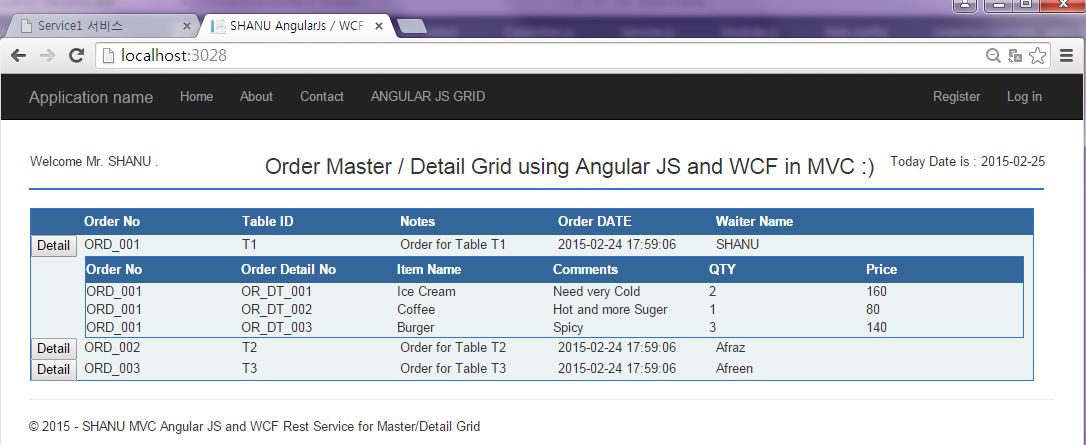
**Title: MVC Angular JS and WCF Rest Service for Master/Detail Grid**

****

**Introduction**

In this article we will create a Master Detail Grid using Angular JS and WCF. In this article we will see about.

1) How to Create WCF Rest service and retrieve data from data base.

2) How to Install Angular JS Package to our MVC application.

3) How to create our Angular JS application to create our own Master Detail Grid.

4) How to use WCS service in Angular JS and bind the data of both Master and Detail to our MVC View.

Note: prerequisite -> Visual Studio 2013 (If you don’t have Visual Studio 2013, you can download from Microsoft website <http://www.visualstudio.com/en-us/products/visual-studio-community-vs> ).

Here we can see some basic and reference links for

**Windows Communication Foundation (WCF):** WCF is a framework for building service-oriented applications.

Service-oriented application – Using protocol the service can be shared and used over a network.

For example let’s consider now we are working on a project and we need to create some common database function and those functions need to be used in multiple projects and projects are in different place and connected via internet.

In this case we can create a WCF service and we can write all our common database function in our WCF service class. We can deploy our WCF in IIS and use the URL in our application to perform DB functions. In code part let’s see how to create a WCF REST service and use it in our Angular JS application.

If you are interested to read more details about WCF then kindly go through this links.

<https://msdn.microsoft.com/en-in/library/dd203052.aspx>

<http://www.codeproject.com/Articles/803409/REST-enabled-WCF-service>

<http://www.codeproject.com/Articles/664238/Understanding-Contracts-in-WCF>

<https://msdn.microsoft.com/en-us/library/ff183866.aspx>

<http://wcftutorial.net/How_to_create_RESTful_Service.aspx>

**Angular JS**

We might be know about what is MVVM (Model, View, View Model) and MVC (Model, View and Controller).Angular JS is a JavaScript framework which is purely based on HTML CSS and JavaScript .

Similar to MVC and MVVM pattern Angular JS MVW pattern (Model View Whatever)

In our example I have used Model, View and Service. In code part let’s see how to Install and create Angular JS in our MVC Application.

If you are interested to read more details about Angular JS then kindly go through this links.

<http://www.w3schools.com/angular/default.asp>

<http://www.codeproject.com/Articles/826307/AngularJS-With-MVC-Web-API>

<http://mrbool.com/mvc-in-angularjs/28962>

<http://www.dotnetcurry.com/showarticle.aspx?ID=1051>

<http://www.c-sharpcorner.com/UploadFile/rahul4_saxena/crud-operation-in-mvc4-using-angularjs-and-wcf-rest-services/>

<http://stephenwalther.com/archive/2015/01/13/asp-net-5-and-angularjs-part-2-using-the-mvc-6-web-api>

**Code Part:**

**1) Create Database and Table:** We will create a Order Master and Order Detail table to be used for Master/Detail Grid data bind.

Below is the script to create a database, table and sample insert query.

Run this script in your SQL Server .I have used SQL Server 2008 R2.

--create DataBase

Create Database OrderManagement

-- Create OrderMasters Table

CREATE TABLE [dbo].[OrderMasters](

[Order\_No] [varchar](20) NOT NULL,

[Table\_ID] [varchar](20) NOT NULL,

[Description] [varchar](200) NOT NULL,

[Order\_DATE] [datetime] NOT NULL,

[Waiter\_Name] [varchar](20) NOT NULL

CONSTRAINT [PK\_OrderMasters] PRIMARY KEY CLUSTERED

(

[Order\_No] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

-- Insert OrderMasters sample data

INSERT INTO [OrderMasters]

([Order\_No],[Table\_ID] ,[Description],[Order\_DATE],[Waiter\_Name])

VALUES

('ORD\_001','T1','Order for Table T1',GETDATE(),'SHANU' )

INSERT INTO [OrderMasters]

([Order\_No],[Table\_ID] ,[Description],[Order\_DATE],[Waiter\_Name])

VALUES

('ORD\_002','T2','Order for Table T2',GETDATE(),'Afraz' )

INSERT INTO [OrderMasters]

([Order\_No],[Table\_ID] ,[Description],[Order\_DATE],[Waiter\_Name])

VALUES

('ORD\_003','T3','Order for Table T3',GETDATE(),'Afreen')

CREATE TABLE [dbo].[OrderDetails](

[Order\_Detail\_No] [varchar](20) NOT NULL,

[Order\_No] [varchar](20) CONSTRAINT fk\_OrderMasters FOREIGN KEY REFERENCES OrderMasters(Order\_No),

[Item\_Name] [varchar](20) NOT NULL,

[Notes] [varchar](200) NOT NULL,

[QTY] INT NOT NULL,

[Price] INT NOT NULL

CONSTRAINT [PK\_OrderDetails] PRIMARY KEY CLUSTERED

(

[Order\_Detail\_No] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

--Now let’s insert the 3 items for the above Order No 'Ord\_001'.

