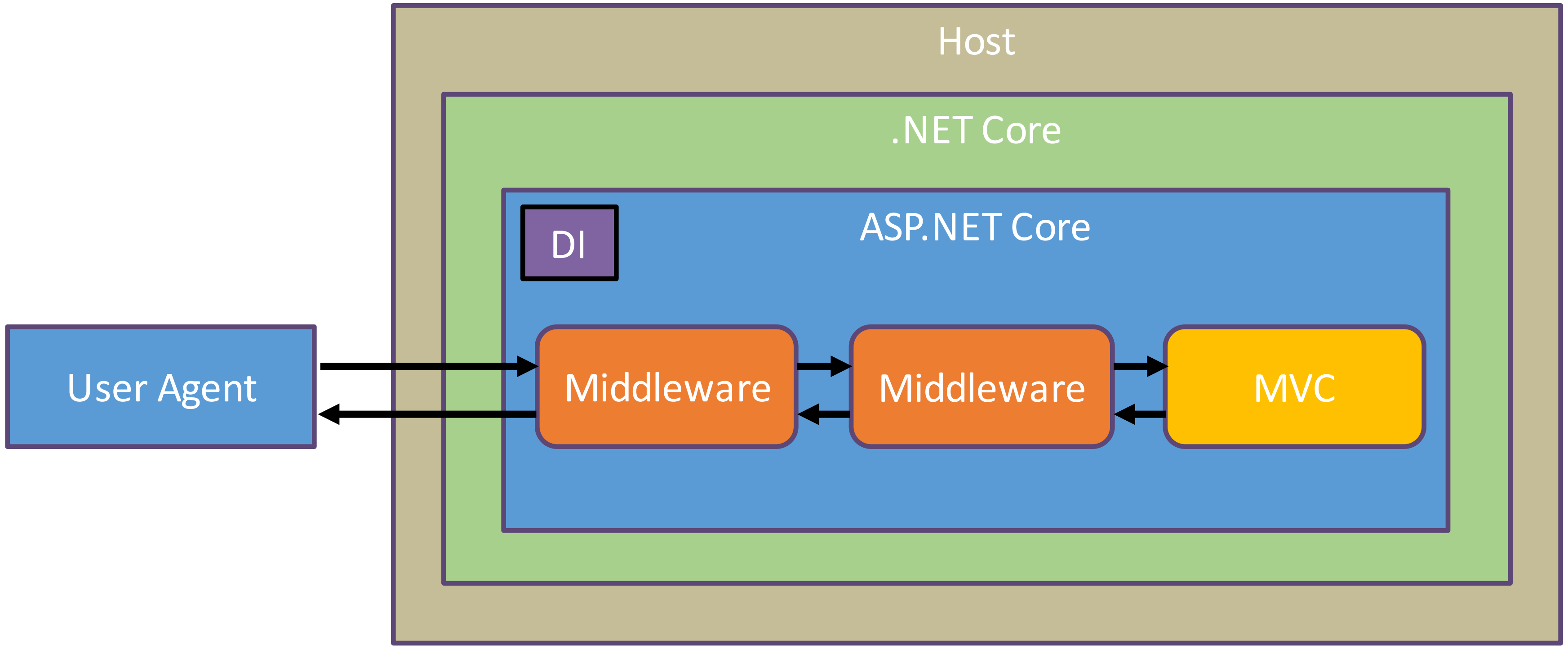
ASP.NET Core, User Authentication, and User Authorization

What is ASP.NET Core?

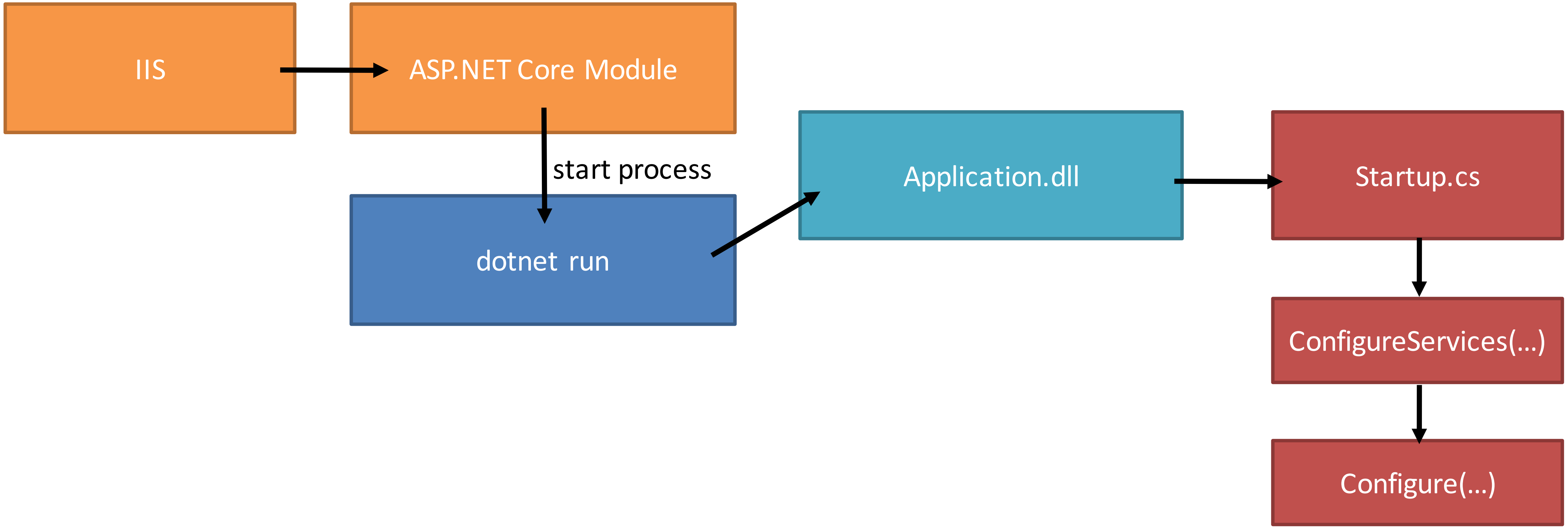
- **Microsoft's new web framework**
  - Built on top of .NET Core
  - Designed to be cross-platform
- **Middleware-based pipeline architecture**
  - Components that provide services for web applications
  - Many features packaged as middleware
- **Familiar HttpContext programming model**
  - But all new
- **Hosting is provided by Kestrel**
  - libuv-based HTTP server

ASP.NET Core Architecture

- **ASP.NET Core is the runtime (hosted by .NET Core)**
- **MVC is Microsoft's primary application framework**
  - combines web UI & API



