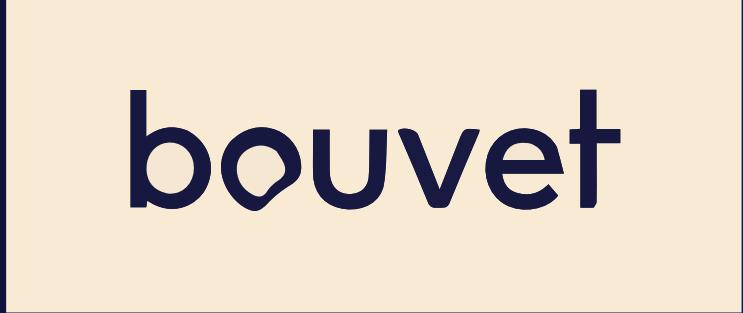


# ID AND ACCESSCONTROL

**Kenneth Fossen**

**Dragefjellet**



**bouvet**



# GITHUB REPO

<https://github.com/kenneth-fossen/kefo-aspnet-security.git>

# \$ WHOAMI

```
$ whoami
{
  name: Kenneth Fossen,
  dep: Dragefjellet@Bouvet,
  email: kenneth.fossen@bouvet.no,
  edu: [
    Bachelor i Datatryggleik
    Master i Programvare Utvikling
  ],
  work: [
    Helse Vest IKT Drift & Sikkerhet
  ]
  current: [
    Software Developer,
    Security Champion,
    #rustaceans
  ]
}
```

Dette er Boisy →



# AGENDA

- **Part i - Theory Covers**
  - **Why this is important**
  - **Acronyms**
  - **Access Control Types**
  - **Access Control in ASP.NET WebAPI**
- **Part i - Coding**

# AGENDA - CONT

- Part ii - Theory Covers
  - What is OAuth / OpenID Connect
  - Tokens (JWT, ID Tokens, Access Tokens)
  - Scopes
  - Flows
    - Implicit Flow
    - Auth Code Grant
- Part ii - Coding

# **PART I**

## **Theory AccessControl**

# WHY IS THIS IMPORTANT?

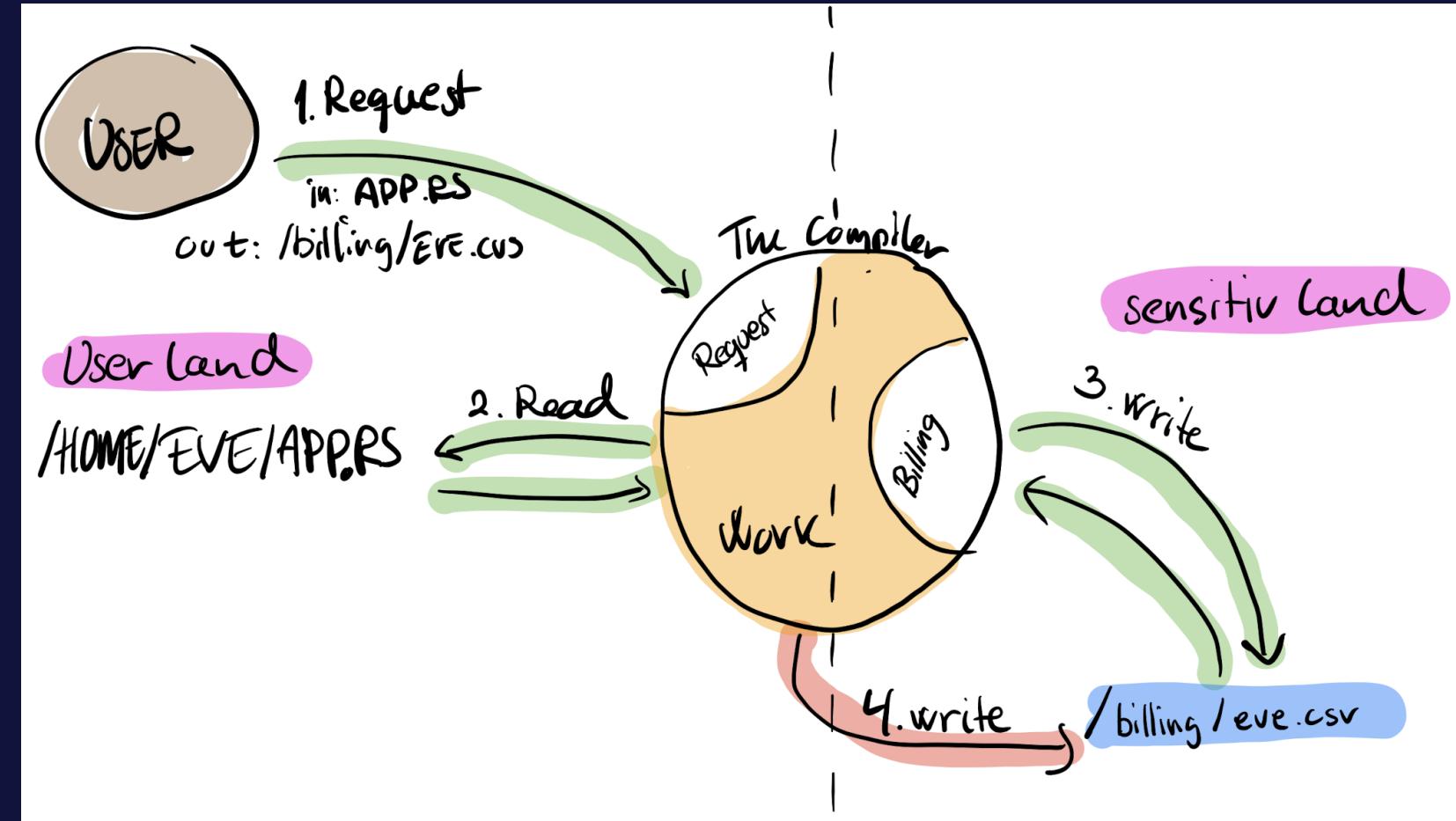
- OWASP Top 10 2021
  - No 1 issue
  - Moving up from the fifth position,
  - 94% of applications were tested for some form of broken access control"
  - Confused Deputy problem
  - PoLP