INSERT INTO [OrderDetails]

([Order\_Detail\_No],[Order\_No],[Item\_Name],[Notes],[QTY] ,[Price])

VALUES

('OR\_DT\_001','ORD\_001','Ice Cream','Need very Cold',2 ,160)

INSERT INTO [OrderDetails]

([Order\_Detail\_No],[Order\_No],[Item\_Name],[Notes],[QTY] ,[Price])

VALUES

('OR\_DT\_002','ORD\_001','Coffee','Hot and more Suger',1 ,80)

INSERT INTO [OrderDetails]

([Order\_Detail\_No],[Order\_No],[Item\_Name],[Notes],[QTY] ,[Price])

VALUES

('OR\_DT\_003','ORD\_001','Burger','Spicy',3 ,140)

INSERT INTO [OrderDetails]

([Order\_Detail\_No],[Order\_No],[Item\_Name],[Notes],[QTY] ,[Price])

VALUES

('OR\_DT\_004','ORD\_002','Pizza','More Chees and Large',1 ,350)

INSERT INTO [OrderDetails]

([Order\_Detail\_No],[Order\_No],[Item\_Name],[Notes],[QTY] ,[Price])

VALUES

('OR\_DT\_005','ORD\_002','Cola','Need very Cold',3 ,50)

INSERT INTO [OrderDetails]

([Order\_Detail\_No],[Order\_No],[Item\_Name],[Notes],[QTY] ,[Price])

VALUES

('OR\_DT\_006','ORD\_003','IDLY','Hot',3 ,40)

INSERT INTO [OrderDetails]

([Order\_Detail\_No],[Order\_No],[Item\_Name],[Notes],[QTY] ,[Price])

VALUES

('OR\_DT\_007','ORD\_003','Thosa','Hot',3 ,50)

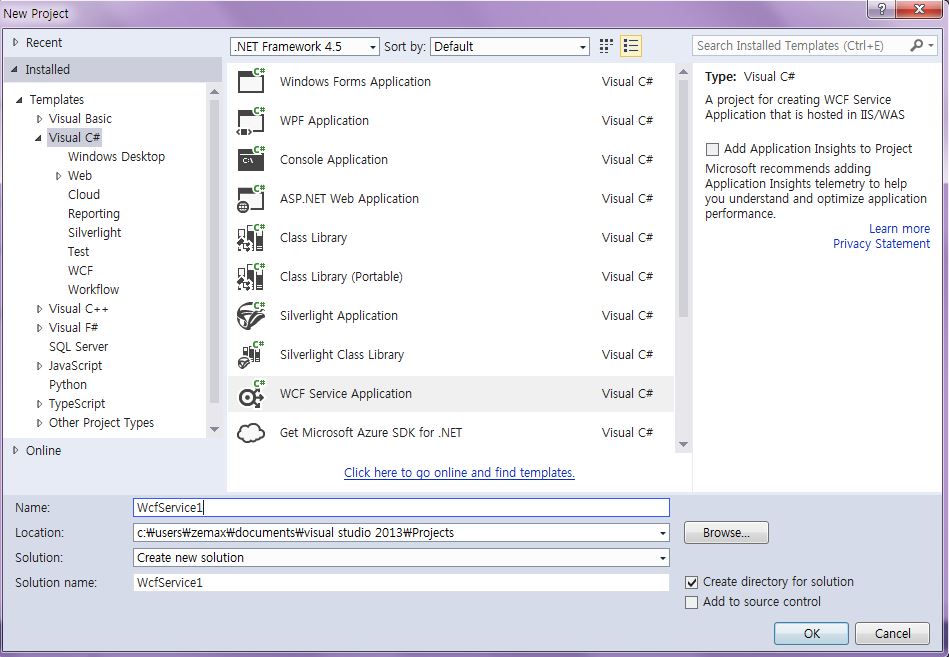
-- To Select and test Order Master and Details

Select \* FROM OrderMasters

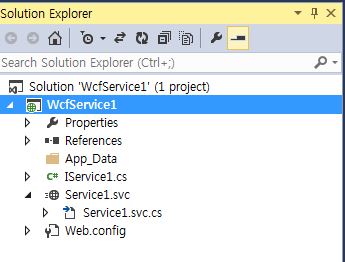
Select \* From OrderDetails

**2) Create WCF REST Service:**

Open Visual Studio 2013 -> Select File- New- Project – Select WCF Service Application -> Select your Project path and Name your WCF service and click ok.



Once we created our WCF Service we can see “IService.CS” and “Service1.svc” in Solution Explorer as below.



**IService.CS - >** In “**IService.CS”** we can see 3 Contract by default

[ServiceContract] - > Describes the Methods or any operations available for service. Service Contract is an Interface and methods can be declared inside the Service Interface using Operation Contract attribute.

[OperationContract] -> is similar to web service [WEBMETHOD]

[DataContract] -> describes the data exchange between the client and the service.

[ServiceContract]

The below code will be automatically created for all the IService.CS file.We can change and write our own code here.

public interface IService1

{

[OperationContract]

string GetData(int value);

[OperationContract]

CompositeType GetDataUsingDataContract(CompositeType composite);

// TODO: Add your service operations here

}

// Use a data contract as illustrated in the sample below to add composite types to service operations.

[DataContract]

public class CompositeType

{

bool boolValue = true;

string stringValue = "Hello ";

[DataMember]

public bool BoolValue

{

get { return boolValue; }

set { boolValue = value; }

}

[DataMember]

public string StringValue

{

get { return stringValue; }

set { stringValue = value; }

}

}

**Data Contract**

In our Example we need to get both Order Master and Order Details from data base, so I have created two Data Contract “OrderMasterDataContract” and “OrderDetailDataContract”

Here we can see we have declarared all our Table column name as Data Member.

public class OrderDataContract

{

[DataContract]

public class OrderMasterDataContract

{

[DataMember]

public string Order\_No { get; set; }

[DataMember]

public string Table\_ID { get; set; }

[DataMember]

public string Description { get; set; }

[DataMember]

public string Order\_DATE { get; set; }

[DataMember]

public string Waiter\_Name { get; set; }

}

[DataContract]

public class OrderDetailDataContract

{

[DataMember]

public string Order\_Detail\_No { get; set; }

[DataMember]

public string Order\_No { get; set; }

[DataMember]

public string Item\_Name { get; set; }

[DataMember]

public string Notes { get; set; }

[DataMember]

public string QTY { get; set; }

[DataMember]

public string Price { get; set; }

}

}

**Service Contract**

In **Operation Contract** we can see “WebInvoke” and “WebGet” which is to retrieve the data from the database in REST Serivce.

