
{
    _context.Expenses.Add(objExpense);
    await _context.SaveChangesAsync();
    return objExpense;
}

[HttpPatch]
[Route("UpdateExpense/{id}")]
0 references
public async Task<Expense> UpdateExpense([FromRoute] int id, [FromBody] Expense objExpense)
{
    // Ensure the ID in the object matches the ID in the URL
    objExpense.Id = id;
    _context.Entry(objExpense).State = EntityState.Modified;
    await _context.SaveChangesAsync();
    return objExpense;
}
```

```
        [HttpDelete]
        [Route("DeleteExpense/{id}")]
        0 references
        public async Task<bool> DeleteExpense(int id)
        {
            var expense = await _context.Expenses.FindAsync(id);
            if (expense != null)
            {
                _context.Expenses.Remove(expense);
                await _context.SaveChangesAsync();
                return true;
            }
            return false;
        }
    }
}
```

Apply Migrations and Create the Database

Tools->NuGet Package Manager->Package Manager Console

Run command

Run these commands in the Package Manager Console –

Add-Migration InitialCreate

Update-Database

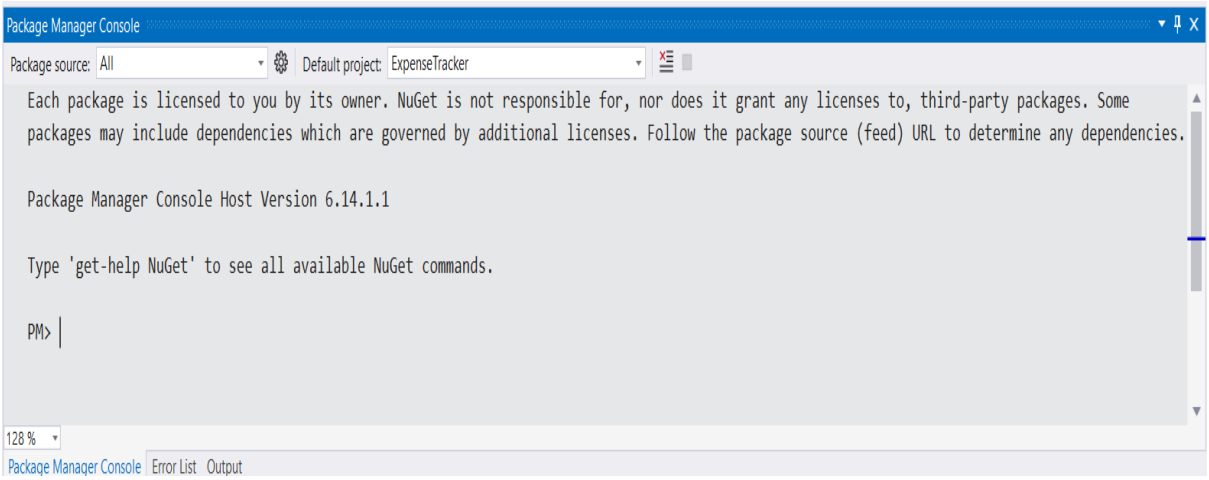
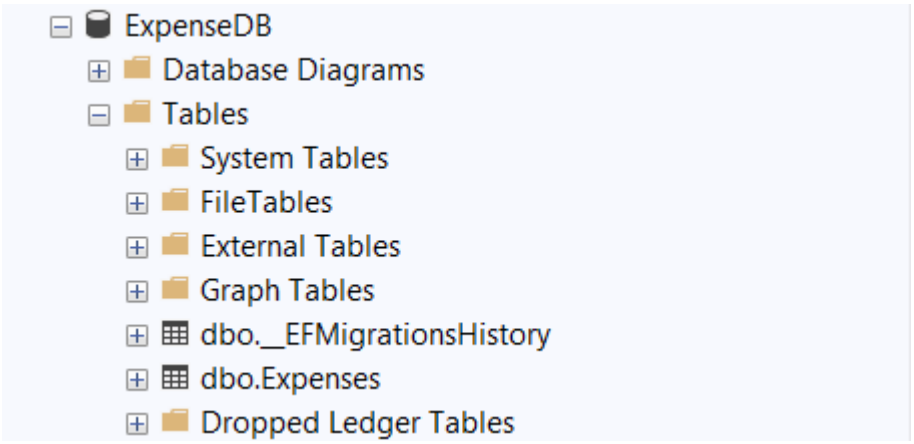
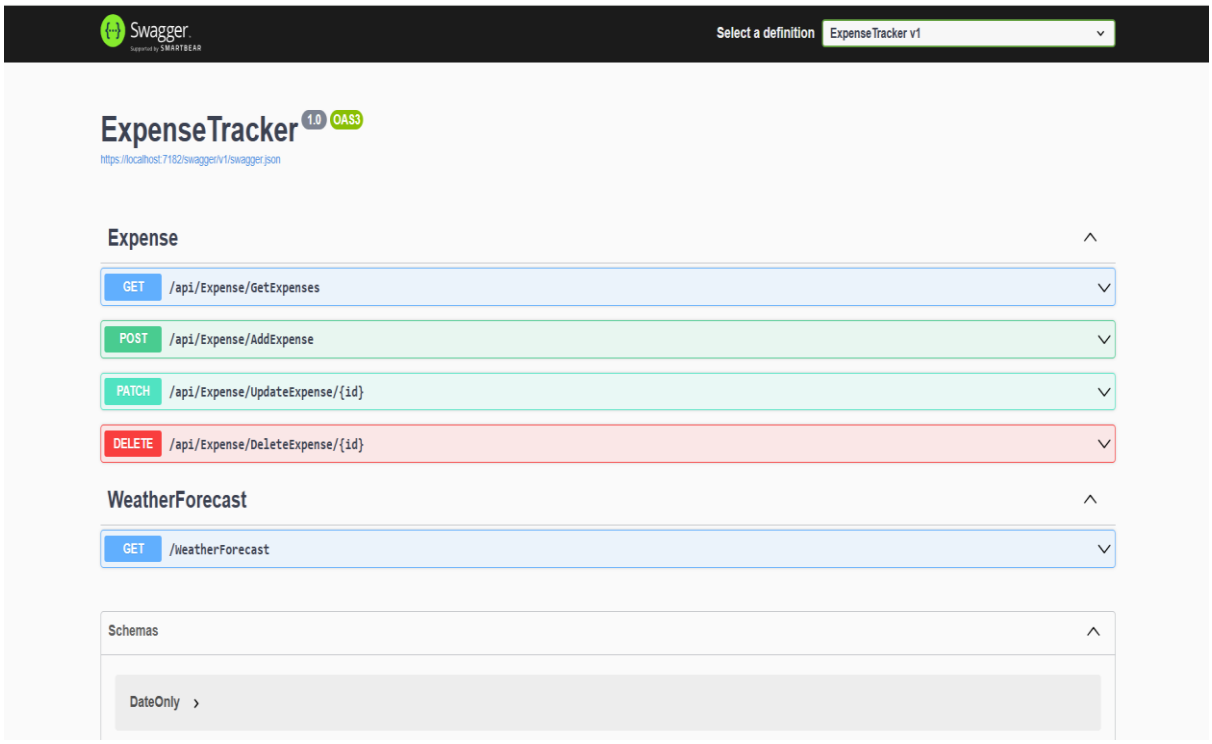


Table will be created in SQL our database inside table –



RUN THE SERVER -

API CONNECTION WILL OPEN.



React Frontend –

Open vs code terminal creates the react application

```
npx create-react-app frontend
```

```
npm start
```

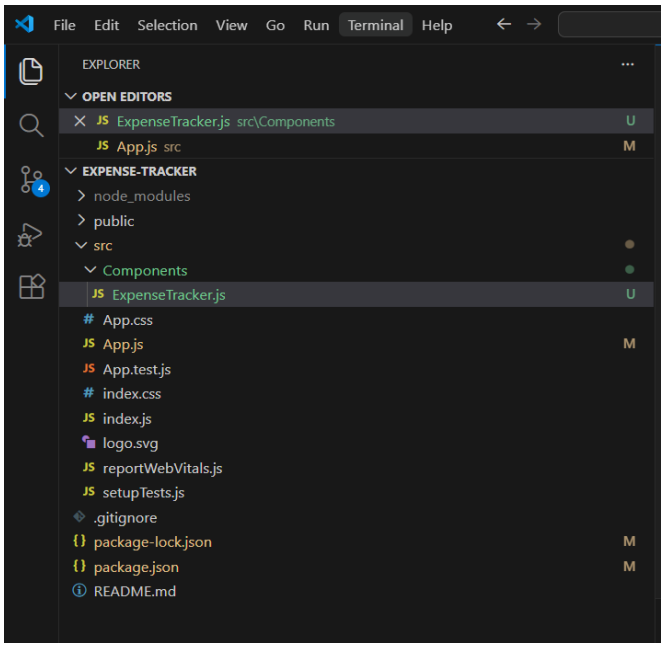
Install the bootstrap command

