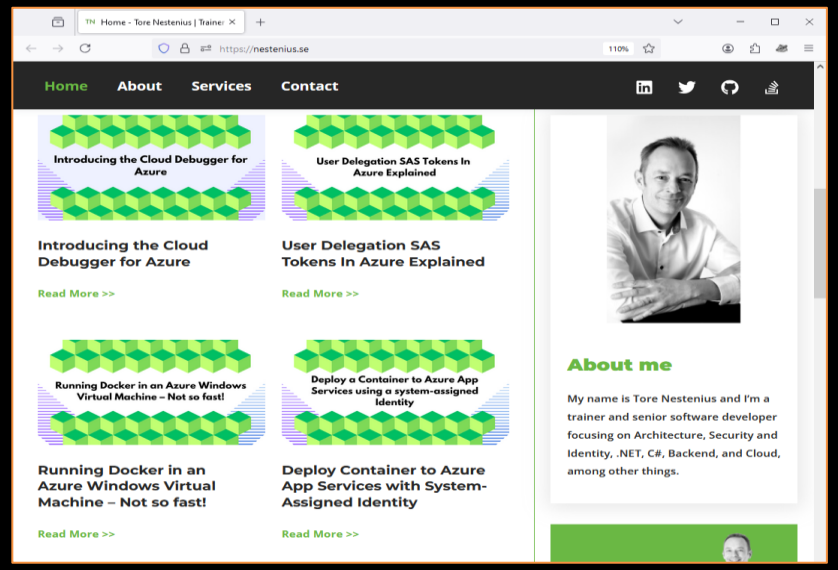
Demystifying authentication in ASP.NET Core

nestenius.se

About Tore Nestenius

Work https://tn-data.se

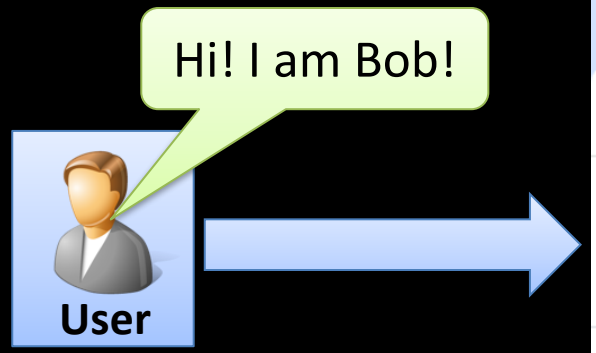


Blog https://nestenius.se

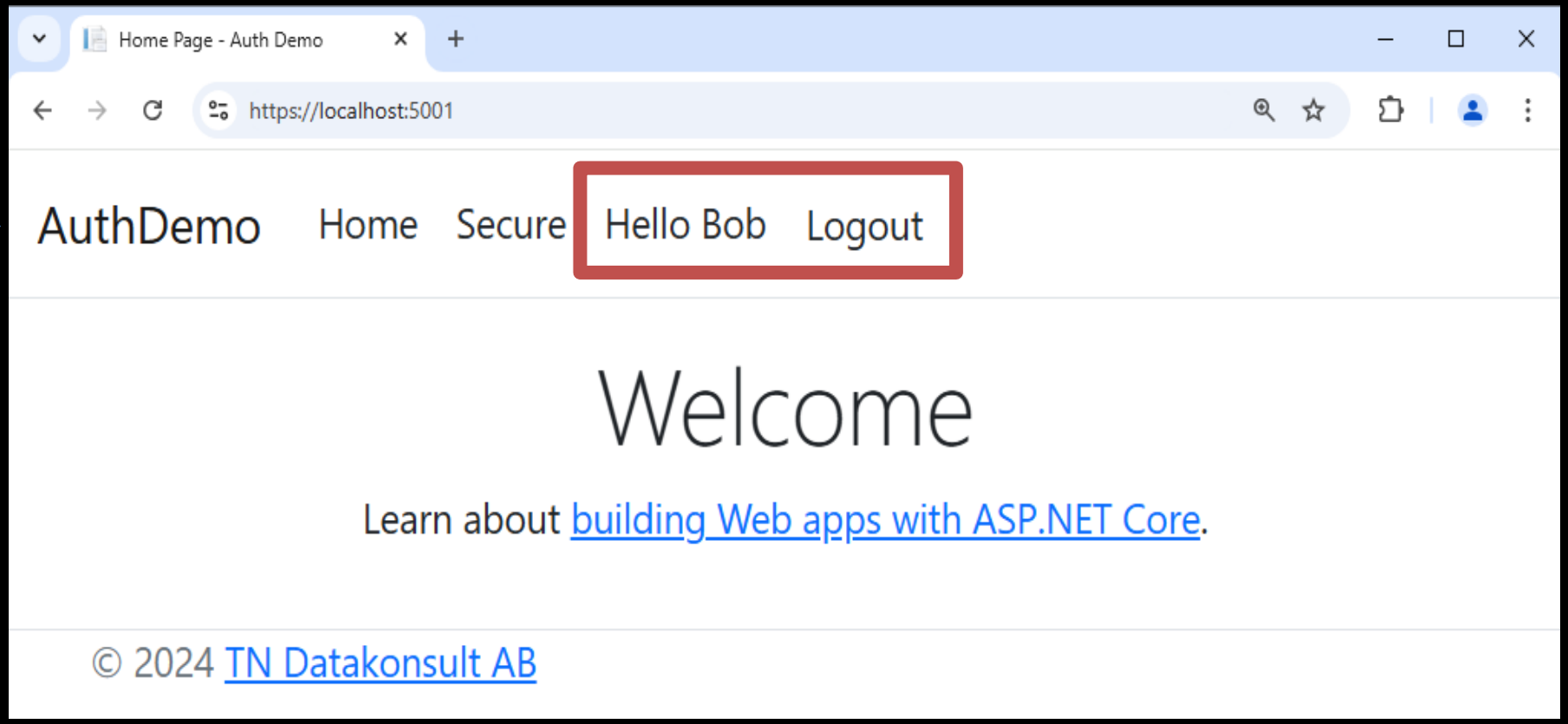
https://meetup.com/net-skane

What is the goal of this presentation?

What does a page need to support this?

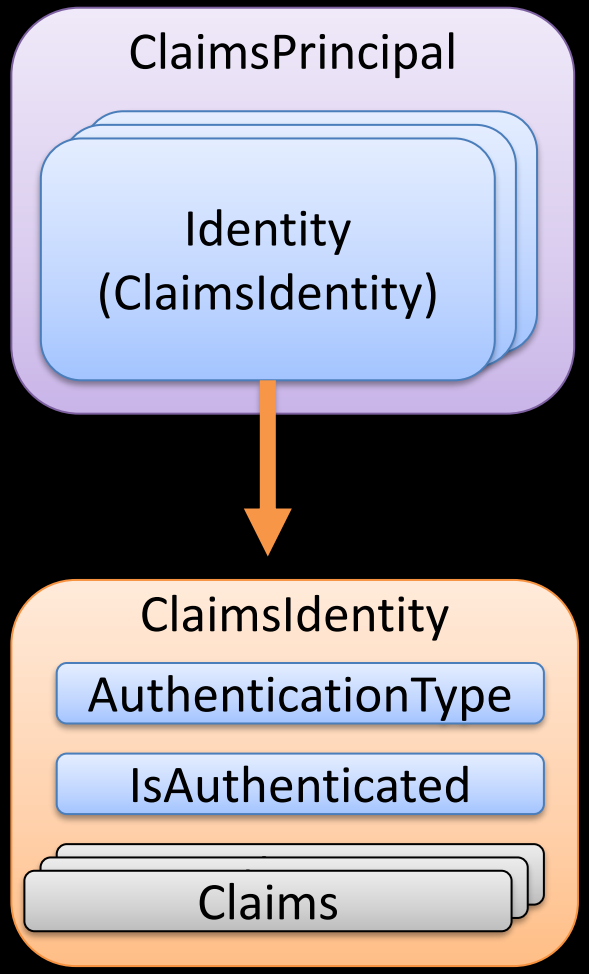


ClaimsPrincipal




```
<a href="/User/Info">
  Hello @(User?.Identity?.Name ?? "Unknown")
</a>
```

ClaimsPrincipal



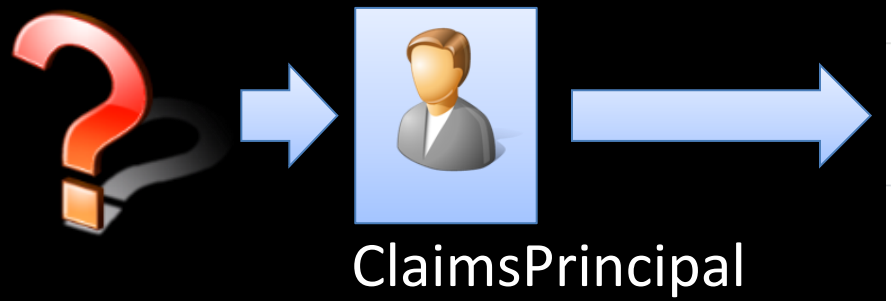

```
public class ClaimsPrincipal : IPrincipal
{
    //List of identities
    List<ClaimsIdentity> _identities = new();

     ...
}
```