RequestFormat = WebMessageFormat.Json,

ResponseFormat = WebMessageFormat.Json,

Here we can see both the request and response format here I have used the Json format.

JSON->JavaScript Object Notation is a lightweight data interchange format

UriTemplate - > Is our Method Name and here our method return type is List.

Here I have declared 3 method as “GetOrderMaster”, “SearchOrderMaster” and “OrderDetails” . “GetOrderMaster” method is used to get the Order Master records .In “OrderDetails” Method Order\_No parameter is used to get the order detail filter by Order No.

[ServiceContract]

public interface IService1

{

[OperationContract]

[WebInvoke(Method = "GET",

RequestFormat = WebMessageFormat.Json,

ResponseFormat = WebMessageFormat.Json,

UriTemplate = "/GetOrderMaster/")]

List<OrderDataContract.OrderMasterDataContract> GetOrderMaster();

[OperationContract]

[WebGet(RequestFormat = WebMessageFormat.Json,

ResponseFormat = WebMessageFormat.Json,

UriTemplate = "/SearchOrderMaster/{Order\_No}")]

OrderDataContract.OrderMasterDataContract SearchOrderMaster(string Order\_No);

[OperationContract]

[WebInvoke(Method = "GET",

RequestFormat = WebMessageFormat.Json,

ResponseFormat = WebMessageFormat.Json,

UriTemplate = "/OrderDetails/{Order\_No}")]

List<OrderDataContract.OrderDetailDataContract> OrderDetails(string Order\_No);

}

**Iservice.Cs -> Complete Source Code**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

namespace Shanu\_WCFDBService

{

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService1" in both code and config file together.

[ServiceContract]

public interface IService1

{

[OperationContract]

[WebInvoke(Method = "GET",

RequestFormat = WebMessageFormat.Json,

ResponseFormat = WebMessageFormat.Json,

UriTemplate = "/GetOrderMaster/")]

List<OrderDataContract.OrderMasterDataContract> GetOrderMaster();

[OperationContract]

[WebGet(RequestFormat = WebMessageFormat.Json,

ResponseFormat = WebMessageFormat.Json,

UriTemplate = "/SearchOrderMaster/{Order\_No}")]

OrderDataContract.OrderMasterDataContract SearchOrderMaster(string Order\_No); [OperationContract]

[WebInvoke(Method = "GET",

RequestFormat = WebMessageFormat.Json,

ResponseFormat = WebMessageFormat.Json,

UriTemplate = "/OrderDetails/{Order\_No}")]

List<OrderDataContract.OrderDetailDataContract> OrderDetails(string Order\_No);

}

public class OrderDataContract

{

[DataContract]

public class OrderMasterDataContract

{

[DataMember]

public string Order\_No { get; set; }

[DataMember]

public string Table\_ID { get; set; }

[DataMember]

public string Description { get; set; }

[DataMember]

public string Order\_DATE { get; set; }

[DataMember]

public string Waiter\_Name { get; set; }

}

[DataContract]

public class OrderDetailDataContract

{

[DataMember]

public string Order\_Detail\_No { get; set; }

[DataMember]

public string Order\_No { get; set; }

[DataMember]

public string Item\_Name { get; set; }

[DataMember]

public string Notes { get; set; }

[DataMember]

public string QTY { get; set; }

[DataMember]

public string Price { get; set; }

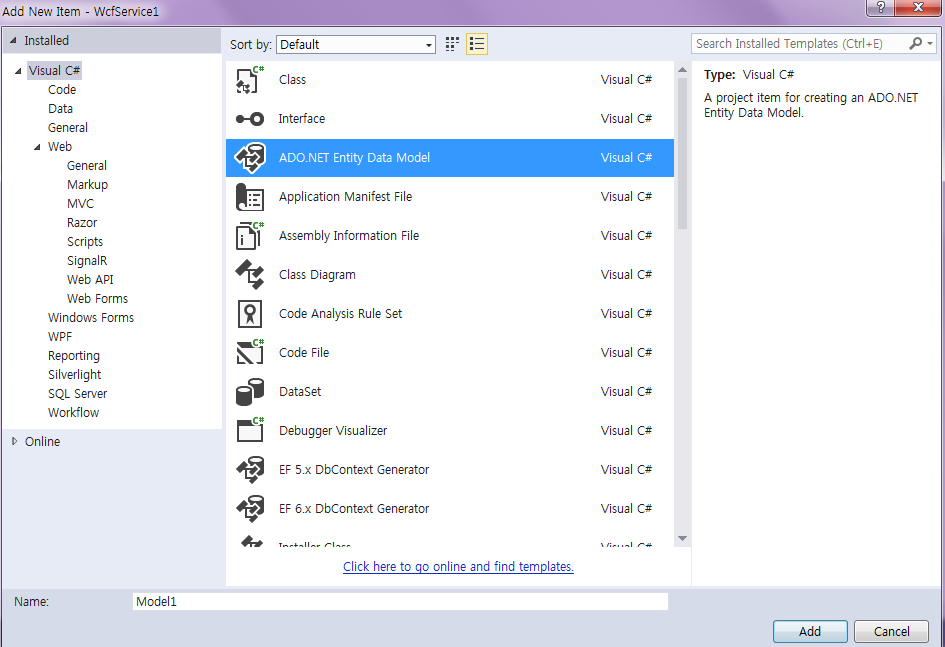
}

}

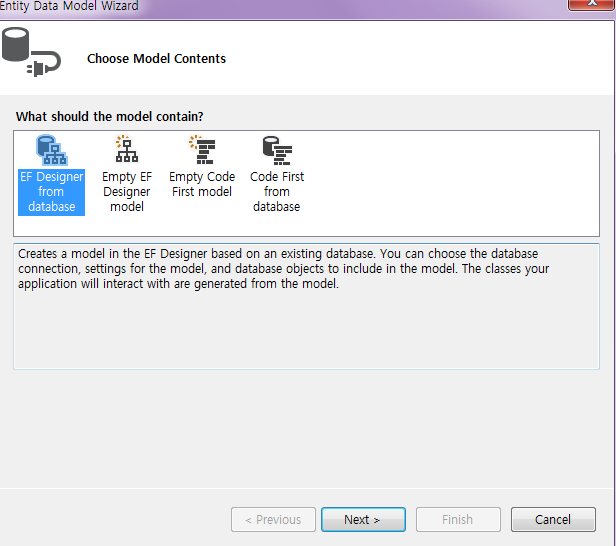
}

**Add Database using ADO.NET Entity Data Model**

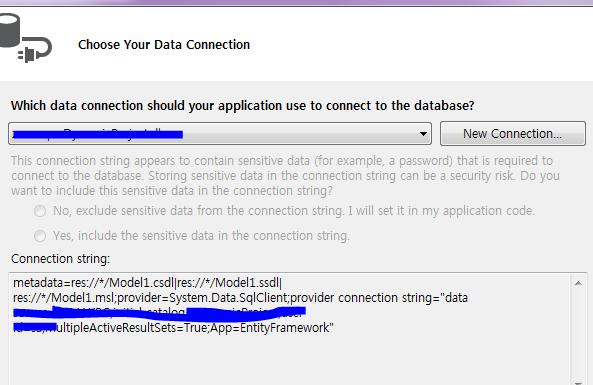
Right click your WCF project and select Add New Item->Select ADO.NET Entity Data Model and click Add.



