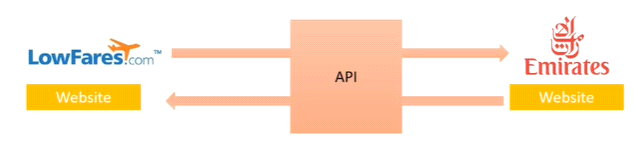
API: Application program interface through which 2 systems can communicate with each other.

Api is a application that takes request from you and then sends that request to the target system telling what you want to do then send the response back to you from the system.

Eg: Booking a tickets from other sites basically need to get the information from the providers system basically this communication happens over API



Creating project:

File > New project> Asp.net web application > select empty template > web Apia bd create

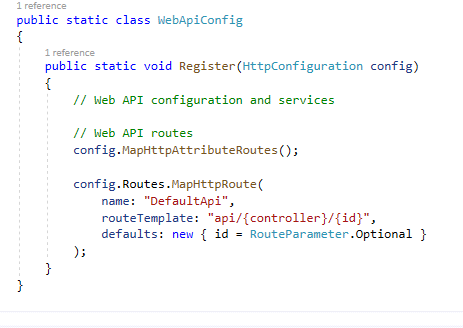
Architecture:

We have properties folder with assembly information





There is App\_Start folder which has webApiConfig class file to register all the routing information



Then we have control defines with all the HTTP verbs which act as end point for the communication to perform action on the class.

HTTP verbs:

In the world of web Api we don’t use this the term crud

Instead we use HTTP verbs

Get to get the data from the source

Put update data from the Api

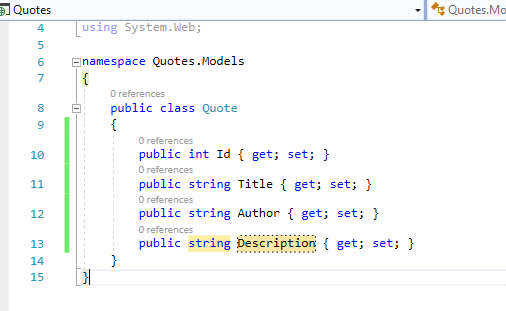
Post insert data into the source

Delete Performs remove operation from the source

Creating a simple webApi:

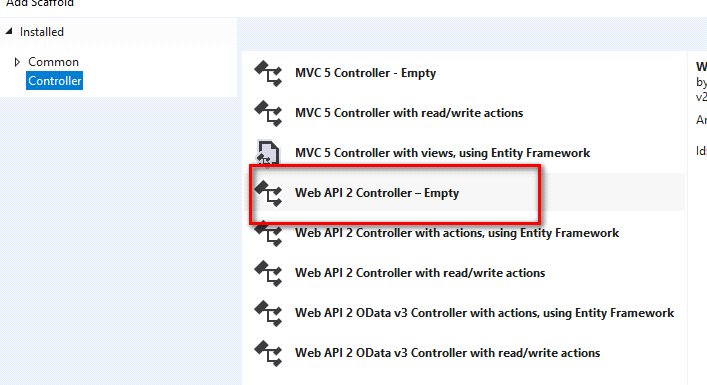
Quotes app

Add a class file called Quote with few properties:

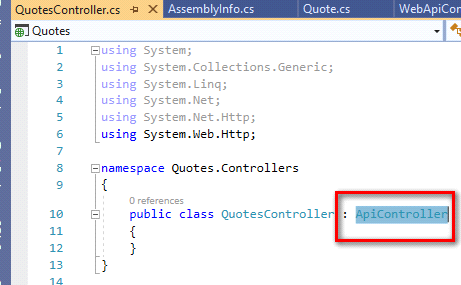


Add new controller

QuotesController select the type of controller as web api2 controller because it extended Api controller abstract class.