How ASP.NET Core Applications start

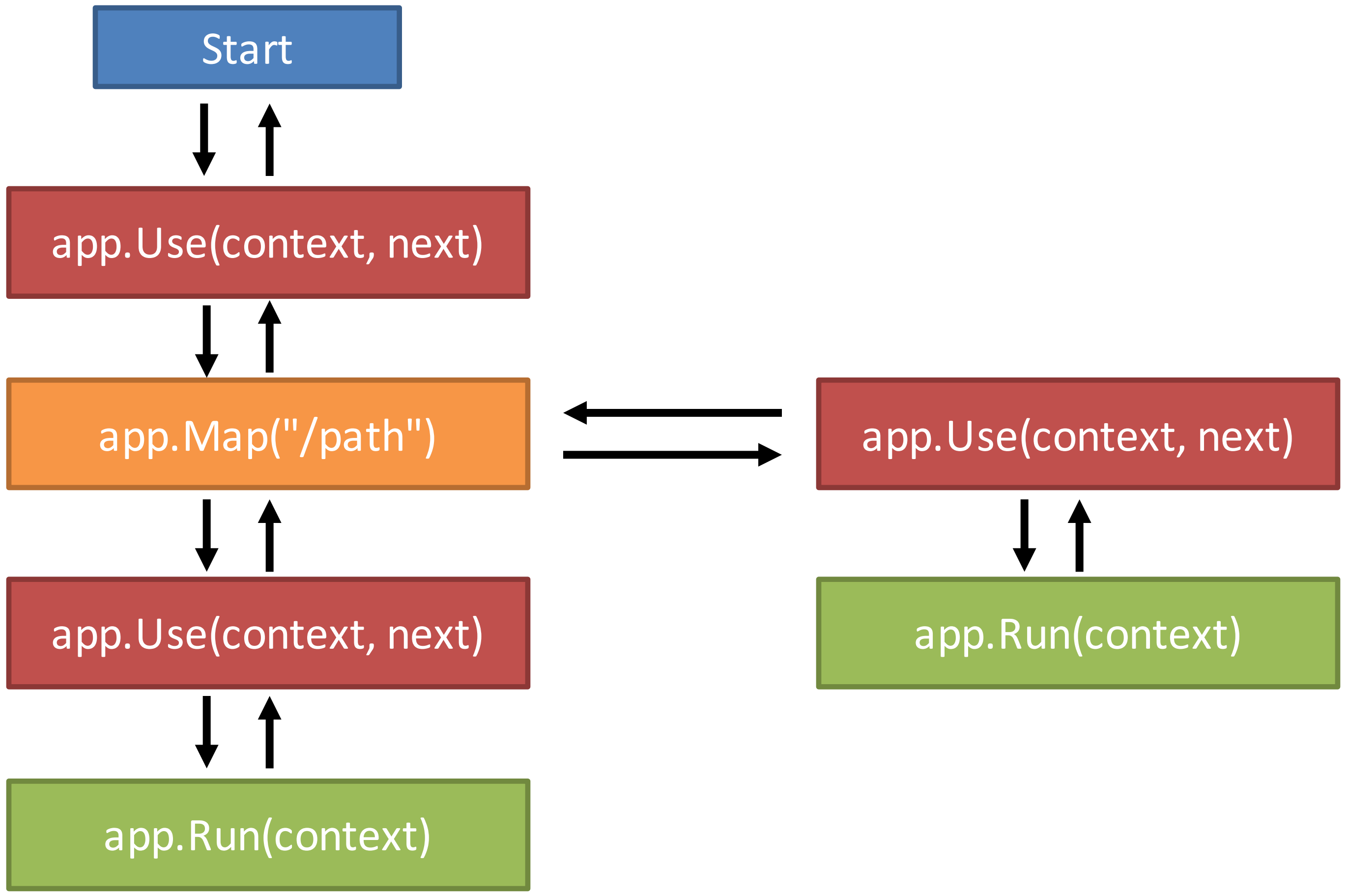


Loading ASP.NET Core

```
public class Program
{
    public static void Main()
    {
        var host = new WebHostBuilder()
            .UseKestrel()
            .UseIISIntegration()
            .UseStartup<Startup>()
            .Build();

        host.Run();
    }
}
```

```
public class Startup
{
    public void Configure(IApplicationBuilder app)
    {
        ...
    }
}
```

Pipeline primitives



Run

```
namespace Microsoft.AspNetCore.Builder
{
    public delegate Task RequestDelegate(HttpContext context);
}
```

```
app.Run(async context =>
{
    await context.Response.WriteAsync("Hello ASP.NET5");
});
```

Map

```
app.Map("/hello", helloApp =>
{
    helloApp.Run(async (HttpContext context) =>
    {
        await context.Response.WriteAsync("Hello ASP.NET5");
    });
});
```

Use

```
app.Use(async (context, next) =>
{
    if (!context.Request.Path.Value.EndsWith("/favicon.ico"))
    {
        Console.WriteLine("pre");
        Console.WriteLine(context.Request.Path);

        await next();

        Console.WriteLine("post");
        Console.WriteLine(context.Response.StatusCode);
    }
    else
    {
        await next();
    }
});
```

Middleware classes

```
app.UseMiddleware<InspectionMiddleware>();
```

```
public class InspectionMiddleware
{
    private readonly RequestDelegate _next;

    public InspectionMiddleware(RequestDelegate next)
    {
        _next = next;
    }

    public async Task Invoke(HttpContext context)
    {
        Console.WriteLine($"request: {context.Request.Path}");
        await _next(context);
    }
}
```

Authentication in ASP.NET Core

- **Various middleware provide authentication features**
  - Cookies for browser based authentication
  - Google, Facebook, and other social authentication
  - OpenId Connect for external authentication
  - JSON web token (JWT) for token-based authentication

AuthenticationManager

- **Central API for coordinating authentication middleware**

```
public abstract class AuthenticationManager
{
   public abstract IEnumerable<AuthenticationDescription> GetAuthenticationSchemes();

   public virtual Task SignInAsync(string authenticationScheme, ClaimsPrincipal principal);
   public virtual Task SignOutAsync(string authenticationScheme);

   public virtual Task<ClaimsPrincipal> AuthenticateAsync(string authenticationScheme);

   public virtual Task ChallengeAsync(string authenticationScheme);
   public virtual Task ForbidAsync();

   // ...
}
```

Cookie Authentication Middleware

- **Forms / Session authentication replacement**

```
public void Configure(IApplicationBuilder app)
{
    app.UseCookieAuthentication(new CookieAuthenticationOptions
    {
        AuthenticationScheme = "Cookies",
        AutomaticAuthenticate = true,
        AutomaticChallenge = true,

        LoginPath = new PathString("/Account/Login"),
        AccessDeniedPath = new PathString("/Account/AccessDenied")
    });
}
```

Cookies: Logging in

- **SignInAsync issues cookie**
  - Authentication scheme parameter indicates which middleware

```
var claims = new Claim[]
{
    new Claim("sub", "37734"),
    new Claim("name", "Brock Allen")
};

var ci = new ClaimsIdentity(claims, "password");
var cp = new ClaimsPrincipal(ci);

await HttpContext.Authentication.SignInAsync("Cookies", cp);
```

Cookies: Logging out

- **SignOutAsync removes cookie**
  - Authentication scheme parameter indicates which middleware

```
await HttpContext.Authentication.SignOutAsync("Cookies");
```

Claims Transformation

- **Per-request manipulation of principal & claims**

```
app.UseClaimsTransformation(context =>
{
    if (context.Principal.Identity.IsAuthenticated)
    {
        CreateApplicationPrincipal(context);
    }

    return Task.FromResult(context.Principal);
});
```

Authorization

- **Complete re-write**
  - support for *unauthorized* vs *forbidden*
  - better separation of business code and authorization logic
  - re-usable policies
  - resource/action based authorization
  - DI enabled

[Authorize]

- **Similar syntax**
  - roles still supported*

```
[Authorize]
public class HomeController : Controller
{
    [AllowAnonymous]
    public IActionResult Index()
    {
        return View();
    }

    [Authorize(Roles = "Sales")]
    public IActionResult About()
    {
        return View(User);
    }
}
```

* …and who thought that would be a good idea?

Authorization policies

Startup

```
services.AddAuthorization(options =>
{
    options.AddPolicy("SalesSenior", policy =>
    {
        policy.RequireAuthenticatedUser();
        policy.RequireClaim("department", "sales");
        policy.RequireClaim("status", "senior");
    });
};
```

Controller

```
[Authorize("SalesSenior")]
public IActionResult Manage()
{
    // stuff
}
```

Custom Requirements

```
public class JobLevelRequirement : IAuthorizationRequirement
{
    public JobLevel Level { get; }

    public JobLevelRequirement(JobLevel level)
    {
        Level = level;
    }
}

public static class StatusPolicyBuilderExtensions
{
    public static AuthorizationPolicyBuilder RequireJobLevel(
      this AuthorizationPolicyBuilder builder, JobLevel level)
    {
        builder.AddRequirements(new JobLevelRequirement(level));
        return builder;
    }
}
```

Handling Requirements

```
public class JobLevelRequirementHandler : AuthorizationHandler<JobLevelRequirement>
{
    private readonly IOrganizationService _service;

    public JobLevelRequirementHandler(IOrganizationService service)
    {
        _service = service;
    }

    protected override void Handle(
        AuthorizationContext context, JobLevelRequirement requirement)
    {
        var currentLevel = _service.GetJobLevel(context.User);

        if (currentLevel == requirement.Level)
        {
            context.Succeed(requirement);
        }
    }
}
```

Resource-based Authorization

Subject

- client ID
- subject ID
- scopes
- more claims

+ DI

Operation

- read
- write
- send via email
- ...

Object

- ID
- owner
- more properties

+ DI

Example: Document resource

```
public class DocumentAuthorizationHandler :
    AuthorizationHandler<OperationAuthorizationRequirement, Document>
{
    public override void Handle(
      AuthorizationContext context,
      OperationAuthorizationRequirement operation,
      Document resource)
    {
        // authorization logic
    }
}
```

Add handler in DI:

```
services.AddTransient<IAuthorizationHandler, DocumentAuthorizationHandler>();
```

Invoking the authorization handler

```
public class DocumentController : Controller
{
    private readonly IAuthorizationService _authz;

    public DocumentController(IAuthorizationService authz)
    {
        _authz = authz;
    }

    public async Task<IActionResult> Update(Document doc)
    {
        if (!await _authz.AuthorizeAsync(User, doc, Operations.Update))
        {
            // forbidden
            return new ChallengeResult();
        }

        // do stuff
    }
}
```

…or from a View

```
@{
    @using Microsoft.AspNetCore.Authorization
    @inject IAuthorizationService _authz
}


@if (await _authz.AuthorizeAsync(User, "SalesOnly"))
{
    <div>
        <a href="/test/salesOnly">Sales only</a>
    </div>
}
```

External Authentication

- **In the box**
  - Google, Twitter, Facebook, Microsoft Account
  - OpenID Connect & JSON Web Tokens

- **New generic OAuth 2.0 middleware makes on-boarding other proprietary providers easier**
  - LinkedIn, Slack, Spotify, WordPress, Yahoo, Github, Instragram, BattleNet, Dropbox, Paypal, Vimeo…

https://github.com/aspnet-contrib/AspNet.Security.OAuth.Providers

Social Identity Providers

- **Enabled with *UseGoogleAuthentication*, et al.**
  - Rely upon cookie authentication middleware

```
app.UseCookieAuthentication(new CookieAuthenticationOptions
{
    AuthenticationScheme = "Cookies",
    AutomaticAuthenticate = true,
});

app.UseGoogleAuthentication(new GoogleOptions
{
    AuthenticationScheme = "Google",
    SignInScheme = "Cookies",
    ClientId = "998042782978...",
    ClientSecret = "HsnwJri_53zn7..."
});
```

Social Identity Providers

- **ChallengeAsync triggers redirect for login**
  - Control URL user returns to with *AuthenticationProperties*
  - MVC *ChallengeResult* works with action result architecture

```
var props = new AuthenticationProperties
{
    RedirectUri = "/Home/Secure"
};
await HttpContext.Authentication.ChallengeAsync("Google", props);

// or if using MVC:

return new ChallengeResult("Google", props);
```