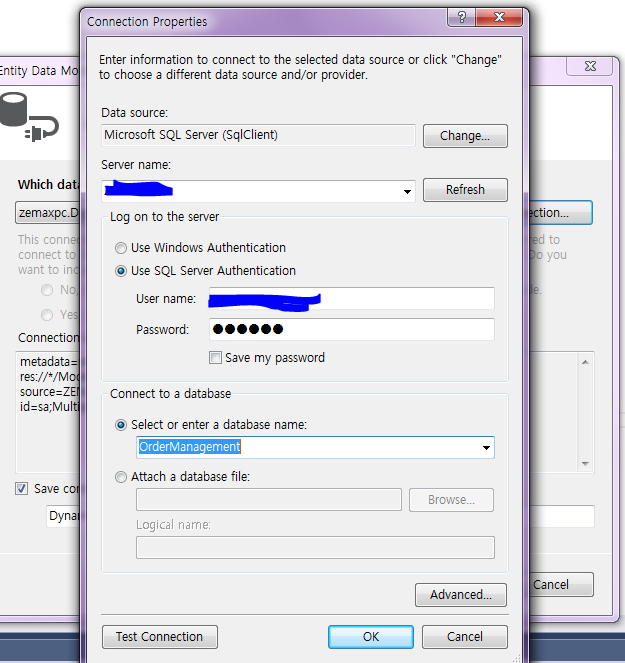
Select EF Designer from Database and click next.



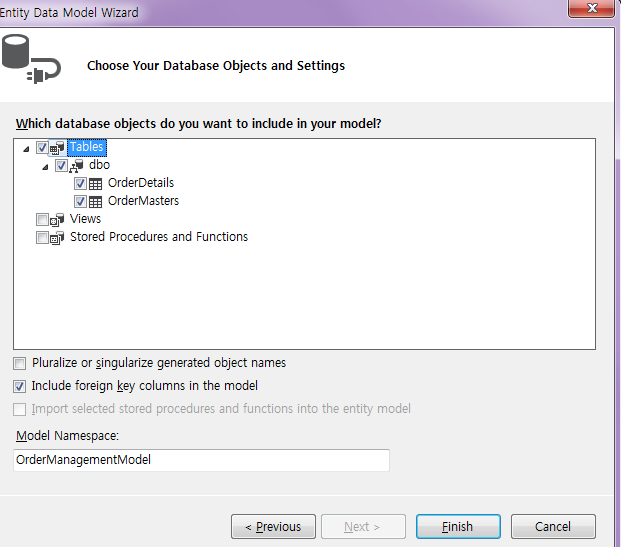
Click New Connection



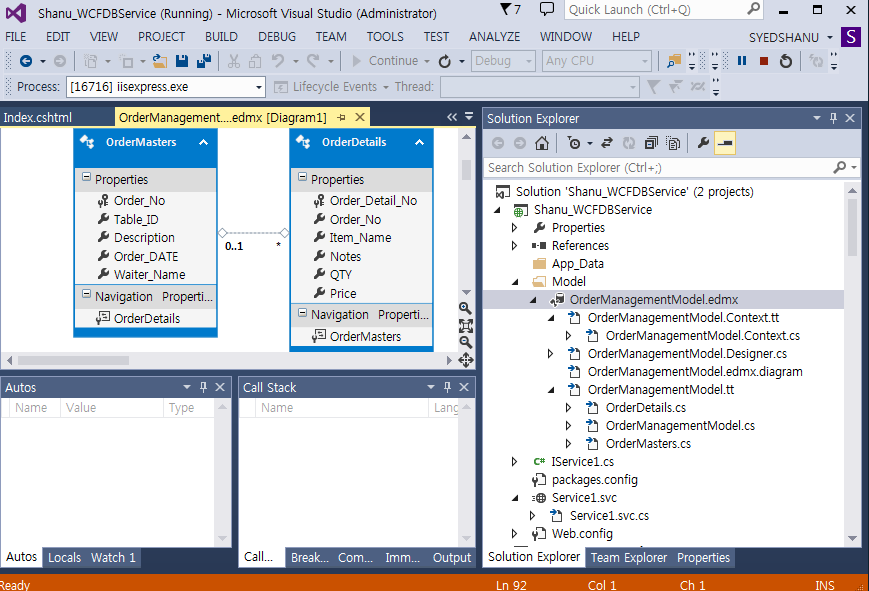
Here we can select our Database Server Name and enter your DB server SQL Server Authentication User ID and Password. We have already created our data base as “OrderManagement” so we can select the database and click ok.



Click next and select our tables need to be used .In our Example we need to use “OrderMasters’ and “orderDetails”.Select both tables and click finish.



Here we can see now we have created our OrderManagementModel.



**Service1.SVC**

In “Service.SVC.CS” implements IService Interface and override and define all the methods of Operation Contract.

