WEB DAY

27 MARZO 2024 7A EDIZIONE

EASY-AUTH: IL MODO PIÙ SEMPLICE PER INTEGRARE L'AUTENTICAZIONE MULTI-PROVIDER NEI TUOI APP SERVICE

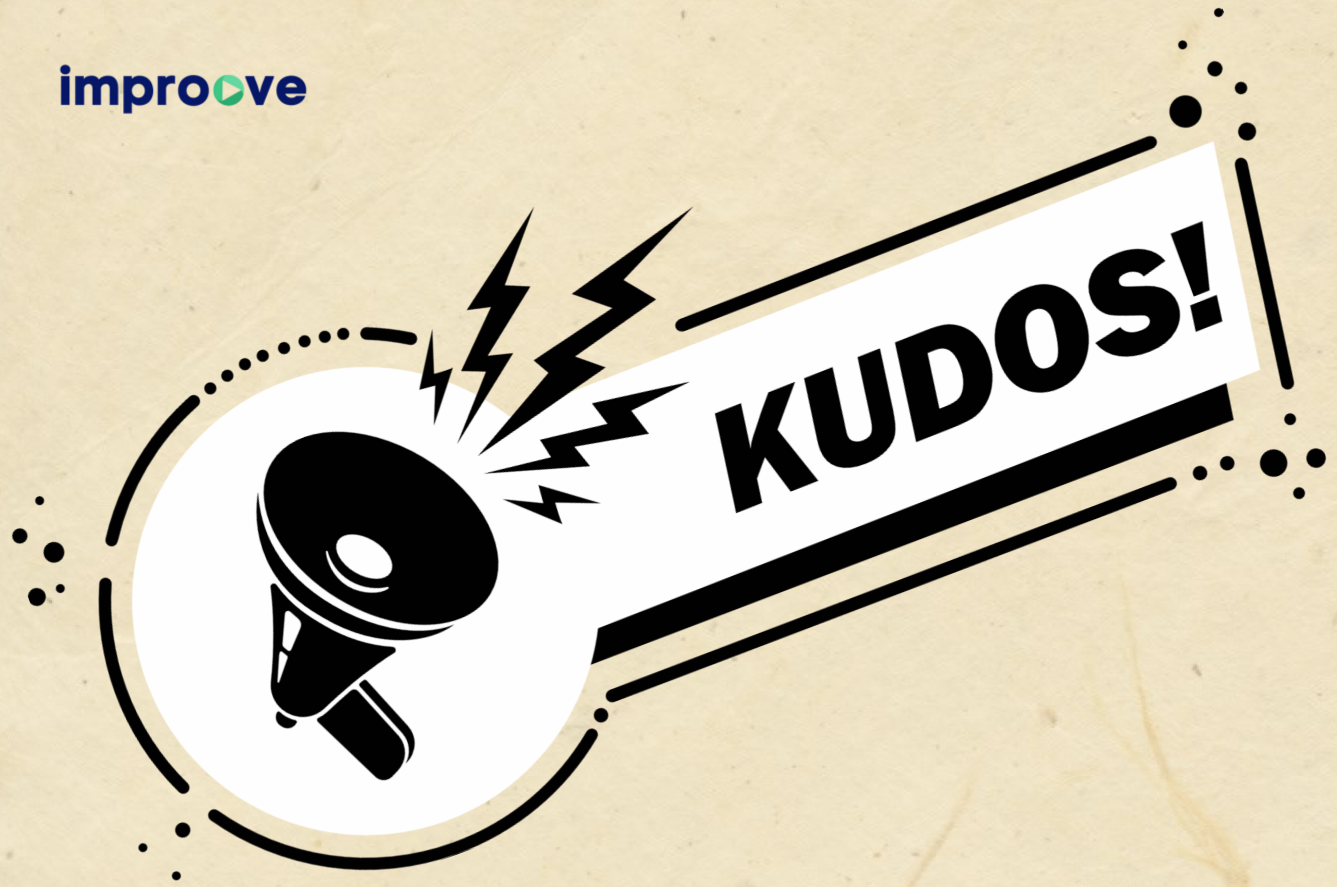
MASSIMO BONANNI

TECHNICAL TRAINER

MICROSOFT

improove

improove



WEB DAY

27 MARZO 2024

7A EDIZIONE

.NET on aws

softwareone

adesso.it

Spreaker From iHeart

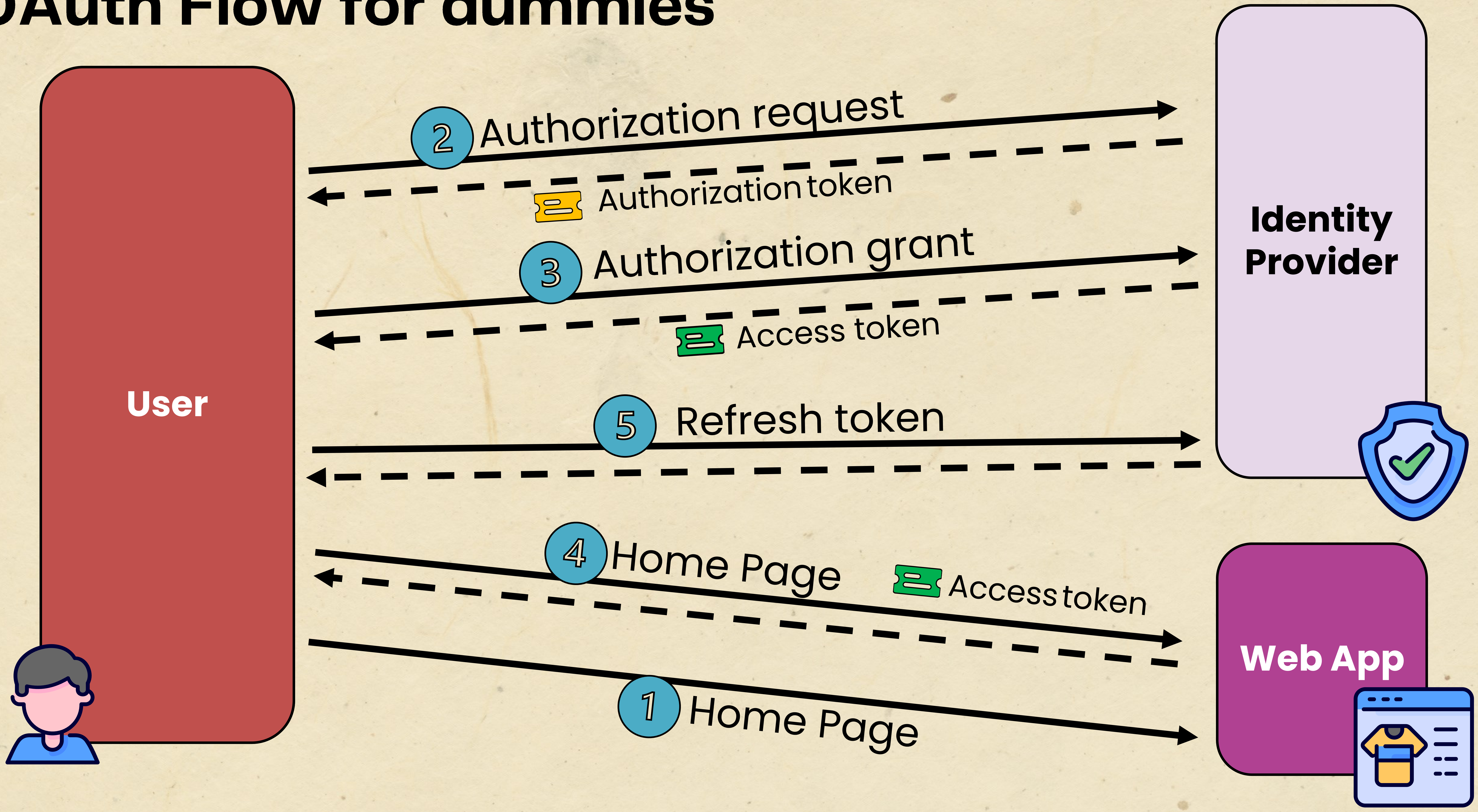
UNIKEY TECH INSIDE

Developer panic in managing authentication