# CONFUSED DEPUTY



Confused deputy problem. (2022, August 5). In Wikipedia.  
[https://en.wikipedia.org/wiki/Confused\\_deputy\\_problem](https://en.wikipedia.org/wiki/Confused_deputy_problem)

# EVIL USER



# POLP / POLA

Principle of Least Privilege / Access

# ACRONYMS

- principal
- subject
- authentication (AuthN)
- authorization (AuthZ)
- scopes
- authority
- claims
- tokens /JWT bearer

# JWT TOKEN

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9  
.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvG4gRG9lIiwiWF0IjoxNTE2MjM5MDIyfQ  
.SflKxwRJSMeKKF2QT4fwpMeJf36P0k6yJV_adQssw5c
```

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
},  
{  
  "sub": "1234567890",  
  "name": "John Doe",  
  "iat": 1516239022,  
}
```

# ACCESS CONTROL TYPES

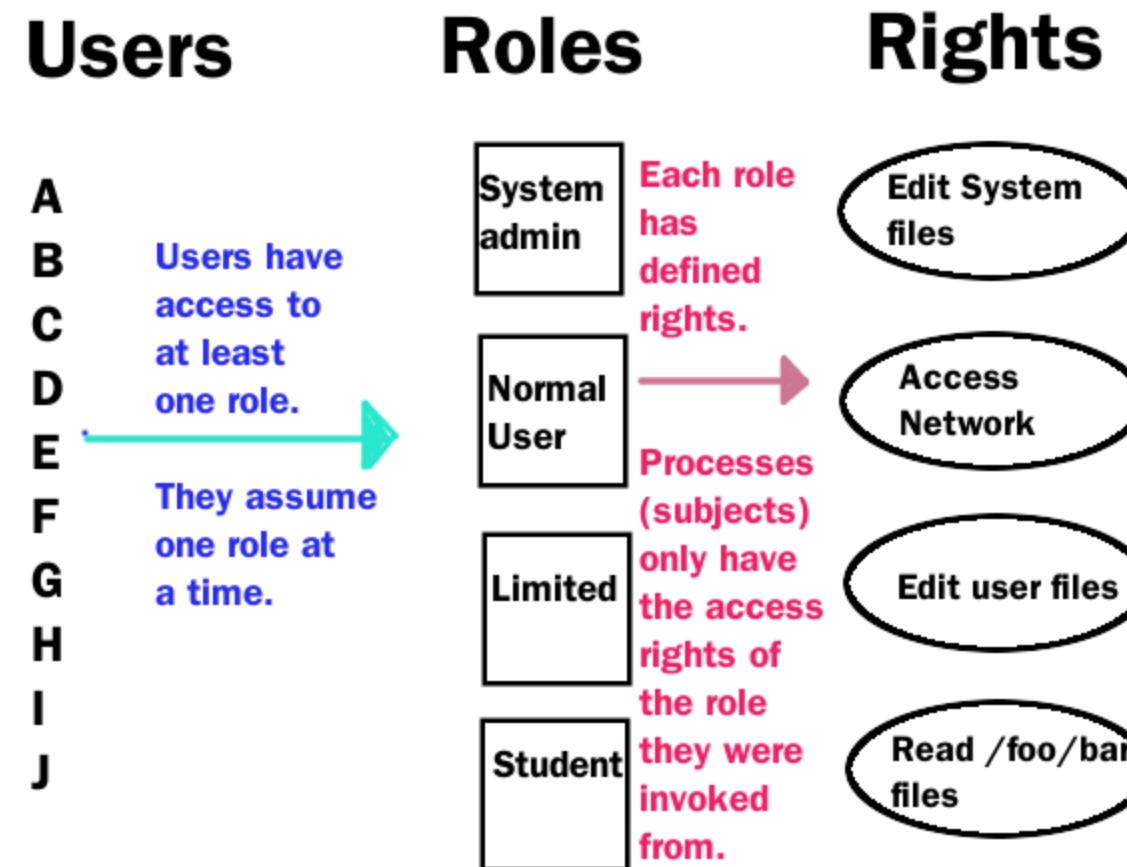
ACL, RBAC, ABAC, CBAC, RBA

# ACCESS CONTROL LISTS

```
> ls -la
total 0
drwxr-xr-x  5 kenneth  staff  160  Apr  5 10:48 .
drwxr-xr-x 153 kenneth  staff 4896  Apr  5 10:48 ..
-r-xr-xr-x  1 kenneth  staff  0   Apr  5 10:48 read_execute_file.txt
-r--r--r--  1 kenneth  staff  0   Apr  5 10:48 read_file.txt
--w-------
 1 kenneth  staff  0   Apr  5 10:48 write_file.txt
-rw--xr--  1 kenneth  staff  0   Apr  5 10:48 own_readwrite_everyone_read.txt
>
```

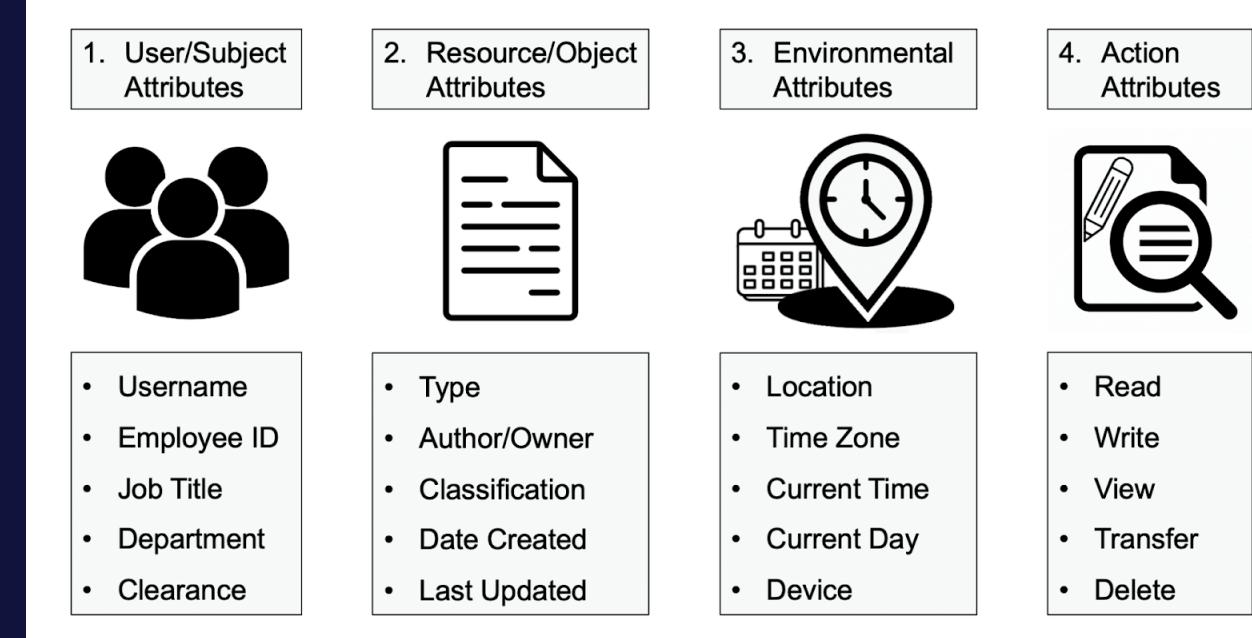
```
{-}{{r-x}{ r-x}{r-x}}
{type}{owner}{group}{everyone}
```

