<https://github.com/nehaasagaonkar/ContactManagementApi/tree/Contact_Management_Api>

This is WebApi solution to manage contact information. There are following projects in the solution:

**Contact.DB**- This class library includes local database for contact management. It has one table named ContactInformation table. Table has following fields-

1. Id- Primary key- Auto incremented
2. FirstName
3. LastName
4. Email
5. Phone
6. Status
7. Created\_date
8. Updated\_date

**Contact.Data**- This class library includes Entity Framework through which application interacts with database.

**Contact\_Model**- This class library has view models and logic of maintaining exception logs.

There are two view models- User and UserConfirmation. The view model “User” represents actual user.

UserConfirmation view model is used to return the result.

There is ApplicationLogger which has static class LogExceptions. The “write” method of LogExceptions class writes exceptions to text file. This log file is stored at the path specified in the web.config file.

<add key="Err\_File\_Path" value="D:\ExceptionLog\Log.txt"/>

**Contact.API**- This is API and needs to be set as startup project. Following are the main things:

1. **Dependancy injection**- This is implemented by using Unity.WebApi. Types are registered in UnityConfig.cs file.

In Startup.cs file DependencyResolver is configured.

1. **Token based authentication**- Owin’s authentication is used for token based authentication.

TokenAuthorizationProvider.cs file is present in the folder named Provider. In that file, user name and password is set. Also, application is configured to allow cross origin.

In the Startup.cs file OAuth is configured. “/token” URL generates the token.

1. **API controller**- ContactsController has the action methods mapped to HTTP verbs to maintain contact information such as GET, POST, PATCH and DELETE. All method have “api” prefix in the URL so the RoutePrefix is added at controller level. All the methods except GET are authorized. So, for GET method, [AllowAnonymous] attribute is added.

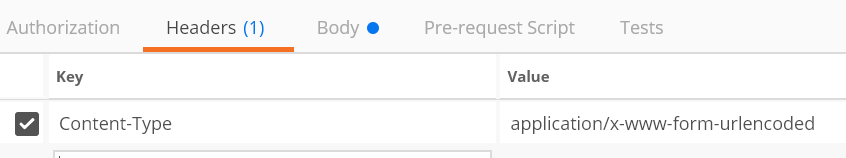
Below is the functionality-

Postman application is used to browse the URLs.