```
npm i bootstrap (or) npm list bootsrap
```

Fetch The Api/Backend Connection Install the Command

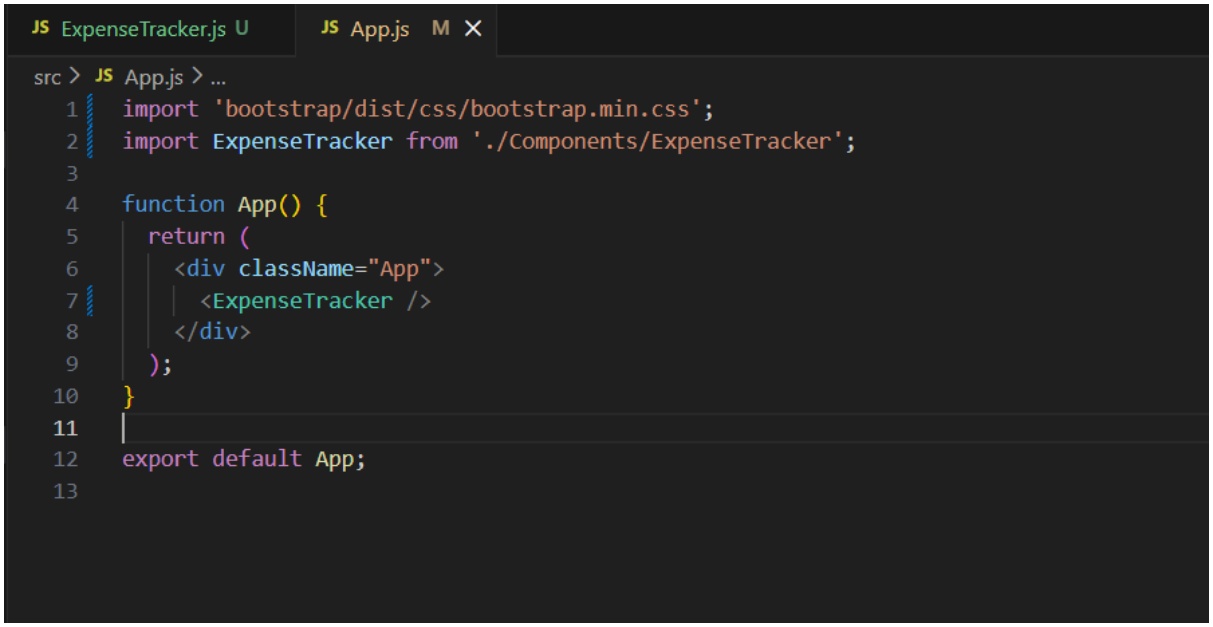
```
npm i axios
```

Open a New Component folder -> give **ExpenseTracker.js** file



App.js

```
import 'bootstrap/dist/css/bootstrap.min.css';
import ExpenseTracker from './Components/ExpenseTracker';

function App() {
  return (
    <div className="App">
      <ExpenseTracker />
    </div>
  );
}

export default App;
```



ExpenseTracker.js file

Import Statements –

```
import axios from "axios";
import { useEffect, useState } from "react";
```

State Variables –

```
function ExpenseTracker() {
  const [id, setId] = useState("");
  const [date, setDate] = useState("");
  const [description, setDescription] = useState("");
  const [category, setCategory] = useState("");
  const [amount, setAmount] = useState("");
  const [paymentMethod, setPaymentMethod] = useState("");
  const [expenses, setExpenses] = useState([]);
```

Component Load –

```
useEffect(() => {
   (async () => await Load())();
 }, []);
```

Read Operation –

```
async function Load() {
   const result = await axios.get("https://localhost:7182/api/Expense/GetExpenses");
   setExpenses(result.data);
 }
```

Create Operation –

```
async function save(event) {
   event.preventDefault();
   if (!date || !amount) return alert("Please fill in Date and Amount");
   await axios.post("https://localhost:7182/api/Expense/AddExpense", {
     date, description, category, amount, paymentMethod
   });
   alert("Expense Saved");
   clear();
   Load();
 }
```

Update Operation –

```
async function update(event) {
    event.preventDefault();
    await axios.patch("https://localhost:7182/api/Expense/UpdateExpense/" + id, {
      id, date, description, category, amount, paymentMethod
    });
    alert("Expense Updated");
    clear();
    Load();
  }
```

Delete Operation –

```
 async function deleteExp(id) {
    if (window.confirm("Are you sure you want to delete this record?")) {
      await axios.delete("https://localhost:7182/api/Expense/DeleteExpense/" + id);
      Load();
    }
  }
```

Edit Operation –

```
function edit(exp) {
    setId(exp.id);
    setDate(exp.date.split('T')[0]);
    setDescription(exp.description);
    setCategory(exp.category);
    setAmount(exp.amount);
    setPaymentMethod(exp.paymentMethod);
  }
```

Clear Operation –

```
function clear() {
   setId(""); setDate(""); setDescription(""); setCategory(""); setAmount("");
setPaymentMethod("");
  }
```

Calculating Total Expenses –

```
const totalAmount = expenses.reduce((sum, item) => sum + parseFloat(item.amount || 0), 0);
```

UI Design code –

```
return (
   <div style={{
      backgroundColor: "#f8f9fa",
      minHeight: "100vh",
      padding: "40px 0",
      color: "#343a40"
   }}>
     <div className="container">
      {/* Header section */}
      <div className="mb-5">
        <h2 style={{ fontWeight: "700", color: "#2c3e50", borderLeft: "5px solid #4e73df",
paddingLeft: "15px" }}>
          Expense Tracker
        </h2>
        <p className="text-muted">Tracking my expenses and manage my finances.</p>
      </div>
```

```
<div className="row g-4">
  {/* Sidebar Form */}
  <div className="col-lg-4">
    <div className="card border-0 shadow-sm p-4" style={{ borderRadius: "12px" }}>
      <h5 className="mb-4" style={{ fontWeight: "600" }}>{id ? "Update Transaction" : "New Transaction"}</h5>
      <form>
        <div className="mb-3">
          <label className="small fw-bold text-muted mb-1">DATE</label>
          <input type="date" className="form-control bg-light border-0" value={date} onChange={(e)=>setDate(e.target.value)} />
        </div>
        <div className="mb-3">
          <label className="small fw-bold text-muted mb-1">CATEGORY</label>
          <select className="form-select bg-light border-0" value={category} onChange={(e)=>setCategory(e.target.value)}>
            <option value="">Select Category</option>
            <option value="Food&Groceries">Food & Groceries</option>
            <option value="Rent">Rent</option>
            <option value="Utilities">Utilities</option>
            <option value="Health">Health</option>
            <option value="Education">Education & Fees</option>
            <option value="Transportation">Transportation</option>
            <option value="Maintenance">Maintenance & Repairs</option>
            <option value="Entertainment">Entertainment</option>
            <option value="Shopping">Shopping & Lifestyle</option>
            <option value="EMI & Loans">EMI & Loans</option>
            <option value="Investments">Investments</option>
            <option value="Others">Other</option>

          </select>
```

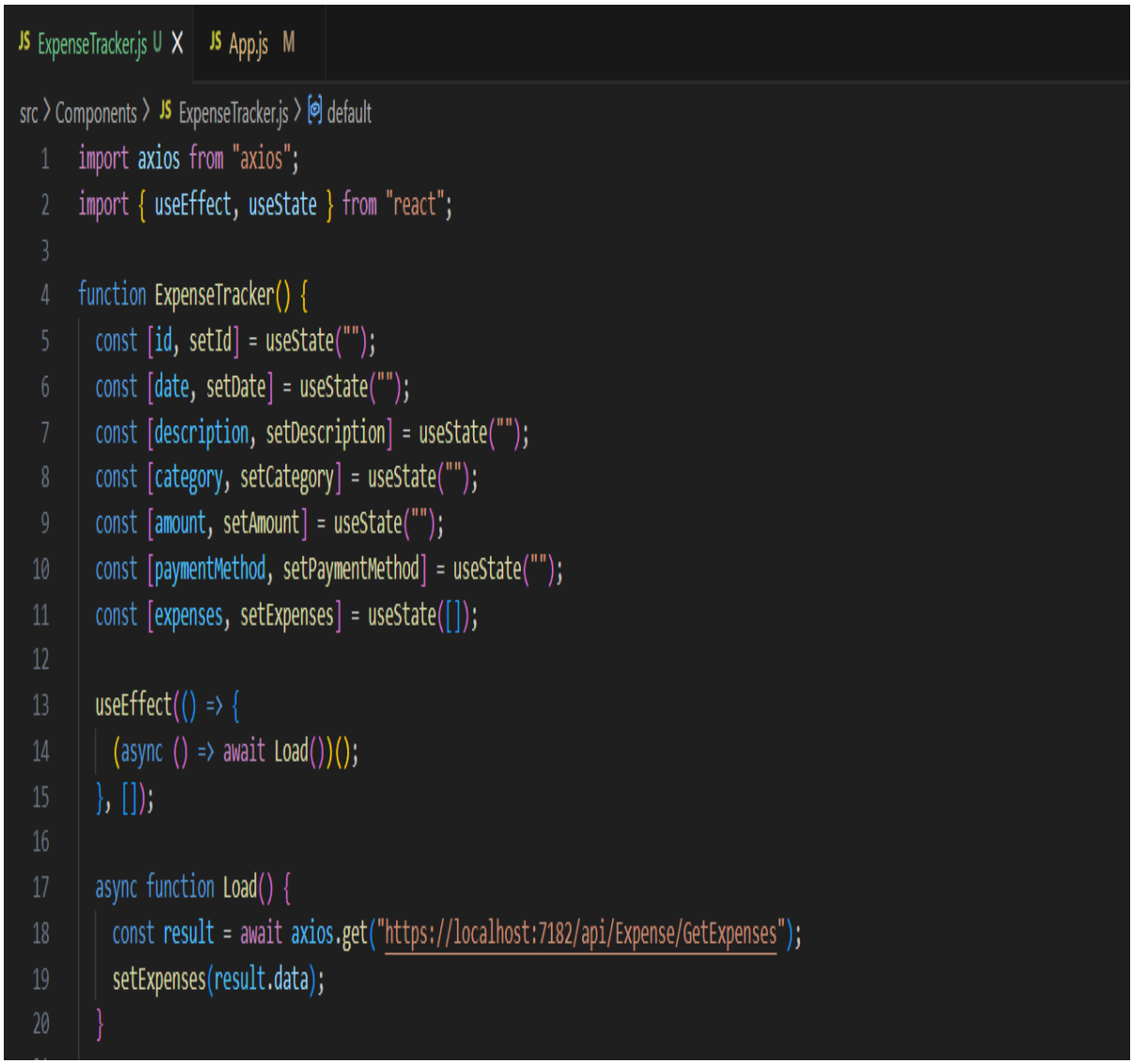
```
            </div>
            <div className="mb-3">
              <label className="small fw-bold text-muted mb-1">AMOUNT (INR)</label>
              <input type="number" className="form-control bg-light border-0"
placeholder="0.00" value={amount} onChange={(e)=>setAmount(e.target.value)} />
            </div>
            <div className="mb-3">
              <label className="small fw-bold text-muted mb-1">DESCRIPTION</label>
              <textarea className="form-control bg-light border-0" rows="2"
value={description} onChange={(e)=>setDescription(e.target.value)}></textarea>
            </div>
            <div className="mb-4">
              <label className="small fw-bold text-muted mb-1">PAYMENT
METHOD</label>
              <input type="text" className="form-control bg-light border-0"
placeholder="Bank Transfer / Card" value={paymentMethod}
onChange={(e)=>setPaymentMethod(e.target.value)} />
            </div>