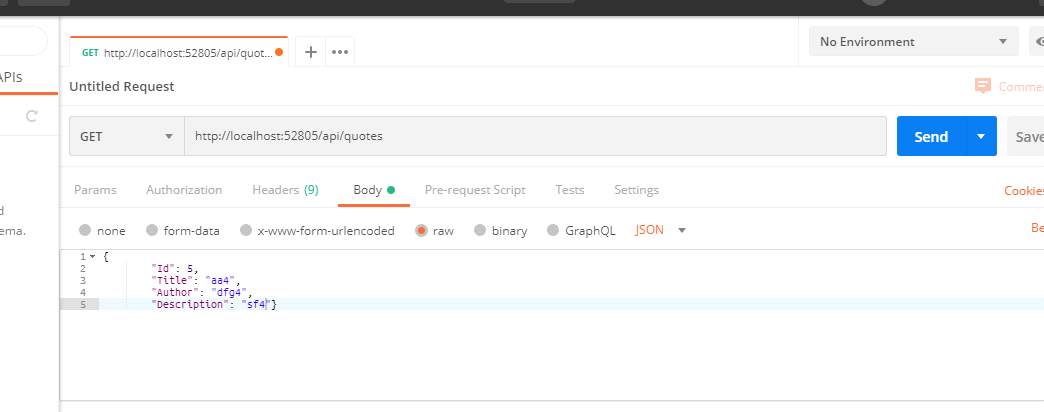
Controller:



Once the controller has action methods call the method from browser using urls

Format will be Api/Controller/Action method

To post the data to the server:



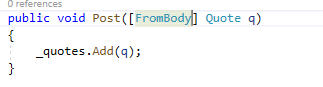
Post is a action verb that we have to use to send date to the server.

Data to be sent as a part of body

So use URL on the postman client with proper action verb.

Select body > type as json> make json data and post that data back to the server

When I post this data I have to tell the system from where I have to read data for this We need to add body attributes as a part of



API with data from the DB EF code first approach:

Step1:

1.Install entity framework

2. Create a db context object that extends from the DbContext

3. Create a object inside the controller to access the data.

4. In code first approach first add the model based on the model tables will be generate on the db

HTTP status code:

Standard response codes which helps in identifying the error occur in the system and it given by the serve. Browser error coded or internet error code.

For he validation of the request we need these codes

To return such error code the return type of my controller action method should be IHttpAction result

Return OK(“messaeg”)

Return StausCode(HttpSTatusCode.Ok)

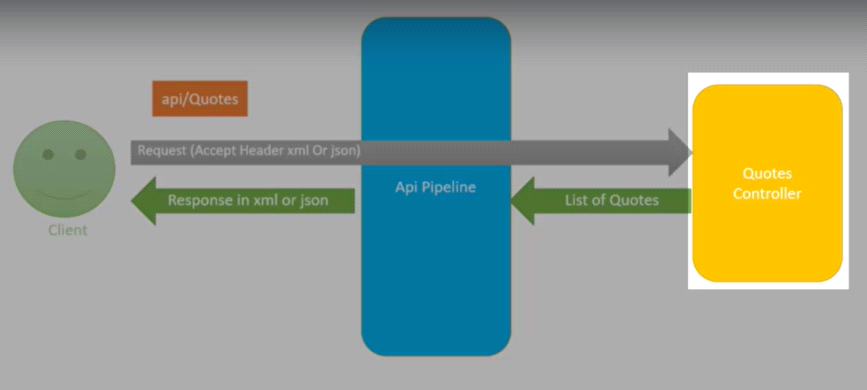
Content negotiation:

One of the feature of the rest api

Client should be able to decide the format of the date to be received from the server XML, Jason

This can be done with the help of Header field value Accept header.

In any rest client If no accept header is defined then the response will come in Jason format



In browser when you make a call to the web service by default the response will be send in Xml format

Media type formatter:

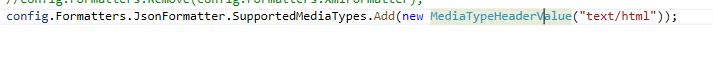
Using this we can format the media type of the data to be sent to the client.

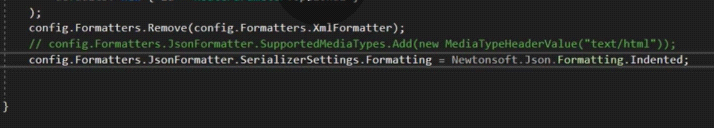


Say a client want data in browser as Jason and from the rest client it should take accept header values.