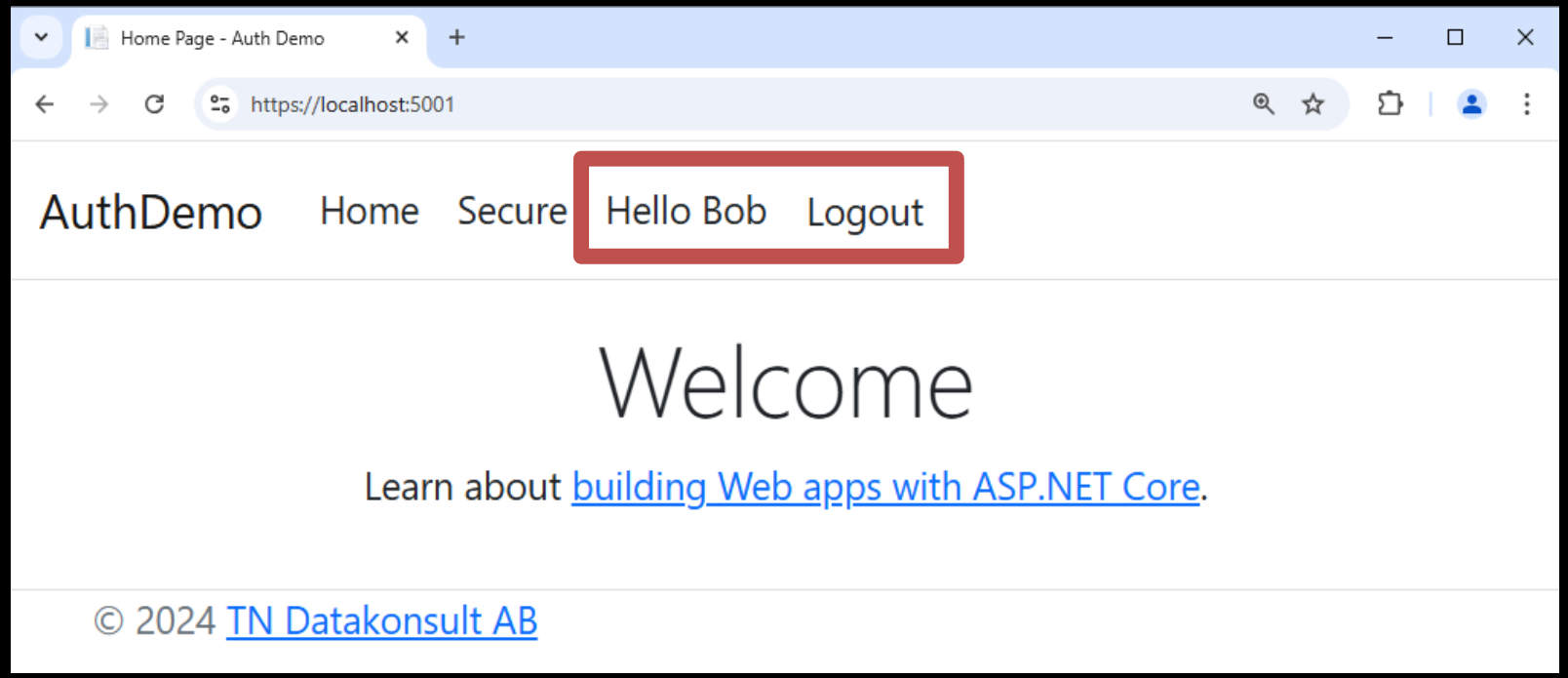
```
public class ClaimsIdentity : IIdentity
{
    //List of claims
    List<Claim> _instanceClaims = new();

    ...
}
```

Claims

```
var claims = new List<Claim>()
        {
            new("sub","1234"),
            new("name","Bob"),
            new("email","bob@tn-data.se"),
            new("role","developer")
        };
```

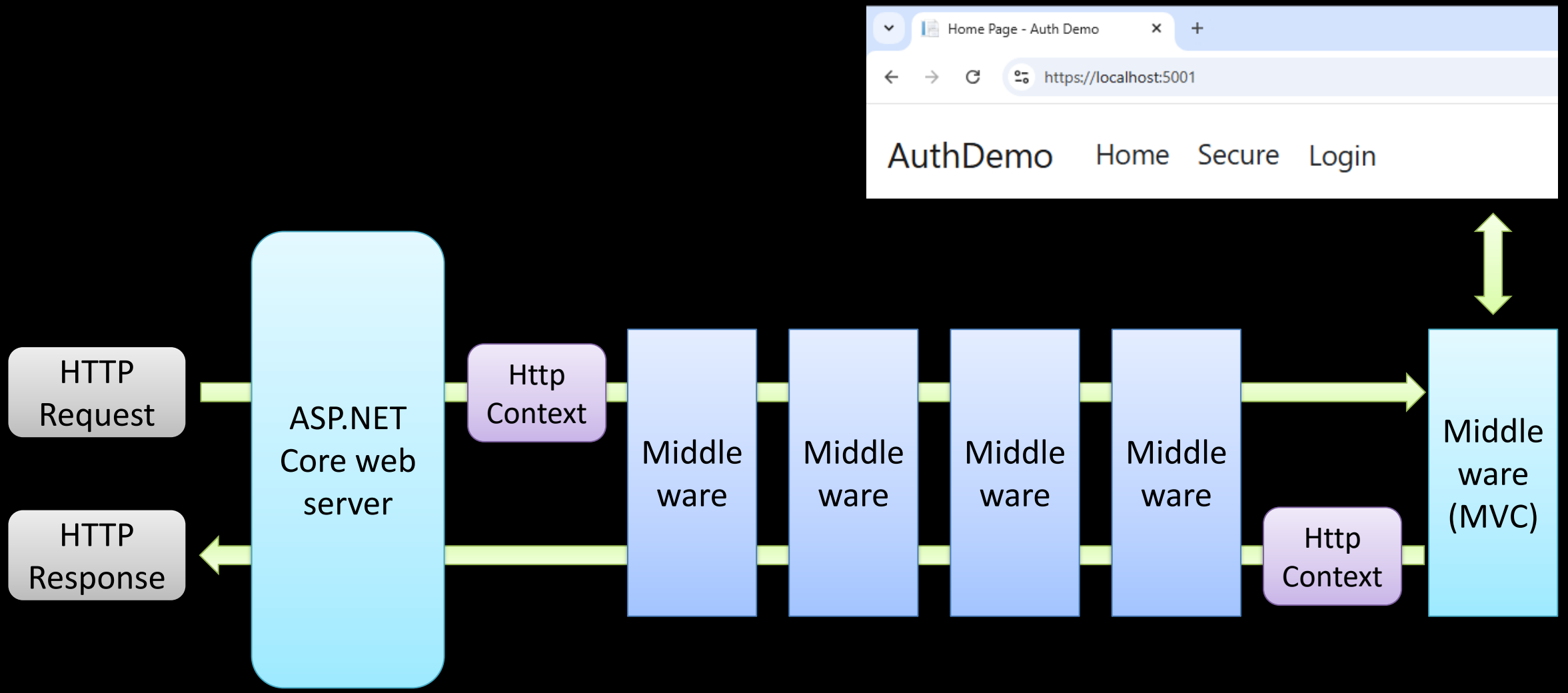
Who creates the ClaimsPrincipal?



