**Asp.Net Web API Project Details**

In CustomerServiceWebAPI Project, I have implemented all basic and important concept of Asp.Net Web API.

I have covered the following key point in this project:

* Content Negotiation
* HttpStatusCode
* Attribute Routing
* Use of [FromBody] and [FromUri]
* Use of Query String Parameter
* Implement Custom Methods
* Exceptional Handling
* Use of Ajax and jquery to call Web API
* Use of MediaTypeFormatter
* Implement Swagger
* Custom Generic Model
* Generic Repository for EF CRUD
* Basic Authentication and Authorization
* Token based authentication and authorization
* Http Verb [HttpGet],[HttpPost],[HttpPut],[HttpDelete]
* Enable Cors (Cross Domain)
* Implement CRUD using Web API1 and Web AP2
* File Uploading in a Document folder
* File Uploading in Database in binary form and downloading file from database
* Use of Session Storage and Local Storage for storing, retrieving and removing.
* Test API using Fiddler, Postman and Swagger

I have created different controller for each concept. Following are the list of Controller and their Details :

**1) CustomerController:**

This is the first controller that I created. This controller is based on Web API 1. In this I have implemented all the basic HTTP Verbs(Http methods):

* GetAll()
* GetById()
* Post()
* Put()
* Delete()

For Post method use following Example:

{

"Person\_Name": "Anita",

"Person\_Age": 23,

"Person\_Occupation": "Engineer",

"Person\_Mail": "[anita@gmail.com](mailto:anita@gmail.com)"

}

**2) WebAPI2Controller:**

This is the same controller as above but in this I have use the return type of all method as IHttpActionResult instead of HttpResponseMessage. IHttpActionResult which is introduced in Web API 2. Benefits of using IHttpActionResult instead of HttpResponseMessage are :

* The code is cleaner and easier to read.
* Unit testing Controller action is much simpler Web API 2.

3)StudentController:

StudentController is an example of Attribute Routing. In this Controller, I have implemented the features of Routing.

* Here I used Attribute Routing by using [Route] attribute. Attribute routing gives more control over the URIs than conventional based routing.
* Creating URI patterns like hierarchies of resources is very difficult with conventional based routing.
* Here I used [RoutePrefix] attribute to specify the "Common Route Prefix" at the controller level to eliminate the need to repeat that common route prefix on every controller action method.
* Here I also used Attribute Routing Constraint.

**4)GenericController:**

In this Controller, I have used Generic repository to perform CRUD operation in EF. Generic Repository and its interface are present in Generic Repository Folder of the project. Generic Repository is useful because we don’t have to implement the same CRUD operation for each entity in the project. It will avoid the repetition of code and hence improve code quality and save time.

**5) CustomerUsingGenericModelController:**

In this Controller, I have used the reference of Custom Generic Model which will return the five properties:

* HttpStatusCode
* Status
* Message
* Entity
* Exception

Each method of this controller will return this five properties.

**6) UsingOptimizeGenericModelController:**

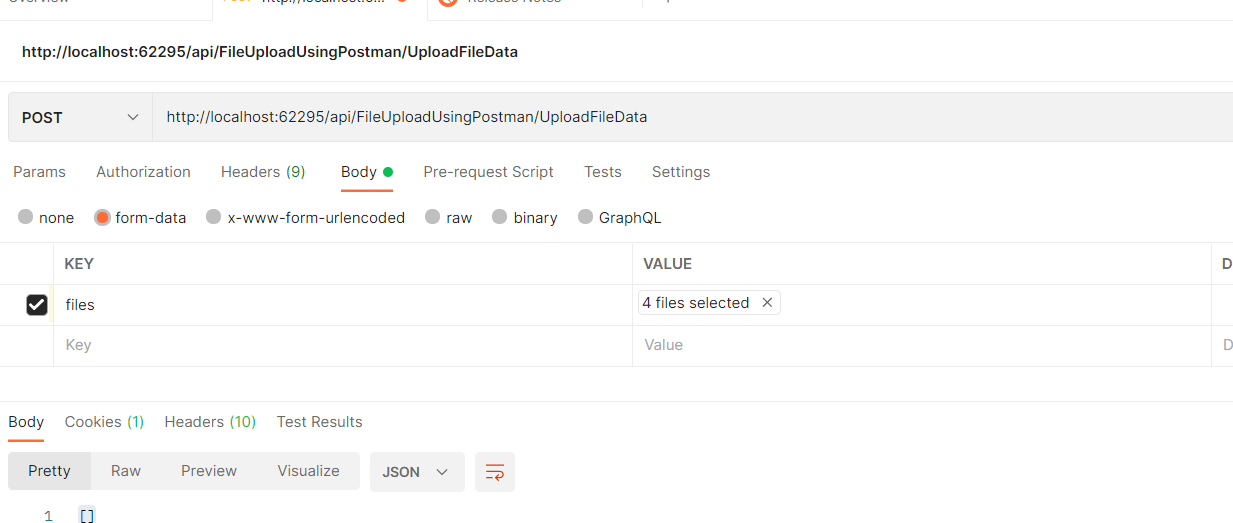
This controller works same as above controller but in this I have used the optimize Generic Model as a reference which will also return the above five properties.