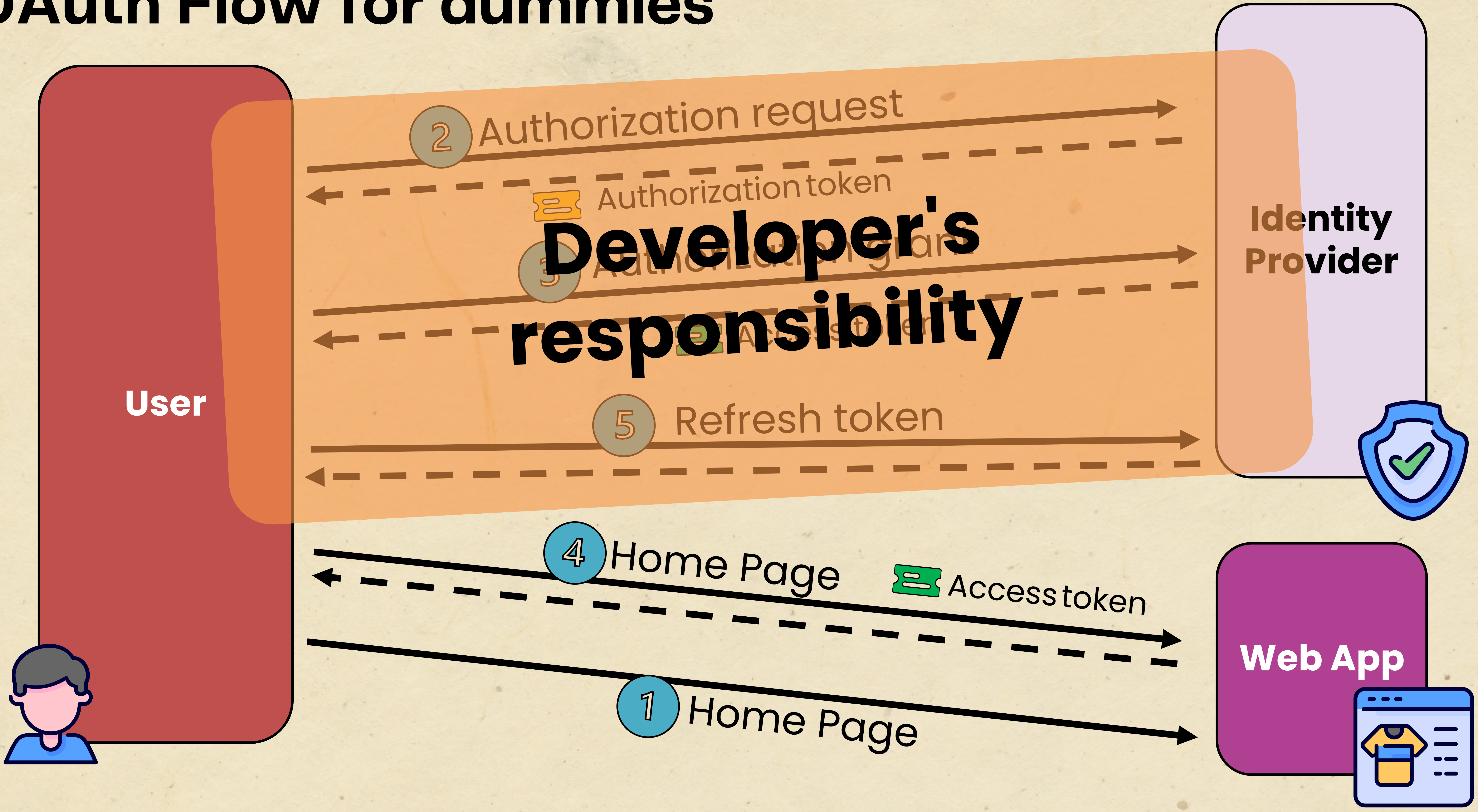
Developer panic in managing authentication

- Authentication can be a complex and time-consuming task for developers
- It requires knowledge of different authentication protocols such as OAuth 2.0 and OpenID Connect
- Authentication involves managing sensitive user information such as passwords and access tokens
- Managing authentication requires a high level of security expertise and can be prone to errors
- Failure to properly manage authentication can lead to security breaches and data leaks

OAuth Flow for dummies



OAuth Flow for dummies



What is Easy-Auth?

Azure App Service provides **built-in** authentication capabilities (sometimes referred to as **"Easy Auth"**)

You can sign in users and access data by **writing minimal or no code** in your App Service, Azure Functions, and Container Apps.