Mixing local and external Authentication

- **Typically need registration logic for users from social providers**
  - Use additional cookie middleware for processing registration

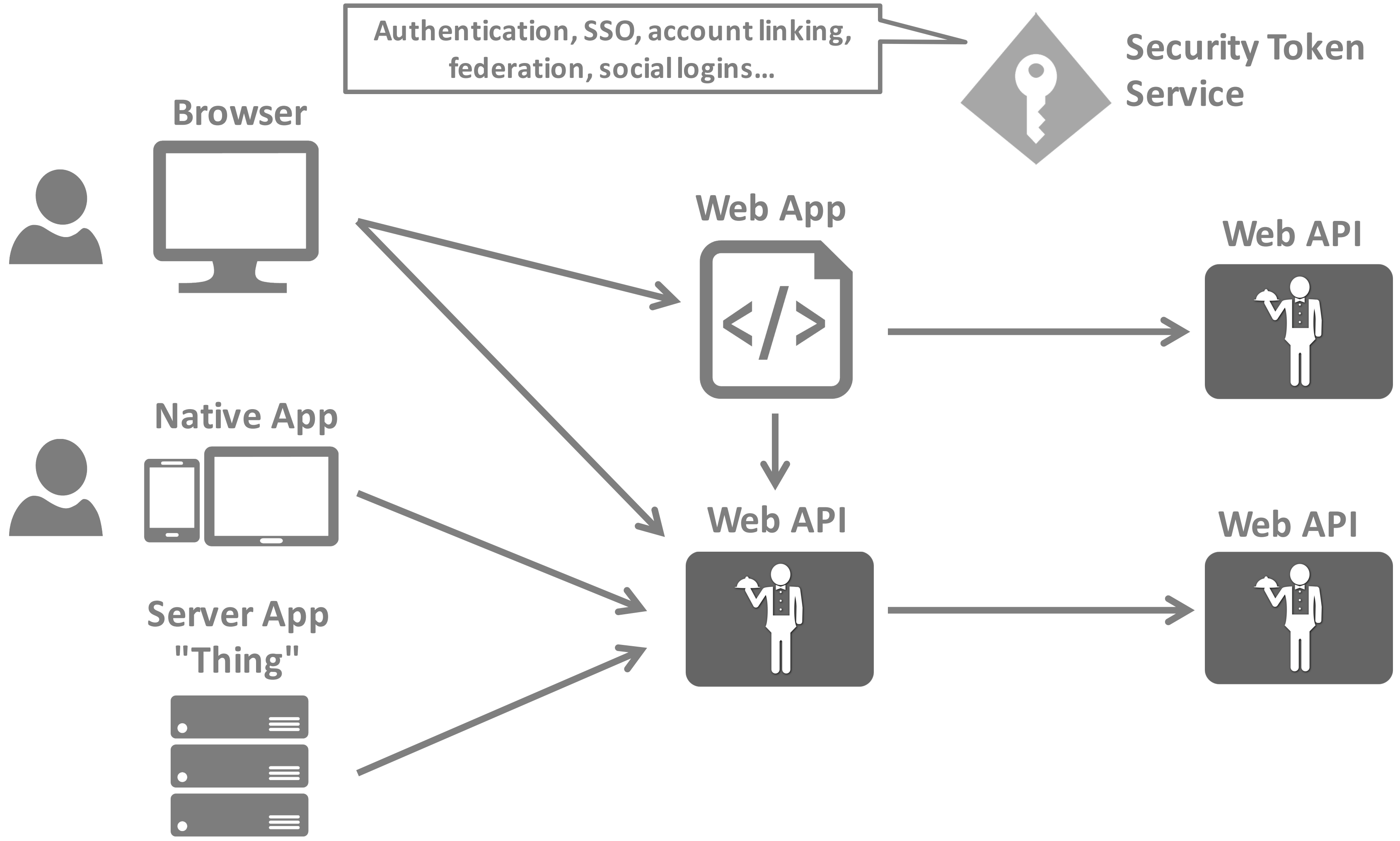
```
app.UseCookieAuthentication(new CookieAuthenticationOptions
{
    AuthenticationScheme = "Temp",
    AutomaticAuthenticate = false,
    AutomaticChallenge = false
});
app.UseGoogleAuthentication(new GoogleOptions
{
    AuthenticationScheme = "Google",
    SignInScheme = "Temp",
    ClientId = "998042782978...",
    ClientSecret = "HsnwJri_53zn7..."
});
```

Mixing local and external Authentication

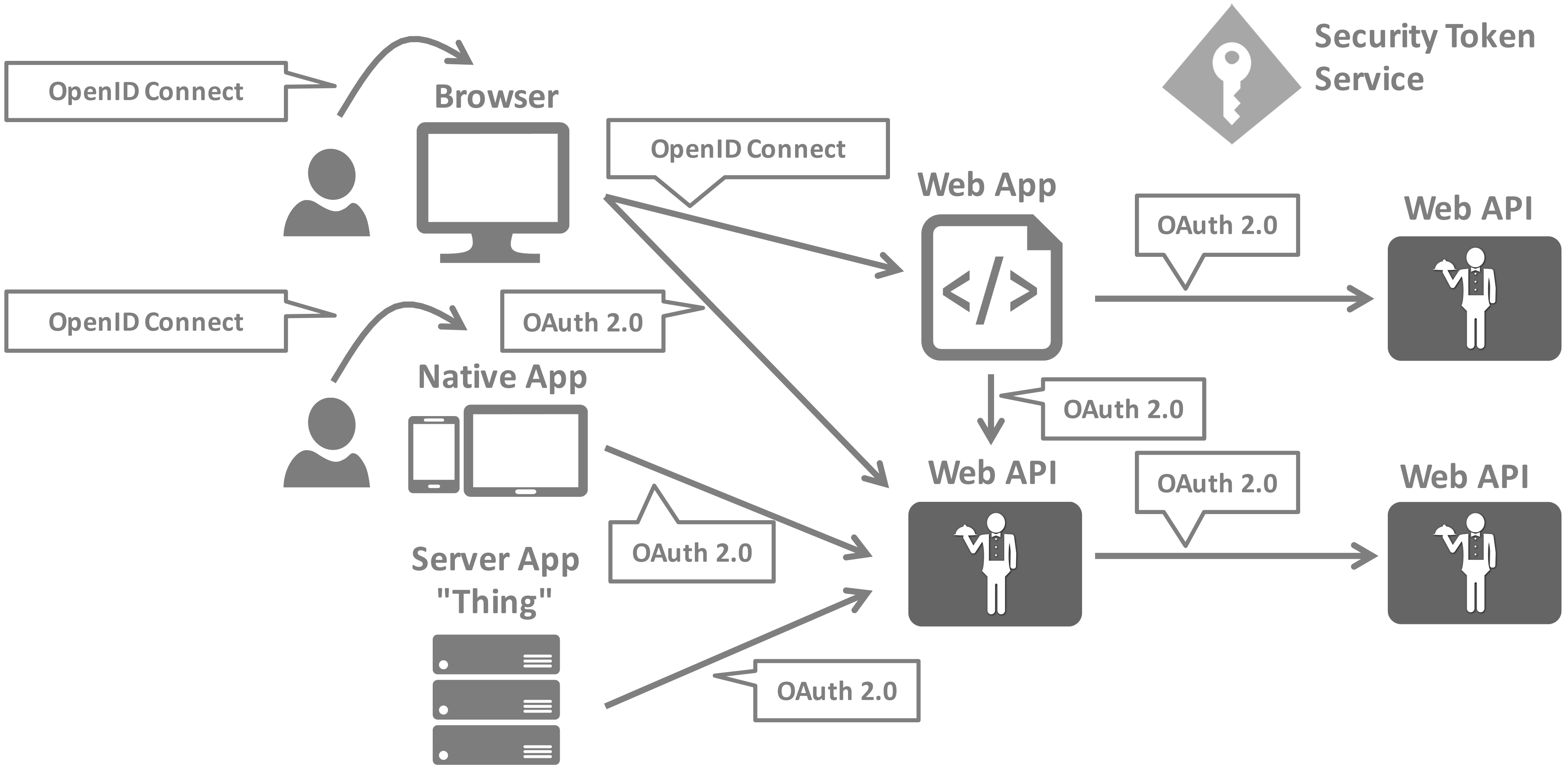
- **Redirect page performs local account registration logic**
  - *AuthenticateAsync* triggers cookie middleware
  - Create local account or load existing account
  - Use primary cookie middleware to log user in (and remove temp cookie)

```
var tempUser = await HttpContext.Authentication.AuthenticateAsync("Temp");
var userIdClaim = tempUser.FindFirst(ClaimTypes.NameIdentifier);
var provider = userIdClaim.Issuer;
var userId = userIdClaim.Value;

// create local account if new, or load existing local account

var user = new ClaimsPrincipal(...);
await HttpContext.Authentication.SignInAsync("Cookies", user);
await HttpContext.Authentication.SignOutAsync("Temp");
```