**7) FileUploadController:**

This controller will take one or multiple files of any type from user and upload in the Document folder of the project.

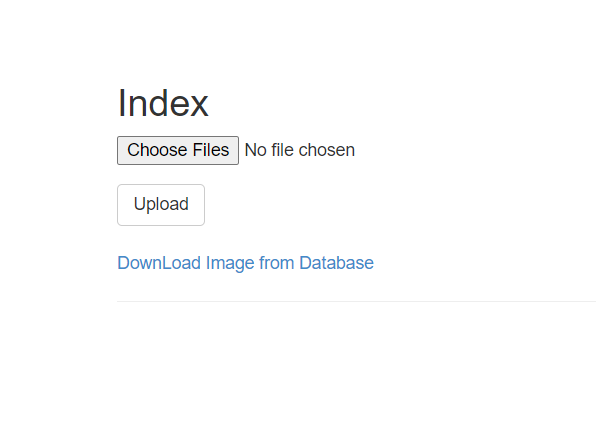
**8) FileUploadUsingPostmanController:**

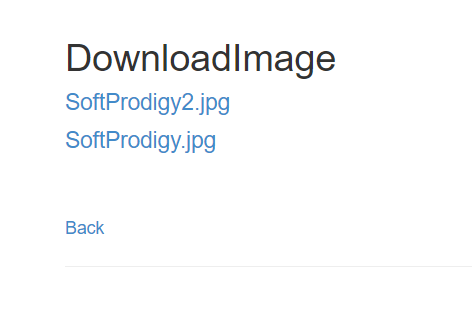
This controller will take one or multiple files from Postman and upload in the Document folder of the project.



**9) FileUploadInDatabaseController:**

This is a MVC controller which will take one or multiple file from user and store it in the Datafile Table of Intern\_DB database in the binary form and then it provides one link of Download Image from Database name after clicking it shows the list of all the file name present in database table Datafile.





After clicking on this file name URL, it will download the specific file on the basis of id.

**10) FileUploadUsingHttpPostedFileBaseController:**

This MVC Controller works same as above but it is using HttpPostedFileBase object to upload file in database. Also, in this controller it will check the file length if it is greater than 8388608 than it will throw exception and also it checks the file type, if it other than .jpg, .jpeg, .png than it will throw exception.