Model Validation:

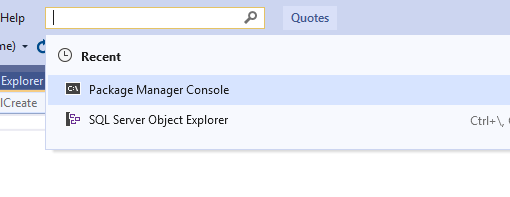
Achieved with adding data attributes to the model properties with the help of data annotation





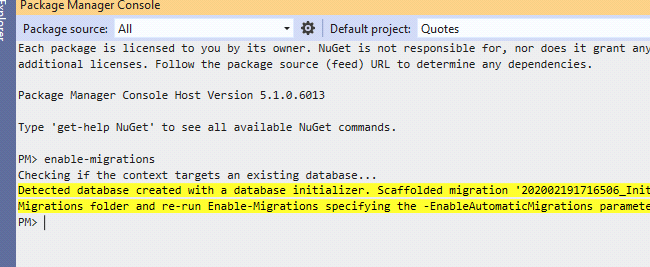
Migration in Code first:

For this enable migration in the project



Run command:

Pm> enable-migrations



Add migration for new db changes

Pm>add-migration QuoteTypeAdded hit enter

Once done

Update the data base so that changes reflect in DB

Pm>update-database

Check if the model is valid

If(!ModelState.IsValid)

{

BadRequest(ModelState)

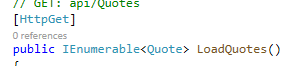
}

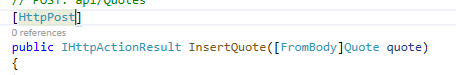
Custom method names in Webapi

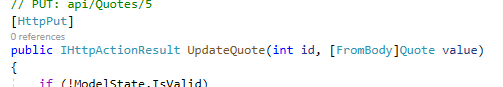
Bu default Action verb name is used as Name of the methods which act as end point for the communication.

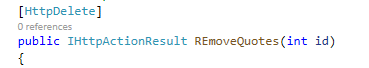
As long as action verb name is prefix for your method its still valid

We can give custom method name for our HTTP action verbs But you have to decorate your method with respective action verb attribute.









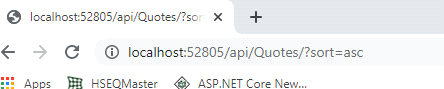
Routing in web Api:

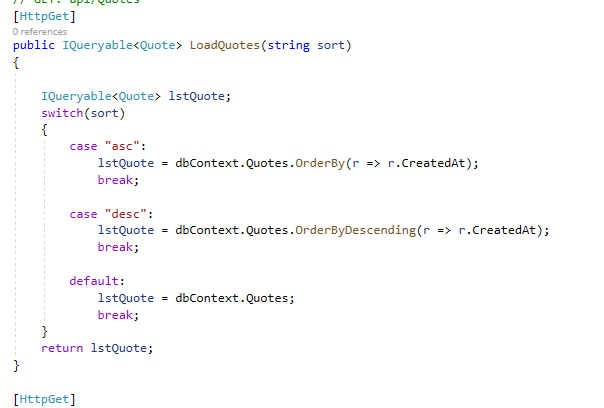
Attribute routing

By default a route template is defined I the WebApiConfig so basically your application will only handle the routes with specified url pattern.

Say I want to access a method using custom url then use attribute routing where in we specific the custom rout for an action method.

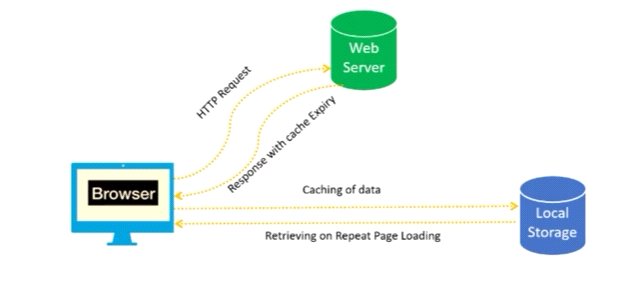
Sorting of the data:





Caching in WebApi:

Technique of storing frequently accessed information in local memory for certain time period. Eliminate the Multiple db calls to improve the performance when the same data is accessed multiple times



Implementation needs Open source cache package manger.

TO cache output we need caching component on the project.

