In this video we will discuss **customizing and using asp.net identity server with Web API**. This is continuation to [Part 21](http://csharp-video-tutorials.blogspot.com/2016/11/aspnet-web-api-user-registration.html). Please watch [Part 21](http://csharp-video-tutorials.blogspot.com/2016/11/aspnet-web-api-user-registration.html) from [ASP.NET Web API tutorial](https://www.youtube.com/playlist?list=PL6n9fhu94yhW7yoUOGNOfHurUE6bpOO2b) before proceeding.  
  
In our previous video we have seen that the database that the identity system creates has a strange name.   
aspnet-EmployeeService-20161122014700.mdf   
  
  
  
First let us understand the different tables that we have in the database and their purpose

| **Table** | **Description** |
| --- | --- |
| \_\_MigrationHistory | The presence of this table tells us that it is using entity framework |
| AspNetRoles | Store roles information. We do not have any roles yet so this table is empty |
| AspNetUserClaims | We do not have claims, so this table will also be empty |
| AspNetUserLogins | This table is for third party authentication providers like Twitter, Facebook. Microsoft etc. Information about those logins will be stored in this table |
| AspNetUserRoles | This is a mapping table which tells us which users are in which roles |
| AspNetUsers | This table stores the registered users of our application |

At this point the obvious question that comes to our mind is, is it possible to change the database name. If so, how can we do it?  
  
The **DefaultConnection**string in web.config controls the name of the generated database. To change the name of the generated database change the **DefaultConnection**string in web.config.   
  
For example, if you want to name the database - **UsersDB**, change the DefaultConnection string as shown below

<add name="DefaultConnection"

  connectionString="Data Source=(LocalDb)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\UsersDB.mdf;Initial Catalog=UsersDB;Integrated Security=True"

  providerName="System.Data.SqlClient" />

Save the changes and when you run the application, asp.net identity system will create a database with name **UsersDB**. But at this point the database is still created in **App\_Data** folder. Another question that comes to our mind is can we create this database in SQL Server.  
  
The answer is, YES we can. To create the database in SQL Server, change the **DefaultConnection**string in web.config to point to your SQL Server. Notice I have changed   
Data Source value from (LocalDb)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\UsersDB.mdf to (local). You can also use . (DOT) instead of the word (local).  
  
With the above change, the connection string in web.config is as shown below

<add name="DefaultConnection"

      connectionString="Data Source=(local);Initial Catalog=UsersDB;Integrated Security=True"

      providerName="System.Data.SqlClient" />

At this point when we save the changes and run the application the database (UsersDB) is created in SQL Server instead of App\_Data folder. Make sure, you refresh the Databases folder in SQL Server Management Studio to see the newly created UsersDB database.  
  
Is it mandatory for the Identity tables to be in a separate database. Can't we have them created in an existing database.  
No it is not mandatory for the Identity tables to be in a separate database. You can have them created by Identity framework in an existing database by just making your connection string point to your existing database instead of a separate database.  
  
For example, if we want the identity tables to be created in EmployeeDB, we will change the DefaultConnection string to point to EmployeeDB as shown below.

<add name="DefaultConnection"

     connectionString="Data Source=(local);Initial Catalog=EmployeeDB;Integrated Security=True"

     providerName="System.Data.SqlClient" />

ASP.NET Identity system can be used with all of the ASP.NET frameworks, such as ASP.NET MVC, Web Forms, Web API, and SignalR. It can be used when you are building web, phone, store, or hybrid applications.  
  
In our next video we will discuss **implementing Login page**.