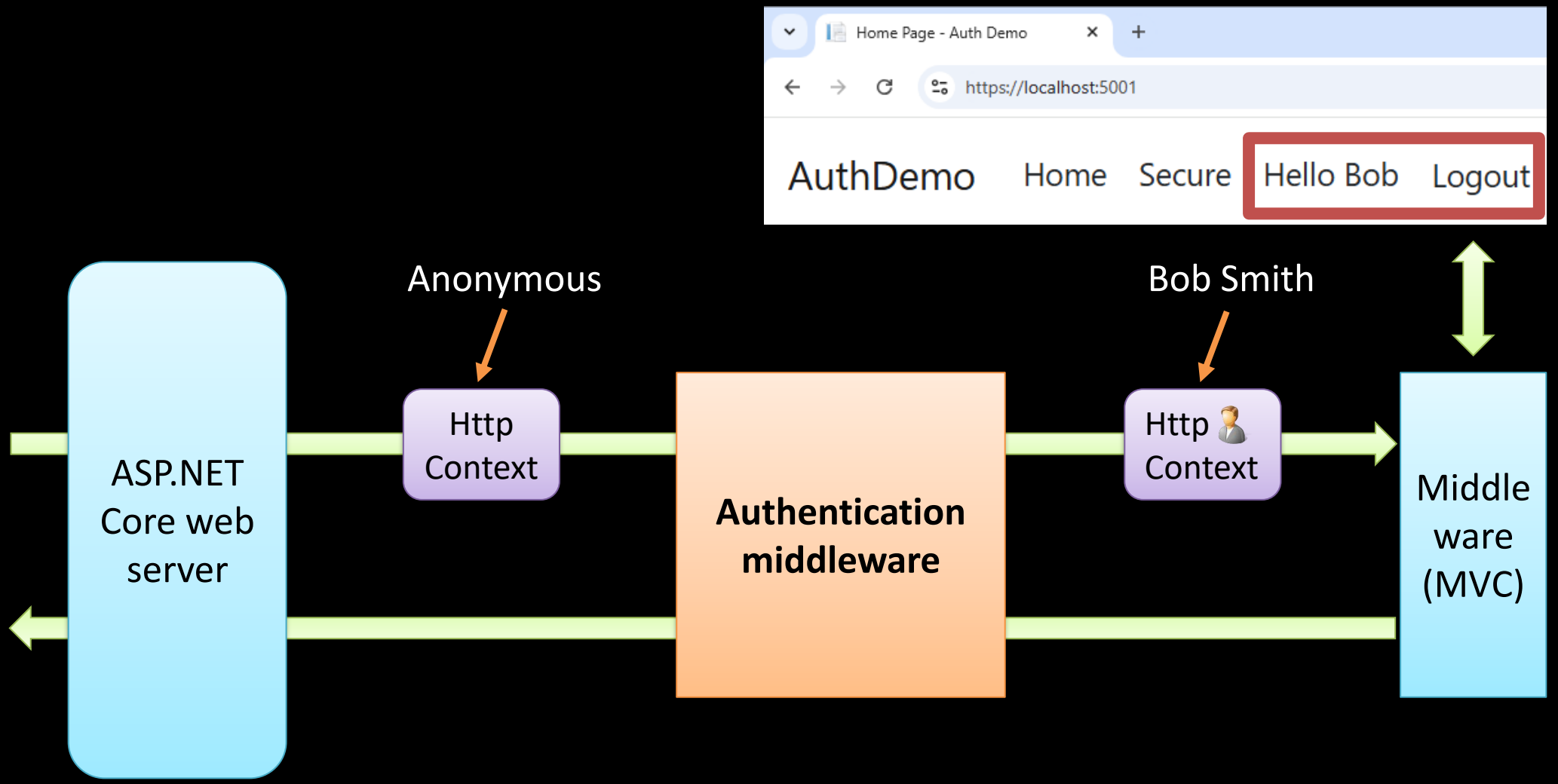


ASP.NET Core Request Pipeline

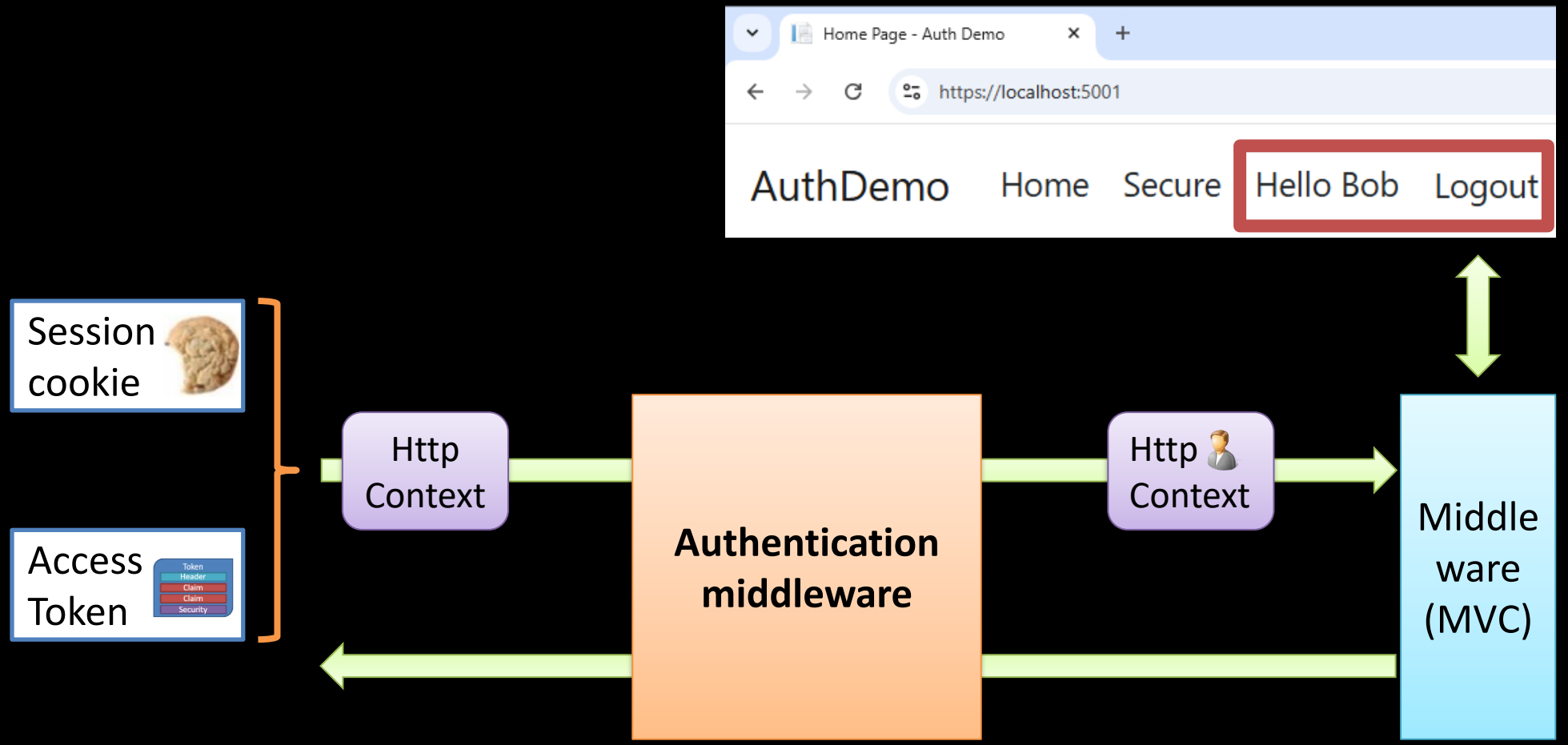
ASP.NET Core Request Pipeline



Who sets the user?



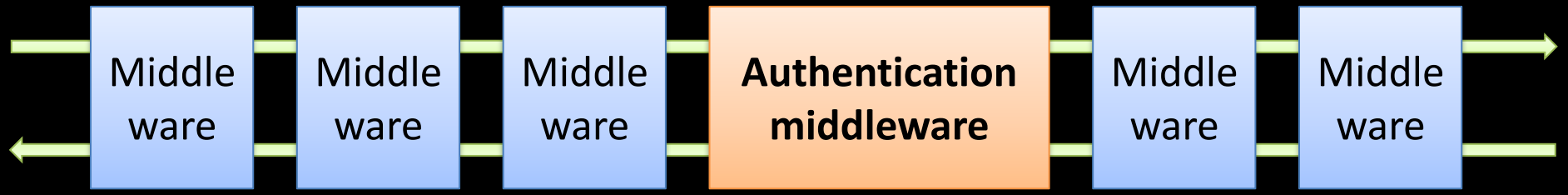
Where does the user data come from?



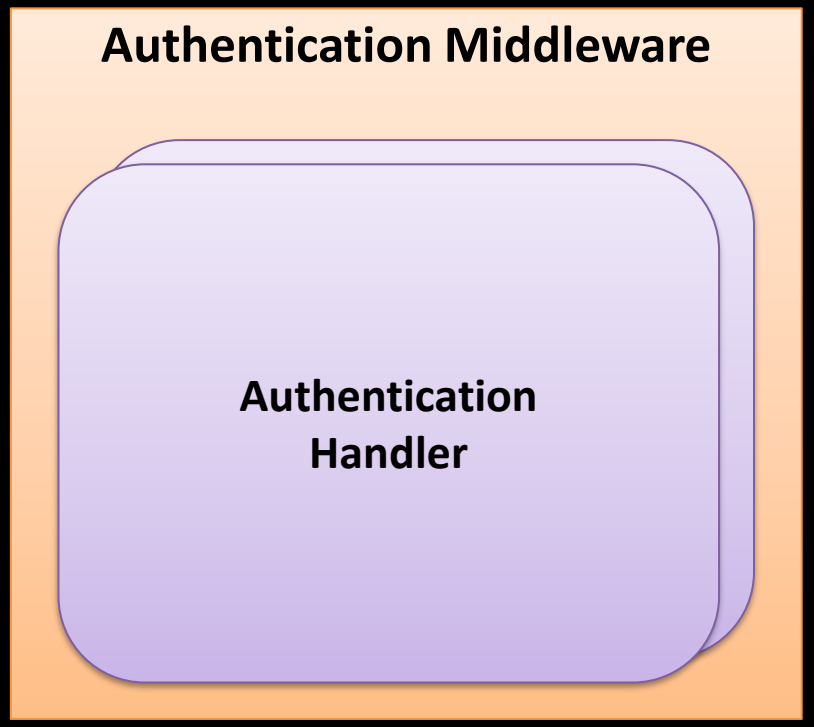
Live coding #1

```
builder.Services.AddAuthentication();

...

app.UseAuthentication();
```

What is inside Authentication?