For Example Here we can see I have implemented the IService1 in Service1 class .Created the Object for our Entity model and in GetOrderMaster using LINQ Query ,I have select the data from OrderMasters table and result was been added to the list.

public class Service1 : IService1

{

OrderManagementEntities OME;

public Service1()

{

OME = new OrderManagementEntities();

}

public List<OrderDataContract.OrderMasterDataContract> GetOrderMaster()

{

var query = (from a in OME.OrderMasters

select a).Distinct();

List<OrderDataContract.OrderMasterDataContract> orderMasterList = new List<OrderDataContract.OrderMasterDataContract>();

query.ToList().ForEach(rec =>

{

orderMasterList.Add(new OrderDataContract.OrderMasterDataContract

{

Order\_No = Convert.ToString(rec.Order\_No),

Table\_ID = rec.Table\_ID,

Description = rec.Description,

Order\_DATE = Convert.ToString(rec.Order\_DATE),

Waiter\_Name = rec.Waiter\_Name

});

});

return orderMasterList;

}

}

**“Service.SVC.CS” - Complete Source Code**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Runtime.Serialization;

using System.ServiceModel;

using System.ServiceModel.Web;

using System.Text;

using Shanu\_WCFDBService.Model;

namespace Shanu\_WCFDBService

{

public class Service1 : IService1

{

OrderManagementEntities OME;

public Service1()

{

OME = new OrderManagementEntities();

}

public List<OrderDataContract.OrderMasterDataContract> GetOrderMaster()

{

var query = (from a in OME.OrderMasters

select a).Distinct();

List<OrderDataContract.OrderMasterDataContract> orderMasterList = new List<OrderDataContract.OrderMasterDataContract>();

query.ToList().ForEach(rec =>

{

orderMasterList.Add(new OrderDataContract.OrderMasterDataContract

{

Order\_No = Convert.ToString(rec.Order\_No),

Table\_ID = rec.Table\_ID,

Description = rec.Description,

Order\_DATE = Convert.ToString(rec.Order\_DATE),

Waiter\_Name = rec.Waiter\_Name

});

});

return orderMasterList;

}

public OrderDataContract.OrderMasterDataContract SearchOrderMaster(string Order\_No)

{

OrderDataContract.OrderMasterDataContract OrderMaster = new OrderDataContract.OrderMasterDataContract();

try

{

var query = (from a in OME.OrderMasters

where a.Order\_No.Equals(Order\_No)

select a).Distinct().FirstOrDefault();

OrderMaster.Order\_No = Convert.ToString(query.Order\_No);

OrderMaster.Table\_ID = query.Table\_ID;

OrderMaster.Description = query.Description;

OrderMaster.Order\_DATE = Convert.ToString(query.Order\_DATE);

OrderMaster.Waiter\_Name = query.Waiter\_Name;

}

catch (Exception ex)

{

throw new FaultException<string>

(ex.Message);

}

return OrderMaster;

}

public List<OrderDataContract.OrderDetailDataContract> OrderDetails(string Order\_No)

{

var query = (from a in OME.OrderDetails

where a.Order\_No.Equals(Order\_No)

select a).Distinct();

List<OrderDataContract.OrderDetailDataContract> OrderDetailList = new List<OrderDataContract.OrderDetailDataContract>();

query.ToList().ForEach(rec =>

{

OrderDetailList.Add(new OrderDataContract.OrderDetailDataContract

{

Order\_Detail\_No = Convert.ToString(rec.Order\_Detail\_No),

Order\_No = Convert.ToString(rec.Order\_No),

Item\_Name = rec.Item\_Name,

Notes = rec.Notes,

QTY = Convert.ToString(rec.QTY),

Price = Convert.ToString(rec.Price)

});

});

return OrderDetailList;

}

}

}

**Web.Config:**

In WCF project “Web.Config”

Change the

1) Change <add binding="basicHttpsBinding" scheme="https" /> to <add binding="webHttpBinding" scheme="http" />

2) Replace the </behaviors> to

<endpointBehaviors>

<behavior>

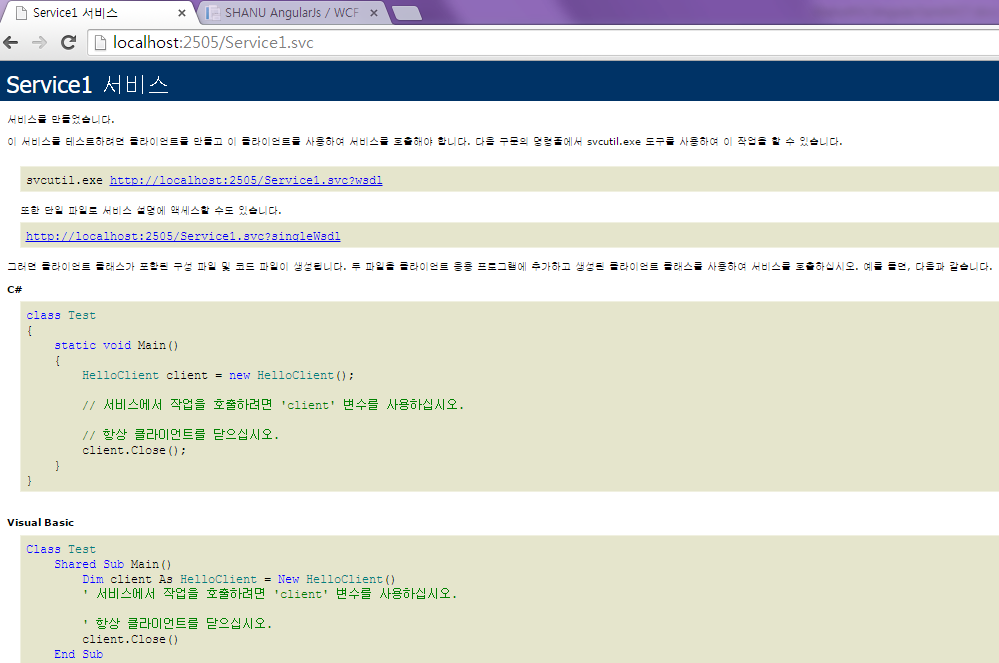
<webHttp helpEnabled="True"/>

</behavior>

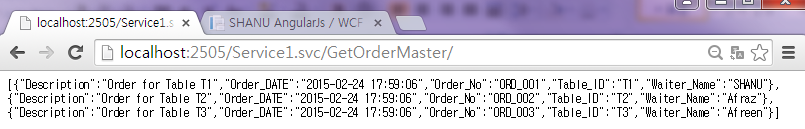
</endpointBehaviors>

</behaviors>

**Run WCF Service: ->** Now we have created our WCF Rest service , let’s run and test our service.



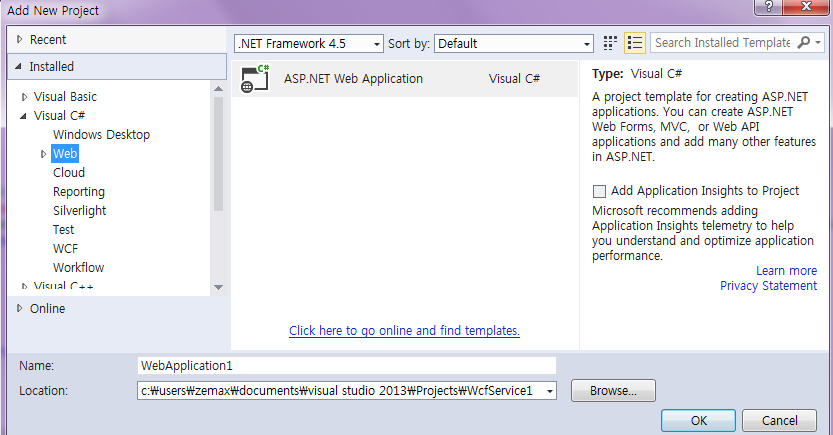
In our service URL we can add our method name and we can see the JSON result data from data base.