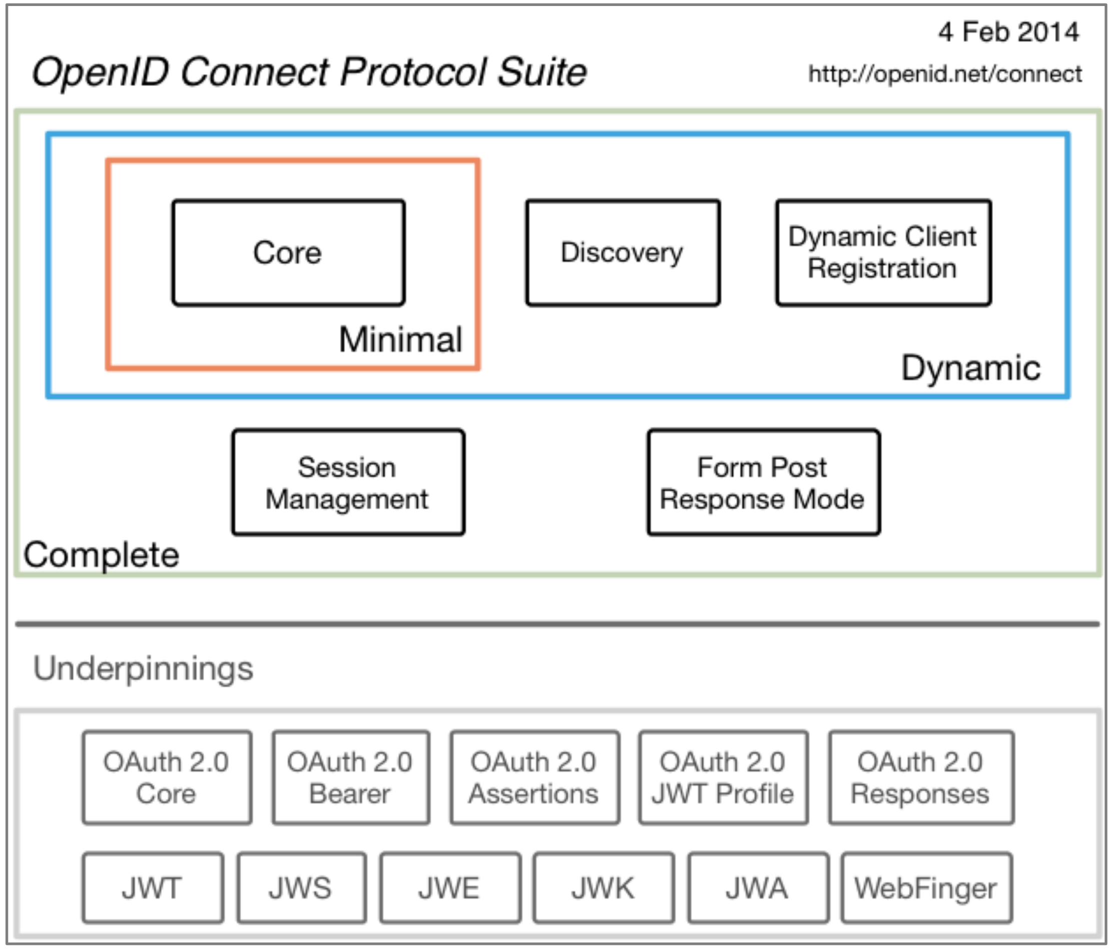
The way forward…



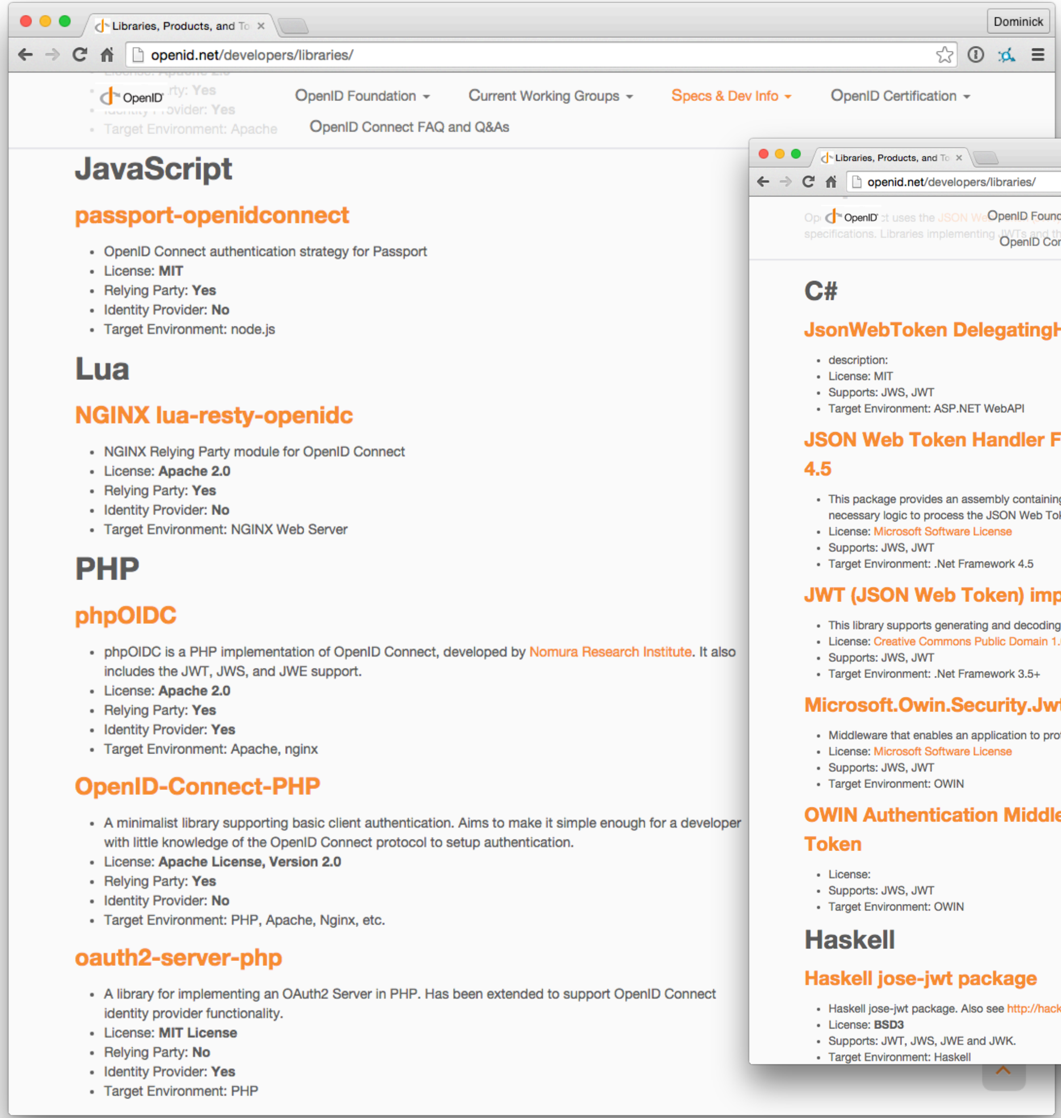
Security Protocols



http://openid.net/connect/



Libraries & Implementations



JavaScript

passport-openidconnect

- OpenID Connect authentication strategy for Passport
- License: **MIT**
- Relying Party: **Yes**
- Identity Provider: **No**
- Target Environment: node.js

Lua

NGINX lua-resty-openidc

- NGINX Relying Party module for OpenID Connect
- License: **Apache 2.0**
- Relying Party: **Yes**
- Identity Provider: **No**
- Target Environment: NGINX Web Server

PHP

phpOIDC

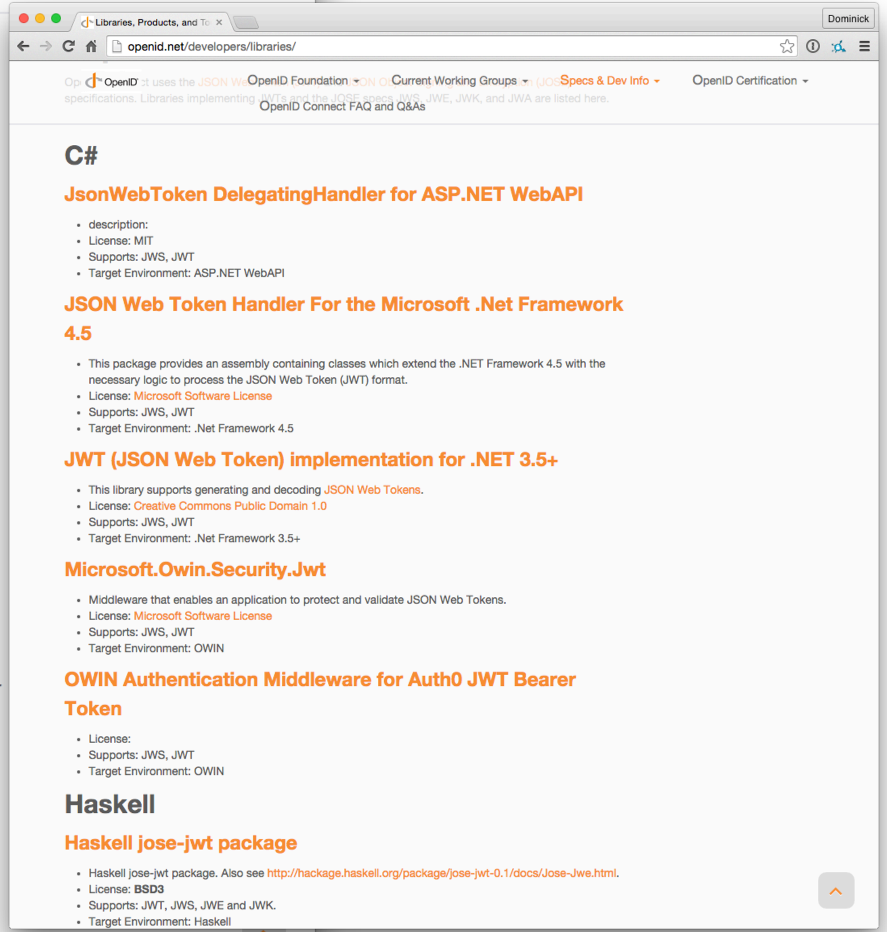
- phpOIDC is a PHP implementation of OpenID Connect, developed by Nomura Research Institute. It also includes the JWT, JWS, and JWE support.
- License: **Apache 2.0**
- Relying Party: **Yes**
- Identity Provider: **Yes**
- Target Environment: Apache, nginx