# ROLE-BASED ACCESS CONTROL

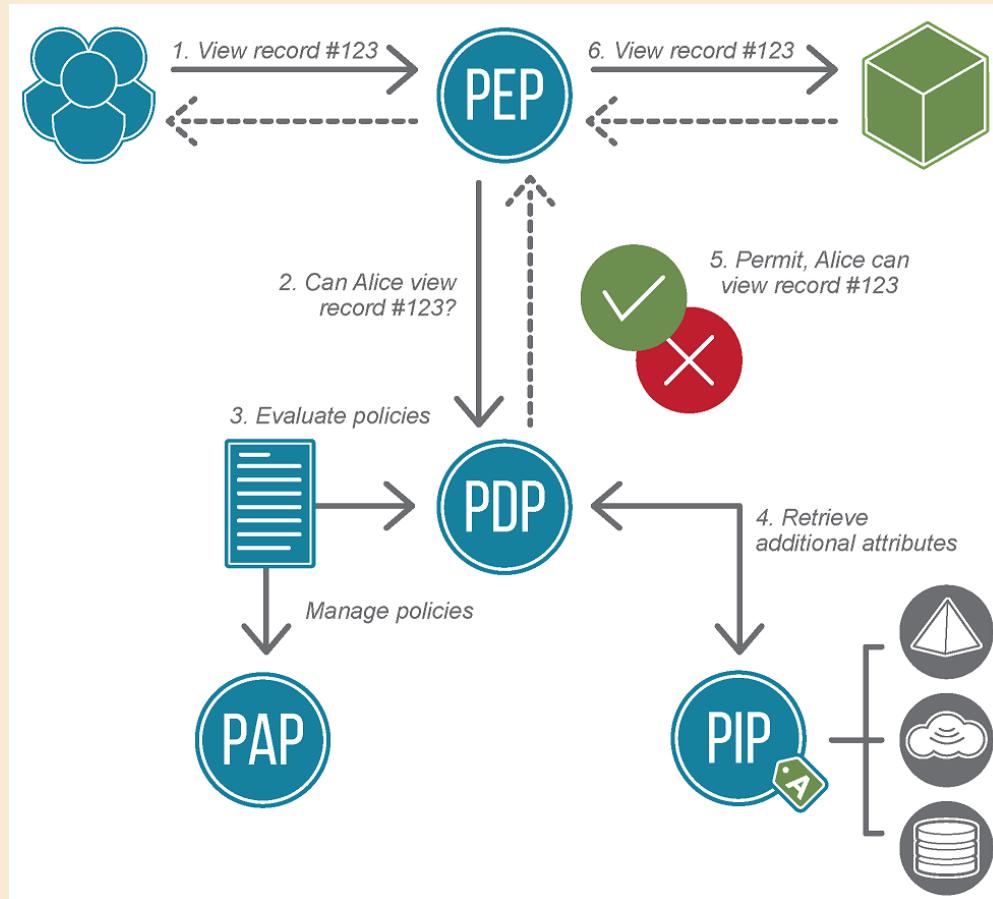


# ATTRIBUTE-BASED ACCESS CONTROL

- Policy Decision Point (PDP)
- Policy Enforcement Point (PEP)
- Policy Information Point (PIP)



