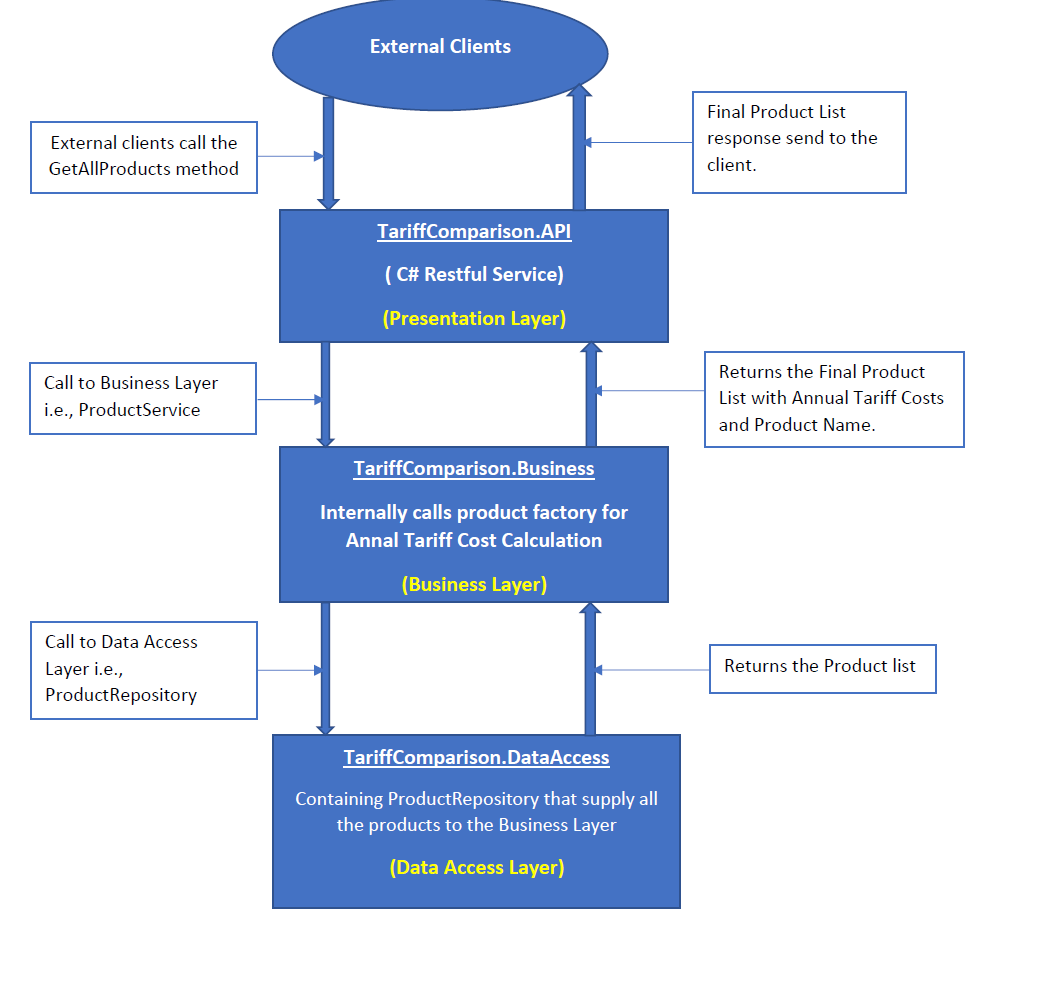
**Technical Design - Tariff Comparison Service**

**Solution Flow Diagram**



**Solution Details:** The solution is design using layered architecture in a loosely coupled way. The IOC is used to establish the loose coupling among the classes. The entire solution has been divided into following layers

1. **TariffComparison.DataAccess** **(Data Access Layer):** This data access layer is implemented using Repository pattern, consists of the mock data of product list. The actual database implementation was not implemented, but Repository is flexible to adapt real database changes easily. The screen shot of this layer is as follows

Graphical user interface, text, application

Description automatically generated

1. **TariffComparison.Business (Business Layer):** This business layer is responsible for overall business calculation of products. Currently only two types of products are available into the system which are “basic electricity tariff” and “Packaged tariff” respectively. But this layer is designed as factory pattern which can accommodate new products easily. The details of individual components are as follows

* **ITariffCostCalculator.cs**: Consists of non-implemented method which servers as contract for concrete products. The method CalculateAnnualCosts (double consumptionInKWh) accept consumption In KWh and returns the calculated annual cost.
* **BasicElectricityTariffCostCalculator.cs:** Consists of concreate implementation of CalculateAnnualCosts method of ITariffCostCalculator. This calculates the annual cost of **“basic electricity tariff”** type of product.
* **PackagedTariffCostCalculator.cs:** Consists of concreate implementation of CalculateAnnualCosts method of ITariffCostCalculator. This calculates the annual cost of **“Packaged tariff”** type of product.
* **TariffCostCalculatorFactory.cs:** All Product Tariff Calculation Families are instantiated inside this class.
* **ProductService.cs:** Implements IProductService interface, works as manager class to delegates the request to specific factory product for annual tariff costs calculation.