            <div className="d-grid gap-2">
               {!id ? (
                  <button className="btn btn-primary shadow-sm" onClick={save} style={{
backgroundColor: "#4e73df", border: "none", padding: "12px" }}>Add Entry</button>
               ) : (
                  <button className="btn btn-dark shadow-sm" onClick={update} style={{
padding: "12px" }}>Update Entry</button>
               )}
              <button type="button" className="btn btn-outline-secondary btn-sm border-0"
onClick={clear}>Reset Form</button>
            </div>
          </form>
        </div>
      </div>
```

```
        {/* Table Area */}
        <div className="col-lg-8">
          {/* Summary Banner */}
          <div className="card border-0 shadow-sm p-4 mb-4" style={{ backgroundColor:
"#ffffff", borderRadius: "12px" }}>
            <div className="row align-items-center">
              <div className="col border-end">
                <span className="small text-muted text-uppercase fw-bold">Total
Expenditures</span>
                <h3 className="mb-0" style={{ color: "#e74c3c", fontWeight: "700"
}}>${totalAmount.toLocaleString(undefined, {minimumFractionDigits: 2})}</h3>
              </div>
              <div className="col ps-4">
                <span className="small text-muted text-uppercase fw-bold">Active
Records</span>
                <h3 className="mb-0" style={{ color: "#2c3e50", fontWeight: "700"
}}>{expenses.length}</h3>
              </div>
            </div>
          </div>

          {/* Data Table */}
          <div className="card border-0 shadow-sm" style={{ borderRadius: "12px",
overflow: "hidden" }}>
            <div className="table-responsive">
              <table className="table table-hover mb-0">
                <thead style={{ backgroundColor: "#f1f4f8" }}>
                  <tr>
                    <th className="p-3 small text-muted">TRANSACTION</th>
                    <th className="p-3 small text-muted">CATEGORY</th>
                    <th className="p-3 small text-muted">AMOUNT</th>
```

```
                <th className="p-3 small text-muted text-end">CONTROLS</th>
              </tr>
            </thead>
            <tbody className="bg-white">
              {expenses.map((exp) => (
                <tr key={exp.id}>
                  <td className="p-3">
                    <div className="fw-bold">{exp.description || "Unspecified"}</div>
                    <div className="text-muted small">{new
Date(exp.date).toLocaleDateString()} | {exp.paymentMethod}</div>
                  </td>
                  <td className="p-3">
                    <span className="badge bg-light text-secondary border px-3 py-2 fw-
normal">
                      {exp.category}
                    </span>
                  </td>
                  <td className="p-3 fw-bold">
                    ${parseFloat(exp.amount).toFixed(2)}
                  </td>
                  <td className="p-3 text-end">
                    <button className="btn btn-sm btn-outline-primary me-2 border-0"
onClick={() => edit(exp)}>Edit</button>
                    <button className="btn btn-sm btn-outline-danger border-0" onClick={()
=> deleteExp(exp.id)}>Delete</button>
                  </td>
                </tr>
              ))}
            </tbody>
          </table>
        </div>
      </div>
```

```
        </div>
      </div>
    </div>
  </div>
 );
}

export default ExpenseTracker;
```

In VS Code -



```
async function save(event) {
  event.preventDefault();
  if (!date || !amount) return alert("Please fill in Date and Amount");
  await axios.post("https://localhost:7182/api/Expense/AddExpense", {
    date, description, category, amount, paymentMethod
  });
  alert("Expense Saved");
  clear();
  Load();
}

async function update(event) {
  event.preventDefault();
  await axios.patch("https://localhost:7182/api/Expense/UpdateExpense/" + id, {
    id, date, description, category, amount, paymentMethod
  });
  alert("Expense Updated");
  clear();
  Load();
}

async function deleteExp(id) {
  if (window.confirm("Are you sure you want to delete this record?")) {
    await axios.delete("https://localhost:7182/api/Expense/DeleteExpense/" + id);
    Load();
  }
}
```

```
function edit(exp) {
  setId(exp.id);
  setDate(exp.date.split('T')[0]);
  setDescription(exp.description);
  setCategory(exp.category);
  setAmount(exp.amount);
  setPaymentMethod(exp.paymentMethod);
}

function clear() {
  setId(""); setDate(""); setDescription(""); setCategory(""); setAmount(""); setPaymentMethod("");
}

const totalAmount = expenses.reduce((sum, item) => sum + parseFloat(item.amount || 0), 0);
```

```
return (
  <div style={{
      backgroundColor: "#f8f9fa",
      minHeight: "100vh",
      padding: "40px 0",
      color: "#343a40"
  }}>
    <div className="container">
      {/* Header section */}
      <div className="mb-5">
        <h2 style={{ fontWeight: "700", color: "#2c3e50", borderLeft: "5px solid #4e73df", paddingLeft: "15px" }}>
          Expense Tracker
        </h2>
        <p className="text-muted">Tracking my expenses and manage my finances.</p>
      </div>

      <div className="row g-4">
        {/* Sidebar Form */}
        <div className="col-lg-4">
          <div className="card border-0 shadow-sm p-4" style={{ borderRadius: "12px" }}>
            <h5 className="mb-4" style={{ fontWeight: "600" }}>{id ? "Update Transaction" : "New Transaction"}</h5>
            <form>
              <div className="mb-3">
                <label className="small fw-bold text-muted mb-1">DATE</label>
                <input type="date" className="form-control bg-light border-0" value={date} onChange={(e)=>setDate(e.target.value)} />
              </div>
              <div className="mb-3">
                <label className="small fw-bold text-muted mb-1">CATEGORY</label>
```

```
<select className="form-select bg-light border-0" value={category} onChange={(e)=>setCategory(e.target.value)}>
  <option value="">Select Category</option>
  <option value="Food&Groceries">Food & Groceries</option>
  <option value="Rent">Rent</option>
  <option value="Utilities">Utilities</option>
  <option value="Health">Health</option>
  <option value="Education">Education & Fees</option>
  <option value="Transportation">Transportation</option>
  <option value="Maintenance">Maintenance & Repairs</option>
  <option value="Entertainment">Entertainment</option>
  <option value="Shopping">Shopping & Lifestyle</option>
  <option value="EMI & Loans">EMI & Loans</option>
  <option value="Investments">Investments</option>
  <option value="Others">Other</option>

</select>
</div>
<div className="mb-3">
<label className="small fw-bold text-muted mb-1">AMOUNT (INR)</label>
<input type="number" className="form-control bg-light border-0" placeholder="0.00" value={amount} onChange={(e)=>setAmount(e.target.value)} />
</div>
<div className="mb-3">
<label className="small fw-bold text-muted mb-1">DESCRIPTION</label>
<textarea className="form-control bg-light border-0" rows="2" value={description} onChange={(e)=>setDescription(e.target.value)}></textarea>
</div>
<div className="mb-4">
<label className="small fw-bold text-muted mb-1">PAYMENT METHOD</label>
<input type="text" className="form-control bg-light border-0" placeholder="Bank Transfer / Card" value={paymentMethod} onChange={(e)=>setPaymentMethod(e.target.value)}
</div>
```

```
        <div className="d-grid gap-2">
            {!id ? (
                <button className="btn btn-primary shadow-sm" onClick={save} style={{ backgroundColor: "#4e73df", border: "none", padding: "12px" }}>Add Entry</button>
            ) : (
                <button className="btn btn-dark shadow-sm" onClick={update} style={{ padding: "12px" }}>Update Entry</button>
            )}
            <button type="button" className="btn btn-outline-secondary btn-sm border-0" onClick={clear}>Reset Form</button>
        </div>
      </form>
    </div>
  </div>