Authentication Handlers




```
//Add the Cookie handler
builder.Services.AddAuthentication()
                .AddCookie();
```

```
//Add the Cookie and OpenID-Connect handler
builder.Services.AddAuthentication()
                .AddCookie()
                .AddOpenIdConnect();
```

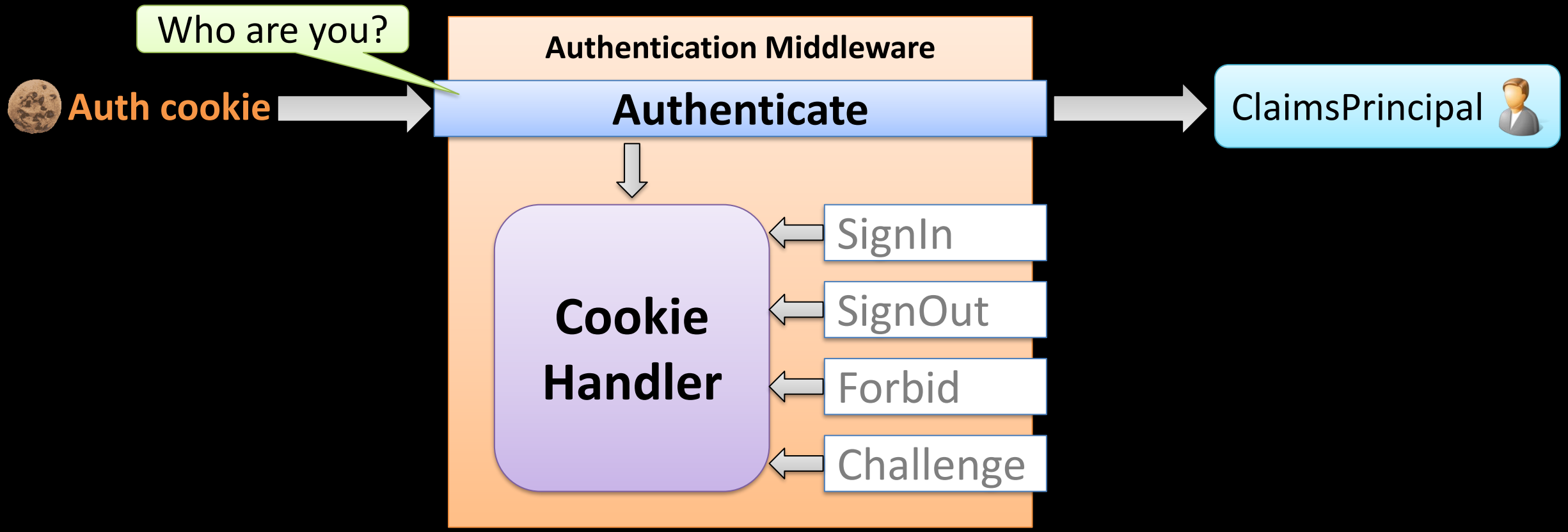
```
// Add the JwtBearer handler (for APIs)
builder.Services.AddAuthentication()
                .AddJwtBearer();
```

Live coding #2

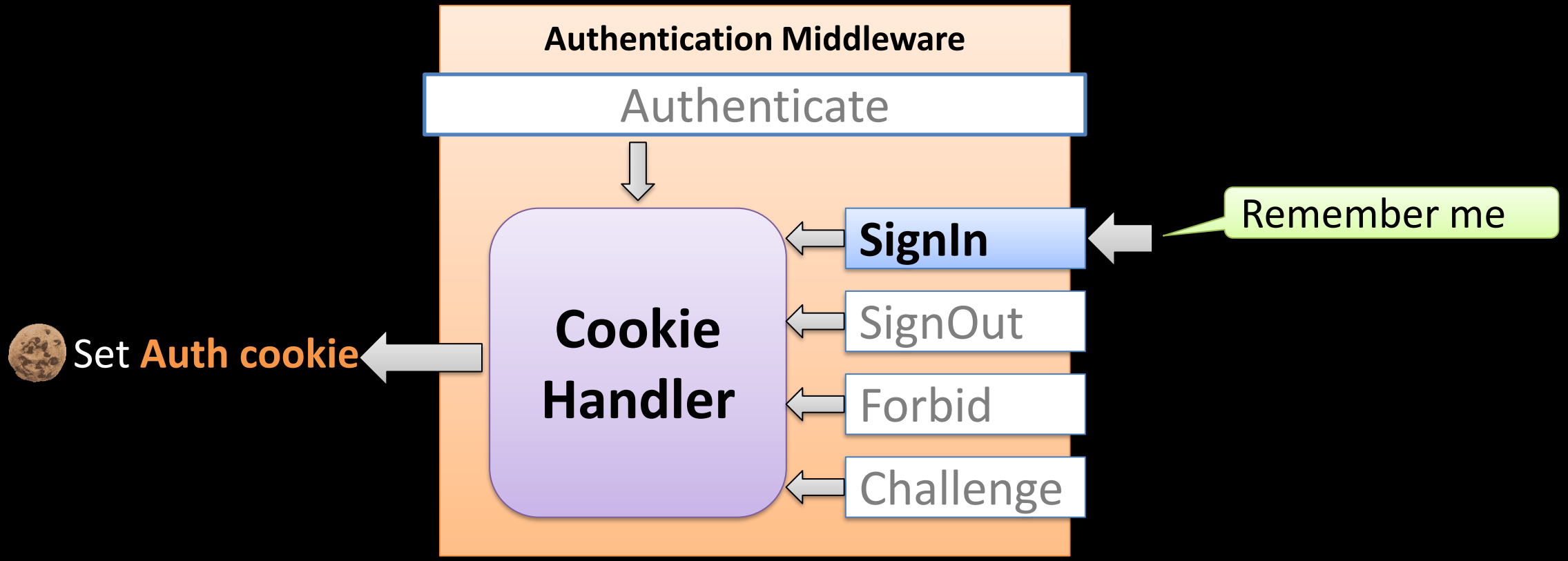
```
builder.Services.AddAuthentication()
                .AddCookie(o =>
                {
                    o.LoginPath = "/user/login";
                    o.LogoutPath = "/user/loggedOut";
                    o.AccessDeniedPath = "/user/AccessDenied";
                });
```

Authentication Operations

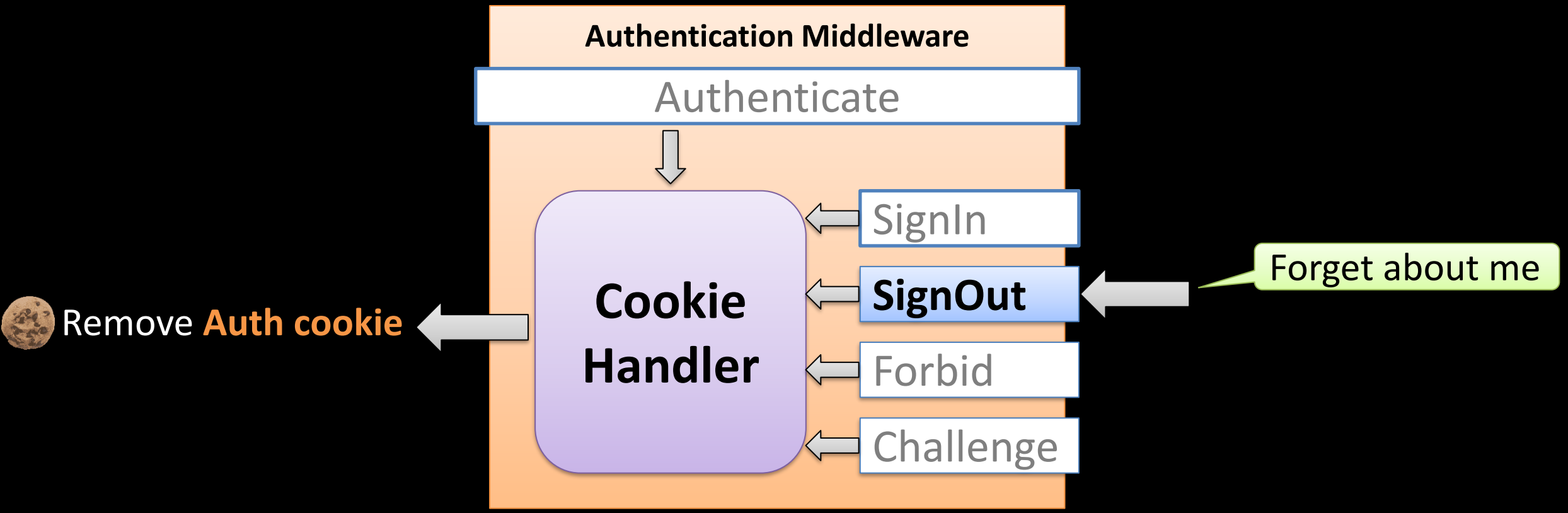
Authenticate



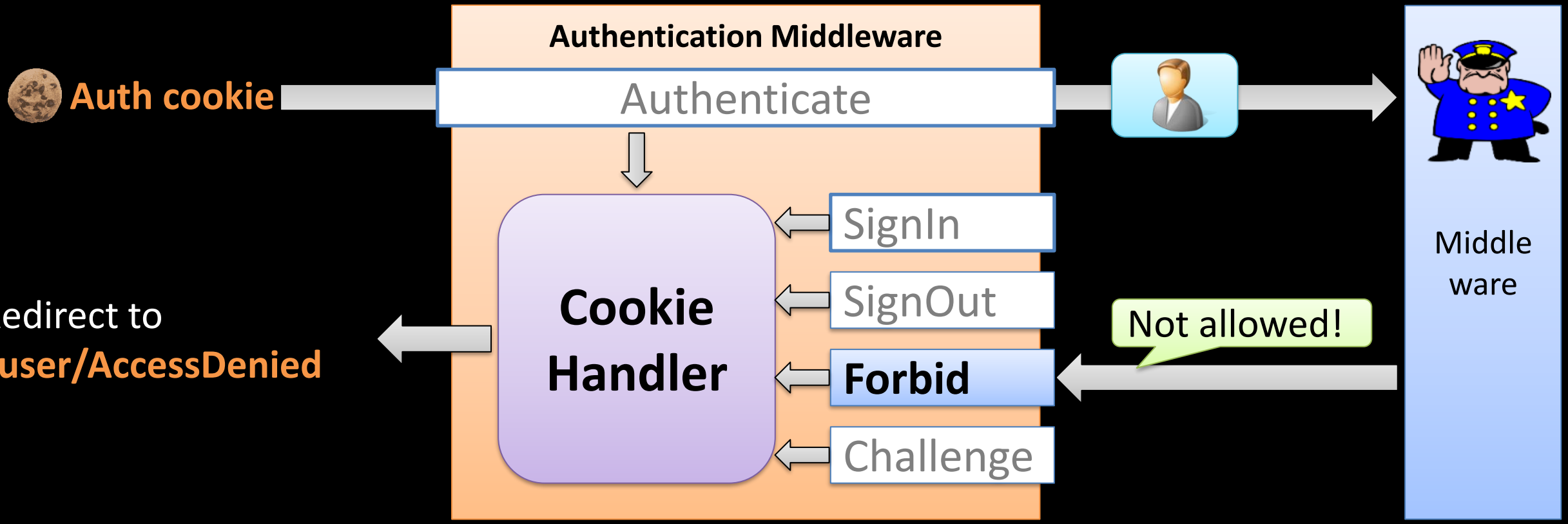
SignIn


Authentication Middleware
Authenticate
Cookie Handler
SignIn
SignOut
Forbid
Challenge
Remember me
Set Auth cookie

