CompareTariff Project Summary

Design Patterns and Principles used:

1. **SOLID Principles:**
   - Single Responsibility Principle (SRP)
   - Open/Closed Principle (OCP)
   - Liskov Substitution Principle (LSP)
   - Interface Segregation Principle (ISP)
   - Dependency Inversion Principle (DIP)
2. **TDD (Test-Driven Development)**

Key Features:

- **Tariff Comparison:** Calculate and compare annual costs for different electricity tariffs based on user consumption.
- **CRUD Operations:**
  - **Get All Tariffs**
  - **Get Tariff by ID**
  - **Add New Tariff**
  - **Update Tariff**
  - **Delete Tariff**
- **Best Tariff Calculation:** Determine the best tariff based on user consumption.
- **Real-Time Updates:** Ensure users have the most current data.
- **Robust Error Handling:** Comprehensive error handling and logging.
- **Middleware Inclusion:** Custom middleware for exception handling and validation.
- **Unit Testing:** Comprehensive unit tests for controllers and services to ensure code reliability.

Project Structure:

- **API Layer:** Handles HTTP requests and responses.
  - **Controllers:** Manages API endpoints.
  - **Middleware:** Custom middleware for exception handling (ExceptionMiddleware) and validation (ValidationExceptionMiddleware).
  - **Validators:** Validation logic for request data.
- **Business Layer:** Contains business logic and service implementations.
  - **Interfaces:** Defines service contracts.
  - **Services:** Implements business logic and calculations.
- **Data Layer:** Manages data access and database operations.
  - **Contexts:** Database context for entity framework.
  - **Interfaces:** Repository interfaces for data access.
  - **Repositories:** Implements data access logic.
- **Shared Layer:** Contains shared models and utilities.

    - **Models:** Data models used across the application.
- **Tests:** Includes unit and integration tests to ensure code quality.
    - **Controller Tests:** Verifies API endpoint functionality.
    - **Service Tests:** Validates business logic and calculations.

Advantages:

- **User-Friendly Interface**
- **Efficient and Scalable**
- **Maintainable and Extensible**
- **Cross-Platform Compatibility**