  {/* Table Area */}
  <div className="col-lg-8">
    {/* Summary Banner */}
    <div className="card border-0 shadow-sm p-4 mb-4" style={{ backgroundColor: "#ffffff", borderRadius: "12px" }}>
        <div className="row align-items-center">
            <div className="col border-end">
                <span className="small text-muted text-uppercase fw-bold">Total Expenditures</span>
                <h3 className="mb-0" style={{ color: "#e74c3c", fontWeight: "700" }}>${totalAmount.toLocaleString(undefined, {minimumFractionDigits: 2})}</h3>
            </div>
            <div className="col ps-4">
                <span className="small text-muted text-uppercase fw-bold">Active Records</span>
                <h3 className="mb-0" style={{ color: "#2c3e50", fontWeight: "700" }}>{expenses.length}</h3>
            </div>
        </div>
    </div>
```

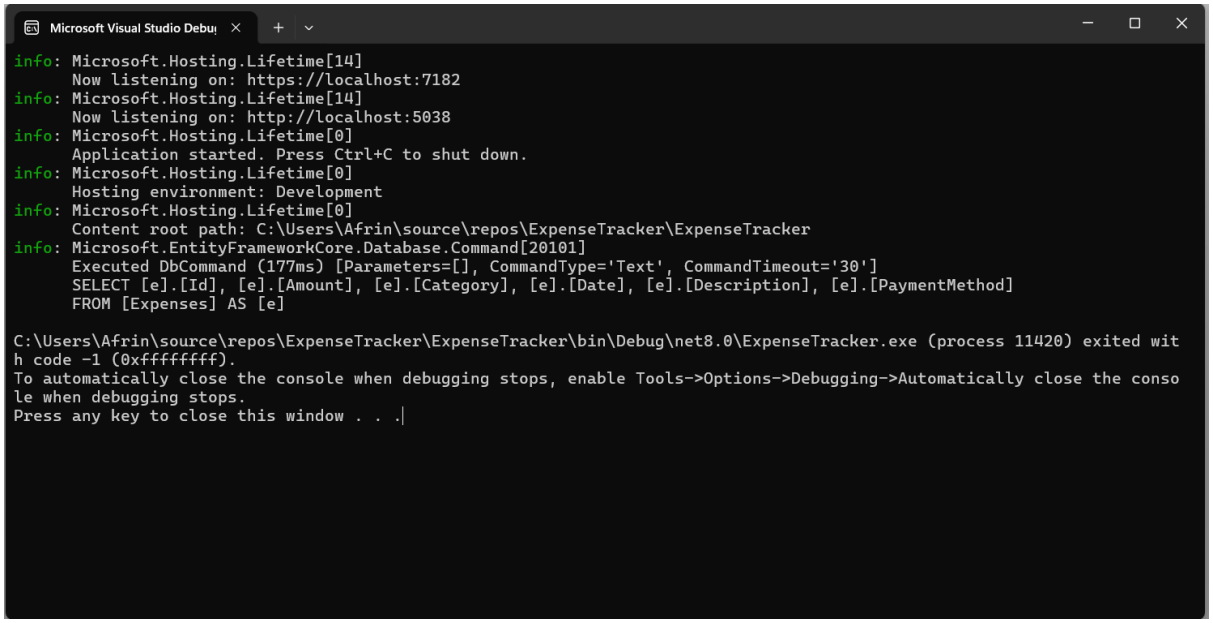
```jsx
{/* Data Table */}
<div className="card border-0 shadow-sm" style={{ borderRadius: "12px", overflow: "hidden" }}>
  <div className="table-responsive">
    <table className="table table-hover mb-0">
      <thead style={{ backgroundColor: "#f1f4f8" }}>
        <tr>
          <th className="p-3 small text-muted">TRANSACTION</th>
          <th className="p-3 small text-muted">CATEGORY</th>
          <th className="p-3 small text-muted">AMOUNT</th>
          <th className="p-3 small text-muted text-end">CONTROLS</th>
        </tr>
      </thead>
      <tbody className="bg-white">
        {expenses.map((exp) => (
          <tr key={exp.id}>
            <td className="p-3">
              <div className="fw-bold">{exp.description || "Unspecified"}</div>
              <div className="text-muted small">{new Date(exp.date).toLocaleDateString()} | {exp.paymentMethod}</div>
            </td>
            <td className="p-3">
              <span className="badge bg-light text-secondary border px-3 py-2 fw-normal">
                {exp.category}
              </span>
            </td>
            <td className="p-3 fw-bold">
              ${parseFloat(exp.amount).toFixed(2)}
            </td>
            <td className="p-3 text-end">
              <button className="btn btn-sm btn-outline-primary me-2 border-0" onClick={() => edit(exp)}>Edit</button>
              <button className="btn btn-sm btn-outline-danger border-0" onClick={() => deleteExp(exp.id)}>Delete</button>
            </td>
          </tr>
        ))}
      </tbody>
    </table>
  </div>
```

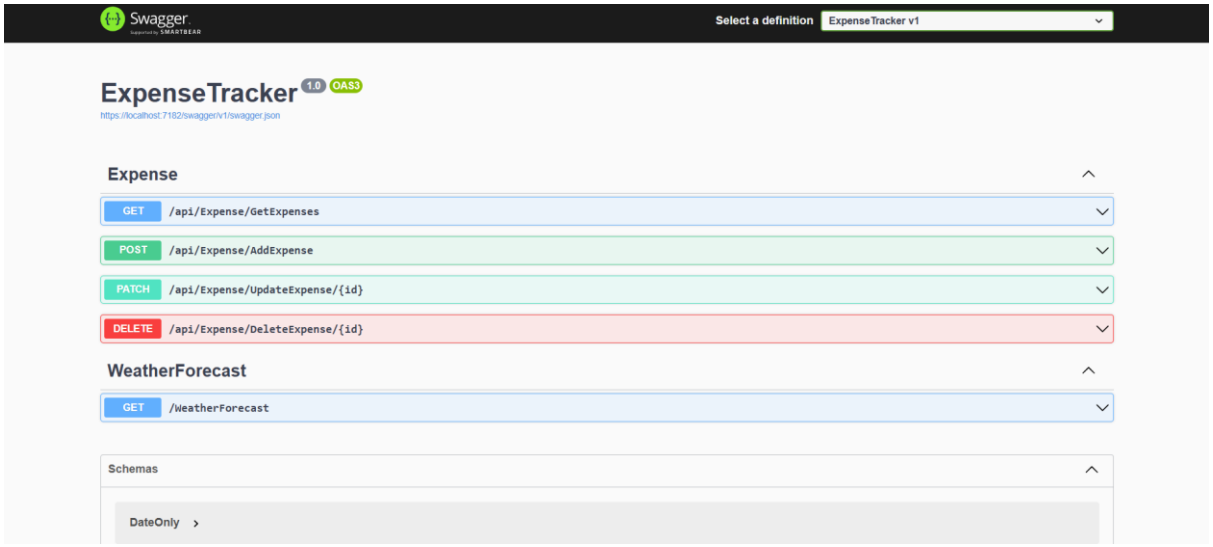
```jsx
            </div>
          </div>
        </div>
      </div>
    </div>
  );
}

export default ExpenseTracker;
```

Output –

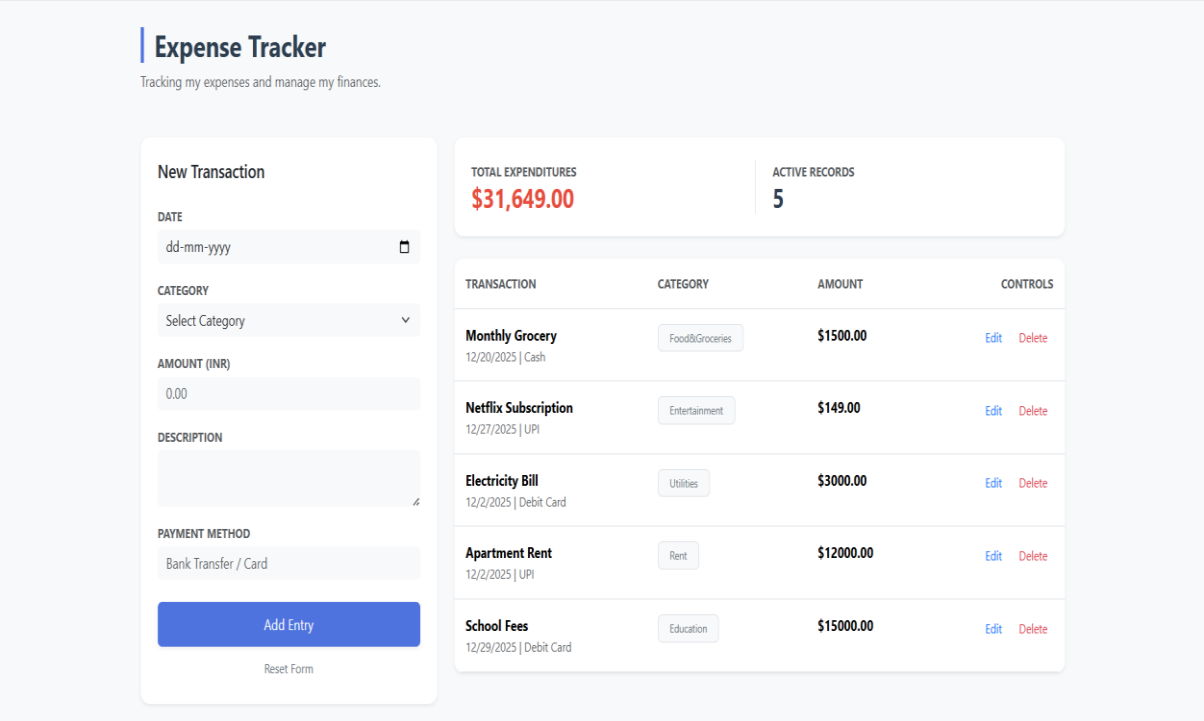
```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Compiled successfully!