**11) AuthenticationTokenController:**

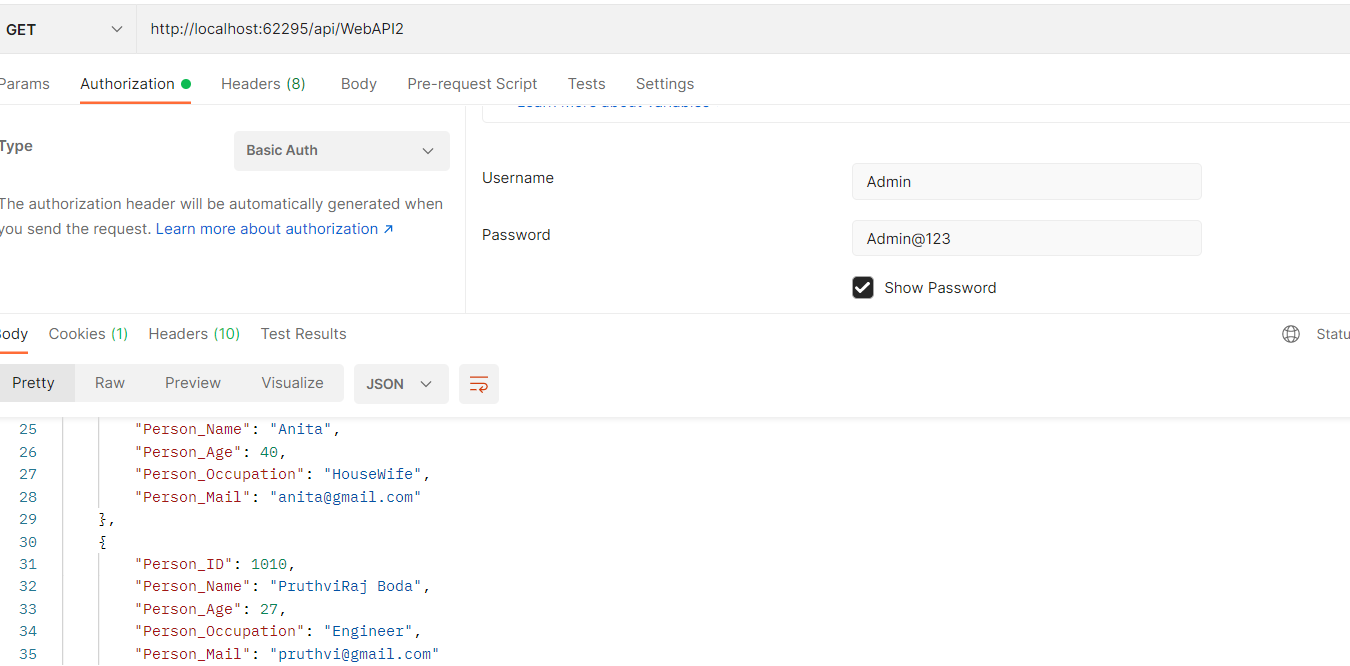
This is an MVC controller which will only returns some views which are related to Authentication Token.

Now For Authentication I have used two approaches

1. Basic Authentication
2. Bearer Authentication (token-based authentication)

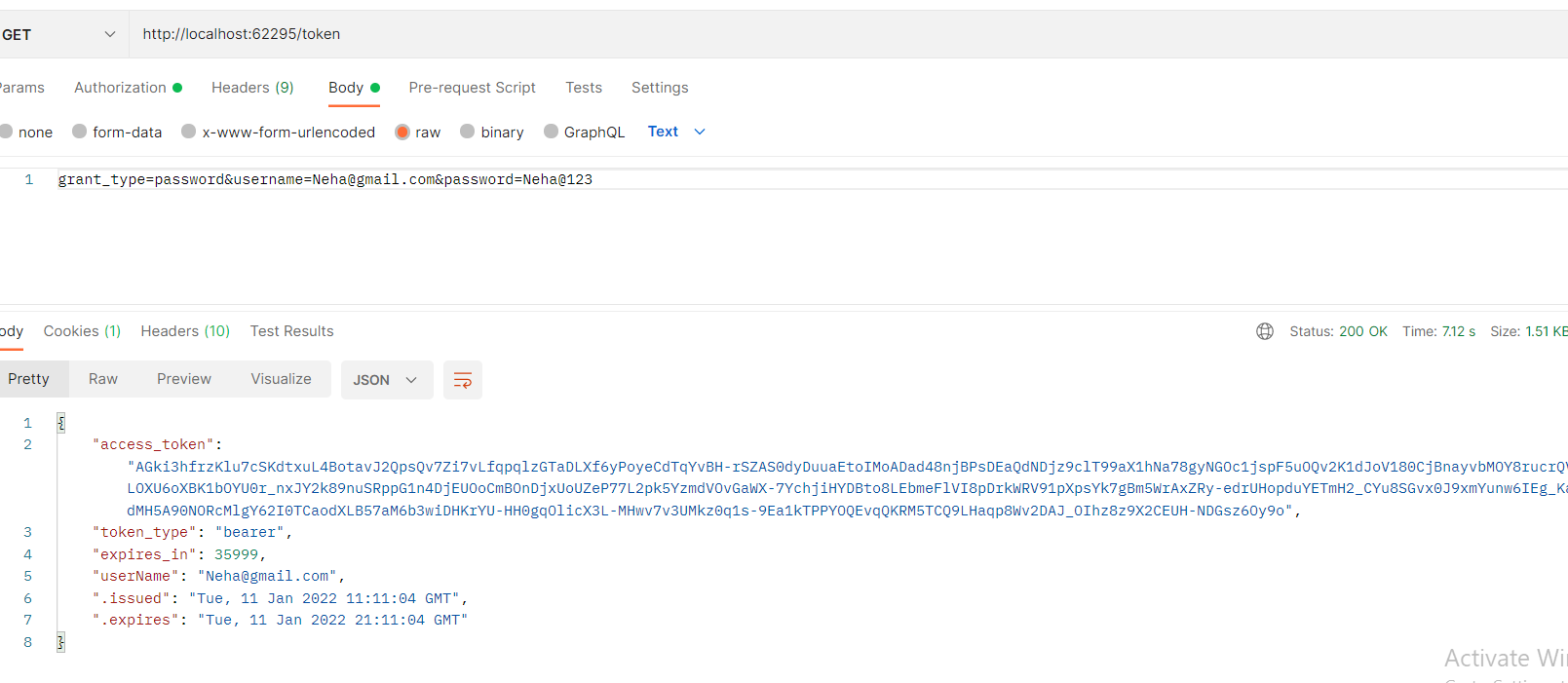
1)For Basic Authentication, I have create one custom filter class in Models folder of project with name BasicAuthenticationFilter. This filter first checks whether the Authorization in Header is null or not. If it is null then it will show unauthorized. If not then it will check matching record of username and password in the LoginDetails Table of Intern\_DB Database.