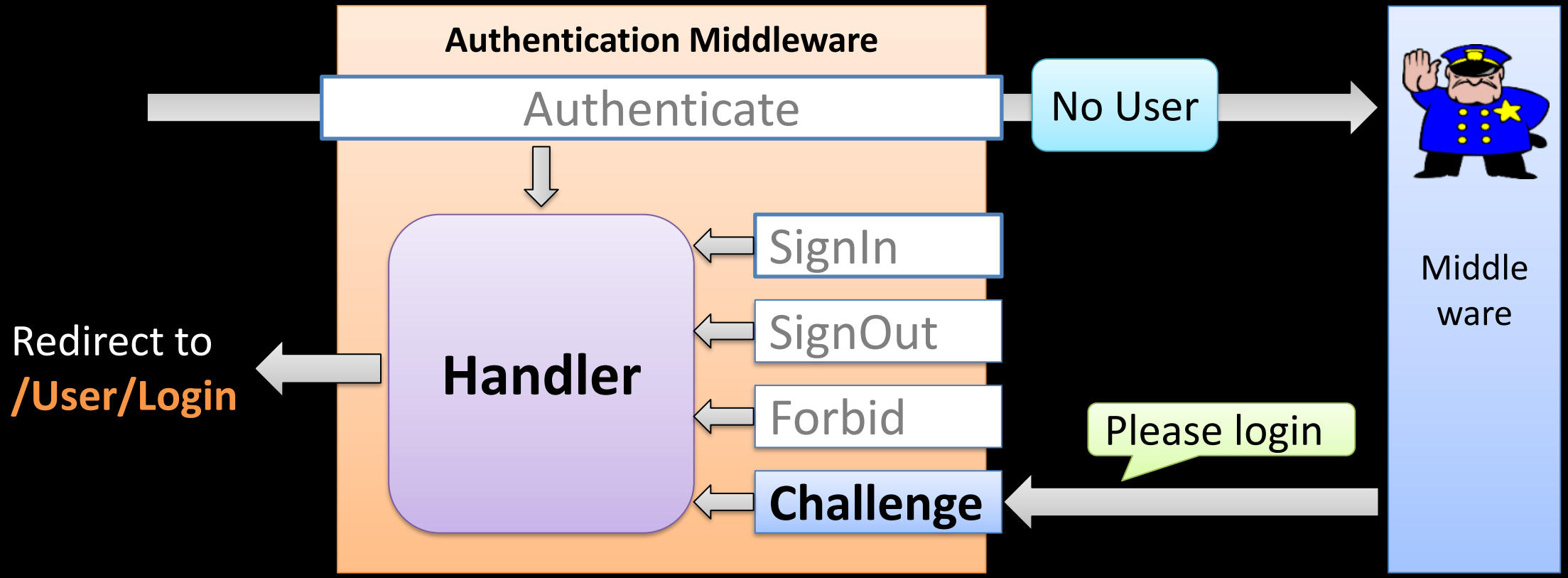
SignOut


Authentication Middleware
Authenticate
Cookie Handler
SignIn
SignOut
Forbid
Challenge
Forget about me
Remove Auth cookie

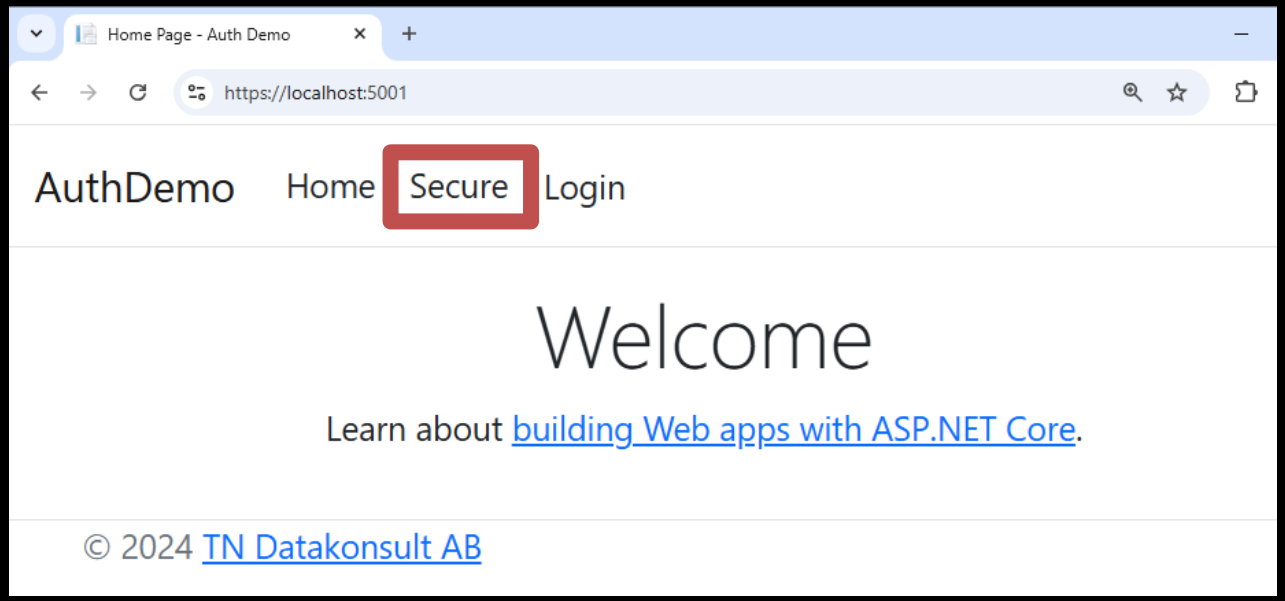
Forbid


Authentication Middleware
Auth cookie
Authenticate
Middle ware
Cookie Handler
SignIn
SignOut
Forbid
Challenge
Not allowed!
Redirect to
/user/AccessDenied

Challenge


Authentication Middleware
Authenticate
No User
Middle ware
Handler
SignIn
SignOut
Forbid
Challenge
Please login
Redirect to
/User/Login