You can now view expense-tracker in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.8.103:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

Backend server running :

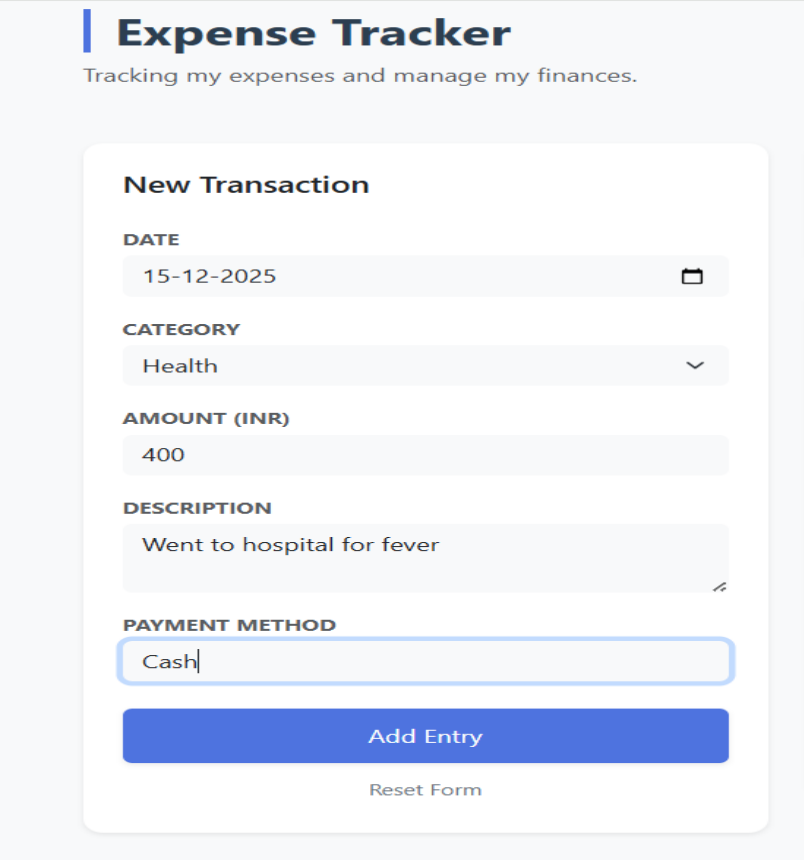




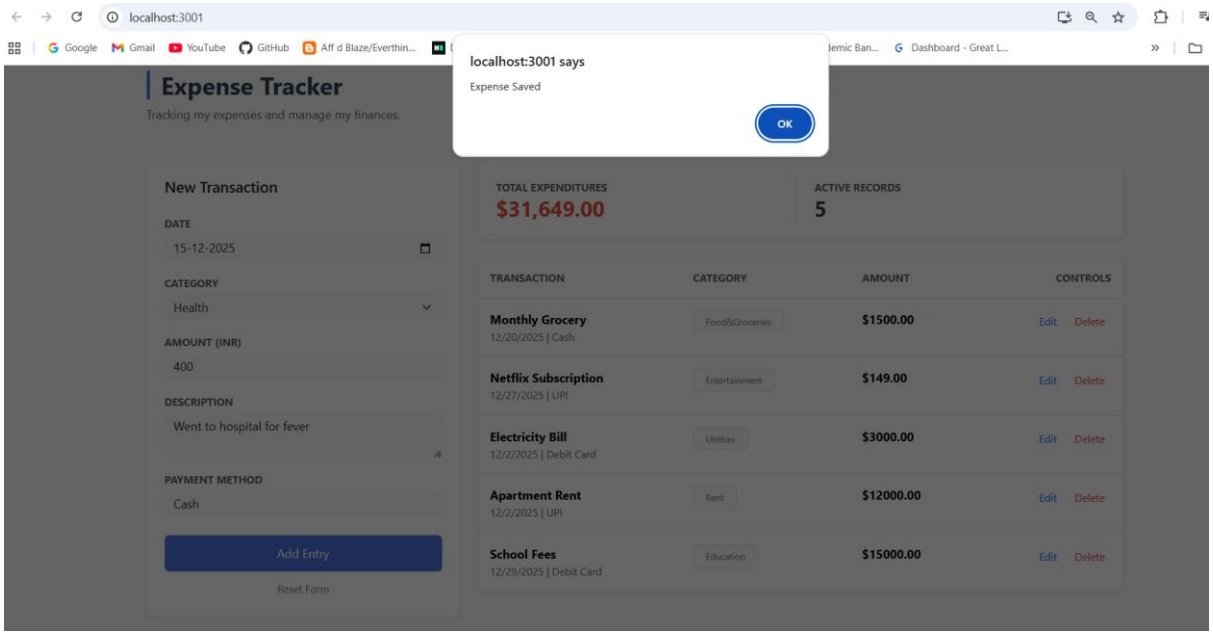
Developed Page –



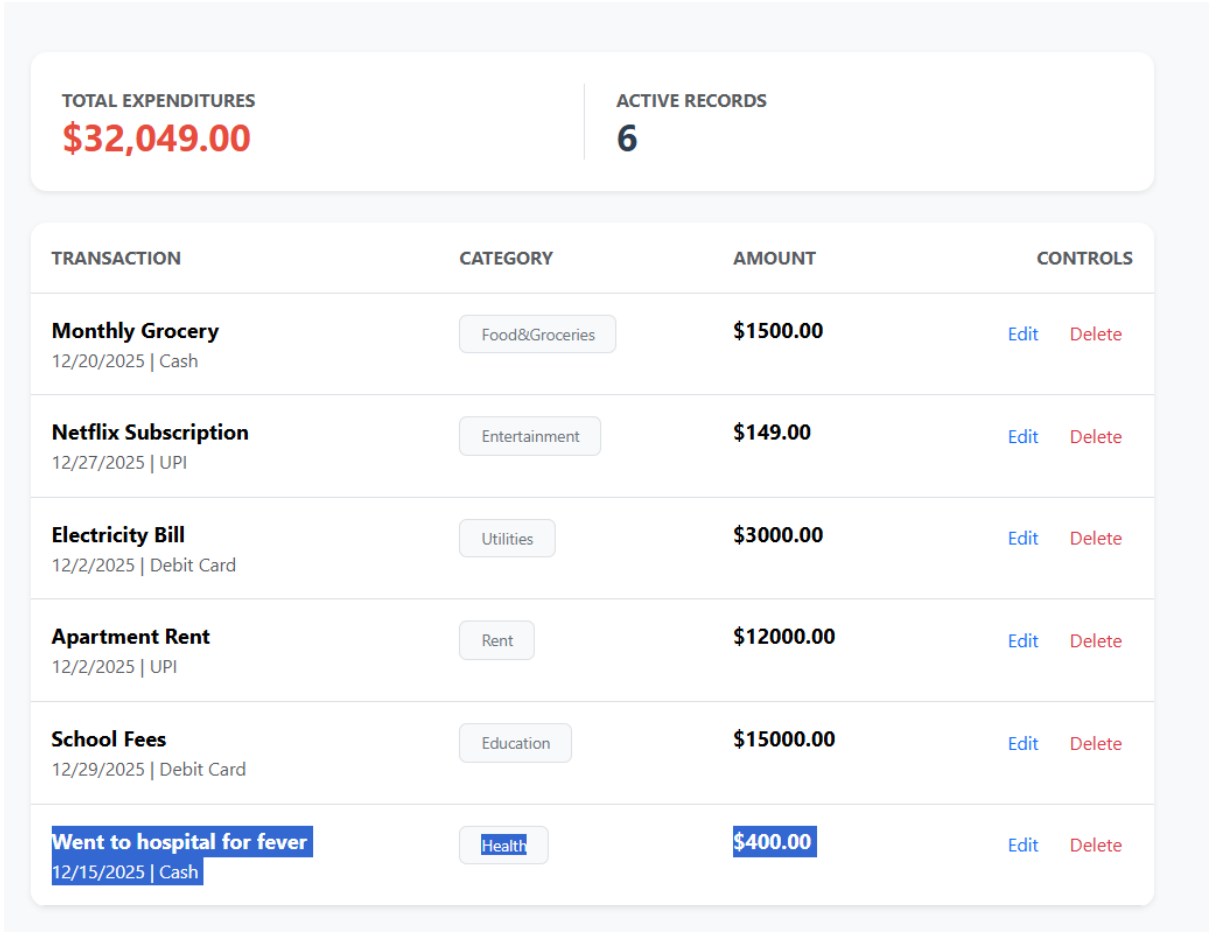
Adding Details –



After clicking Add Entry Button –

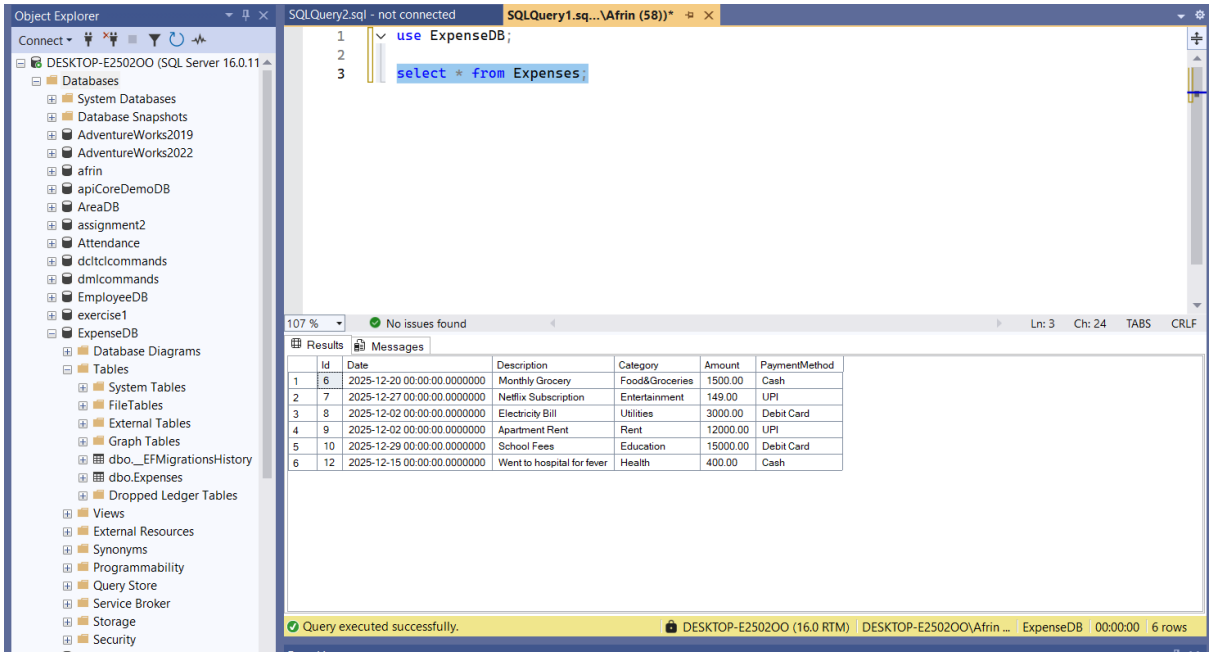


Updated Form –