Open source chache output for the Asp.net

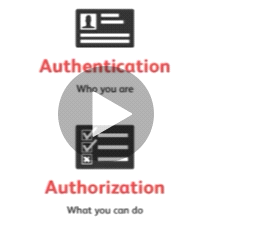
StrathWeb.CacheOuput.WebApi2

Once this is done decorate your action method with the OutputCache attribute.

ClientTime span data will be cached for so many seconds on client side

and serverTime span

Authentication and authorization:



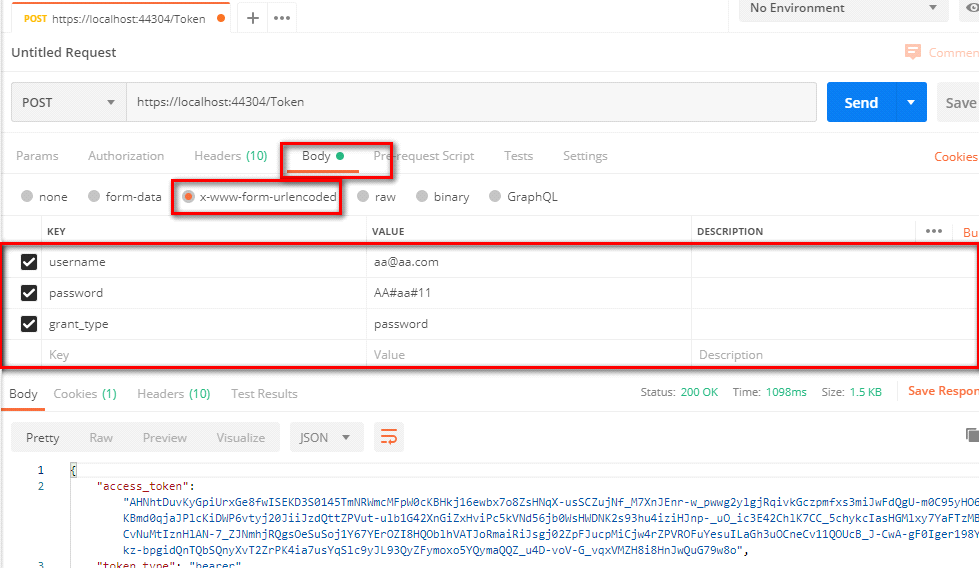
With authorization we basically give permission to the user what he can do in application this is achieved by proving the token to the user using which he can access the resources.

In web API token or access token is needed to grant permission to access the resources.

Once the user is created on the system next step is to get the authorization token for the user. Which then can be used for the Authorization to access the resources.

Token is got from the token end point defined is authentication code

We need to specify user name and password in form data

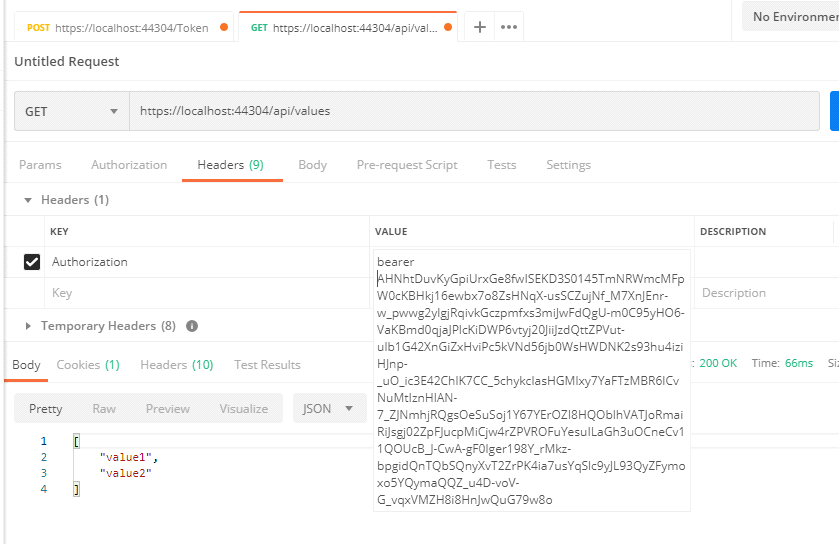


Along with this some grant type need to be specified so that the token is generated

Once the token is generated this means user is authenticated and token is ready to access the system.

So we can access the system with this.