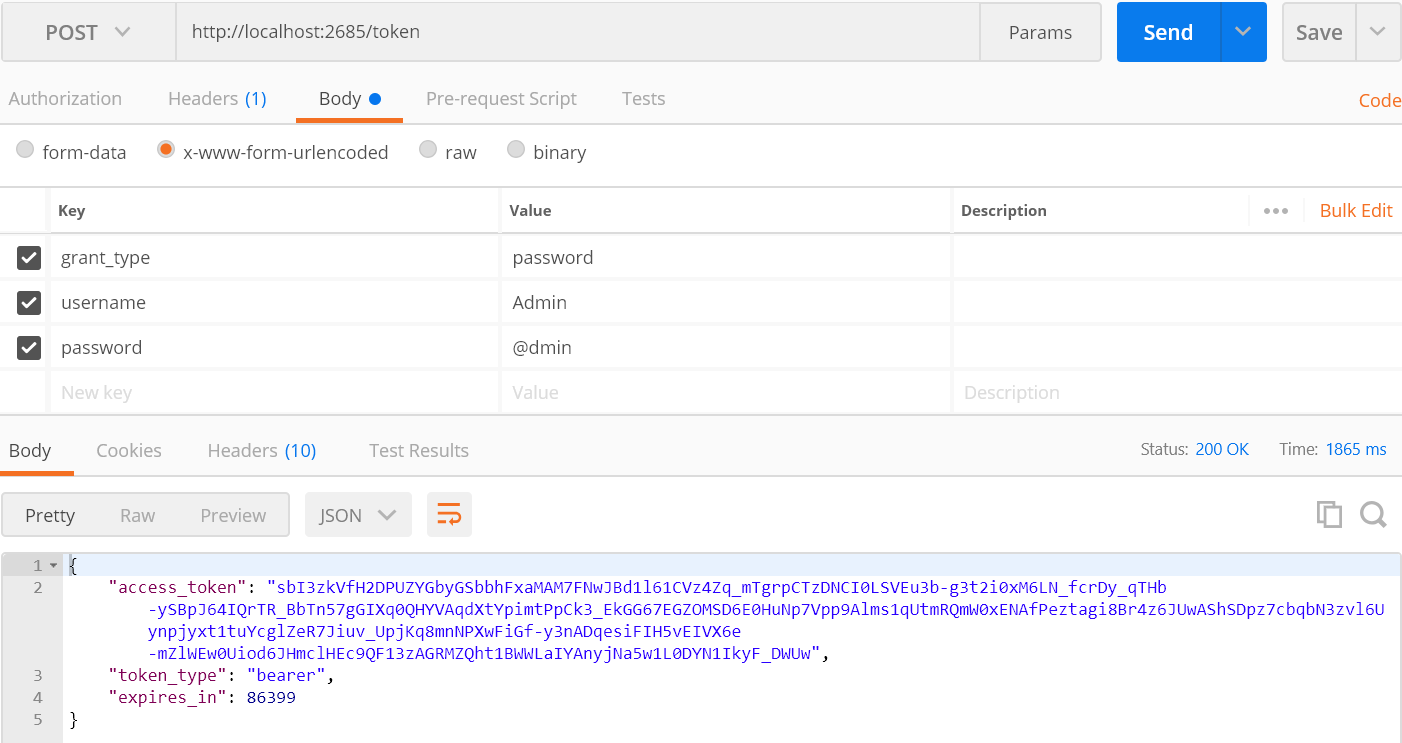
1. Generate a token-

URL- <http://localhost:2685/token>

Header-

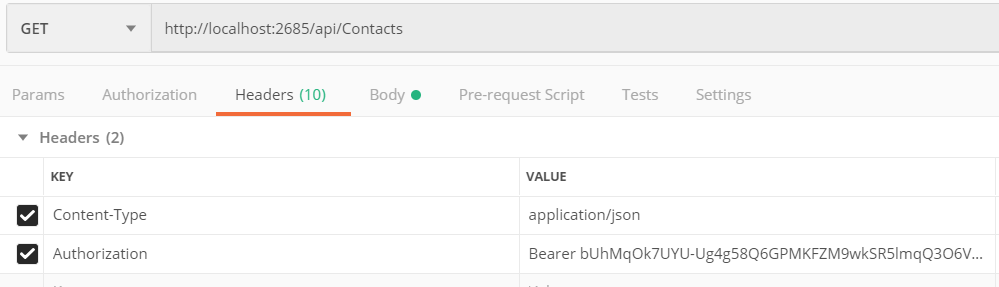


OutPut



1. List contacts-

**GET**- [http://localhost:2685/api/](http://localhost:2685/api/list)Contacts



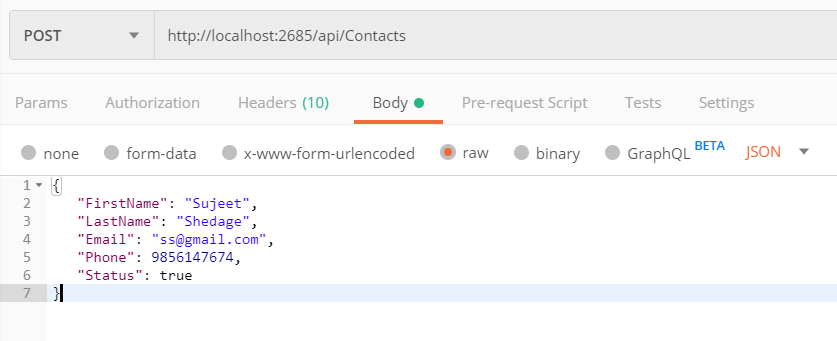
Output-



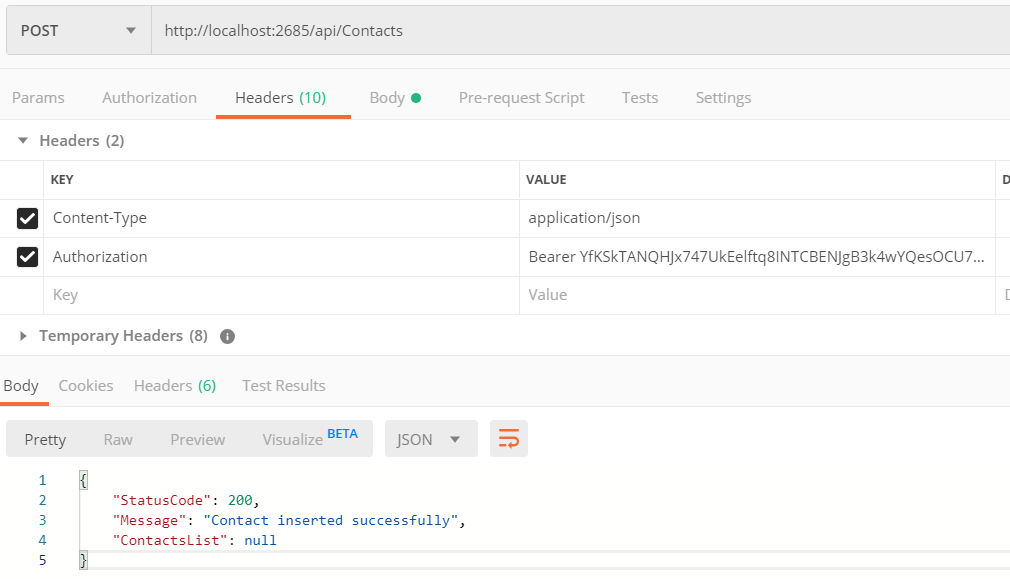