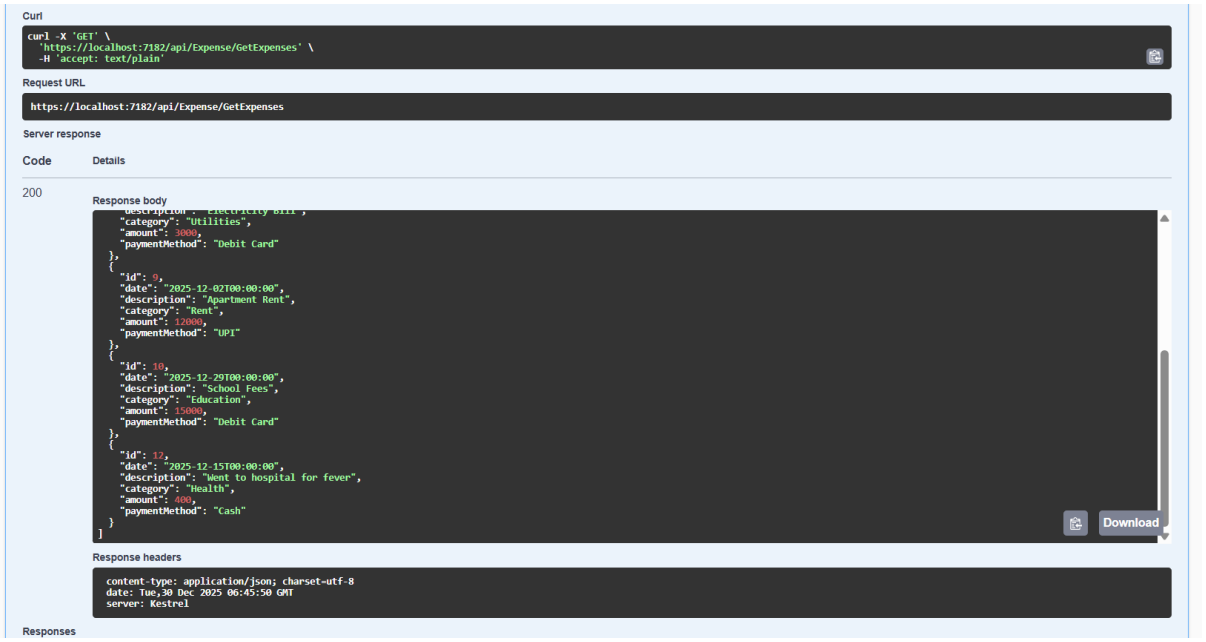


•

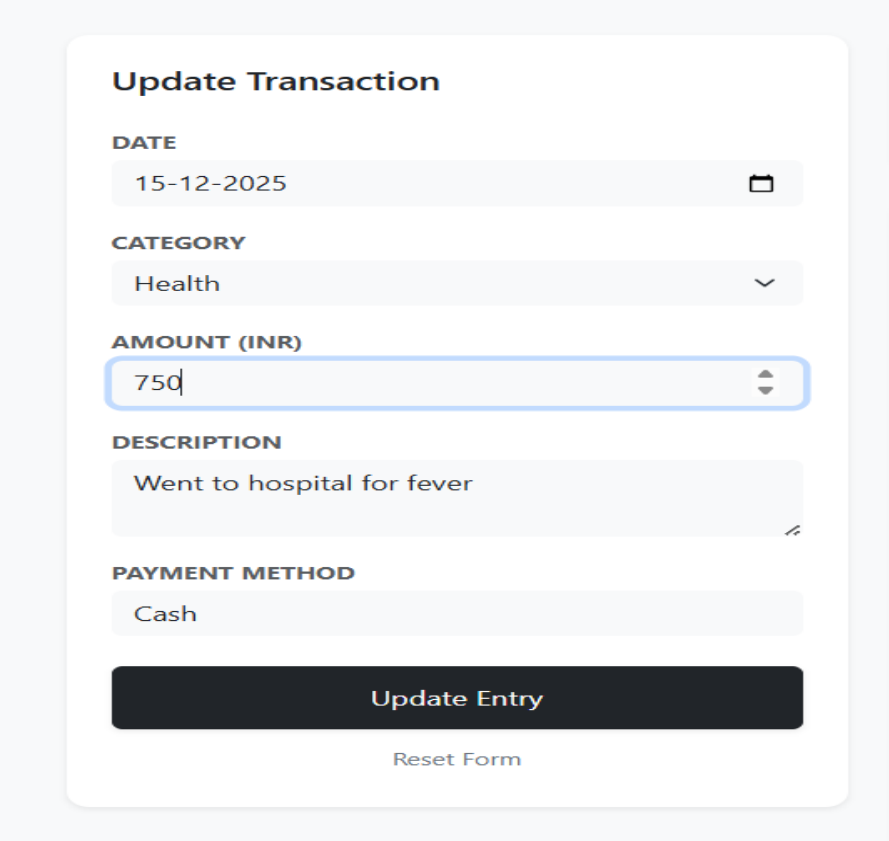
Updated Table in SQL –



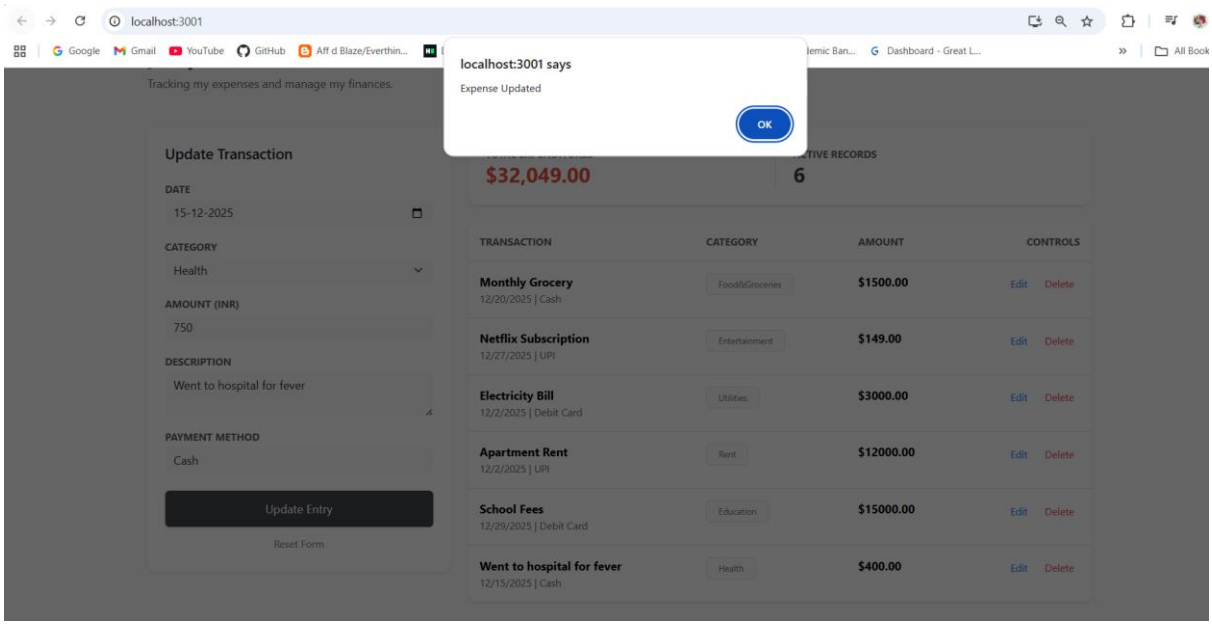
In Swagger –



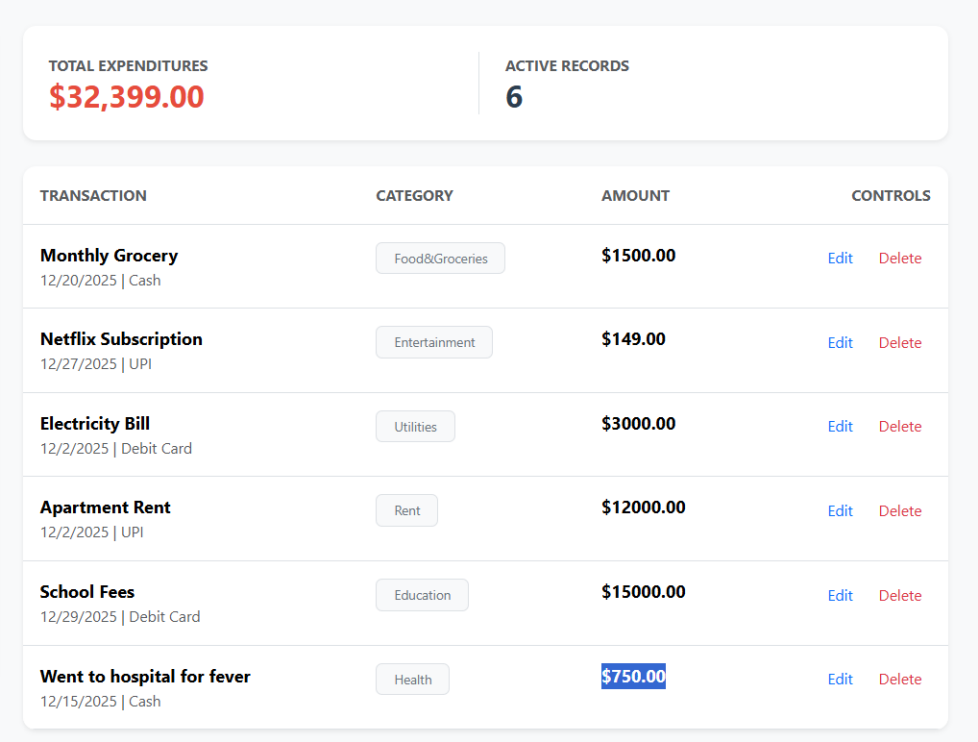
Editing Details –



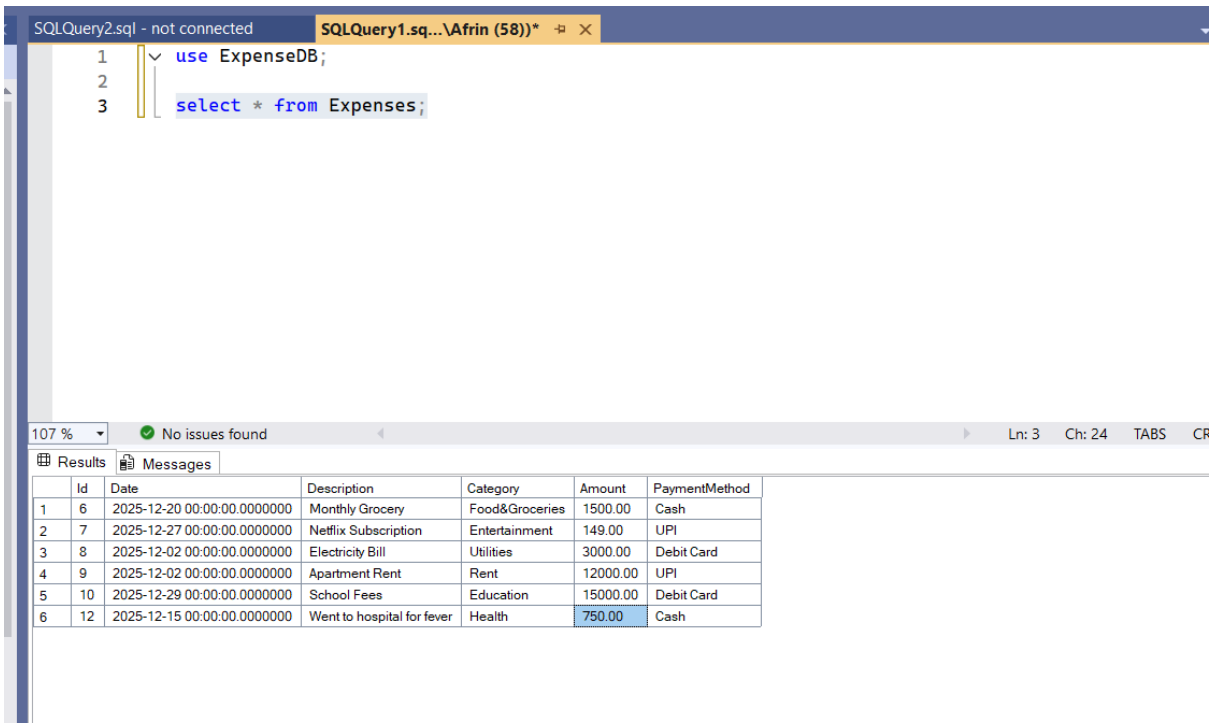
After clicking Update Entry –



Updated Form -

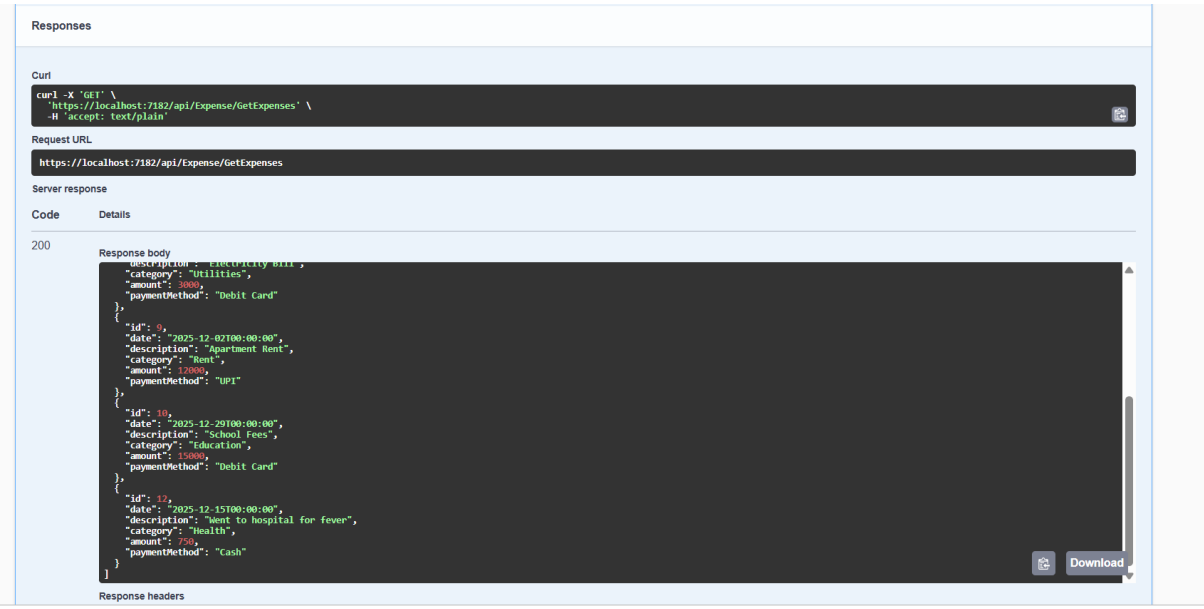


In Sql Server –

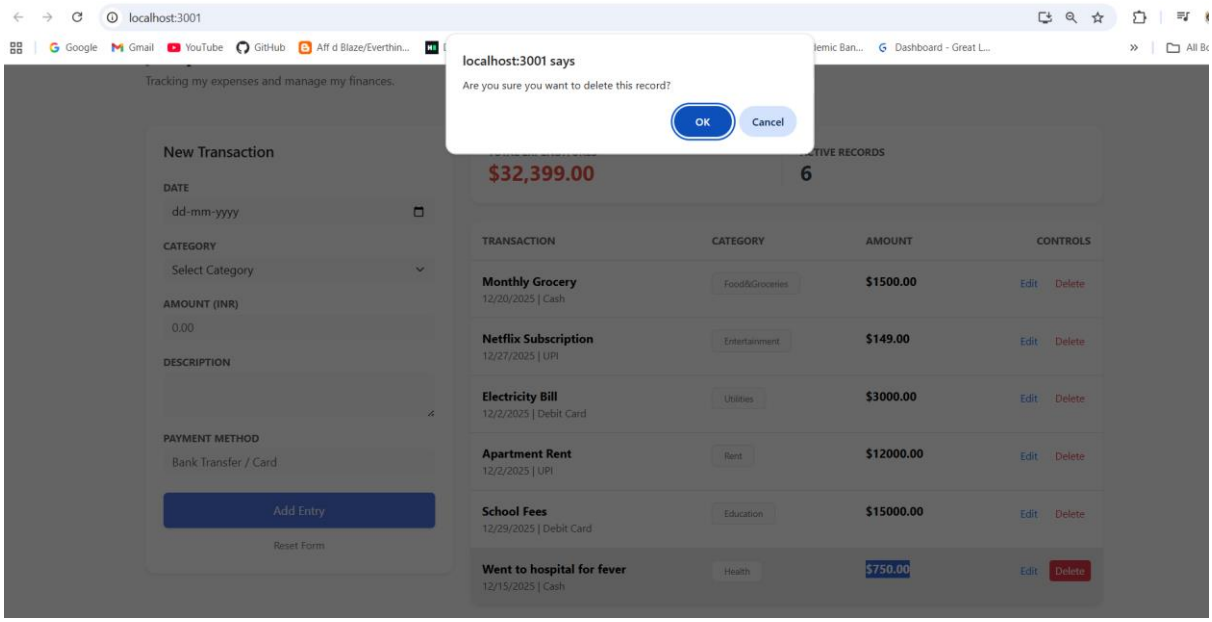


	Id	Date	Description	Category	Amount	PaymentMethod
1	6	2025-12-20 00:00:00.0000000	Monthly Grocery	Food&Groceries	1500.00	Cash
2	7	2025-12-27 00:00:00.0000000	Netflix Subscription	Entertainment	149.00	UPI
3	8	2025-12-02 00:00:00.0000000	Electricity Bill	Utilities	3000.00	Debit Card
4	9	2025-12-02 00:00:00.0000000	Apartment Rent	Rent	12000.00	UPI
5	10	2025-12-29 00:00:00.0000000	School Fees	Education	15000.00	Debit Card
6	12	2025-12-15 00:00:00.0000000	Went to hospital for fever	Health	750.00	Cash

