**1. Project Structure**

dotnetapp/

│

├── Controllers/

│ └── ExpenseController.cs

│

├── Models/

│ └── Expense.cs

│

├── Services/

│ └── ExpenseService.cs

│

├── Program.cs

└── dotnetapp.csproj

**2. Models**

**Models/Expense.cs**

namespace dotnetapp.Models

{

public class Expense

{

public int ExpenseId { get; set; }

public string Description { get; set; }

public decimal Amount { get; set; }

public DateTime Date { get; set; }

public string Category { get; set; }

}

}

**3. Service Implementation**

**Services/ExpenseService.cs**

using dotnetapp.Models;

namespace dotnetapp.Services

{

public class ExpenseService

{

private readonly List<Expense> \_expenses;

public ExpenseService()

{

\_expenses = new List<Expense>

{

new Expense { ExpenseId = 1, Description = "Lunch", Amount = 15.00m, Date = DateTime.Now.AddDays(-7), Category = "Food" },

new Expense { ExpenseId = 2, Description = "Groceries", Amount = 50.00m, Date = DateTime.Now.AddDays(-14), Category = "Food" },

new Expense { ExpenseId = 3, Description = "Bus Ticket", Amount = 2.50m, Date = DateTime.Now.AddDays(-21), Category = "Transport" }

};

}

public List<Expense> GetAllExpenses() => \_expenses;

public Expense GetExpenseById(int id) => \_expenses.FirstOrDefault(e => e.ExpenseId == id);

public Expense CreateExpense(Expense newExpense)

{

int newId = \_expenses.Any() ? \_expenses.Max(e => e.ExpenseId) + 1 : 1;

newExpense.ExpenseId = newId;

\_expenses.Add(newExpense);

return newExpense;

}

public bool UpdateExpense(int id, Expense updatedExpense)

{

var existing = \_expenses.FirstOrDefault(e => e.ExpenseId == id);

if (existing == null) return false;

existing.Description = updatedExpense.Description;

existing.Amount = updatedExpense.Amount;

existing.Date = updatedExpense.Date;

existing.Category = updatedExpense.Category;

return true;

}

public bool DeleteExpense(int id)

{

var expense = \_expenses.FirstOrDefault(e => e.ExpenseId == id);

if (expense == null) return false;

\_expenses.Remove(expense);

return true;

}

}

}

**4. Controller**

**Controllers/ExpenseController.cs**

using Microsoft.AspNetCore.Mvc;

using dotnetapp.Models;

using dotnetapp.Services;

namespace dotnetapp.Controllers

{

[ApiController]

[Route("api/[controller]")]

public class ExpenseController : ControllerBase

{

private readonly ExpenseService \_service;

public ExpenseController(ExpenseService service)

{

\_service = service;

}

/// <summary>

/// Get all expenses

/// </summary>

[HttpGet]

public IActionResult GetAllExpenses()

{

var expenses = \_service.GetAllExpenses();

if (!expenses.Any()) return NoContent(); // 204

return Ok(expenses); // 200

}

/// <summary>

/// Get expense by ID

/// </summary>

[HttpGet("{id}")]

public IActionResult GetExpenseById(int id)

{

var expense = \_service.GetExpenseById(id);

if (expense == null) return NotFound(); // 404

return Ok(expense); // 200

}

/// <summary>

/// Create a new expense

/// </summary>

[HttpPost]

public IActionResult CreateExpense([FromBody] Expense newExpense)

{

if (newExpense == null) return BadRequest(); // 400

var createdExpense = \_service.CreateExpense(newExpense);

return CreatedAtAction(nameof(GetExpenseById), new { id = createdExpense.ExpenseId }, createdExpense); // 201

}

/// <summary>

/// Update an existing expense

/// </summary>

[HttpPut("{id}")]

public IActionResult UpdateExpense(int id, [FromBody] Expense updatedExpense)

{

if (updatedExpense == null) return BadRequest(); // 400

var result = \_service.UpdateExpense(id, updatedExpense);

if (!result) return NotFound(); // 404

return NoContent(); // 204

}

/// <summary>

/// Delete an expense

/// </summary>

[HttpDelete("{id}")]

public IActionResult DeleteExpense(int id)

{

var deleted = \_service.DeleteExpense(id);

if (!deleted) return NotFound(); // 404

return NoContent(); // 204

}

}

}

**5. Program.cs (Dependency Injection & Swagger)**

**Program.cs**

using dotnetapp.Services;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container

builder.Services.AddSingleton<ExpenseService>();

builder.Services.AddControllers();

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var app = builder.Build();

app.UseSwagger();

app.UseSwaggerUI(c =>

{

c.SwaggerEndpoint("/swagger/v1/swagger.json", "Expense Management API V1");

});

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run("http://localhost:8080");

**6. HTTP Status Codes Summary**

| **Operation** | **Status Code** |
| --- | --- |
| Get all expenses (found) | 200 OK |
| Get all expenses (none) | 204 No Content |
| Get expense by ID (found) | 200 OK |
| Get expense by ID (not found) | 404 Not Found |
| Create expense (valid) | 201 Created |
| Create expense (invalid payload) | 400 Bad Request |
| Update expense (found) | 204 No Content |
| Update expense (not found) | 404 Not Found |
| Delete expense (found) | 204 No Content |
| Delete expense (not found) | 404 Not Found |

**7. Sample Test Data**

{

"description": "Dinner",

"amount": 20.00,

"date": "2025-10-12T00:00:00",

"category": "Food"

}