# ABAC INFRASTRUCTURE



## Exploring Capability-based security in software design with Rust

---

*Author:* Kenneth Fossen

*Supervisor:* Håkon Robbestad Gylterud

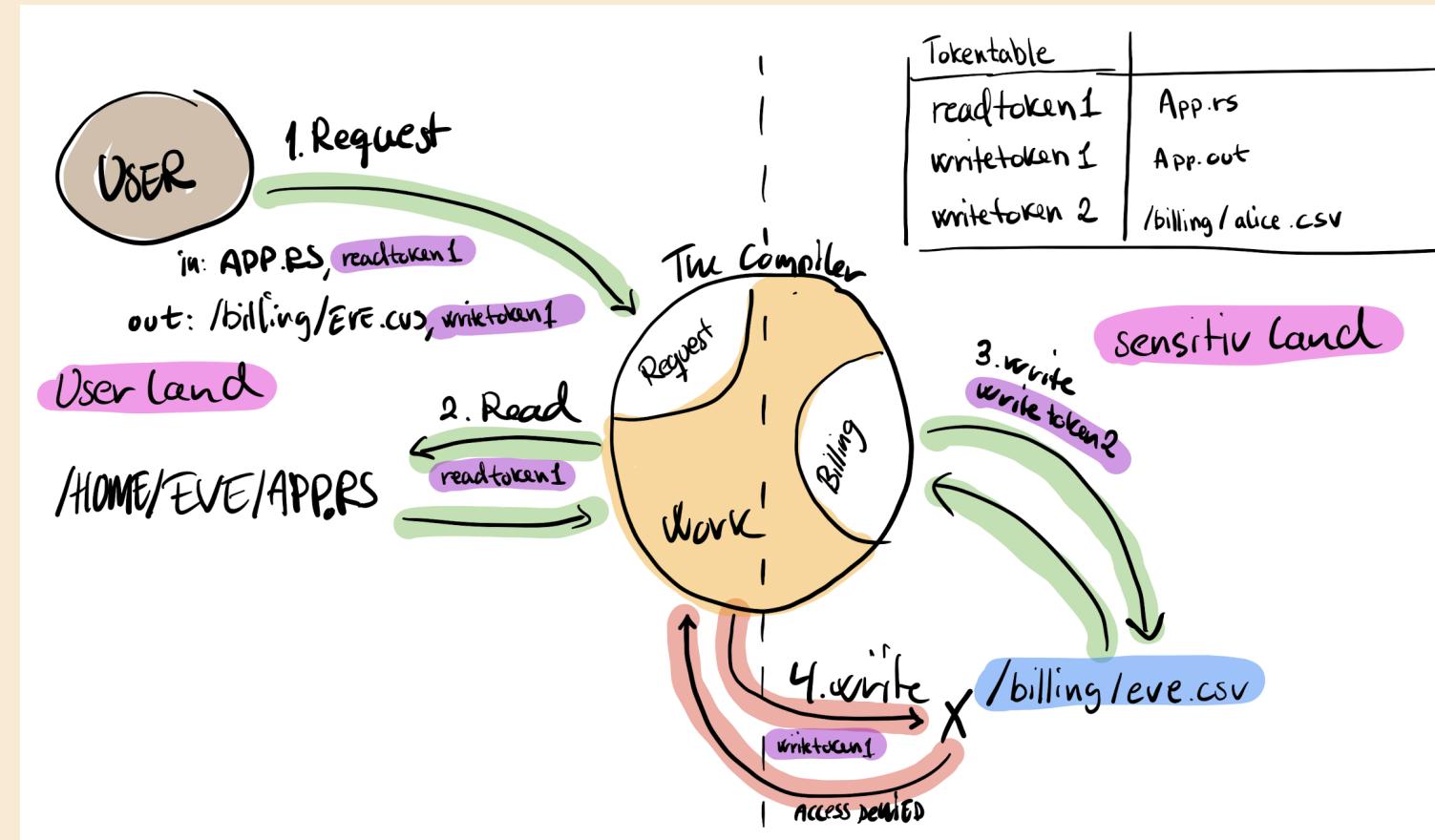
# CAPABILITY-BASED ACCESS CONTROL



UNIVERSITETET I BERGEN  
*Det matematisk-naturvitenskapelige fakultet*

June, 2022

# CAPABILITIES



# RISK-BASED AUTHENTICATION

<https://riskbasedauthentication.org/>

# ACCESS CONTROL IN ASP.NET WEBAPI

- ClaimsBased
- PolicyBased

ABAC is what Microsoft calls: policy-based access control (PBAC) or claims-based access control (CBAC)

# JWT

```
{  
  "aud": "api://1851a809-314c-4d44-9844-382ce9f64f85",  
  "iss": "https://sts.windows.net/e5d441e9-b0fb-4458-a4d7-15d2b4cf193/",  
  "iat": 1681404264,  
  "nbf": 1681404264,  
  "exp": 1681408164,  
  "idp": "https://sts.windows.net/e5d441e9-b0fb-4458-a4d7-15d2b4cf193/",  
  "oid": "b3da45cf-872e-41af-b1b6-713852002cac",  
  "scp": [  
    "profile",  
    "email",  
  ],  
  "roles": [  
    "ConsoleClientRole"  
  ],  
  "sub": "b3da45cf-872e-41af-b1b6-713852002cac",  
  //..  
}
```

# CLAIMS

```
iss:https://sts.windows.net/e5d441e9-b0fb-4458-a4d7-15d2b4cf193/
iat:1681444727
nbf:1681444727
exp:1681448627
aio:E2ZgYFhy8uvSUM2J0PWhC2NffE7RAQA=
appid:f44c4125-7493-4ea0-8a91-77a15e9995df
appidacr:1
idp:https://sts.windows.net/e5d441e9-b0fb-4458-a4d7-15d2b4cf193/
oid:b3da45cf-872e-41af-b1b6-713852002cac
rh:0.AYEA6UHU5fuwWESk1xXStM_BkwmoURhMMURNmEQ4L0n2T4WBAAA.
roles:ConsoleClientRole
sub:b3da45cf-872e-41af-b1b6-713852002cac
tid:e5d441e9-b0fb-4458-a4d7-15d2b4cf193
uti:eqdCjChDh0qEAdbXP-o3AA
ver:1.0
```

# CLAIMSTRANSFORMATION

```
public class SecuredApiTransformClaims : IClaimsTransformation
public Task<ClaimsPrincipal> TransformAsync(ClaimsPrincipal principal)
```