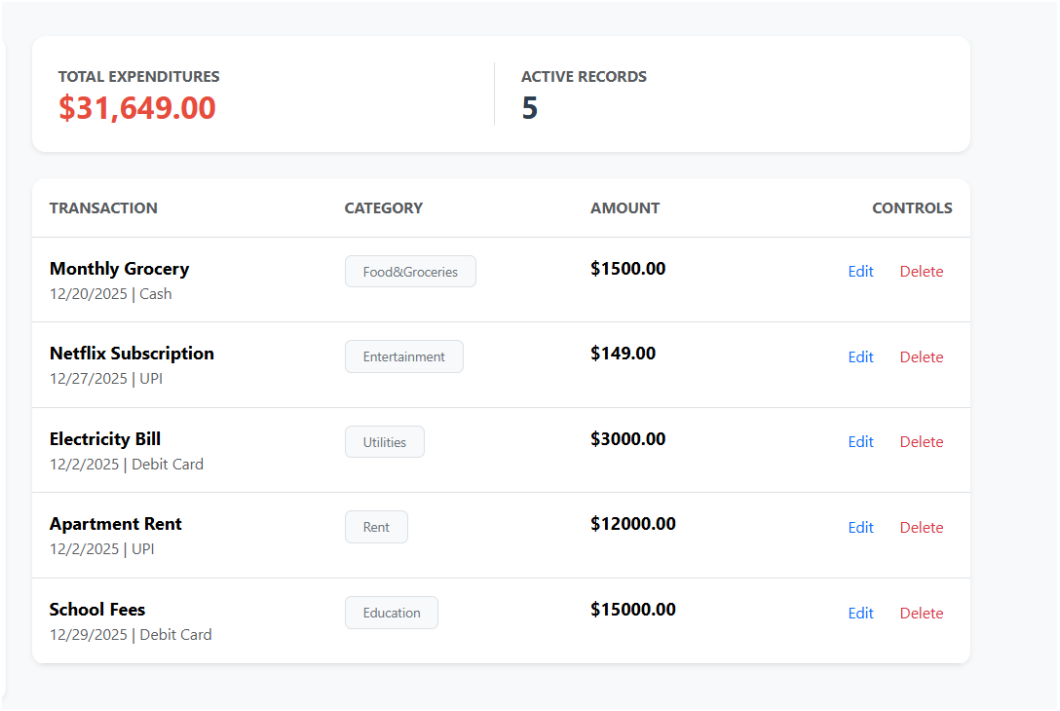
In Swagger –



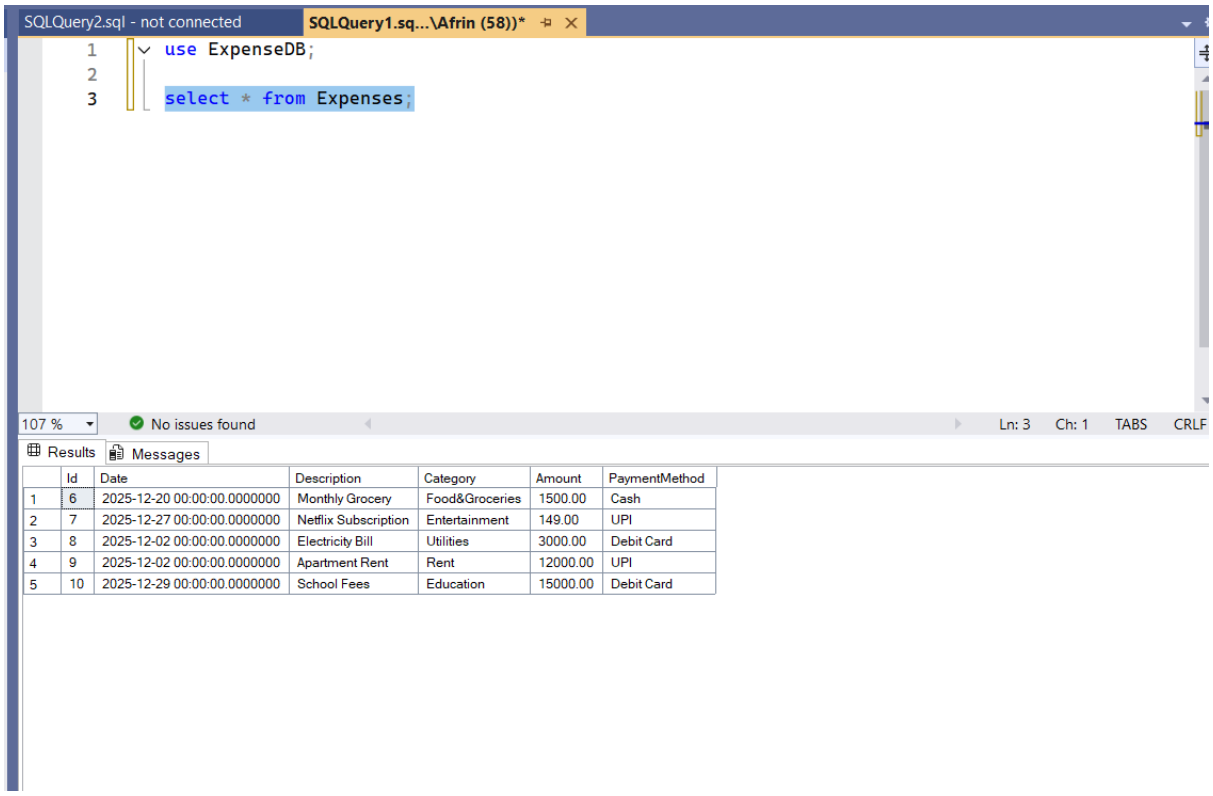
Deleting Data –



Updated form after Deleting –

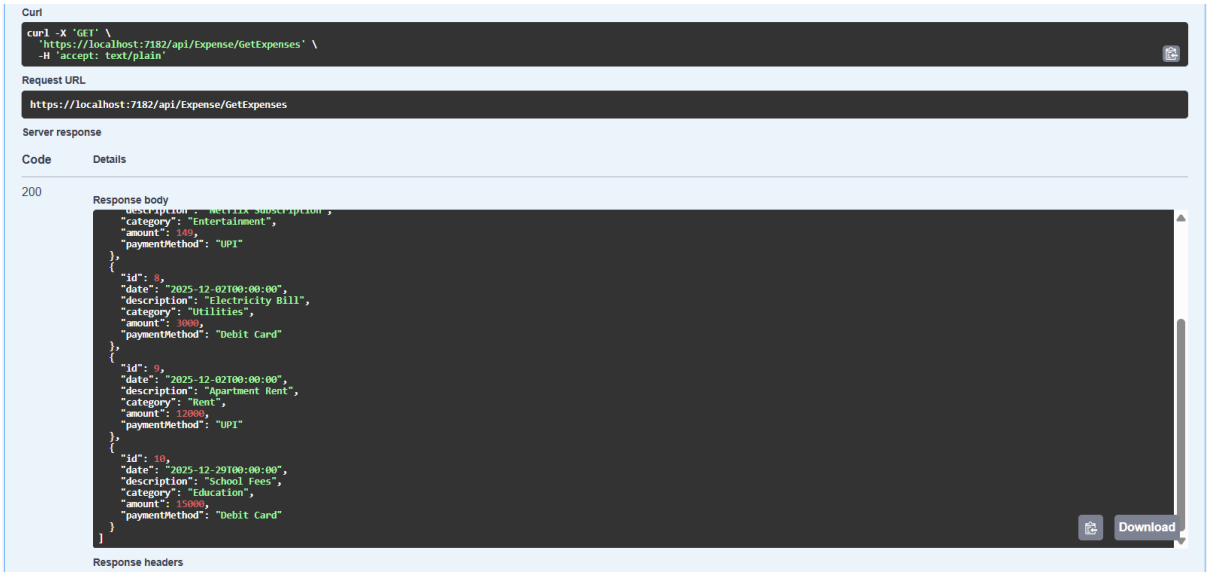


In SQL Server –



	Id	Date	Description	Category	Amount	PaymentMethod
1	6	2025-12-20 00:00:00.0000000	Monthly Grocery	Food&Groceries	1500.00	Cash
2	7	2025-12-27 00:00:00.0000000	Netflix Subscription	Entertainment	149.00	UPI
3	8	2025-12-02 00:00:00.0000000	Electricity Bill	Utilities	3000.00	Debit Card
4	9	2025-12-02 00:00:00.0000000	Apartment Rent	Rent	12000.00	UPI
5	10	2025-12-29 00:00:00.0000000	School Fees	Education	15000.00	Debit Card

In Swagger –


Curl
curl -X 'GET' \
'https://localhost:7182/api/Expense/GetExpenses' \
-H 'accept: text/plain'
Request URL
https://localhost:7182/api/Expense/GetExpenses
Server response
Code
Details
200
Response body
"category": "Entertainment",
"amount": 149,
"paymentMethod": "UPI"
"id": 8,
"date": "2025-12-02T00:00:00",
"description": "Electricity Bill",
"category": "Utilities",
"amount": 3000,
"paymentMethod": "Debit Card"
"id": 9,
"date": "2025-12-02T00:00:00",
"description": "Apartment Rent",
"category": "Rent",
"amount": 12000,
"paymentMethod": "UPI"
"id": 10,
"date": "2025-12-29T00:00:00",
"description": "School Fees",
"category": "Education",
"amount": 15000,
"paymentMethod": "Debit Card"
]
Download
Response headers