I have applied this [BasicAuthenticationFilter] filter attribute to the GetAllCustomer() method of WebAPI2Controller. So, to access this method it is necessary to authenticate user by providing username and password.

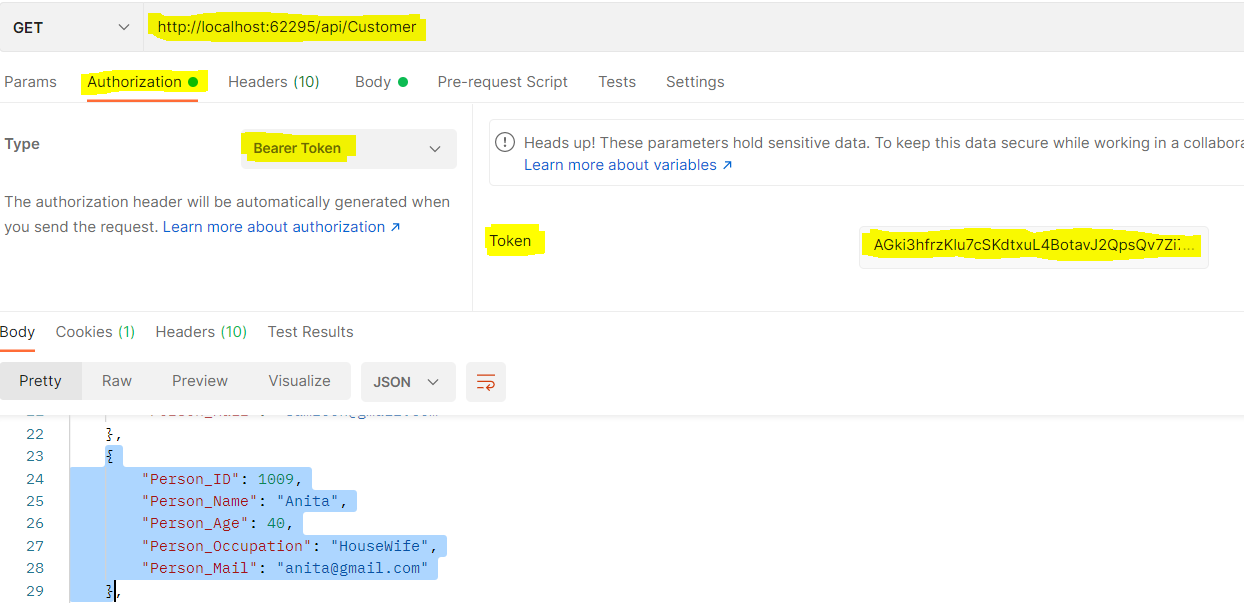


2)For Bearer or token-based Authentication, there are two ways to use this authentication:

* Through Browser, we will first register user then after registration, we have to login, after login it will create token and store it in local Storage. After login, it will check if token is present or not and if it is present then it will show Customer data on the button click and after logout it will remove access token from local storage so that user cannot access data again without login.
* Through Postman, we have to first generate token in the postman.



After getting token, we will use that token and access the Customer Data



Here, I have applied this authentication on GetAllCustomers() method of CustomerController. So, to access customer Data, we have to first authenticate user by using access token.

To enable Cross – Domain to share our resources, we will use the cors package. For that we have to first install,

Package – Microsoft.AspNet.WebApi.Cors

After installing, do the following configuration in WbApiConfig.cs file:

