برای پیاده سازی احراز هویت با JWT در ASP.Net MVC مراحل زیر را انجام می دهیم:

1- ابتدا دستور زیر را در Package Manager Console اجرا میکنیم **Install-Package System.IdentityModel.Tokens.Jwt**

2- کلاسی با نام دلخواه از نوع internal در root پروژه جهت انجام کانفیگ jwt ایجاد کرده و این کلاس از DelegatingHandler جهت کانفیگ سیستم ارث بری میکند. محتویات کلاس نام برده در زیر قابل مشاهده است.

```
using System;
using System.Collections.Generic;
using System.IdentityModel.Tokens.Jwt;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Threading;
using System.Threading.Tasks;
using System.Web;
using Microsoft.IdentityModel.Tokens;

namespace JWT_Pharmacy
{
    internal class TokenHandler : DelegatingHandler
    {
        private static bool TryRetrieveToken(HttpRequestMessage request, out string
token)
        {
            token = null;
            IEnumerable<string> authzHeaders;
            if (!request.Headers.TryGetValues("Authorization", out authzHeaders) ||
authzHeaders.Count() > 1)
            {
                return false;
            }
            var bearerToken = authzHeaders.ElementAt(0);
            token = bearerToken.StartsWith("Bearer ") ? bearerToken.Substring(7) :
bearerToken;
            return true;
        }

        protected override Task<HttpResponseMessage> SendAsync(HttpRequestMessage
request, CancellationToken cancellationToken)
        {
            HttpStatusCode statusCode;
            string token;
            if (!TryRetrieveToken(request, out token))
            {
                statusCode = HttpStatusCode.Unauthorized;

                return base.SendAsync(request, cancellationToken);
            }
```

```
            try
            {
                //امنیتی کلید
                const string sec =
"401b09eab3c013d4ca54922bb802bec8fd5318192b0a75f201d8b3727429090fb337591abd3e44453b954555
b7a0812e1081c39b740293f765eae731f5a65ed1";
                var now = DateTime.UtcNow;
                var securityKey = new
Microsoft.IdentityModel.Tokens.SymmetricSecurityKey(System.Text.Encoding.Default.GetBytes
(sec));


                SecurityToken securityToken;
                JwtSecurityTokenHandler handler = new JwtSecurityTokenHandler();
                TokenValidationParameters validationParameters = new
TokenValidationParameters()
                {
                    ValidAudience = "http://localhost:50191",
                    ValidIssuer = "http://localhost:50191",
                    ValidateLifetime = true,
                    ValidateIssuerSigningKey = true,
                    LifetimeValidator = this.LifetimeValidator,
                    IssuerSigningKey = securityKey
                };

                Thread.CurrentPrincipal = handler.ValidateToken(token,
validationParameters, out securityToken);
                HttpContext.Current.User = handler.ValidateToken(token,
validationParameters, out securityToken);

                return base.SendAsync(request, cancellationToken);
            }
            catch (SecurityTokenValidationException e)
            {
                statusCode = HttpStatusCode.Unauthorized;
            }
            catch (Exception ex)
            {
                statusCode = HttpStatusCode.InternalServerError;
            }
            return Task<HttpResponseMessage>.Factory.StartNew(() => new
HttpResponseMessage(statusCode) { });
        }
        //خیر یا شده منقضی توکن که مورد این بررسی جهت
        public bool LifetimeValidator(DateTime? notBefore, DateTime? expires,
SecurityToken securityToken, TokenValidationParameters validationParameters)
        {
            if (expires != null)
            {
                if (DateTime.UtcNow < expires) return true;
            }
            return false;
        }

    }
}
```

3- تابع createToken را طبق زیر در محل دلخواه نوشته و در سرویس لاگین خود، پس از احراز هویت کاربر بوسیله آن توکن را میسازیم:

```
private string createToken(string username)
        {
            DateTime issuedAt = DateTime.UtcNow;
            DateTime expires = DateTime.UtcNow.AddDays(7);

            var tokenHandler = new JwtSecurityTokenHandler();

            ClaimsIdentity claimsIdentity = new ClaimsIdentity(new[]
            {
                new Claim(ClaimTypes.Name, username)
            });

            const string sec =
"401b09eab3c013d4ca54922bb802bec8fd5318192b0a75f201d8b3727429090fb337591abd3e44453b954555
b7a0812e1081c39b740293f765eae731f5a65ed1";
            var now = DateTime.UtcNow;
            var securityKey = new
Microsoft.IdentityModel.Tokens.SymmetricSecurityKey(System.Text.Encoding.Default.GetBytes
(sec));
            var signingCredentials = new
Microsoft.IdentityModel.Tokens.SigningCredentials(securityKey,
Microsoft.IdentityModel.Tokens.SecurityAlgorithms.HmacSha256Signature);

            var token =
                (JwtSecurityToken)
                    tokenHandler.CreateJwtSecurityToken(issuer: "http://localhost:50191",
audience: "http://localhost:50191",
                        subject: claimsIdentity, notBefore: issuedAt, expires: expires,
signingCredentials: signingCredentials);
            var tokenString = tokenHandler.WriteToken(token);

            return tokenString;
      }
```

4- در بخش App_Start در کلاس WebApiConfig.cs کد زیر را در متد Registerاضافه میکنیم تا کانفیگ خود را بشناسانیم:

```
config.MessageHandlers.Add(new TokenHandler());
```

5- حال در Header تمام درخواست هایی که به اکشنی فرستاده میشود که اتریبیوت Authorize را دارند، باید توکن به فرمت (Bearer [token]) قرار بگیرد.