# AUTHN AND AUTHZ

```
app.UseAuthentication()  
app.UseAuthorization()
```

# EXAMPLE: MIDDLEWARE ORDER

```
var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
...
var app = builder.Build();

// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
    app.UseMigrationsEndPoint();
}
else
{
    app.UseExceptionHandler("/Error");
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();
// app.UseCookiePolicy();

app.UseRouting();
// app.UseRequestLocalization();
// app.UseCors();

app.UseAuthentication(); // Always before Authorization
app.UseAuthorization();
// app.UseSession();
// app.UseResponseCompression();
// app.UseResponseCaching();

app.MapRazorPages();
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();
```

# [AUTHORIZE] AND POLICIES

[AllowAnonymous]

[Authorize]

[Authorize(Policy = "ConsoleClientRole")]

[Authorize(Roles = "HRManager,Finance")]

[Authorize(Roles = "PowerUser")]

[Authorize(Roles = "ControlPanelUser")]

```
builder.Services.AddAuthorization(options =>
{
    options.AddPolicy("ConsoleClientRole", policy =>
    {
        policy.RequireRole("ConsoleClientRole");
    });
});
```

# PART I - BREAK BEFORE CODING



# PART I - CODING

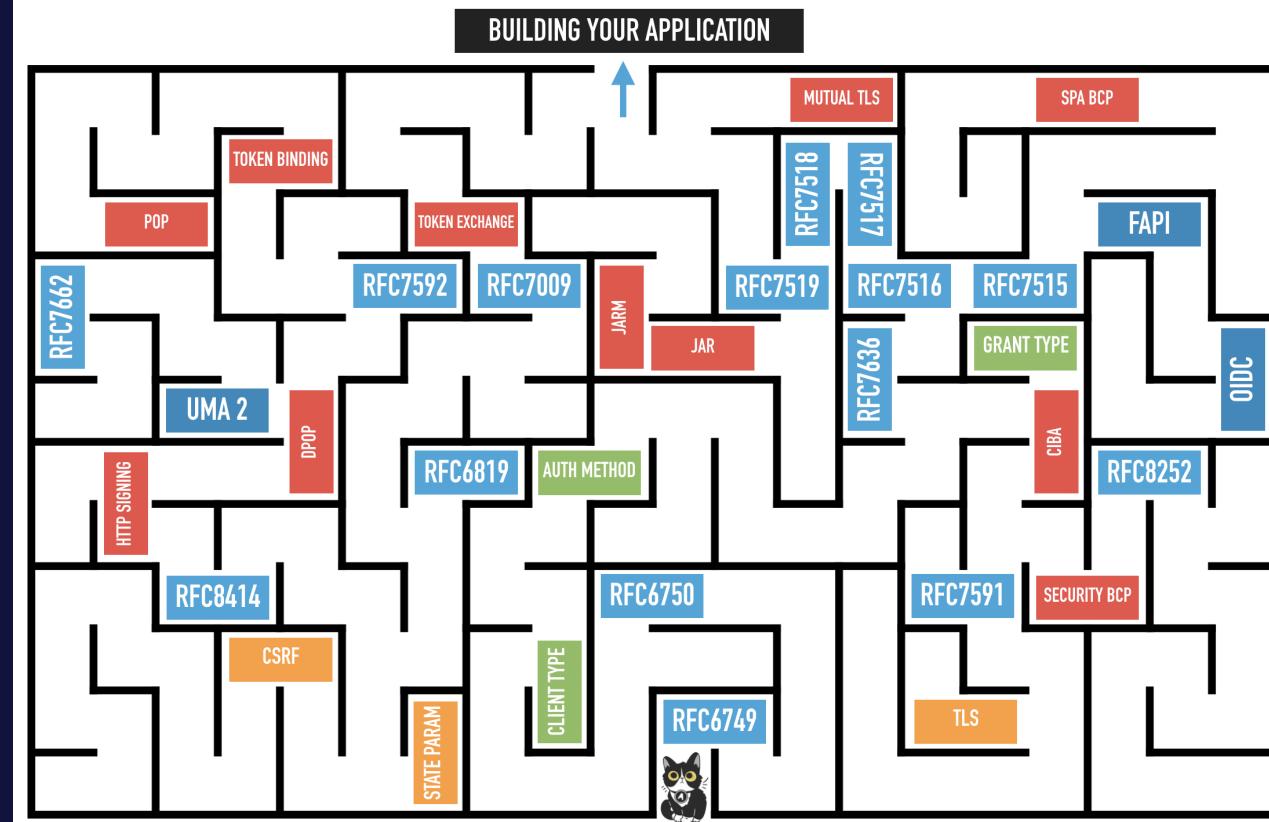
Tasks for Part i

# **PART II**

## **Theory OAuth and Azure**

# OAuth

- What is OAuth



Source: <https://aaronparecki.com/2019/12/21/its-time-for-oauth-2-dot-1>

# OAUTH ACRONYMS

- Audience
- Issuer
- Scope
- Subject

# JWT

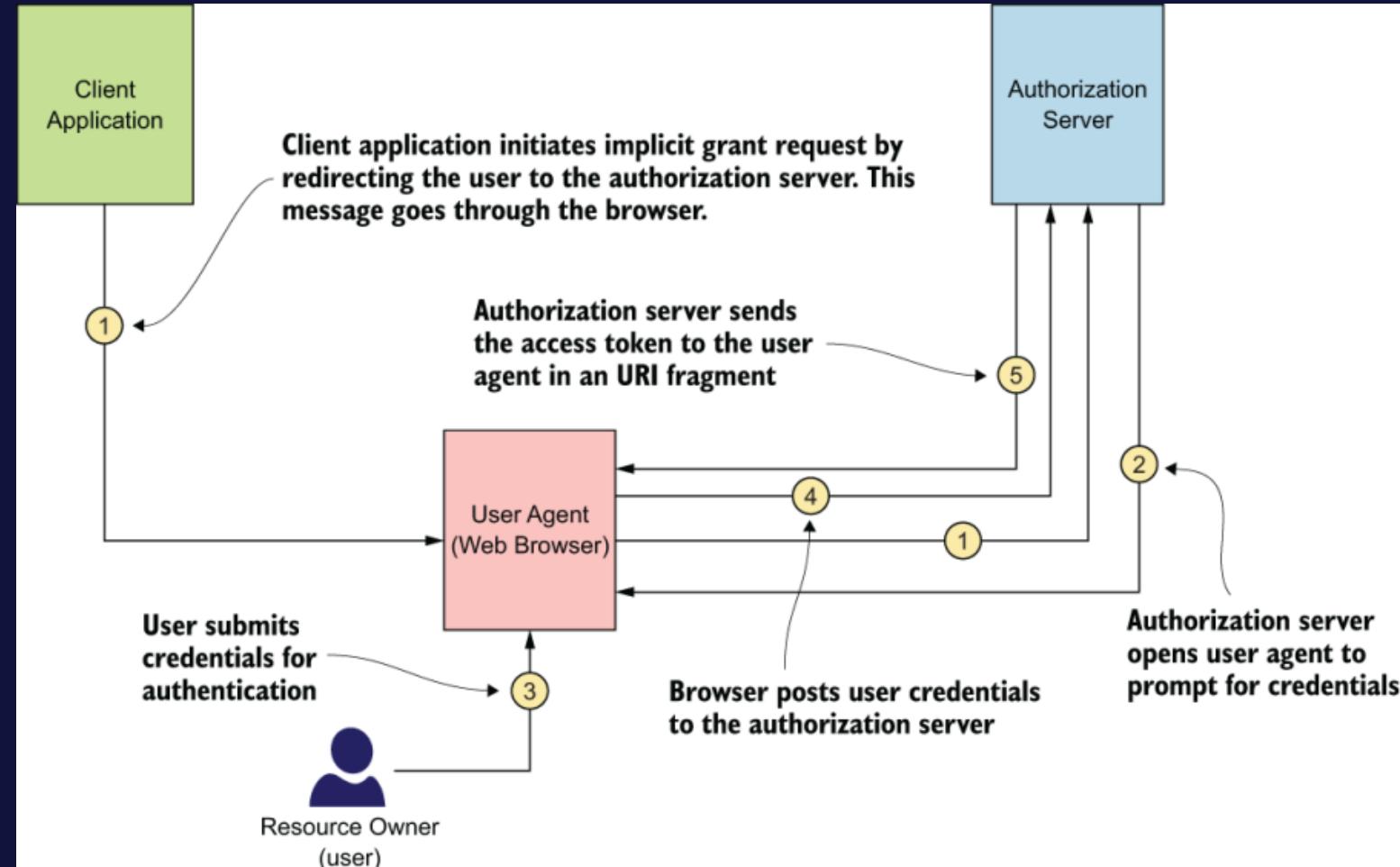
```
{  
  "aud": "api://1851a809-314c-4d44-9844-382ce9f64f85",  
  "iss": "https://sts.windows.net/e5d441e9-b0fb-4458-a4d7-15d2b4cf193/",  
  "iat": 1681404264,  
  "nbf": 1681404264,  
  "exp": 1681408164,  
  "idp": "https://sts.windows.net/e5d441e9-b0fb-4458-a4d7-15d2b4cf193/",  
  "oid": "b3da45cf-872e-41af-b1b6-713852002cac",  
  "scp": [  
    "profile",  
    "email",  
  ],  
  "roles": [  
    "ConsoleClientRole"  
  ],  
  "sub": "b3da45cf-872e-41af-b1b6-713852002cac",  
  //..  
}
```

- Tokens
  - Id Tokens
  - Access Tokens
  - Scopes

# FLOWS

- Implicit Grant Flow
- Authorization Code Grant (PCKE)