OpenID-Connect-PHP

- A minimalist library supporting basic client authentication. Aims to make it simple enough for a developer with little knowledge of the OpenID Connect protocol to setup authentication.
- License: **Apache License, Version 2.0**
- Relying Party: **Yes**
- Identity Provider: **No**
- Target Environment: PHP, Apache, Nginx, etc.

oauth2-server-php

- A library for implementing an OAuth2 Server in PHP. Has been extended to support OpenID Connect identity provider functionality.
- License: **MIT License**
- Relying Party: **No**
- Identity Provider: **Yes**
- Target Environment: PHP



C#

JsonWebToken DelegatingHandler for ASP.NET WebAPI

- description:
- License: MIT
- Supports: JWS, JWT
- Target Environment: ASP.NET WebAPI

JSON Web Token Handler For the Microsoft .Net Framework 4.5

- This package provides an assembly containing classes which extend the .NET Framework 4.5 with the necessary logic to process the JSON Web Token (JWT) format.
- License: Microsoft Software License
- Supports: JWS, JWT
- Target Environment: .Net Framework 4.5

JWT (JSON Web Token) implementation for .NET 3.5+

- This library supports generating and decoding JSON Web Tokens.
- License: Creative Commons Public Domain 1.0
- Supports: JWS, JWT
- Target Environment: .Net Framework 3.5+

Microsoft.Owin.Security.Jwt

- Middleware that enables an application to protect and validate JSON Web Tokens.
- License: Microsoft Software License
- Supports: JWS, JWT
- Target Environment: OWIN

OWIN Authentication Middleware for Auth0 JWT Bearer Token

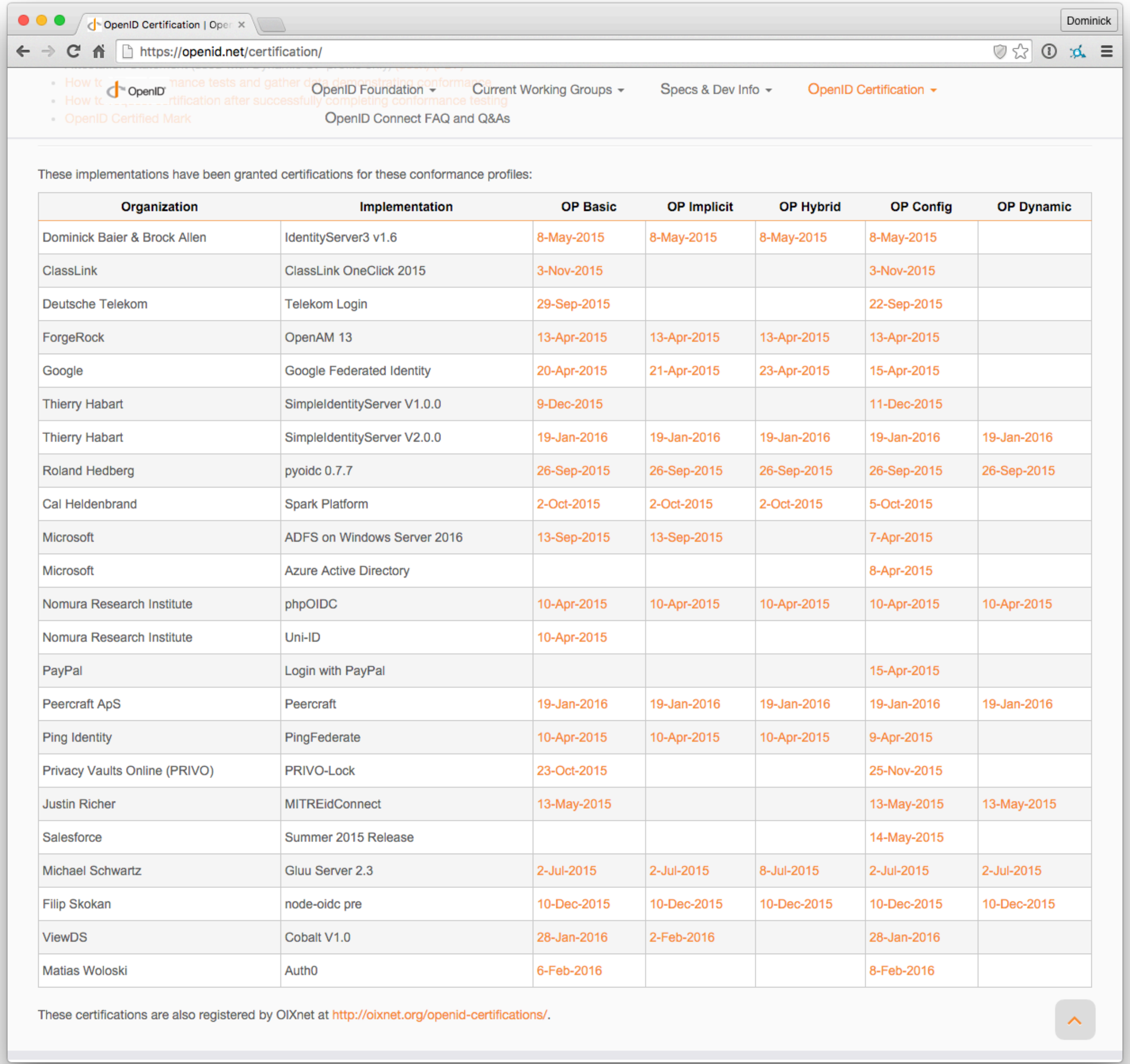
- License:
- Supports: JWS, JWT
- Target Environment: OWIN

Haskell

Haskell jose-jwt package

- Haskell jose-jwt package. Also see http://hackage.haskell.org/package/jose-jwt-0.1/docs/Jose-Jwe.html.
- License: **BSD3**
- Supports: JWT, JWS, JWE and JWK.
- Target Environment: Haskell

OpenID Connect Certification

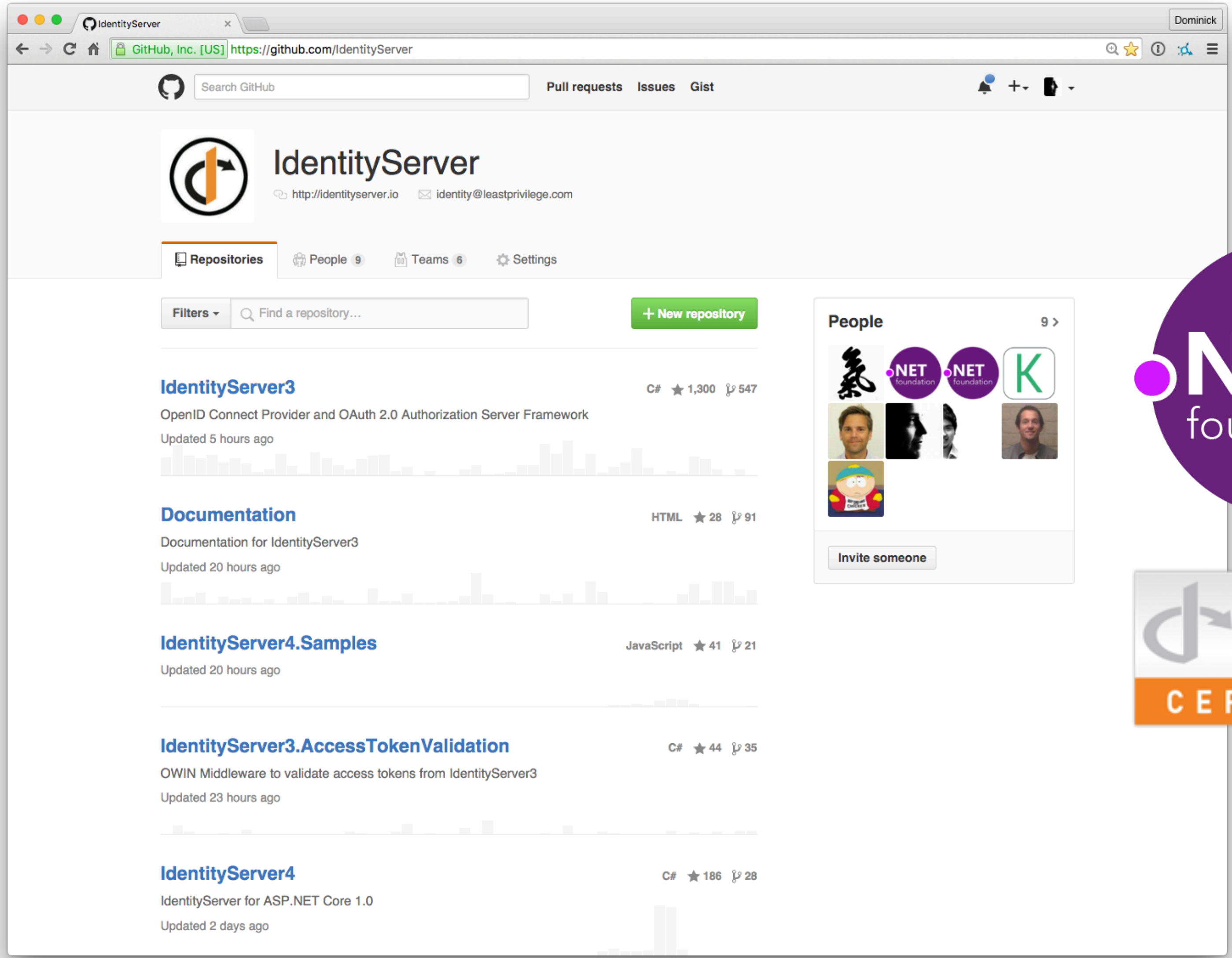


These implementations have been granted certifications for these conformance profiles:

Organization	Implementation	OP Basic	OP Implicit	OP Hybrid	OP Config	OP Dynamic
Dominick Baier & Brock Allen	IdentityServer3 v1.6	8-May-2015	8-May-2015	8-May-2015	8-May-2015	
ClassLink	ClassLink OneClick 2015	3-Nov-2015			3-Nov-2015	
Deutsche Telekom	Telekom Login	29-Sep-2015			22-Sep-2015	
ForgeRock	OpenAM 13	13-Apr-2015	13-Apr-2015	13-Apr-2015	13-Apr-2015	
Google	Google Federated Identity	20-Apr-2015	21-Apr-2015	23-Apr-2015	15-Apr-2015	
Thierry Habart	SimpleIdentityServer V1.0.0	9-Dec-2015			11-Dec-2015	
Thierry Habart	SimpleIdentityServer V2.0.0	19-Jan-2016	19-Jan-2016	19-Jan-2016	19-Jan-2016	19-Jan-2016
Roland Hedberg	pyoidc 0.7.7	26-Sep-2015	26-Sep-2015	26-Sep-2015	26-Sep-2015	26-Sep-2015
Cal Heldenbrand	Spark Platform	2-Oct-2015	2-Oct-2015	2-Oct-2015	5-Oct-2015	
Microsoft	ADFS on Windows Server 2016	13-Sep-2015	13-Sep-2015		7-Apr-2015	
Microsoft	Azure Active Directory				8-Apr-2015	
Nomura Research Institute	phpOIDC	10-Apr-2015	10-Apr-2015	10-Apr-2015	10-Apr-2015	10-Apr-2015
Nomura Research Institute	Uni-ID	10-Apr-2015				
PayPal	Login with PayPal				15-Apr-2015	
Peercraft ApS	Peercraft	19-Jan-2016	19-Jan-2016	19-Jan-2016	19-Jan-2016	19-Jan-2016
Ping Identity	PingFederate	10-Apr-2015	10-Apr-2015	10-Apr-2015	9-Apr-2015	
Privacy Vaults Online (PRIVO)	PRIVO-Lock	23-Oct-2015			25-Nov-2015	
Justin Richer	MITREidConnect	13-May-2015			13-May-2015	13-May-2015
Salesforce	Summer 2015 Release				14-May-2015	
Michael Schwartz	Gluu Server 2.3	2-Jul-2015	2-Jul-2015	8-Jul-2015	2-Jul-2015	2-Jul-2015
Filip Skokan	node-oidc pre	10-Dec-2015	10-Dec-2015	10-Dec-2015	10-Dec-2015	10-Dec-2015
ViewDS	Cobalt V1.0	28-Jan-2016	2-Feb-2016		28-Jan-2016	
Matias Woloski	Auth0	6-Feb-2016			8-Feb-2016	

These certifications are also registered by OIXnet at http://oixnet.org/openid-certifications/.

IdentityServer



IdentityServer

http://identityserver.io identity@leastprivilege.com

Repositories | People 9 | Teams 6 | Settings

IdentityServer3 — C# ★ 1,300 547
OpenID Connect Provider and OAuth 2.0 Authorization Server Framework
Updated 5 hours ago

Documentation — HTML ★ 28 91
Documentation for IdentityServer3
Updated 20 hours ago

IdentityServer4.Samples — JavaScript ★ 41 21
Updated 20 hours ago

IdentityServer3.AccessTokenValidation — C# ★ 44 35
OWIN Middleware to validate access tokens from IdentityServer3
Updated 23 hours ago

IdentityServer4 — C# ★ 186 28
IdentityServer for ASP.NET Core 1.0
Updated 2 days ago