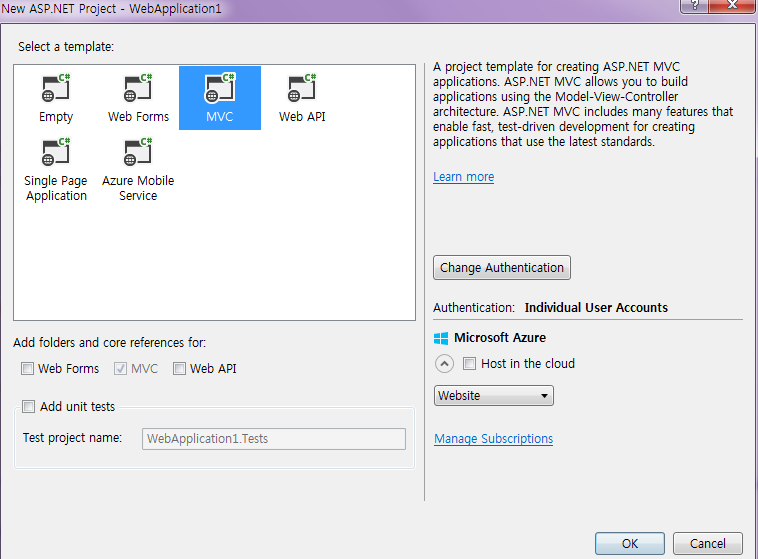
So now we have completed our WCF and now it’s time to create our MVC Angular JS application.

We can add new project to our existing project and create new MVC web Application as below.

Right click your Solution and Click Add New Project -> Enter your project Name and Click OK.

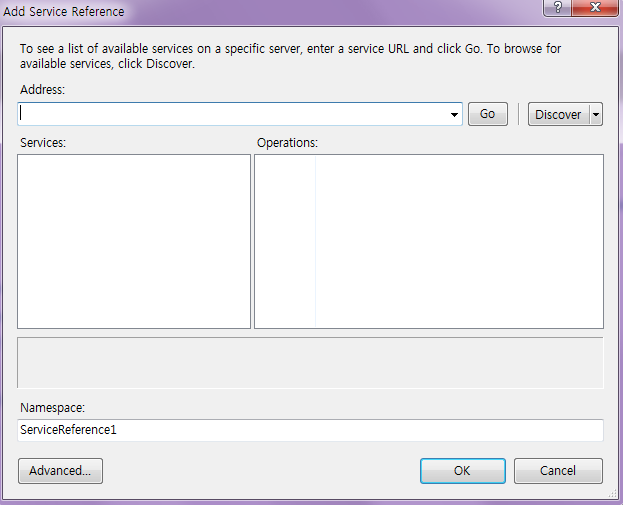


Select MVC and Click ok



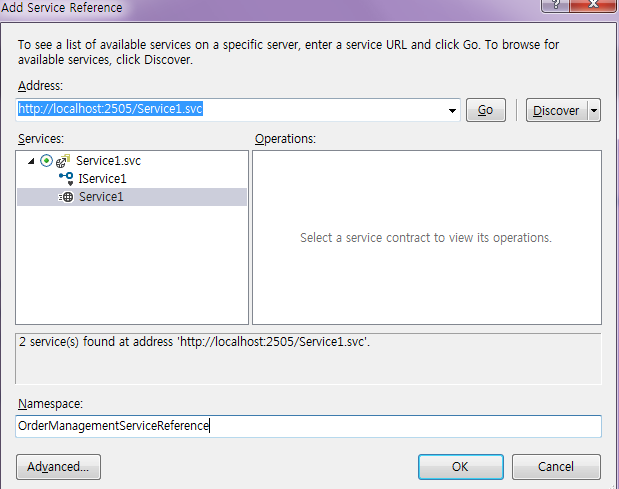
Now we have created our MVC application and, it’s time to add our WCF Service and install the Angular JS package to our solution.

**Add WCF Service:** Right click MVC Solution and Click Add->Click Service Reference.

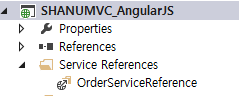


Enter your WCF URL and click GO .here my WCF URL is <http://localhost:2505/Service1.svc>

Add your name and click ok.

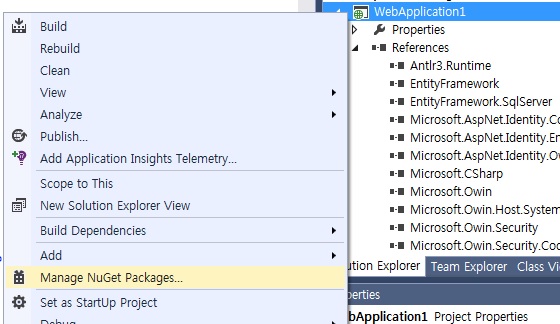


Now we have successfully add our WCF Service to our MVC Application.

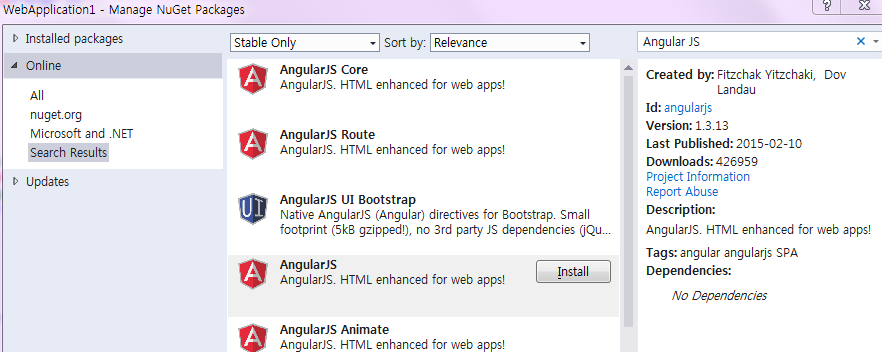


**Steps to Install Angular JS package**

Right Click your MVC project and Click-> Manage NuGet Packages

****

Select Online and Search for Angular JS. Select the AngularJs and click Install.