The screen shot of this layer is as follows

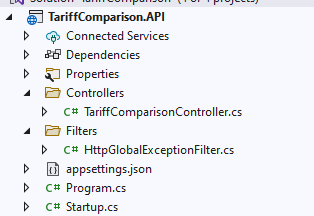
**Graphical user interface, text, application

Description automatically generated**

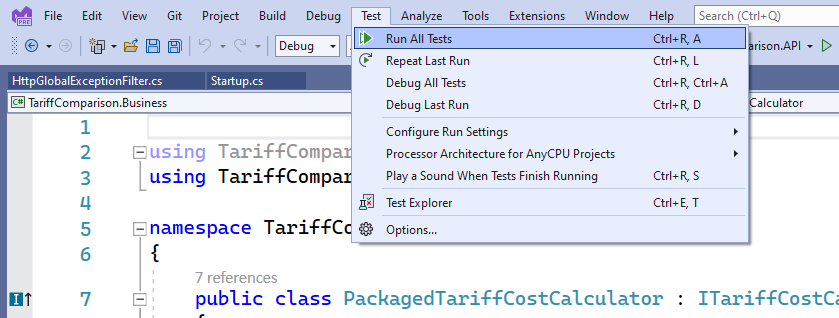
1. **TariffComparison.API:** Presentation Layer ashost for RESTful service where outside communication happens. External clients are interacted with the service using “GetAllProducts” endpoint and pass the consumption as an input parameter and service returns the list of all the products. The core components of service are as follows

* **TariffComparisonController.cs** – It consists of Api method which internally calls ProductService’s “GetAllProducts” method.
* **Startup.cs:** Bootstrapper to the service which initializes the required dependencies like Repository, Business Service, Routing and Swagger Api etc.
* **HttpGlobalExceptionFilter.cs :** Global exception handler to the application.

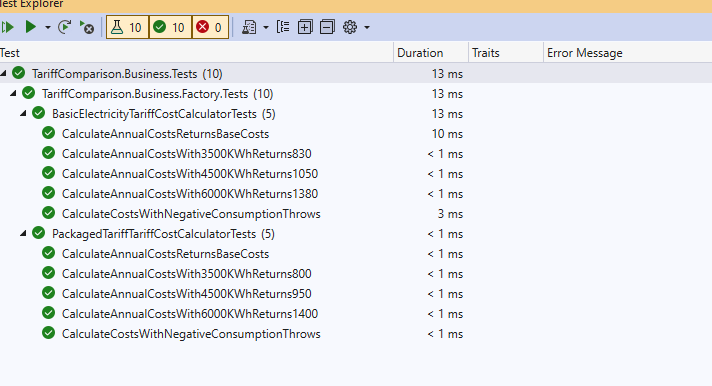
The screen shot of this layer is as follows



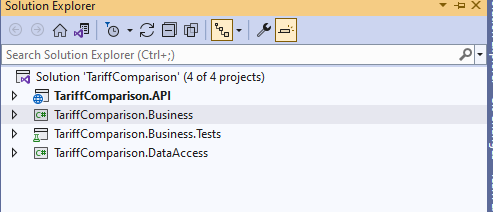
1. **TariffComparison.Business.Tests :** This solution consists of the possible Unit Tests for “BasicElectricityTariffCostCalculator” and “PackagedTariffCostCalculator” Tariff Calculators. All Tests can be run using Run All Tests as per below.



The Test Results will appear as



Complete Solution’s Layers are as follows



**Technologies used:** The “Tariff Comparison” solution is developed using **C# and .NET 5.0** as a

Restful Service which returns the product list. In addition, **Visual Studio 2022 Preview** is used to develop an entire solution.

**Service Endpoint**

<http://localhost:5041/api/TariffComparison/GetAllProducts>

**Request with consumption = 3500** <http://localhost:5041/api/TariffComparison/GetAllProducts?consumption=3500>

**Response**:



**Request with consumption = 4500**

<http://localhost:5041/api/TariffComparison/GetAllProducts?consumption=4500>

**Response**



**Request with consumption = 6000**

<http://localhost:5041/api/TariffComparison/GetAllProducts?consumption=6000>

**Response**

