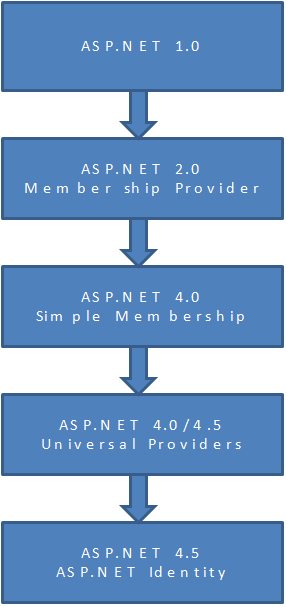
**ASP.NET Identity**

Microsoft released ASP.NET Identity as the new membership system for ASP.NET applications. Using ASP.NET Identity we can customize user profile data which otherwise was very hard in previous membership providers. We can add Login/ Logout functionality to an application very easily. ASP.NET Identity helps us to create and manage user accounts.  It enables users to self-register on the site. It allows users to select preferred user id and password. One of the most helpful features that is introduced is social media account logins, Users can now register using their social login accounts such as Facebook, google, twitter etc. ASP.NET MVC Identity uses [entity framework with Code first](http://tektutorialshub.com/entity-framework-tutorial/) approach with SQL SERVER as default database and [OWIN](http://tektutorialshub.com/owin/).

### History of ASP.NET Membership system

ASP.NET membership provider introduced initially in .Net 2.0 used forms authentication system to authenticate users to the web application that used an authentication cookie.  When a user logs in to a web application, the credentials provided by him/her are verified by querying the information from the back-end data source.  The Membership provider’s forms Authentication then creates a cookie and embeds the user credentials into it, now all the Subsequent requests coming to the application must contain this cookie which is validated by the forms authentication. If the cookie is found and is valid, then the request is granted and the user is navigated to the default home screen else he/she is presented a login page and is prompted to enter valid credentials.

ASP.NET 2.0 introduced the Membership provider. Simple Membership made its entry in ASP.NET 4.0 (made primarily for ASP MVC) followed by Universal Membership provider in ASP.NET 4/4.5, now all these Membership providers used an abstract base class to manage the user credentials. Internally they all provided security by using Forms Authentication.

[](http://tektutorialshub.com/asp-net-identity/history-of-membership-in-asp-net/)

History of Membership in ASP.NET

**Problems, Traits and limitations:**

ASP.NET Identity is latest membership technology introduced in .NET 4.5. Its purpose is to replace prevailing Membership and **SimpleMembership** systems; hence both Membership and **SimpleMembership** are now obsolete, moreover new membership system does not have backward compatibility. ASP.NET identity works only with .Net Framework 4.5 or later. If for some reason you cannot upgrade your systems to use identity compatible framework i.e. 4.5, then the existing old Membership or **SimpleMembership** systems must be used.

New ASP.NET Identity is available as NuGet package and is also included by default in Visual Studio 2013. Publishing new features as a NuGet package helps to obtain and upgrade to new versions easily in the future. **Microsoft.AspNet.Identity.Core** namespace includes ASP.NET identity.

There are many problems with old ASP.NET Membership system. But, at time when it's released it was great help, when compared with idea to customize and implement complete membership system yourself.

**Problems:**

First, ASP.NET Membership system is hard to customize. It database used is SQL SERVER which is the only database supported to store data about users. It has predefined user's information table and it is not simple to extend default functionality to add custom user fields, or to move data to other kind of database. ASP Membership database schema is designed for SQL Server DB only and it is hard to move data to other data sources, especially to databases that are non-relational.

Second, functionalities like Sign-In and Sign-Out are based on Forms Authentication, so it can't use OWIN. ASP.NET Identity membership has fixed these problems by providing a much simpler and easier way for customization user profiles, changing the data storage and also as it works with OWIN so it’s abstract now and does not depend on IIS.

The second implementation of Membership was **SimpleMembership, it** was introduced with Visual Studio 2010 SP1 and WebMatrix. One of the main advantages of **SimpleMembership** over classic Membership system is that it made customization of user profiles easier. But, the other problems such as using other kind of databases, integrating with social logins and external membership providers, working with OWIN etc., are still persisted. Also, **SimpleMembership** is made primarily for MVC and Web Forms applications still used old ASP.NET Membership system

**ASP.NET Identity new features and advantages**

ASP.NET Identity membership is latest approach of Microsoft to make membership system more efficient, filling all the gaps and putting things on right way. It solves previous problems and also supports newest demands, demands like integrating with external login provider’s ex- Facebook, Google, Twitter or Live login.

Here are few important advantages of ASP.NET Identity:

* Supports a wide range of ASP.NET applications like Web Forms, MVC, Web API, SignalR or web pages, so there is no need to learn different methods for MVC and Web Forms. One membership system now supports all kinds of ASP.NET applications.
* It’s really easy to customize the system. You can add new fields to user profile in no time.
* ASP.NET Identity is highly customizable as it is built on interfaces. If you don't like one part of system, you may just replace it with some other object.
* Entity Framework code first is used as default implementation, now since database schema is completely in your hands, you can easily customize it to change table names, primary keys, data types etc.
* ASP.NET supports wide range of Data sources, ex- SharePoint, NoSQL, Windows Azure Active Directory, Oracle, MySQL etc. Practically any other database can be used with new ASP.NET identity. By default it uses SQL Server.
* ASP.NET Identity is testable. You may run unit tests on it like the way you might have done while developing MVC applications. One of intentions which MVC framework constituted when it was introduced was to enable unit testing in web applications. But, ASP.NET **SimpleMembership** still could not be unit tested. ASP.NET Identity supports Unit testing and is testable. ASP.NET web application with all of its parts including membership system is unit testable now.
* In addition to Roles ASP.NET identity supports Claims for more efficient authorization. Old membership systems only supported Role based authorization. We know that by using Roles we may allow or constraint anonymous users to gain access to different parts/ modules of web application. ASP.NET Identity supports Claims based security along with roles. Roles are like subset of Claims. Along with user roles, information like name, email, phone, user groups pertaining to a user can be used as Claims. So using Claims you may access criteria now can be implemented in different ways that would practically makes sense for you (e.g. allow access only to those users that are part of Sales team). Like everything else Claims in ASP.NET Identity are separate. Authorization, roles, Authentication and claims are all separated for an efficient integration and easier customization.   
    
  So why ASP.NET Identity offers both Claims and Roles based security? Why do we need Roles at all? Why not to use Claims only? The answer would be that Roles are simpler and defined than Claims and not every application needs Claims based security. If you are familiar with Roles and you think that they alone could work well with your requirements, it would be OK to use them.
* ASP.NET Supports external login, Users may now login with their existing social media logins like Facebook, Google, Twitter or Microsoft Live. Almost all the people around the globe use social networking sites and already have accounts on these sites. Majority of people don’t want to create separate accounts on every site and remember login details separately for each one of it. Moreover there can be a probability that the user is not sure whether he really wants to try your website, so it becomes imperative to consider this as a decisive reason whether to allow the users to login from social networks or not .Allowing the users to login through their existing account would be much faster and easier to look inside your website and see what it has to offer.
* ASP.NET Identity doesn't depend upon System.Web namespace that is shipped as a consolidated package with all features. It works with OWIN, Old Membership system was a part of System.Web namespace and used Forms Authenticatio. ASP.NET Identity uses OWIN CookieAuthentication to generate login cookie and it comes as a separate package.