Now we have Installed AngularJS package to our MVC Project. Now let’s create our Angular Js

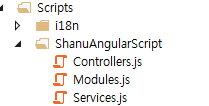
Modules.js

Controllers.js

Services.js

**Steps to Create Angular Js Script Files:**

Right Click Script folder and create your own folder to create our Angular Js Model/Controller and Service JavaScript. In your script folder add three JavaScript file and name as Modules.js, Controllers.js and Services,js as below.



**Modules.js :** here we add the reference to the Angular.js javascript and create a Angular Module named“RESTClientModule”

/// <reference path="../angular.js" />

/// <reference path="../angular.min.js" />

var app;

(function () {

app = angular.module("RESTClientModule", []);

})();

**Services.js:** here we add the reference to the Angular.js JavaScript and our Module.js.

Here we give name to our service and we use this name in controllers.js.Here for Angular service I have given the name as "AngularJs\_WCFService". You can give your own name but careful about changing the name in Controllers.js.Here we can see in method as I have passed the URL of our webservice.

/// <reference path="../angular.js" />

/// <reference path="../angular.min.js" />

/// <reference path="Modules.js" />

app.service("AngularJs\_WCFService", function ($http) {

//Get Order Master Records

this.getOrdermaster = function () {

return $http.get("http://localhost:2505/Service1.svc/GetOrderMaster");

};

//Search Order Master Records

this.getSearchOrder = function (OrderNO) {

return $http.get("http://localhost:2505/Service1.svc/SearchOrderMaster/" + OrderNO);

}

//Search Order Details Records

this.getOrderDetail = function (OrderNO) {

return $http.get("http://localhost:2505/Service1.svc/OrderDetails/" + OrderNO);

}

});

**Controllers.js:** here we add the reference to the Angular.js JavaScript and our Module.js and Services.js.Same like Services.for the controller I have given the name as "AngularJs\_WCFController".

First I get the current data and store the date using $scope.date.

I have created a method as GetOrderMasters() and using the Services module I get the resultant Order Master table and bind the result to the “$scope.OrderMastersDisp = pl.data”. Same like this we create all our rest of method.

/// <reference path="../angular.js" />

/// <reference path="../angular.min.js" />

/// <reference path="Modules.js" />

/// <reference path="Services.js" />

app.controller("AngularJs\_WCFController", function ($scope, $window, AngularJs\_WCFService) {

$scope.date = new Date();

GetOrderMasters();

//To Get All Records

function GetOrderMasters() {

var promiseGet = AngularJs\_WCFService.getOrdermaster();

promiseGet.then(function (pl) {

$scope.OrderMastersDisp = pl.data

},

function (errorPl) {

});

}

Hidetables()

function Hidetables() {

$scope.isRowHidden = false;

}

$scope.get = function (Order) {

if (Order == null) {

return;

}

if (Order.isRowHidden == true) {

Order.isRowHidden = false;

var promiseGet = AngularJs\_WCFService.getOrderDetail(Order.Order\_No);

promiseGet.then(function (pl) {

$scope.OrderDetailDisp = pl.data

},

function (errorPl) {

});

}

else {

Order.isRowHidden = true;

}

}

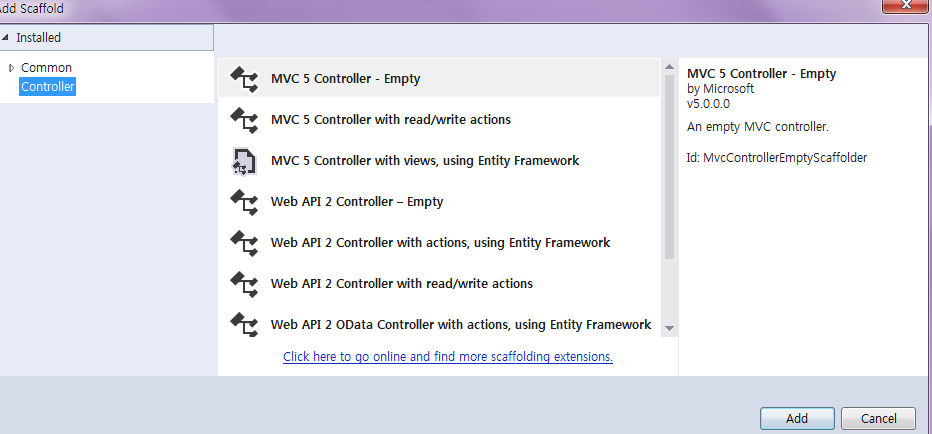
});

So now we have created our Angular Js Module /Controller and Service. So what is next?

**Create MVC Control and View** to display our result.

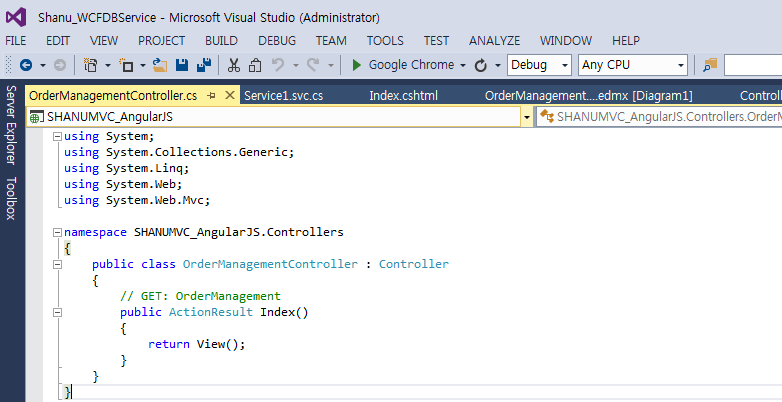
**Add Controller:**

Right click Controllers->Add Controller ->Select MVC 5 Controller –Empty ->Click Add



Change the Controller name and here I have given the name as “OrderManagementController” and click ok.

**Add View:** Right click onController Index and Click Add View.



Name the View as “Index”.

In View Design your page and reference of angular.Js , Modules.js , Services.js and Controllers.js.

In Angular JS we use {{ }} to bind or display the data.

Here we can see first I create one table and for that table.

First in table I have used the data-ng-controller="AngularJs\_WCFController" and here we can see data-ng-controller will be used to bound the data of controller to out html table.

Using <tbody data-ng-repeat="detail in OrderDetailDisp"> we can get all the records and using the <td><span>**{{**order.Order\_No**}}**</span></td> bind all the data inside the table.Same like this we have created for the Inner Table.When the user clicks on Details Button I will display the Order Details table.

