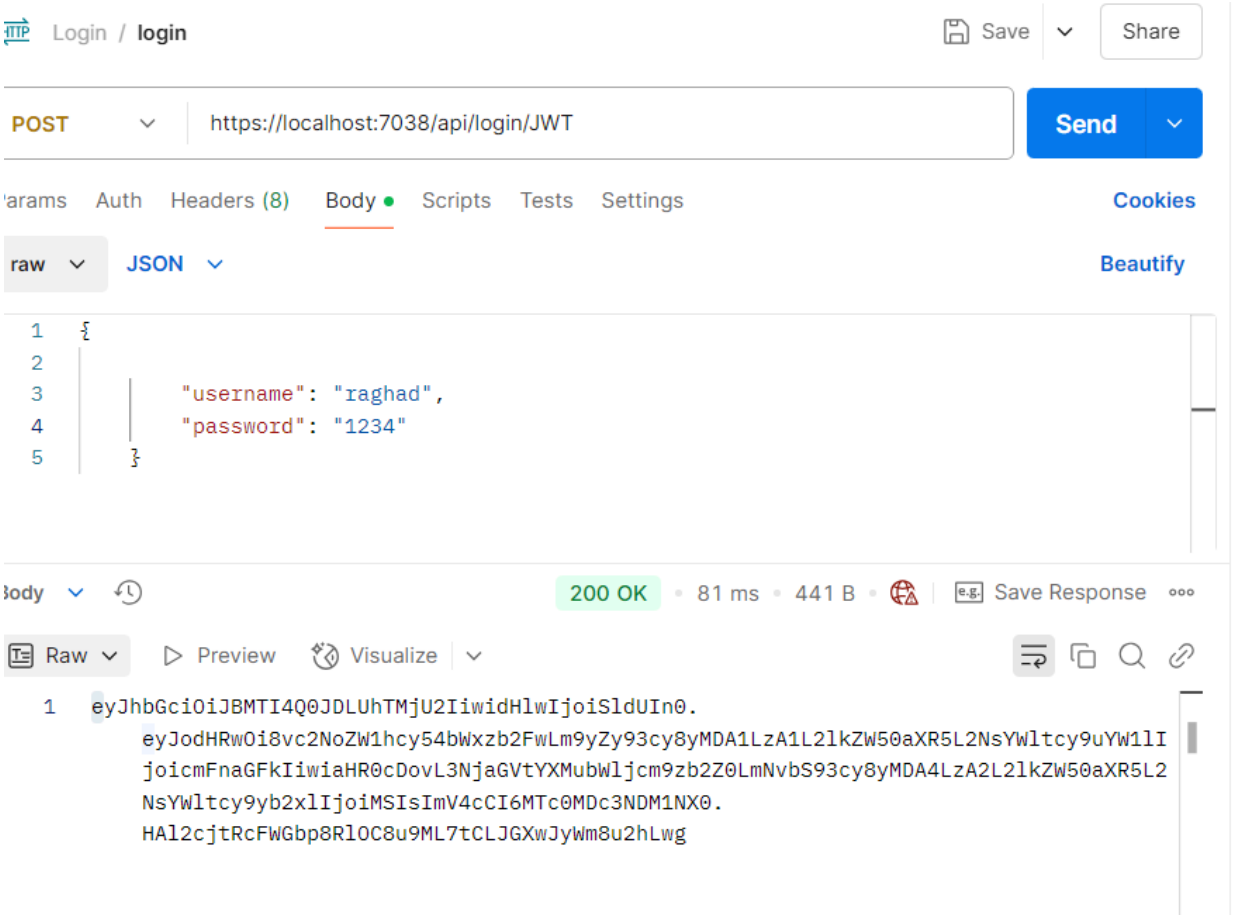
Login authentication and authorization:

```
2 references
public string Auth(Login login) {

    var result = _loginRepository.Auth(login);

    if (result == null)
    {
        return  null;
    }
    else
    {
        var secertKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("superSecretKeyDana@345"));
        var signCredential = new SigningCredentials(secertKey, SecurityAlgorithms.Aes128CbcHmacSha256);
        var claims = new List<Claim>()
        {
            new Claim(ClaimTypes.Name , result.Username),
            new Claim(ClaimTypes.Role , result.Roleid.ToString())
        };
        var tokenOption = new JwtSecurityToken (claims : claims ,
                                                expires: DateTime.Now.AddHours(24),
                                                signingCredentials : signCredential);

        var tokenAsString = new JwtSecurityTokenHandler().WriteToken(tokenOption);
        return tokenAsString;
```

```
2 references
public Login Auth(Login login)
{
    var p = new DynamicParameters();
    p.Add("user_name", login.Username, dbType: DbType.String, direction: ParameterDirection.Input);
    p.Add("pass", login.Password, dbType: DbType.String, direction: ParameterDirection.Input);
    var result = _dbConext.Connection.Query<Login>("login_package.userLogin", p,
        commandType: CommandType.StoredProcedure);
    return result.FirstOrDefault();
}
```

```
}
[HttpPost]
[Route("JWT")]
0 references
public IActionResult Auth([FromBody] Login login)
{
    var token = loginService.Auth(login);
    if (token == null) {
    return Unauthorized();
    }
    else
    {
        return Ok(token);
    }

}
```

The result:




```
namespace demo.API.Controllers
{
    2 references
    public class CheckClaimsAtt: Attribute , IAuthorizationFilter
    {
        private readonly string _claimName;
        private readonly string _claimValue;

        1 reference
        public CheckClaimsAtt(string claimName, string claimValue)
        {
            _claimName = claimName;
            _claimValue = claimValue;
        }
        0 references
        public void OnAuthorization(AuthorizationFilterContext context)
        {
            if (!context.HttpContext.User.HasClaim(_claimName, _claimValue)) {
                context.Result = new ForbidResult();
            }
        }
    }
}
```