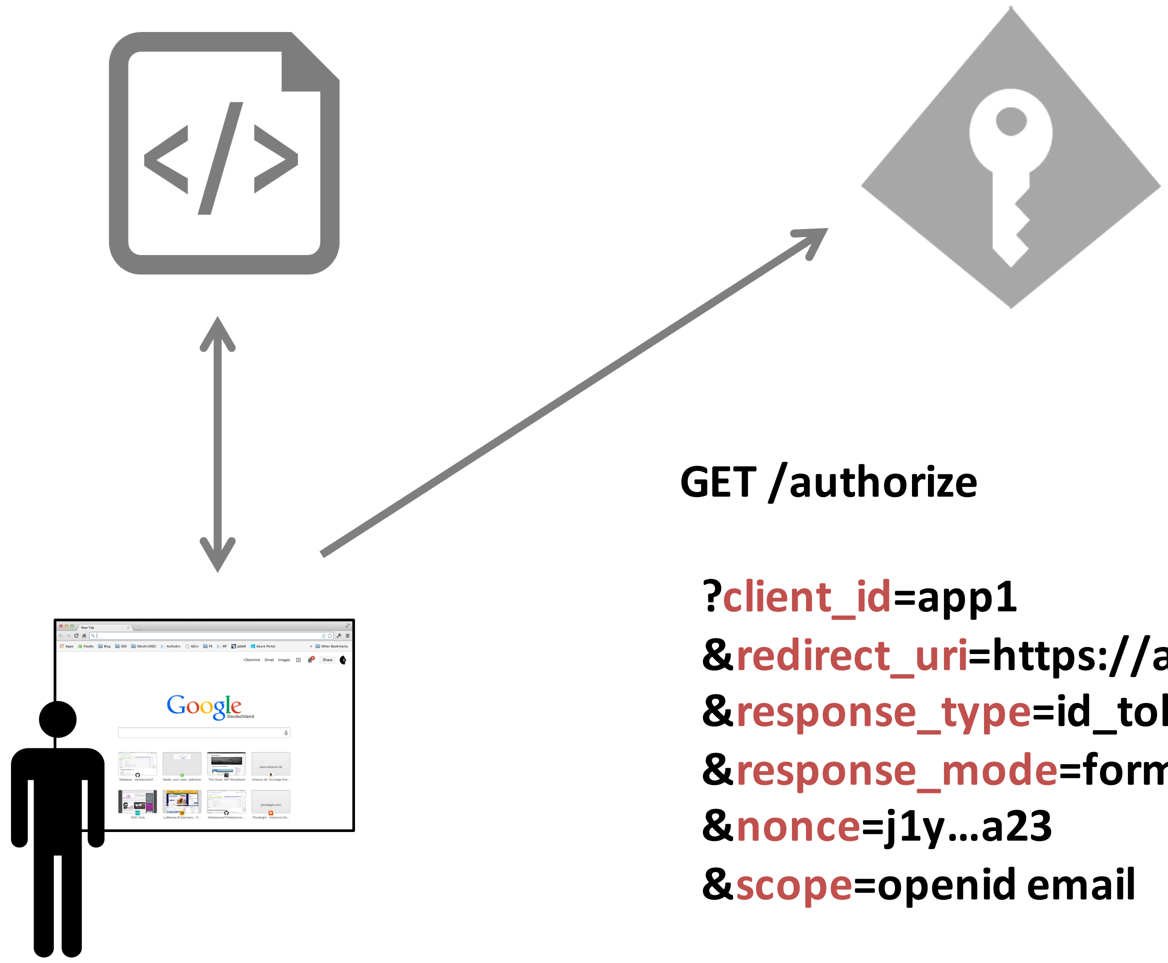
Endpoints

Authorize Endpoint

Token Endpoint

UserInfo Endpoint

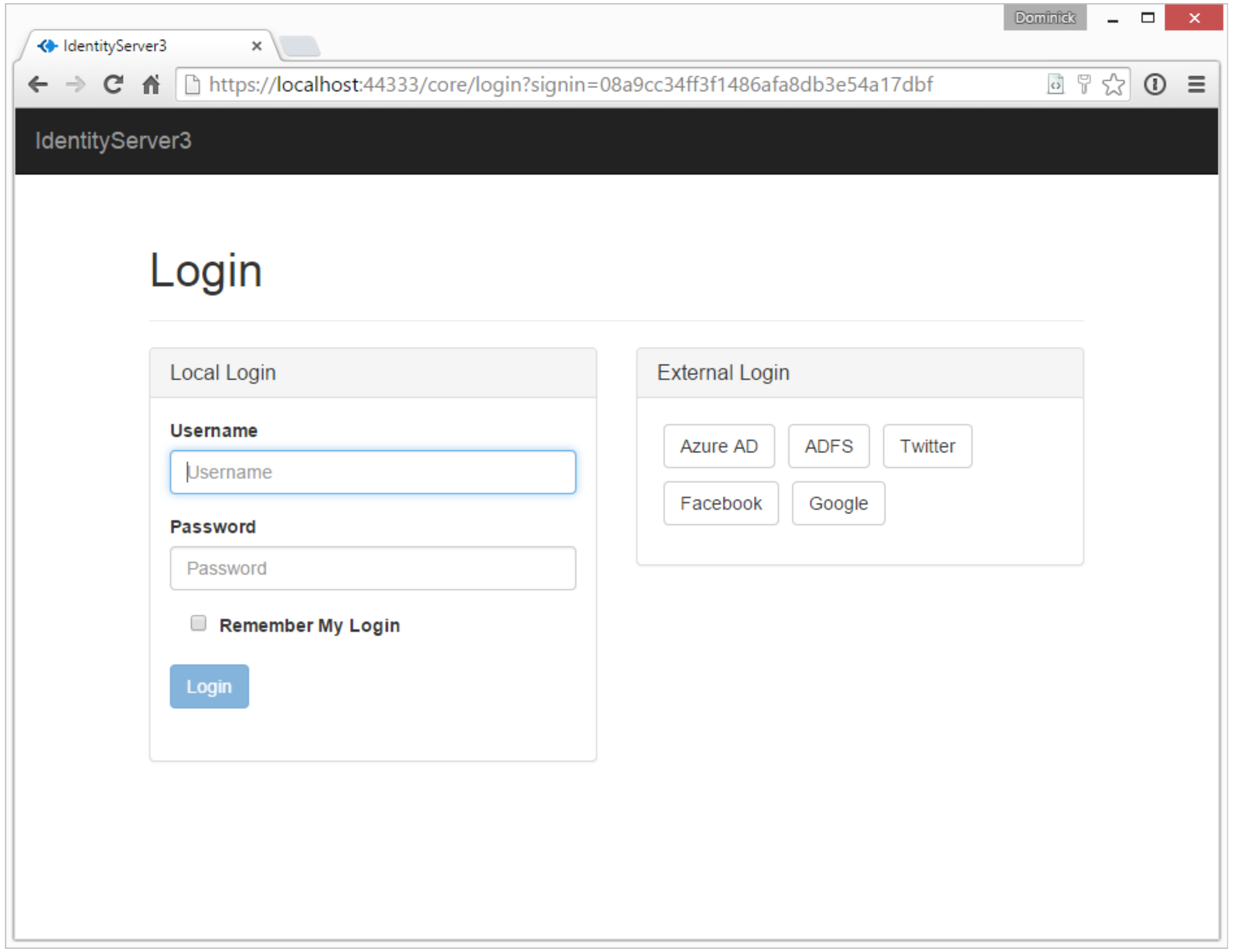
Authentication for Web Applications



GET /authorize

?client_id=app1
&redirect_uri=https://app.com/cb
&response_type=id_token
&response_mode=form_post
&nonce=j1y…a23
&scope=openid email

Authentication



https://localhost:44333/core/login?signin=08a9cc34ff3f1486afa8db3e54a17dbf

IdentityServer3

Login

Local Login

Username

Password

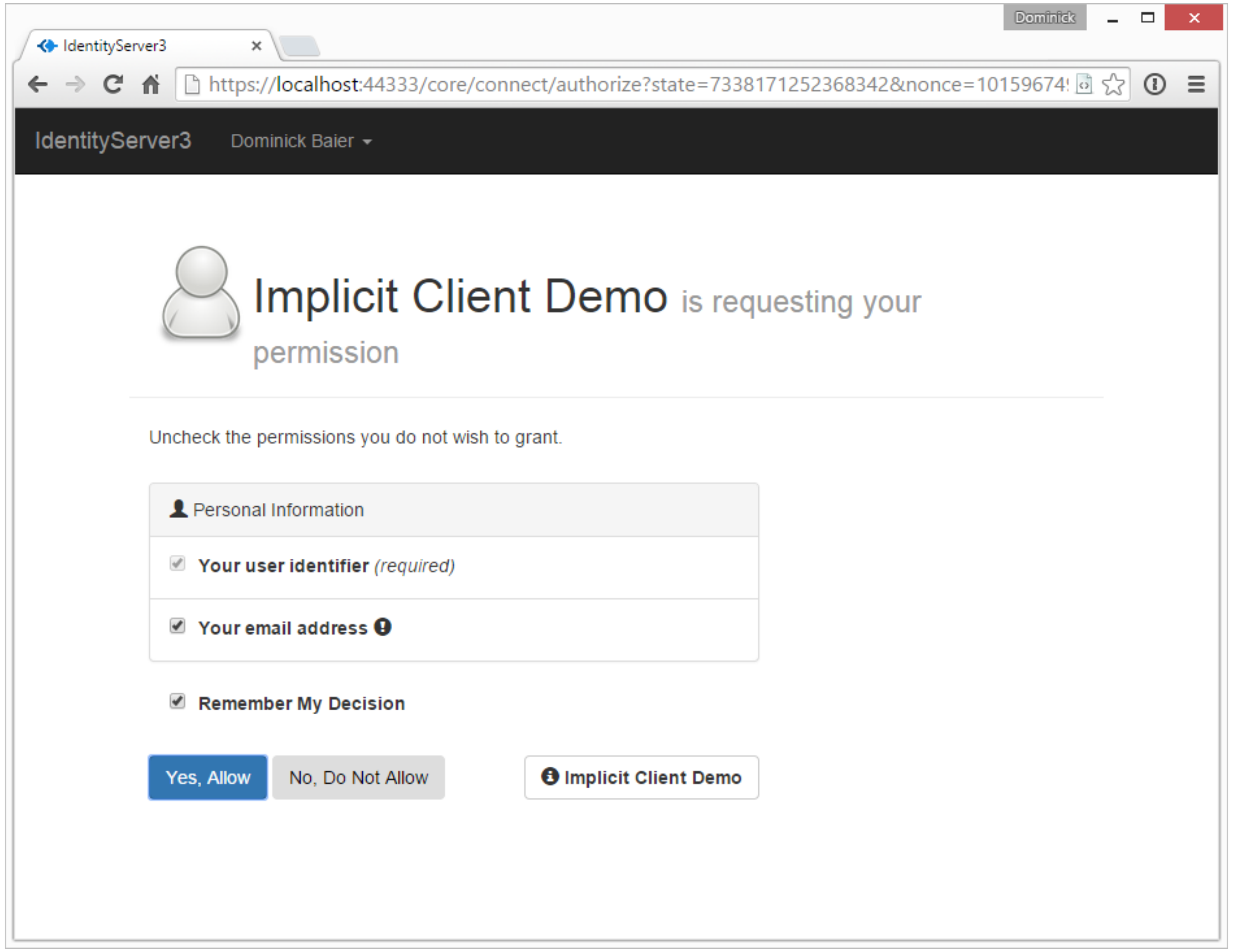
Remember My Login

External Login

Scopes

Scope	Claims
profile	name, family_name, given_name, middle_name, nickname, preferred_username, profile, picture, website, gender, birthdate, zoneinfo, locale, and updated_at
email	email, email_verified
address	address
phone	phone_number, phone_number_verified
offline_access	requests refresh token

Consent



Dominick

IdentityServer3

https://localhost:44333/core/connect/authorize?state=7338171252368342&nonce=10159674!