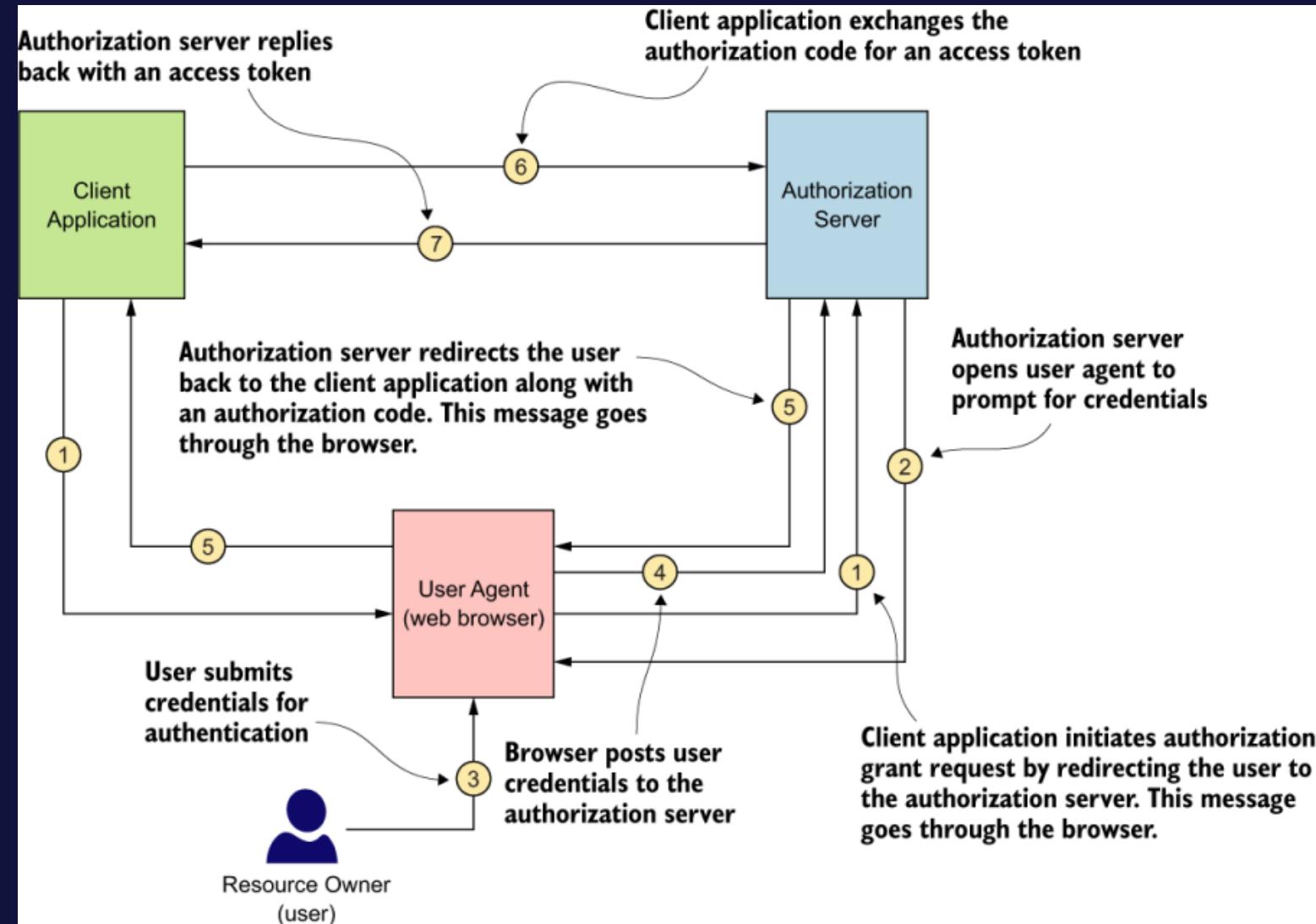
# IMPLICIT GRANT FLOW



# AUTHORIZATION CODE GRANT (PCKE)

- RedirectURIs
  - Common Mistake Open Redirect

# Auth Code Flow



# AZUREAD

- Service Principals / AppRegistrations
  - an identity created for use with applications, hosted services, and automated tools to access Azure resources
  - Users have to manage secrets and passwords
- Managed Identity
  - Same as a service principal, just managed

## PART II - BREAK BEFORE CODING



# PART II - CODING

Coding

Tasks for Part ii

# SOURCES

- OAuth 2.0 implicit grant flow
- Microsoft identity platform and OAuth 2.0 authorization code flow
- ASP.NET Core Middleware
- Azure.Identity Nuget
- Microsoft Identity Platform Documentation
- MsC: Capability Bases Access Control