1. Add a contact

**POST**- [http://localhost:2685/api/ Contacts](http://localhost:2685/api/add)

Input Json-



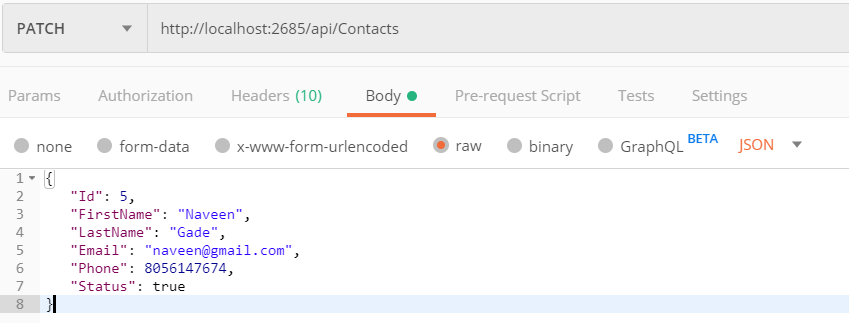
Output-



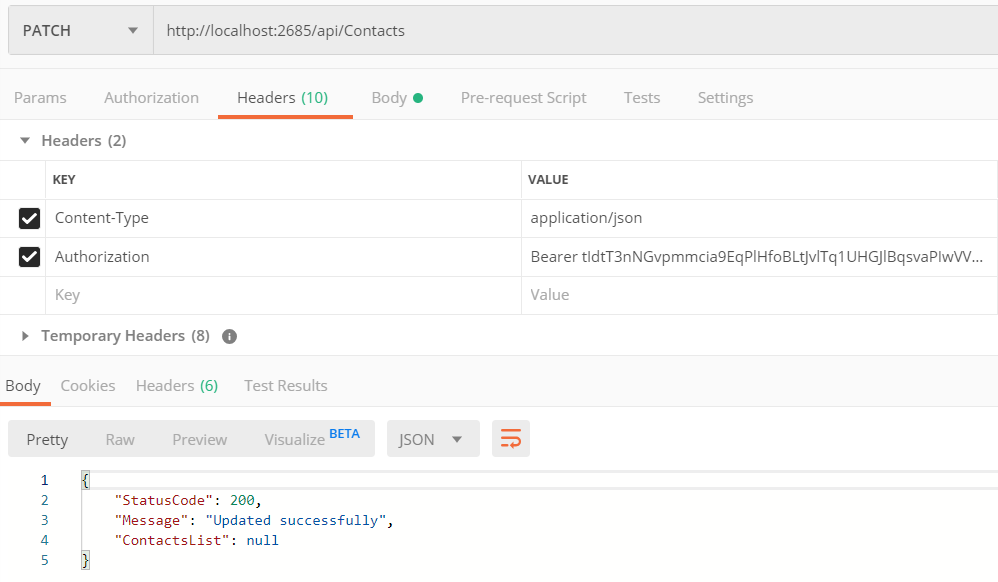
1. Edit contact

**PATCH**- <http://localhost:2685/api/Contacts>

Input JSON



Output-



1. Inactivate a contact

DELETE- <http://localhost:2685/api/Contacts>