Live coding #3

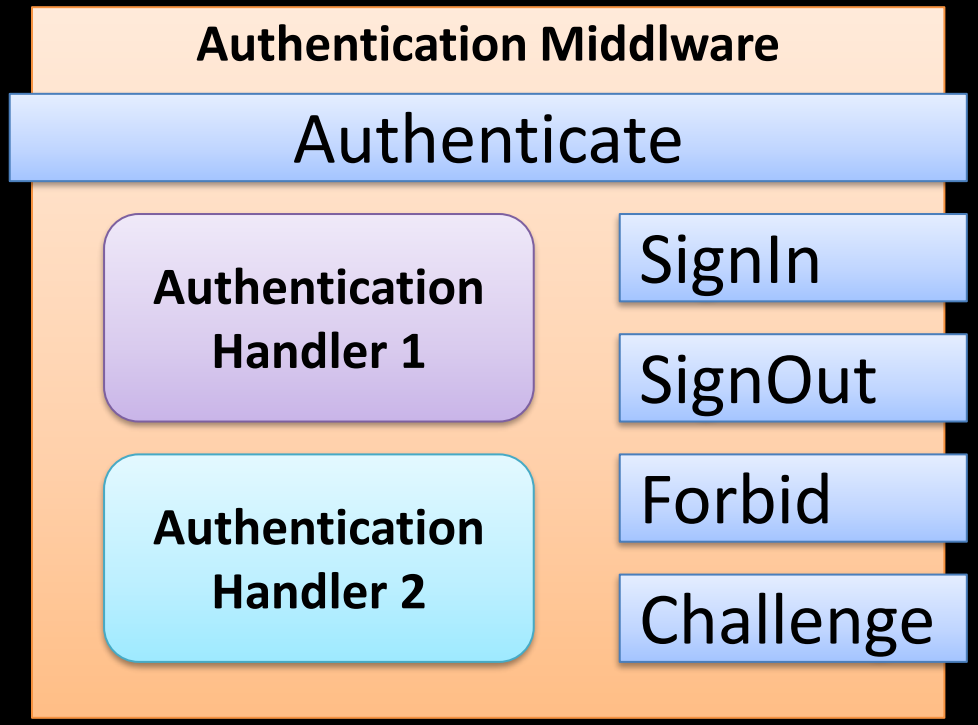


https://localhost:5001/Secure

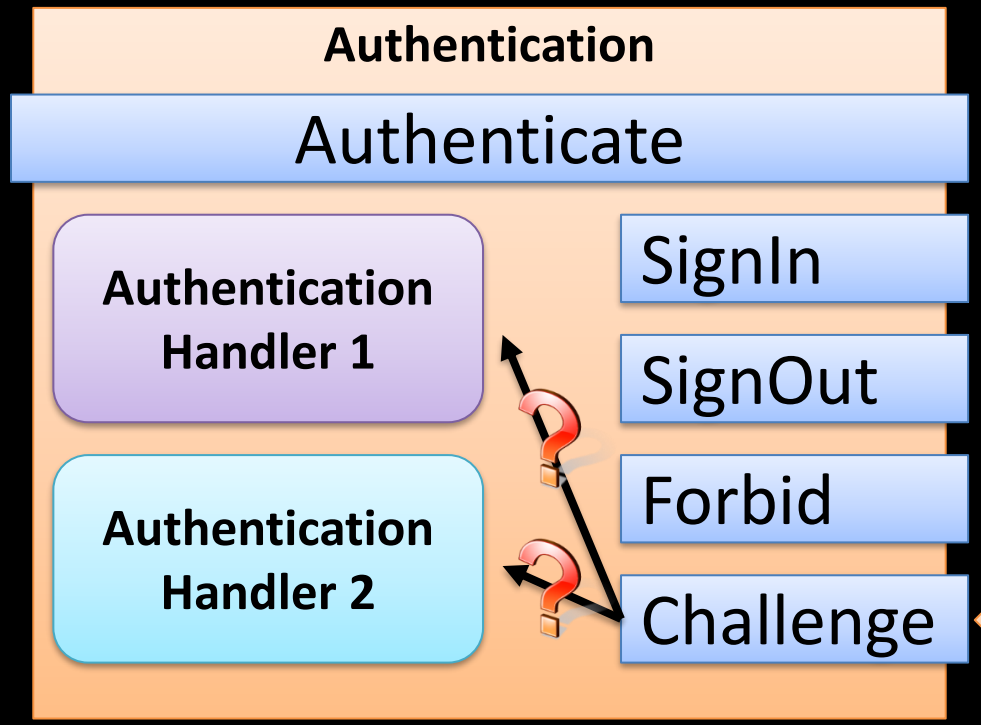
```
public async Task<IActionResult> Index()
{
    if (User.Identity.IsAuthenticated == false)
    {
        await HttpContext.ChallengeAsync();
    }

    return View();
}
```

Schemes!

Schemes


Authentication Middlware
Authenticate
Authentication Handler 1
Authentication Handler 2
SignIn
SignOut
Forbid
Challenge

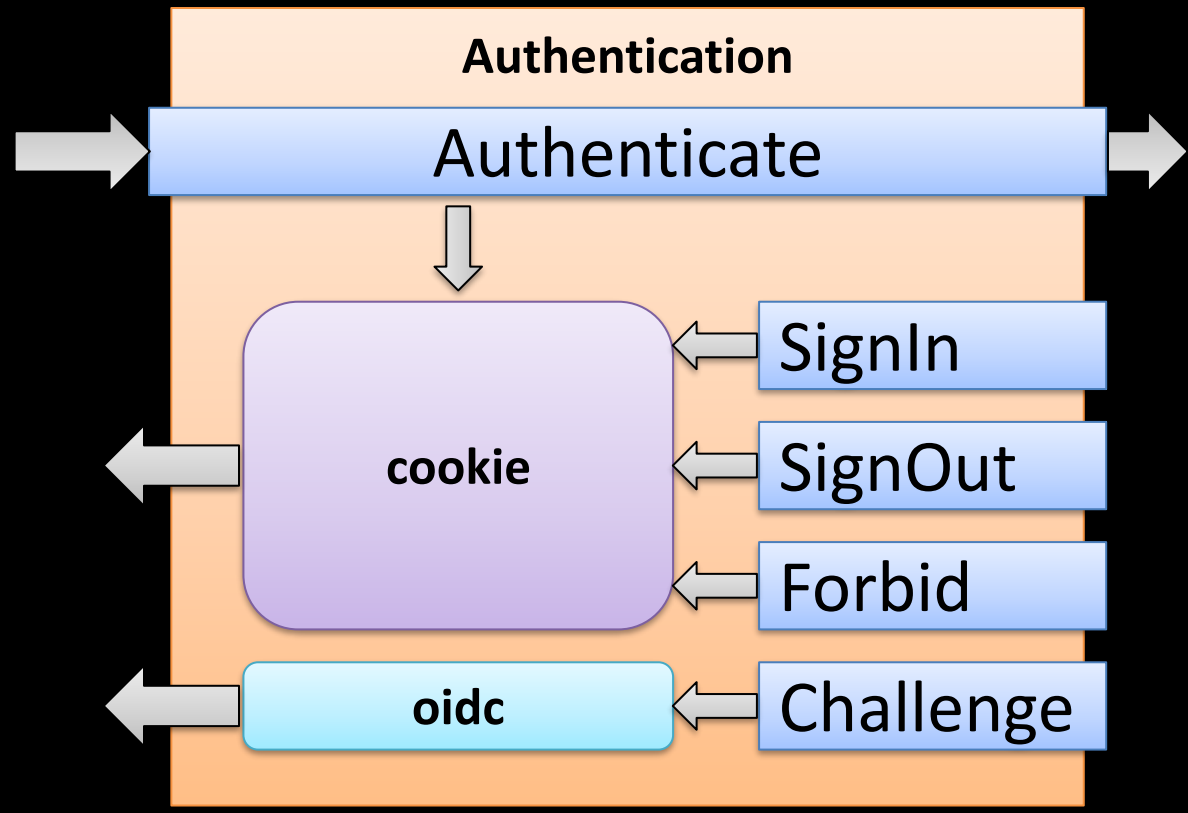
Schemes




```
public async Task<IActionResult> Index()
{
    if (User.Identity.IsAuthenticated == false)
    {
        await HttpContext.ChallengeAsync();
    }

    return View();
}
```

Schemes

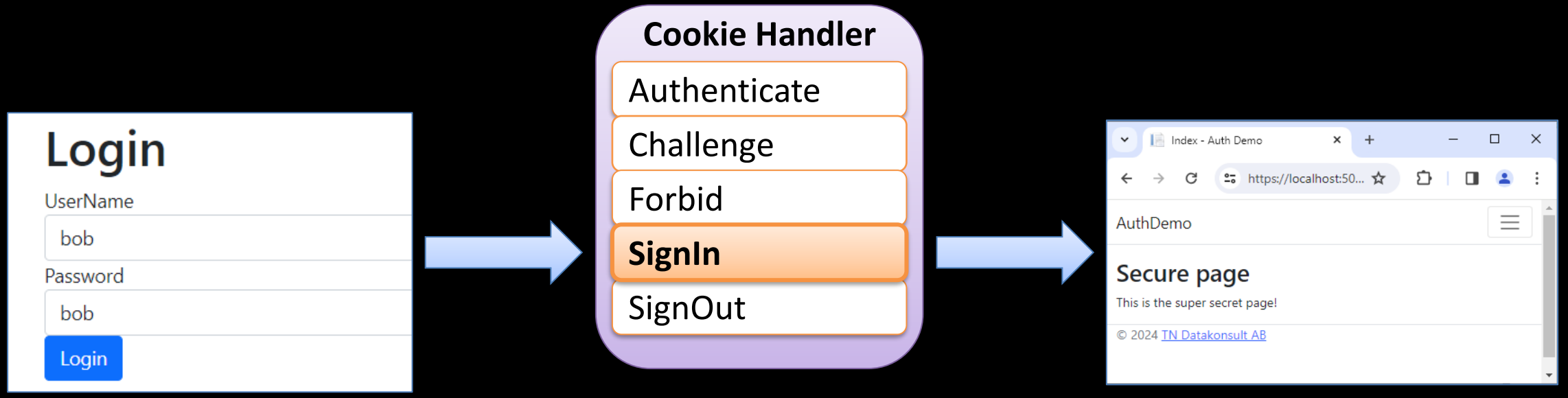
```
builder.Services.AddAuthentication(o =>
{
    o.DefaultAuthenticateScheme = "cookie";
    o.DefaultChallengeScheme = "oidc";
    o.DefaultForbidScheme = "cookie";
    o.DefaultSignInScheme = "cookie";
    o.DefaultSignOutScheme = "cookie";
})
.AddCookie("cookie", opt =>
{
  ...
})
.AddOpenIdConnect("oidc", opt =>
{
   ...
});
```



Live coding #4

```
builder.Services.AddAuthentication(o =>
{
    o.DefaultScheme = "cookie";
})
.AddCookie("cookie", o =>
{
    o.LoginPath = "/user/login";
    o.LogoutPath = "/user/loggedOut";
    o.AccessDeniedPath = "/user/AccessDenied";
});
```

SignIn the user



Validate the username and password

```
[HttpGet]
public IActionResult Login(string ReturnUrl)
{
    return View(new LoginModel() { ReturnUrl = ReturnUrl });
}

[HttpPost]
[ValidateAntiForgeryToken]
public async Task Login(LoginModel loginCredentials)
{
    //...
}
```

```
public class LoginModel
{
    public string UserName  { get; set; };
    public string Password  { get; set; };
    public string ReturnUrl { get; set; };
}
```

Create the list of claims

```
[HttpPost]
[ValidateAntiForgeryToken]
public async Task Login(LoginModel loginCredentials)
{

    //1. Validate username + password

    //2. Load the claims for this user from the DB
    var claims = new List<Claim>()
                {
                    new("sub","1234"),
                    new("name","Bob"),
                    new("email","bob@tn-data.se"),
                    new("role","developer")
                };

    //...

}
```