The setup the request header with this token with type and token value



Sample App for coffee shope:

Menu and submenu relation

To load submenu in main menu call we need to include sub table data or the egger loading.

Error code:

HTTP is stateless protocol client and server communication happens over the internet using this protocol.

When client click on the link or button fro the broweser basically it send a request ot browser this is called Http request and the client will rever t back tot he server with the response called HTTp response.

And msot of the time we see some king of the status code sent from the server for example

404 from the client when the requested page is not found on the server.

All 100: informational

200: success status codes

300: redirectional status code

400 client errors

500 server erorrs.

100: continue

part of the request is received successfully and rest will be sent now

101:switching protocol:

Indicates that the server is changing the protocol to the upgraded version of the protocol.

200: ok

Evrything is ok request is received understood and being processed.

201: created

success and resource has been created.

202: accepted

server accepted the request for processing.

Web Api

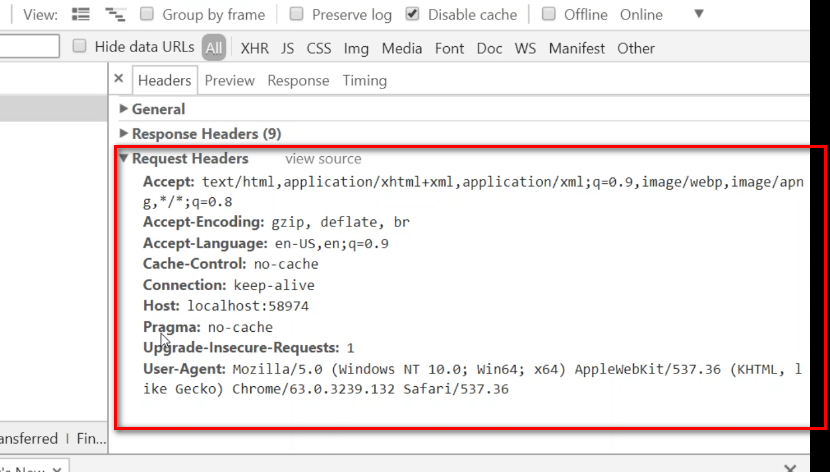
Webs api

Reset principle

1. Follow HTTP protocol:
2. Use HTTP methods put, post, get, delete don’t create your own methods. So that follow uniformity while building APIS
3. When a call made to any of the services returned are

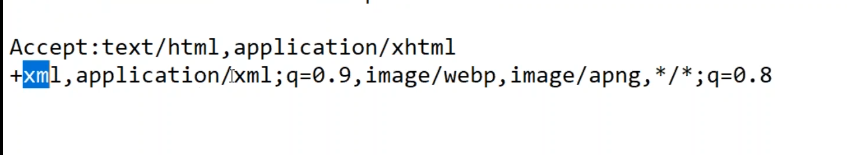
Data + header (technical data)

To send or receive few values use HTTP header information don’t go and create a custom variable (only for the standard header value)



1. Everything/resource on the Web is represented by the unique URL. Represent some resource inside the site
2. Content negotiation: good web api react to the customer request which means consumer can get the data in the required format

Using header value Accept:text/Jason

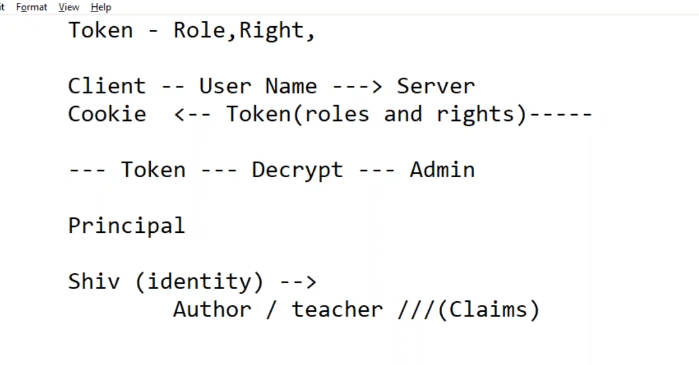


With above line client says these are the format I can read.

Type/Format

Behave as per the client

1. Response talk via status code (success, failure, client error, server error)



Identity tells about who you are

Claims tells about your role (what you claim to be)

Principle holds both claims and identity

