WS-Federation, PKI, WIF, ADFS.

**ADFS** (Active Directory Federation Services): by Microsoft relies on AD for authentication. Can be used in active (SOAP web services) or passive (web sites) scenarios and supports SAML tokens, WS-Federation, WS-Trust and SAML-Protocol. It can be used as an Identity Provider (against AD) or as a Federation Provider.

**WS-Federation** (Web Services Federation): is a protocol that can be used to negotiate the issuance of a token.

**OpenID** Connect is not OpenID 1 or OpenID 2 (both previous versions are deprecated).

OpenID Connect is an identity layer on top of OAuth.

**OAuth** (Open Authorization) is an open standard for token-based authentication and authorization on the Internet.

OAuth 1.0 and OAuth 2.0 allows an end user's account information to be used by third-party services, such as Amazon, Google, Facebook, Microsoft, LinkedIn and Twitter, without exposing the user's password. OAuth acts as an intermediary on behalf of the end user, providing the service with an access token that authorizes specific account information to be shared. The process for obtaining the token is called a flow.

public OpenID Connect providers are Google, Microsoft, Yahoo,PayPal,SalesForce all use JSON.

**OAuth 2.0** include new flows, simplified signatures and short-lived tokens with long-lived authorizations.

in OAuth 1.0, desktop applications or mobile phone applications had to direct the user to open their browser to the desired service, authenticate with the service, and copy the token from the service back to the application. The main criticism here is against the user experience. With OAuth 2.0, there are now new ways for an application to get authorization for a user.

**IdentityServer** is an open source OpenID Connect Provider and OAuth 2.0

<https://docs.microsoft.com/en-us/aspnet/core/security/authentication/social/?view=aspnetcore-2.2>

<https://docs.microsoft.com/en-us/aspnet/mvc/overview/security/create-an-aspnet-mvc-5-app-with-facebook-and-google-oauth2-and-openid-sign-on>

**Differences between JWT and OAuth authentication**?

OAuth 2.0 defines a protocol, i.e. specifies how tokens are transferred, JWT defines a token format.

JWT authentication is not a standard and does not specify how the Client obtains the token in the first place.

OAuth is an authorization protocol that can use JWT as a token.

JWT (JSON Web Tokens) tokens are JSON encoded data structures contains information about issuer, subject (claims), expiration time etc. It is signed for tamper proof and authenticity.

**SSO** is what allows you to sign into Gmail and switch over to Google Calendar or YouTube without typing your password in again.

**How SAML Works**:

SAML SSO works by transferring the user’s identity from one place (the identity provider) to another (the service provider). This is done through an exchange of digitally signed XML documents.

Consider the following scenario: A user is logged into a system that acts as an identity provider. The user wants to log in to a remote application, such as a support or accounting application (the service provider). The following happens:

1. The user accesses the remote application using a link on an intranet, or similar and the application loads.
2. The application identifies the user’s origin (by application subdomain, user IP address, or similar) and redirects the user back to the identity provider, asking for authentication. This is the authentication request.
3. The user either has an existing active browser session with the identity provider or establishes one by logging into the identity provider.
4. The identity provider builds the authentication response in the form of an XML-document containing the user’s username or email address, signs it using an X.509 certificate, and posts this information to the service provider.
5. The service provider, which already knows the identity provider and has a certificate fingerprint, retrieves the authentication response and validates it using the certificate fingerprint.
6. The identity of the user is established and the user is provided with app access.

Service Provider (SP) – This is your application, which will ask an IdP for authentication information when a user tries to log in.

Identity Provider (IdP) – The service that stores the user’s actual credentials – such as Salesforce, OneLogin, or an open-source system like Shibboleth.

To implement SAML with C#, **ComponentSpace** and **ComponentPro’s** Ultimate SAML are two popular commercial SAML components/ libraries available for ASP.NET.

Or Sustainsys/Saml2 is a free open source SAML library for C#. <https://saml2.sustainsys.com/en/2.0/> (Open Source SAML Service Provider Libraries)

**Katana vs OWIN**:

OWIN (Open Web Interface for .NET) itself does not have any tools or libraries. It is just a specification (collection of best practices).

Katana is project name (.NET library, you can get using Nuget) to implement OWIN in Microsoft.

At the moment, Katana has successfully adapted Web API, ASP.NET Identity and Signal R etc ASP.NET frameworks to OWIN.

It enables you to run the same application on all the web servers that support OWIN other than IIS.

ASP.NET Core supports the Open Web Interface for .NET (OWIN) and can use webserver like Kestrell.