Create an Identity + ClaimsPrincipal

```
//1. Validate username + password

//2. Load the claims for this user from the DB
var claims = new List<Claim>()
    {
        new("sub","1234"),
        new("name","Bob"),
        new("email","bob@tn-data.se"),
        new("role","developer")
    };

//3. Create an Identity for these claims
var identity = new ClaimsIdentity(claims: claims,
                                  authenticationType: "pwd",
                                  nameType: "name",
                                  roleType: "role");

//4. Create the principal based on the users Identity
var principal = new ClaimsPrincipal(identity);
```

Auth types
pwd
External
Windows
Jwt
Cookie
api_key
NTLM
Digest
Basic
Negotiate
…

SignIn Principal

```
var principal = new ClaimsPrincipal(identity);

var prop = new AuthenticationProperties()
{
    RedirectUri = loginCredentials.ReturnUrl,
    Items =
    {
        { "IpAddress", "192.168.0.3" },
        { "ComputerName", "MyComputer" },
        { "ApiKey", "Summer2024" }
    }
};

await HttpContext.SignInAsync(scheme: "cookie",
                              principal: principal,
                              properties: prop);
```

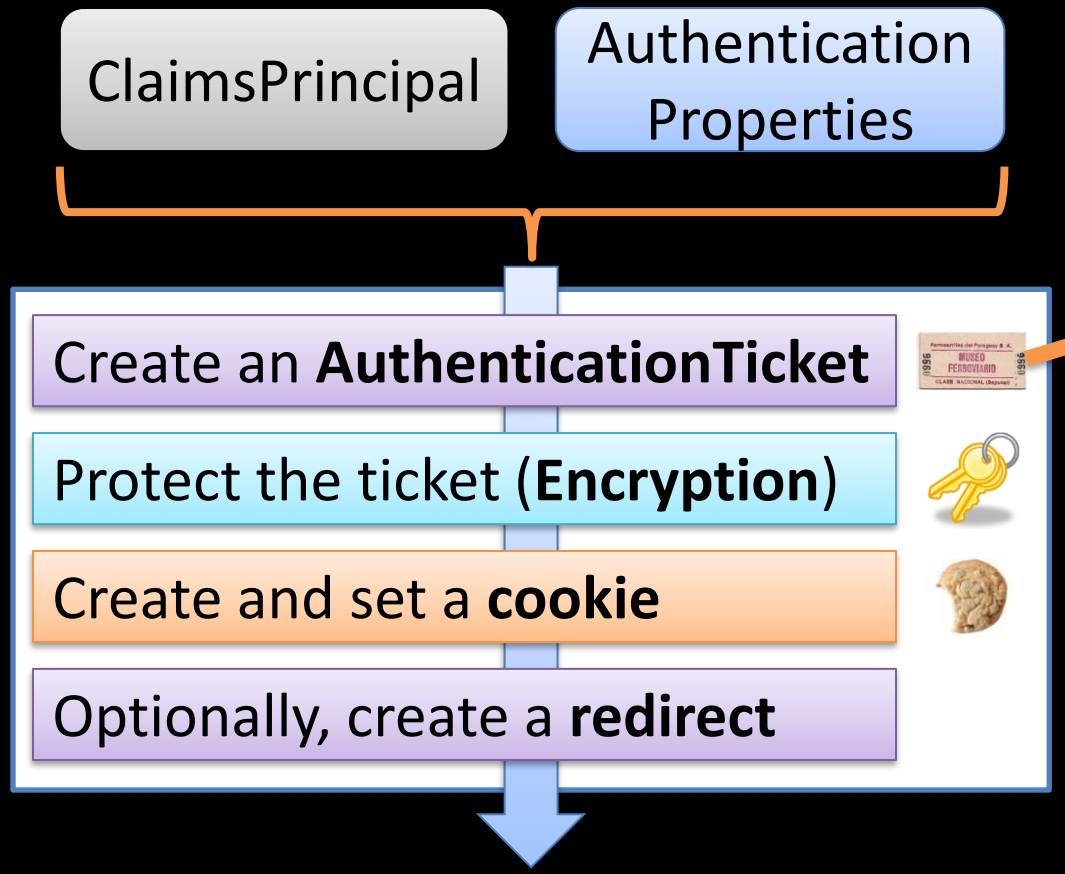
Live coding #5

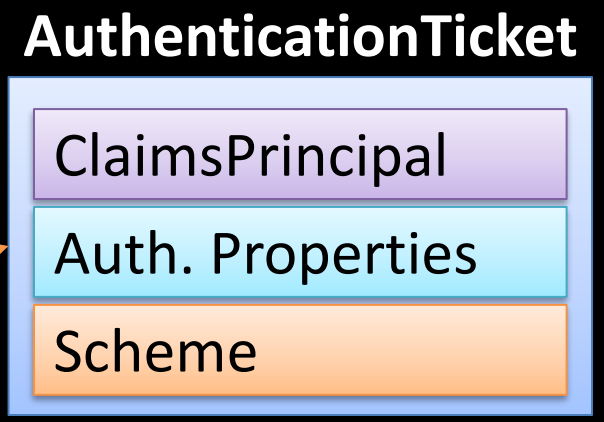
```
//1. Validate username + password

//2. Load the claims for this user from the DB
var claims = new List<Claim>()
{
    new("sub","1234"),
    new("name","Bob"),
    new("email","bob@tn-data.se"),
    new("role","developer")
};

var identity = new ClaimsIdentity(claims: claims,
                                  authenticationType: "pwd",
                                  nameType: "name",
                                  roleType: "role");

var principal = new ClaimsPrincipal(identity);

var prop = new AuthenticationProperties()
{
    RedirectUri = "/",
    Items =
    {
        { "IpAddress", "192.168.0.3" },
        { "ComputerName", "MyComputer" },
        { "ApiKey", "Summer2024" }
    }
};

await HttpContext.SignInAsync(scheme: "cookie",
                              principal: principal,
                              properties: prop);
return LocalRedirect("/");
```

What happens when we call SignInAsync?

```
await HttpContext.SignInAsync(scheme: "cookie",
                              principal: principal,
                              properties: prop);
```

SignInAsync inside the Cookie Handler





What is inside the authentication cookie?

```
Set-Cookie: .AspNetCore.cookie=CfDJ8IgPXRNAZH1EkNA0dd3_JvtpVOohM43sH8lB8MW4
2T1L57tP0RRWmJu8svjUYUwIrYUOW4xo0ikClOR3H87teUK4MYy58NBBAsjc8RDRWhKO6JVz0HuHW1eNSfu
nLJ_OO0bZ1y6kYlF52lkzI8cw8VLPzG4zm33hoynL2JHLTCoWbugN-3kyOLrUSyVJdotB1ANGcvBT-
jz2rAFuOeUbzCdXXDjm98YzW3E99QffLamD1LlrKe7MX1y31NWdxzQ39m4WmGwUNa3b0iHoyDaeSKJvifmz
lMSWT_8o9x4AUtzC6_whIOfPVHXhYkwCXGTwtpIYeb_KOGuAvidb3S3tTkK4m3LHcf5Fx9ajfbE8RC_5FLO
sPxbQiQcF3KGmIUP0dnHmtK7MHczg4UR-OgBh_TA_sKyMoPy9Ak9sa4P-XvMWWyssEkOOzxfHbi6F
VWbq5CNDe6W1QZG6z5PwtwGsVmx4vK8C3_4b9r-HU; path=/; secure; samesite=lax; httponly
```

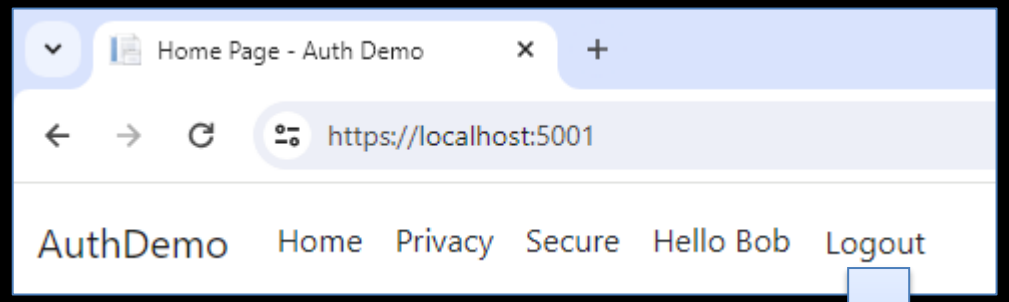
Live coding #6

```
.AddCookie(o =>
{
    o.DataProtectionProvider = new MyDataProtector();
});
```

```
public class MyDataProtector : IDataProtector
{
    public IDataProtector CreateProtector(string purpose)
    {
        return new MyDataProtector();
    }
    public byte[] Protect(byte[] plaintext)
    {
        return plaintext;
    }
    public byte[] Unprotect(byte[] protectedData)
    {
        return protectedData;
    }
}
```

SignOut

SignOut




```
public async Task Logout()
{
    //Sign out from this specific scheme
    await HttpContext.SignOutAsync("cookie");
}
```

Cookie Handler

- Authenticate
- Challenge
- Forbid
- SignIn
- **SignOut**

```
Set-Cookie: .AspNetCore.cookie=; expires=Thu, 01 Jan 1970 00:00:00 GMT; ...
```

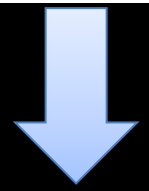
Live coding #7

```
[HttpPost]
public async Task Logout()
{
    await HttpContext.SignOutAsync("cookie");
}
```

LogoutPath

Why are we not redirected?