<html data-ng-app="RESTClientModule">

@{

ViewBag.Title = "SHANU AngularJs / WCF and Master Detail Grid";

}

<body>

<img src="~/Images/blank.gif" alt="" width="1" height="10" />

<table width="99%" style=" border-bottom:3px solid #3273d5;">

<tr>

<td width=" 250">

<table width="99%">

<tr>

<td>

Welcome Mr. **{{**'SHANU'**}}** .

</td>

</tr>

</table>

</td>

<td class="style1" align="center">

<h3>Order Master / Detail Grid using Angular JS and WCF in MVC :)</h3>

</td>

<td align="right">

<div ng-controller="AngularJs\_WCFController">

Today Date is :

**{{**date | date:'yyyy-MM-dd'**}}**

</div>

</td>

</tr>

</table>

<img src="~/Images/blank.gif" alt="" width="1" height="10" />

<table id="tblContainer" data-ng-controller="AngularJs\_WCFController" style='width: 99%;table-layout:fixed;'>

<tr>

<td>

<table style=" background-color:#ECF3F4; border: solid 2px #3273d5; padding: 5px;width: 99%;table-layout:fixed;">

<tr style="height: 30px; background-color:#336699 ; color:#FFFFFF ;">

<th width="60"></th>

<th>Order No</th>

<th>Table ID</th>

<th>Notes</th>

<th>Order DATE</th>

<th>Waiter Name</th>

<th></th>

</tr>

<tbody data-ng-repeat="order in OrderMastersDisp">

<tr>

<td width="60">

<input type="button" id="Detail" value="Detail" data-ng-click="get(order)" />

</td>

<td><span>**{{**order.Order\_No**}}**</span></td>

<td><span>**{{**order.Table\_ID**}}**</span></td>

<td><span>**{{**order.Description**}}**</span></td>

<td><span>**{{**order.Order\_DATE**}}**</span></td>

<td><span>**{{**order.Waiter\_Name**}}**</span></td>

<td></td>

</tr>

<tr id=**{{**order.Order\_No**}}** ng-hide="order.isRowHidden" ng-init="get(order)">

<td> </td>

<td colspan="6">

<table style=" background-color:#ECF3F4; border: solid 2px #3273d5; padding: 5px;width: 99%;table-layout:fixed;">

<tr style="height: 30px; background-color:#336699 ; color:#FFFFFF ;">

<th>Order No</th>

<th>Order Detail No</th>

<th>Item Name</th>

<th>Comments</th>

<th>QTY</th>

<th>Price</th>

</tr>

<tbody data-ng-repeat="detail in OrderDetailDisp">

<tr>

<td><span>**{{**detail.Order\_No**}}**</span></td>

<td><span>**{{**detail.Order\_Detail\_No**}}**</span></td>

<td><span>**{{**detail.Item\_Name**}}**</span></td>

<td><span>**{{**detail.Notes**}}**</span></td>

<td><span>**{{**detail.QTY**}}**</span></td>

<td><span>**{{**detail.Price**}}**</span></td>

</tr>

</tbody>

</table>

</td>

</tr>

</tbody>

</table>

</td>

</tr>

</table>

</body>

</html>

<script src="~/Scripts/angular.js"></script>

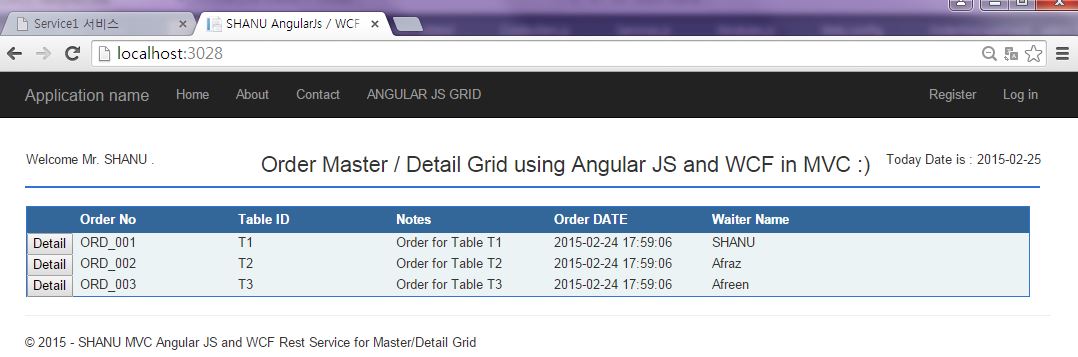
<script src="~/Scripts/ShanuAngularScript/Modules.js"></script>

<script src="~/Scripts/ShanuAngularScript/Services.js"></script>

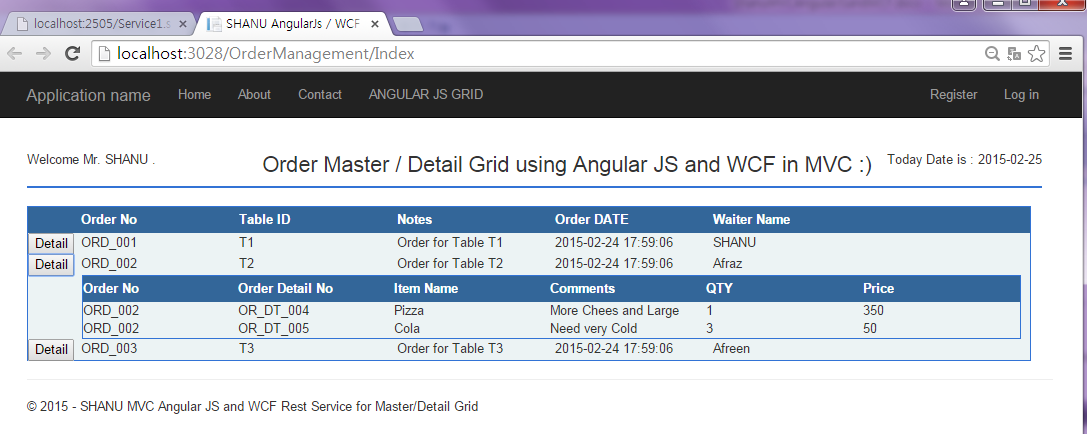
<script src="~/Scripts/ShanuAngularScript/Controllers.js"></script>

**Run your program**

Here we can see that when I run the program .First I display the Order Master records in table.

****

When user clicks on the Detail button I will display the details of order in next row.

****

**Supported Browsers: Chrome and Firefox.**