Shopbridge Web Api for Item

By : Gaurish Abhisheki

20.June.2021

Contents

[Tools and technologies used 2](#_Toc75116786)

[Assumptions 2](#_Toc75116787)

[Step to run 2](#_Toc75116788)

[1] Download workspace and make runnable. 2](#_Toc75116789)

[2] Run the Script given in DbScript.sql. This will create Item table and 4 Sps. 3](#_Toc75116790)

[3] Change the Server and Db information in Web.config (webapiproject) and App.config (testproject) 4](#_Toc75116791)

[Running Screenshots and API access with Postman 5](#_Toc75116792)

[Unit tests with MUnit 9](#_Toc75116793)

[Time on each task and Further enhancements 10](#_Toc75116794)

[Time spent on task 10](#_Toc75116795)

[Further Enhancements 10](#_Toc75116796)

# Tools and technologies used

* Visual Studio premium 2013
* Sql Server 2019
* Adot.NET entity Data Model
* MSUnit testing framework

# Assumptions

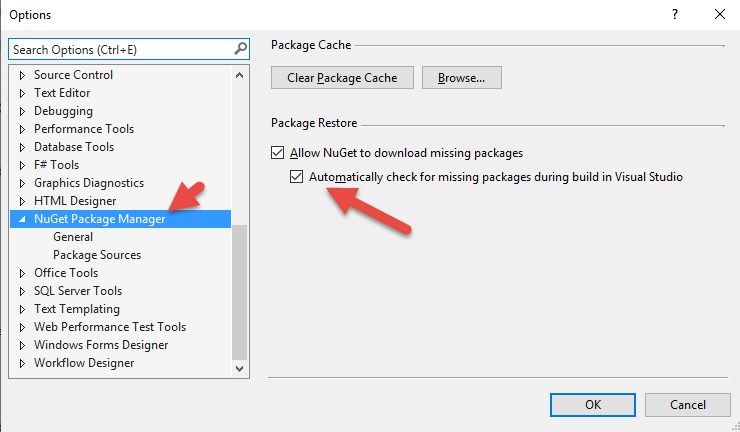
* Item table Id is Identity column with increment 1
* Insert , Update and Delete are in SPs linked to entity model so all the relevant validations or related table updates can be made
* Test and Webapi – Dbs will be different with same structure. So Dev or Prod data is not touched and test data is kept separate.
* Initially 5 recrods are inserted in the ITEM table with Script

# Step to run

## 1] Download workspace and make runnable.

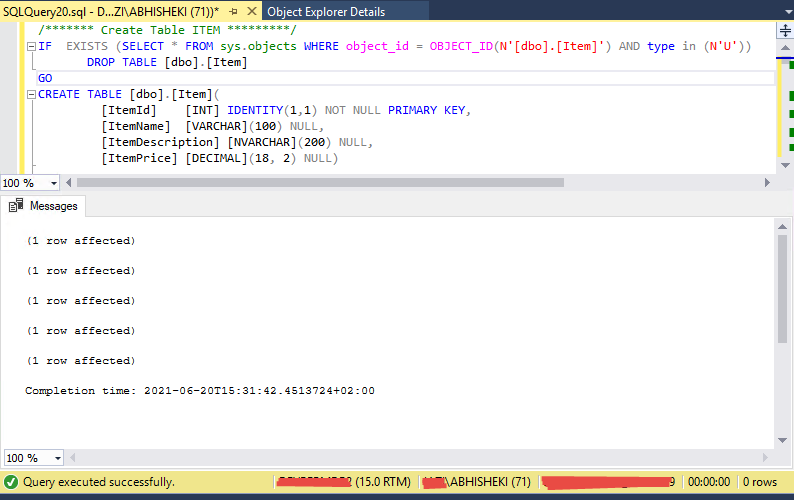
Restore NuGet packages to run the workspace.

Choose the following option under **Tools** > **Options** > **NuGet Package Manager** so Automatically check for missing packages during build in Visual Studio



## 2] Run the Script given in DbScript.sql. This will create Item table and 4 Sps.

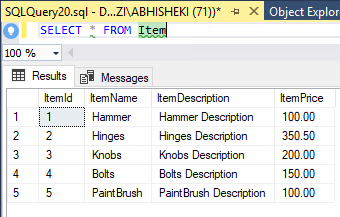


 \*Server and User is purposely hidden

This will create the table ITEM and

|  |  |
| --- | --- |
| Table | Stored procedures |
|  |  |

Initial Data in Item table



## 3] Change the Server and Db information in Web.config (webapiproject) and App.config (testproject)

Web.config

Change in

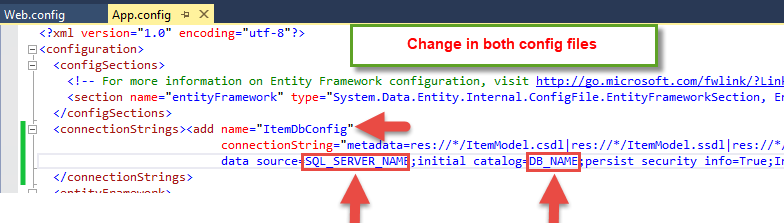
WebApiShopBridge

ShopBridgeTests

App.config

Change in

Update the connection string named “ItemDbConfig” in both Web.config and App.config to appropriate production and test db.



# Running Screenshots and API access with Postman

|  |  |
| --- | --- |
|  |  |
| 1 | Set the Api project as startup project and Run F5 |
| 2 |  |

|  |  |
| --- | --- |
| 3 | **Operation Get Single Item GetItem(1)** |
| 4 | **Operation Get all Items GetItem(null)** |

|  |  |
| --- | --- |
| 5 | **Operation Insert New Item InsertItem**(“Latches”,”Latches Description”,500**)** |
| 6 | **Operation Update existing Item SaveItem(2,”Bottle”,”Coke Glass Bottle”,12)** |

|  |  |
| --- | --- |
| 7 | **Operation Delete existing Item DeleteItem(6)**    **Try again with same number** |

# Unit tests with MUnit

|  |  |
| --- | --- |
| 1 | Initializes records for test |
| 2 | * Build Solution * Visual Studio->Test->Windows -> Test explorer |
| 3 |  |
| 3 | Asserts |
| 4 | Code coverage    \*Machine name is purposly hidden |

# Time on each task and Further enhancements

## Time spent on task

|  |  |  |
| --- | --- | --- |
| Sr Num | Task | Time |
| 1 | Pre development thinking | 1 hr |
| 2 | Data store design | 2 hr |
| 3 | API and service | 3 hr |
| 4 | Unit Test Coverage | 3 hr |

## Further Enhancements

* Constraints like Item Name Unique check
* Price > 0 constraint if required
* Authorization logic
* Proper error handling
* Request logging
* Error logging on the server side file or db