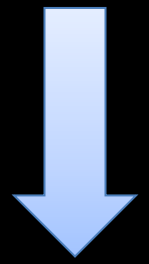
```
builder.Services.AddAuthentication()
                .AddCookie(o =>
                {
                    ...
                    o.LogoutPath = "/user/LoggedOut";
                    ...
                });
```



```
HTTP/1.1 200 OK
Content-Length: 0
Set-Cookie: .AspNetCore.cookie=; expires=Thu, 01 Jan 1970 00:00:00 GMT;...
```

Live coding #8

```
<form action="/User/Logout" method="post">
  <button type="submit">Logout</button>
</form>
```

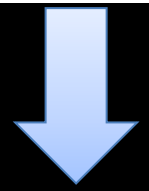


```
<form action="/User/Logout?ReturnUrl=/User/LoggedOut" method="post">
  <button type="submit">Logout</button>
</form>
```

Using ReturnUrl

/User/Logout?ReturnUrl=/User/LoggedOut

```
builder.Services.AddAuthentication()
                .AddCookie(o =>
                {
                    ...
                    o.LogoutPath = "/user/LoggedOut";
                    ...
                });
```

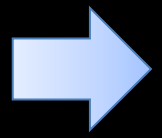


```
HTTP/1.1 200 OK
Content-Length: 0
Set-Cookie: .AspNetCore.cookie=; expires=Thu, 01 Jan 1970 00:00:00 GMT;...
```

Live coding #9

/User/Logout?ReturnUrl=/User/LoggedOut

```
.AddCookie(o =>
{
    ...
   o.LogoutPath = "/user/LoggedOut";
    ...
});
```



```
.AddCookie(o =>
{
    ...
    o.LogoutPath = "/user/LogOut";
    ...
});
```

```
HTTP/1.1 302 Found
Location: /User/LoggedOut
Set-Cookie: .AspNetCore.cookie=; expires=Thu, 01 Jan 1970 00:00:00 GMT;...
```

Alternatives

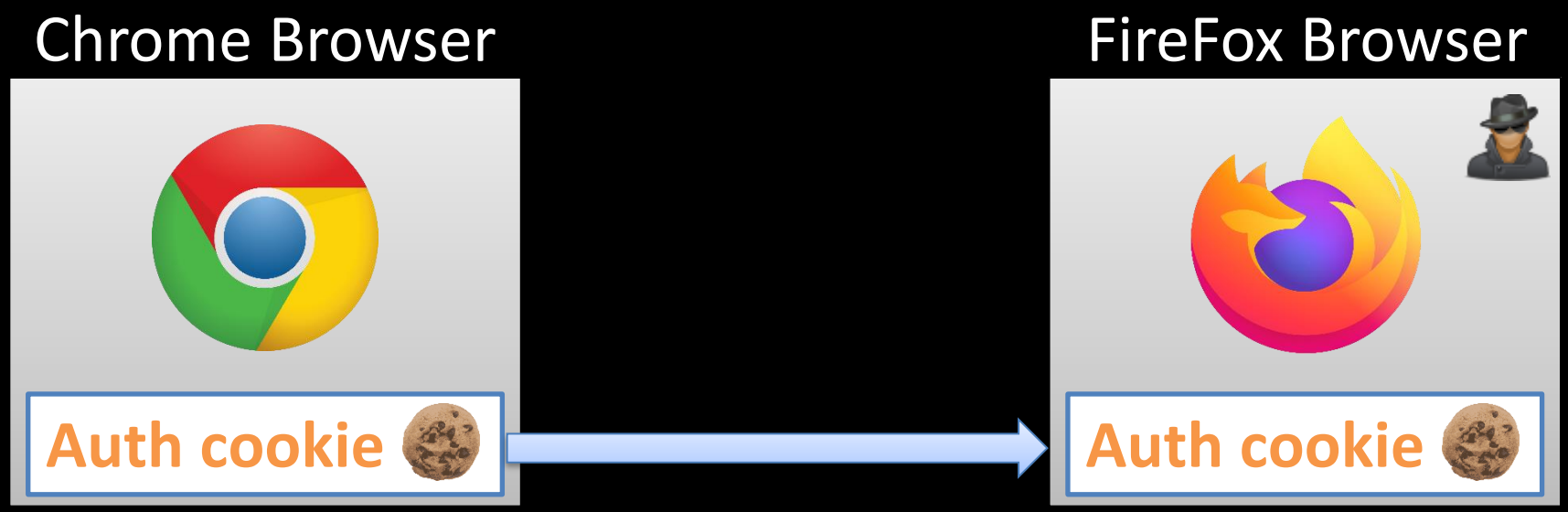
```
[HttpPost]
public async Task Logout()
{
    var properties = new AuthenticationProperties()
    {
        RedirectUri = "/User/LoggedOut"
    };

    await HttpContext.SignOutAsync("cookie", properties);
}
```

```
[HttpPost]
public async Task<IActionResult> Logout()
{
    await HttpContext.SignOutAsync("cookie");

    return LocalRedirect("/User/LoggedOut");
}
```

Are we really signed out?

```
[HttpPost]
public async Task Logout()
{
    await HttpContext.SignOutAsync("cookie");
}
```

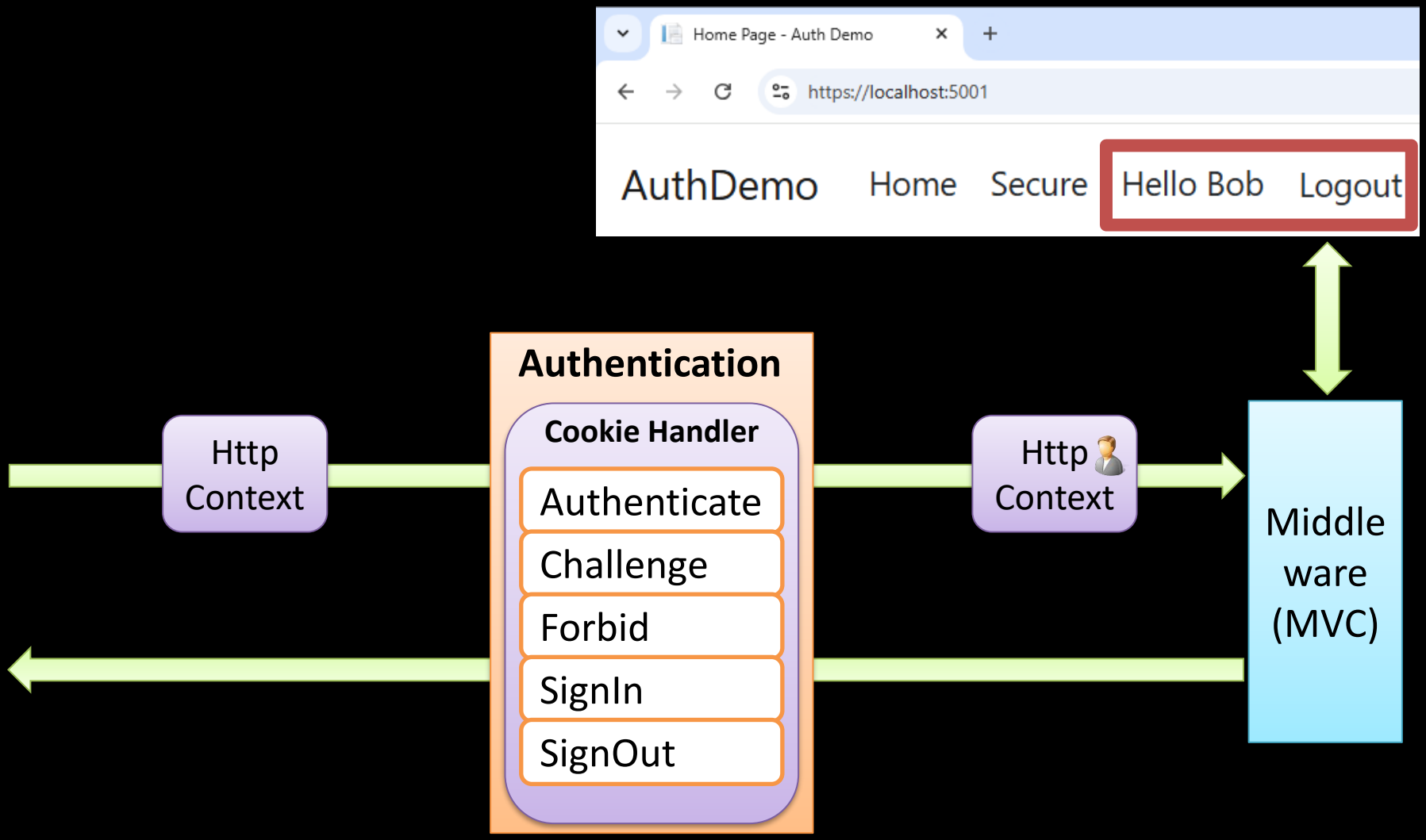
Live coding #10



Large cookies!

```
ŵắs çľắîņș   ŋêx Ľîșʈ Cľắîņ

    ŋêx  șụč    ,,,,     ŋêx șʈsîŋĝ  y̰  ,,,,,
    ŋêx  ŋắņê   Bộč
    ŋêx  êņắîľ   čộč ʈŋ đắʈắ șê
    ŋêx  sộľê   đêŵêľộřês
```

Summary


Home Page - Auth Demo
https://localhost:5001
AuthDemo
Home
Secure
Hello Bob
Logout
Authentication
Cookie Handler
Authenticate
Challenge
Forbid
SignIn
SignOut
Http Context
Http Context
Middle ware (MVC)

QUESTIONS?

Presentation and code
https://github.com/tndataab/PublicBlogContent

Blog
https://nestenius.se

Work
https://tn-data.se