IdentityServer3 Dominick Baier

Implicit Client Demo is requesting your permission

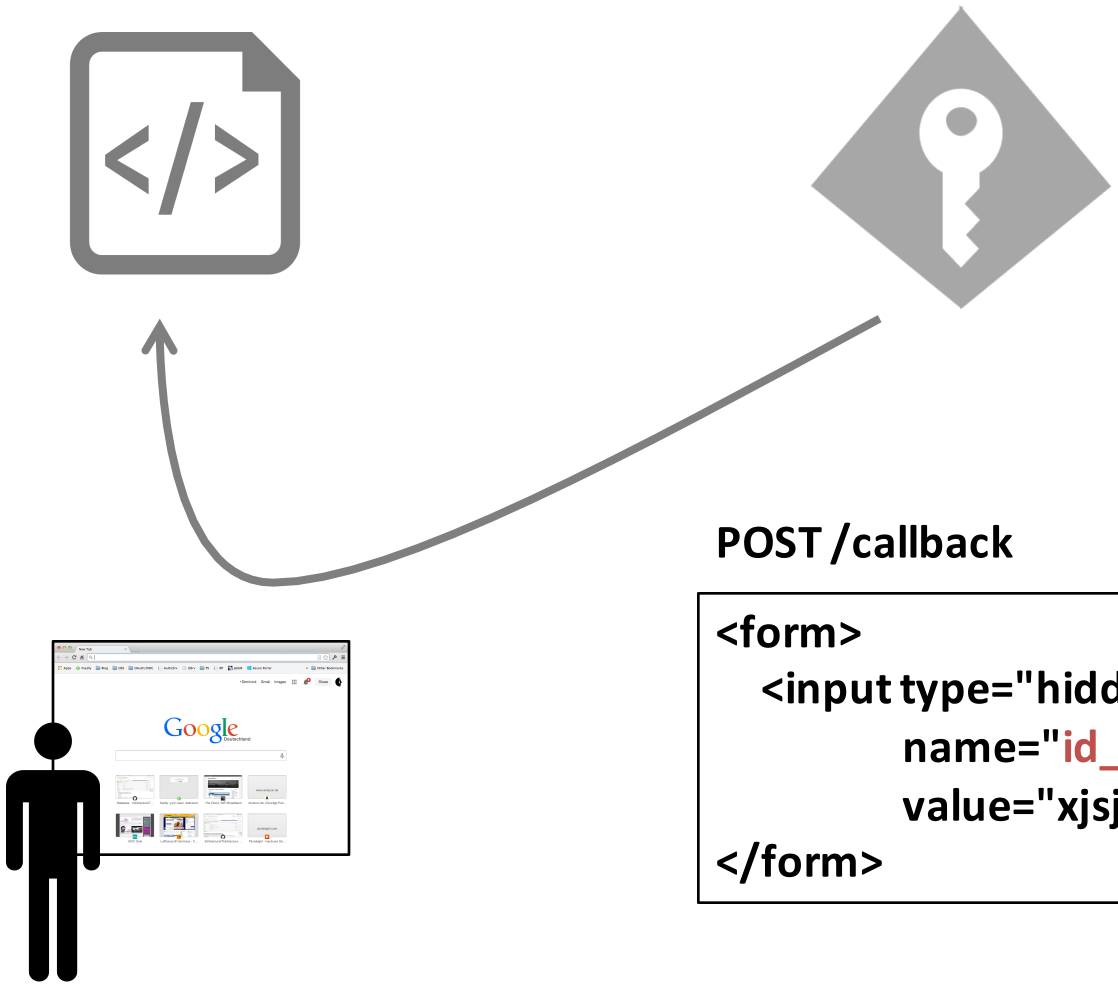
Uncheck the permissions you do not wish to grant.

Personal Information

- [✓] **Your user identifier** *(required)*
- [✓] **Your email address**

- [✓] **Remember My Decision**

Response



POST /callback

```
<form>
   <input type="hidden"
             name="id_token"
             value="xjsj…aas" />
</form>
```

Identity Token

Header

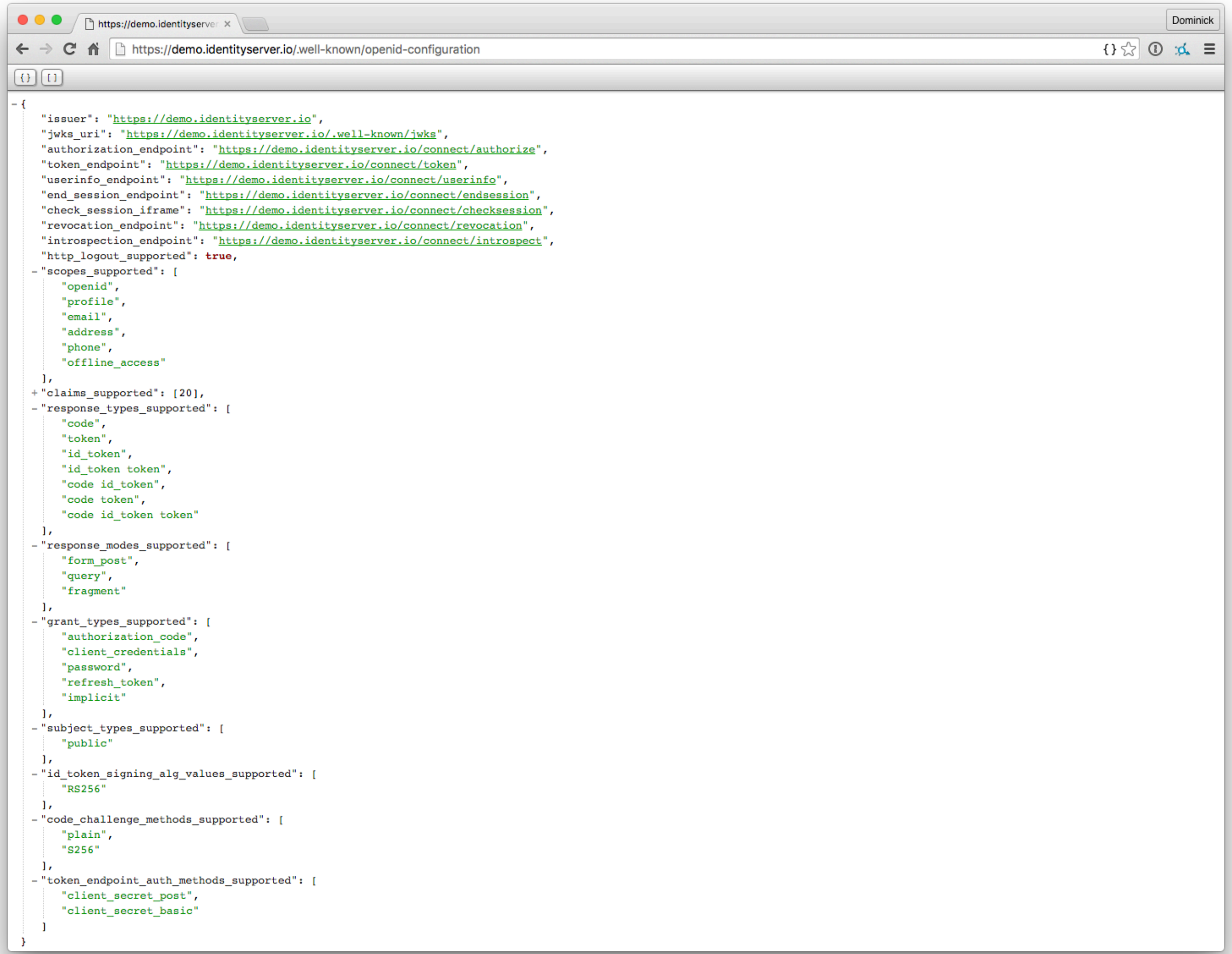
```
{
  "typ": "JWT",
  "alg": "RS256",
  "kid": "mj399j…"
}
```

Payload

```
{
  "iss": "https://idsrv3",
  "exp": 1340819380,
  "aud": "app1",
  "nonce": "j1y…a23",

  "sub": "182jmm199",
  "email": "alice@alice.com",
  "email_verified": true,
  "amr": [ "password" ],
  "auth_time": 12340819300
}
```

$$\underbrace{\texttt{eyJhbGciOiJub25lIn0}}_{\textbf{Header}}\texttt{.}\underbrace{\texttt{eyJpc3MiOiJqb2UiLA0KICJleHAiOjEzMD}}_{\textbf{Payload}}\texttt{.}\underbrace{\texttt{4MTkzODAsDQogImh0dHA6Ly9leGFt}}_{\textbf{Signature}}$$

Discovery


https://demo.identityserve
Dominick
https://demo.identityserver.io/.well-known/openid-configuration

```
- {
    "issuer": "https://demo.identityserver.io",
    "jwks_uri": "https://demo.identityserver.io/.well-known/jwks",
    "authorization_endpoint": "https://demo.identityserver.io/connect/authorize",
    "token_endpoint": "https://demo.identityserver.io/connect/token",
    "userinfo_endpoint": "https://demo.identityserver.io/connect/userinfo",
    "end_session_endpoint": "https://demo.identityserver.io/connect/endsession",
    "check_session_iframe": "https://demo.identityserver.io/connect/checksession",
    "revocation_endpoint": "https://demo.identityserver.io/connect/revocation",
    "introspection_endpoint": "https://demo.identityserver.io/connect/introspect",
    "http_logout_supported": true,
  - "scopes_supported": [
        "openid",
        "profile",
        "email",
        "address",
        "phone",
        "offline_access"
    ],
  + "claims_supported": [20],
  - "response_types_supported": [
        "code",
        "token",
        "id_token",
        "id_token token",
        "code id_token",
        "code token",
        "code id_token token"
    ],
  - "response_modes_supported": [
        "form_post",
        "query",
        "fragment"
    ],
  - "grant_types_supported": [
        "authorization_code",
        "client_credentials",
        "password",
        "refresh_token",
        "implicit"
    ],
  - "subject_types_supported": [
        "public"
    ],
  - "id_token_signing_alg_values_supported": [
        "RS256"
    ],
  - "code_challenge_methods_supported": [
        "plain",
        "S256"
    ],
  - "token_endpoint_auth_methods_supported": [
        "client_secret_post",
        "client_secret_basic"
    ]
}
```

Middleware for OpenID Connect

```
app.UseCookieAuthentication(new CookieAuthenticationOptions
{
    AuthenticationScheme = "Cookies",
    AutomaticAuthenticate = true,
});

JwtSecurityTokenHandler.DefaultInboundClaimTypeMap.Clear();

app.UseOpenIdConnectAuthentication(new OpenIdConnectOptions
{
    AuthenticationScheme = "oidc",
    SignInScheme = "Cookies",

    Authority = "https://url-to-openid-provider",
    ClientId = "your_app_id",
    ResponseType = "id_token",
});
```

Summary

- **ASP.NET Core is a new modular HTTP pipeline**
  - Middleware is central to the architecture
- **Authentication middleware provides user authentication services**
- **AuthenticationManager coordinates authentication middleware**
- **Policy- and resource-based authorization improvements**