# Helix Leisure Application Structure

The solutions is divided into 5 different projects. Below is the brief about each one of those:

1. **HelixLeisureRepository**: This is a class library project has the all the model classes. At this point we just have two:

A. Sale

B. Products

2. **HelixLeisureDataAccess**: This is a class library project and contains the database context and methods to perform database operations. Classes in this project are implemented via interface to decuple the system.

3. **HelixLeisure**: This project is the Web API which connects to **HelixLeisureDataAccess** layer using the models in **HelixLeisureRepository** project. I am using Unity container to decouple the layers to make it more easily maintainable and testable. Exceptions are thrown through a custom exception class so that the client is aware of exact error.

4. **HelixLeisureUnitTest**: This project unit test methods test the methods contained in **HelixLeisureDataAccess**.

5. **HelixLeisureUI**: This is client application created using WPF. This connects to **HelixLeisure** Web API using Http protocol. This application has two tabs, first one displays the list of Sales in a grid and on double clicking on any row it displays the respective product details associated to that specific sale. The second tab is used to POST the JSON string to the Web API.