

## E-Commerce Web API

### Description:

This REST API can create new customer accounts and orders. It mainly operates on two tables Orders and Customers.

Database: E\_Comm

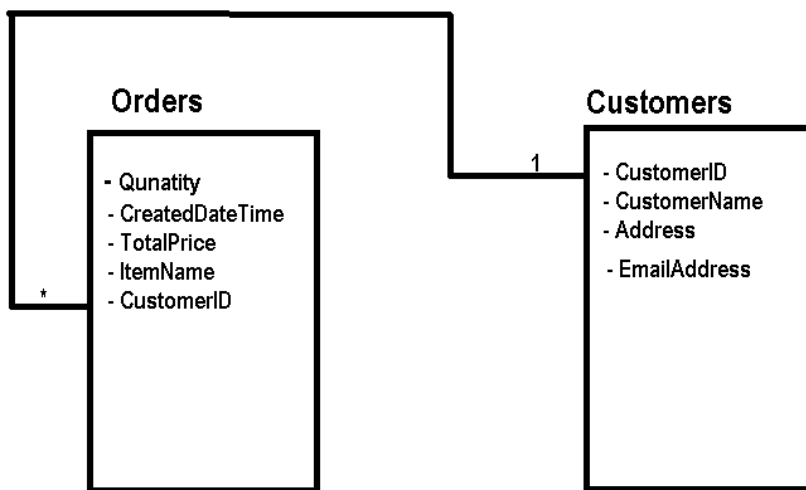


Table columns and relationship between them.

### Table Constraints:

- CustomerID provides the one to many relationships between Customers and Orders table. CustomerID is a PRIMARY KEY in Customers table and FOREIGN KEY in Orders table.
- EmailAddress column is a UNIQUE KEY in Customer Table.

**Design Patterns:** Repository Pattern and Unit of Work.

**Repository** pattern restricts anybody to work directly with the data in the application and creates layers for database operations, business logic.

**Unit Of Work** is used to warp all the changes done Orders and Customer tables in a single transaction.

**EmailValidationAttribute** : A custom a AttributeFilter is designed by implementing predefined ActionFilterAttribute class provided by .NET which is used to check whether the given Email Address from client is already in use or not.

**CustomExceptionAttribute** : A custom ExceptionFilterAttribute is designed to by implementing ExceptionFilterAttribute class provided by .NET and it handles all the exception that are uncaught. And it is applied on Controller level.

**Paging Of Orders** : This functionality is basically implemented using LINQ given Page Index and Page Size as input.

#### **Challenges:**

- The main challenge that I faced was during the serialization of the Entity object to a Json String. So instead of passing the object directly for serializing. I have created JsonMapperClasses for Customer and Order entity model class. So, all the properties are populated to JsonMapperClass object using LINQ and then are serialized to json string.

#### **Comments:**

- Didn't felt necessary to use Stored Procedure anywhere as all the database operation can be done easily through Entity Framework.
- Unit testing still needs to be written.