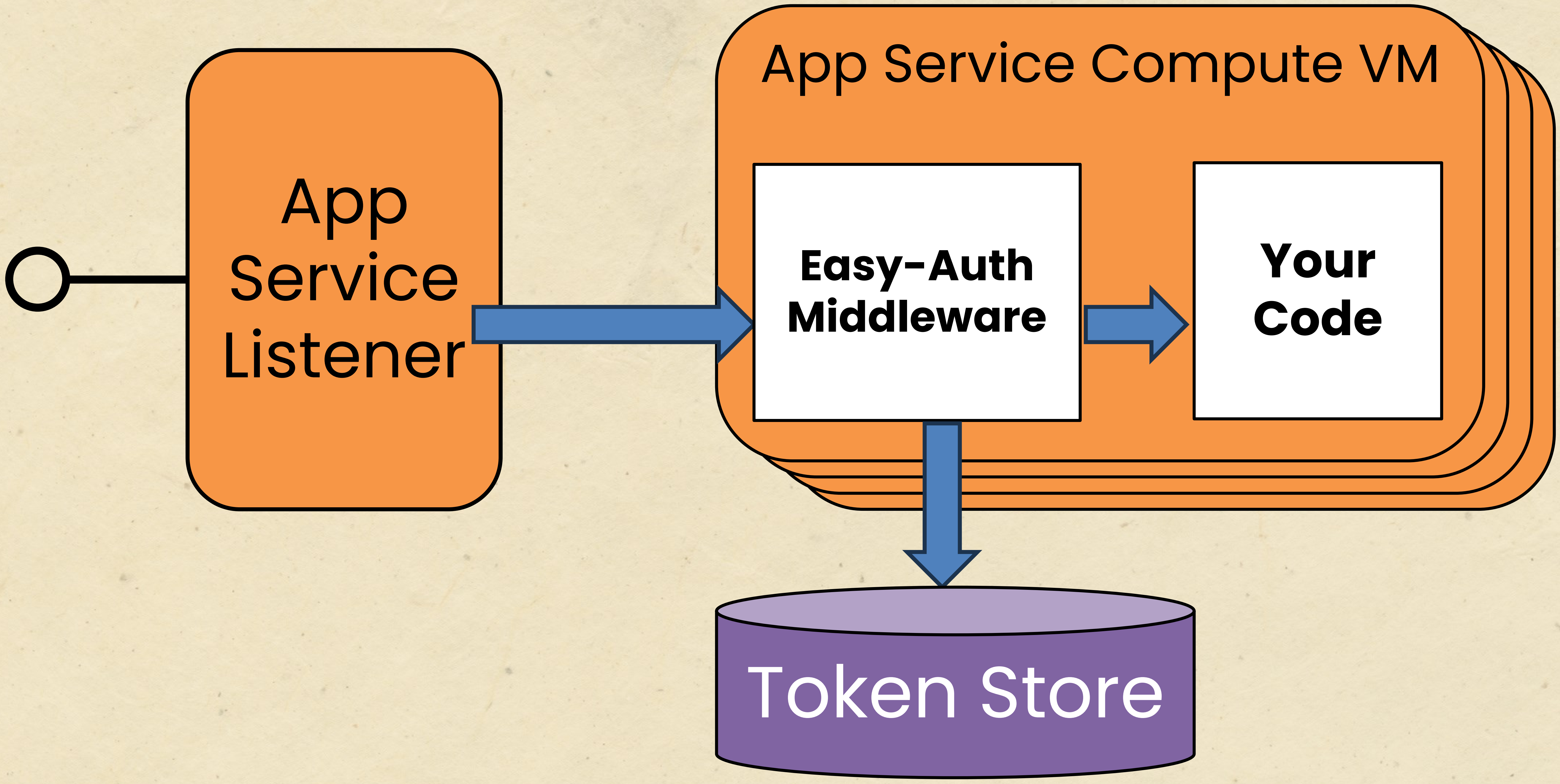
Why use Easy-Auth

Allows you to integrate a variety of auth capabilities into your web app or API without implementing them yourself.

It's built directly into the platform and doesn't require any particular language, SDK, security expertise, or even any code to utilize.

You can integrate with multiple login providers. For example, Microsoft Entra ID, Facebook, Google, Twitter.

Easy-Auth architecture



Easy-Auth architecture

Easy-Auth Middleware

The platform middleware handles several things for your app:

- ✓ Authenticates users and clients with the specified identity provider(s)
- ✓ Validates, stores, and refreshes OAuth tokens issued by the configured identity provider(s)
- ✓ Manages the authenticated session
- ✓ Injects identity information into HTTP request headers

Easy-Auth architecture

Easy-Auth middleware runs as a native IIS module in the same sandbox as your application.

When it's enabled, every incoming HTTP request passes through it before being handled by your application.

The relevant information that your app needs is passed through using request headers

Easy-Auth middleware runs in a separate container, isolated from your application code (uses Ambassador Pattern to perform similar functionality as on Windows).

Because it does not run in-process, no direct integration with specific language frameworks is possible.

The relevant information that your app needs is passed through using request headers.

User claims in your code

Easy-Auth processes tokens from authenticated users **injecting their claims** directly into your code's **request headers**.

These headers **can't** be set by external requests and are exclusively added by App Service.

`X-MS-CLIENT-PRINCIPAL`	A Base64 encoded JSON representation of available claims
`X-MS-CLIENT-PRINCIPAL-ID`	An identifier for the caller set by the identity provider
`X-MS-CLIENT-PRINCIPAL-NAME`	A human-readable name for the caller set by the identity provider, e.g. Email Address, User Principal Name
`X-MS-CLIENT-PRINCIPAL-IDP`	The name of the identity provider used by App Service Authentication

Token Store

In your authenticated app, you must write code to collect, store, and refresh tokens.

App Service provides a built-in token store.

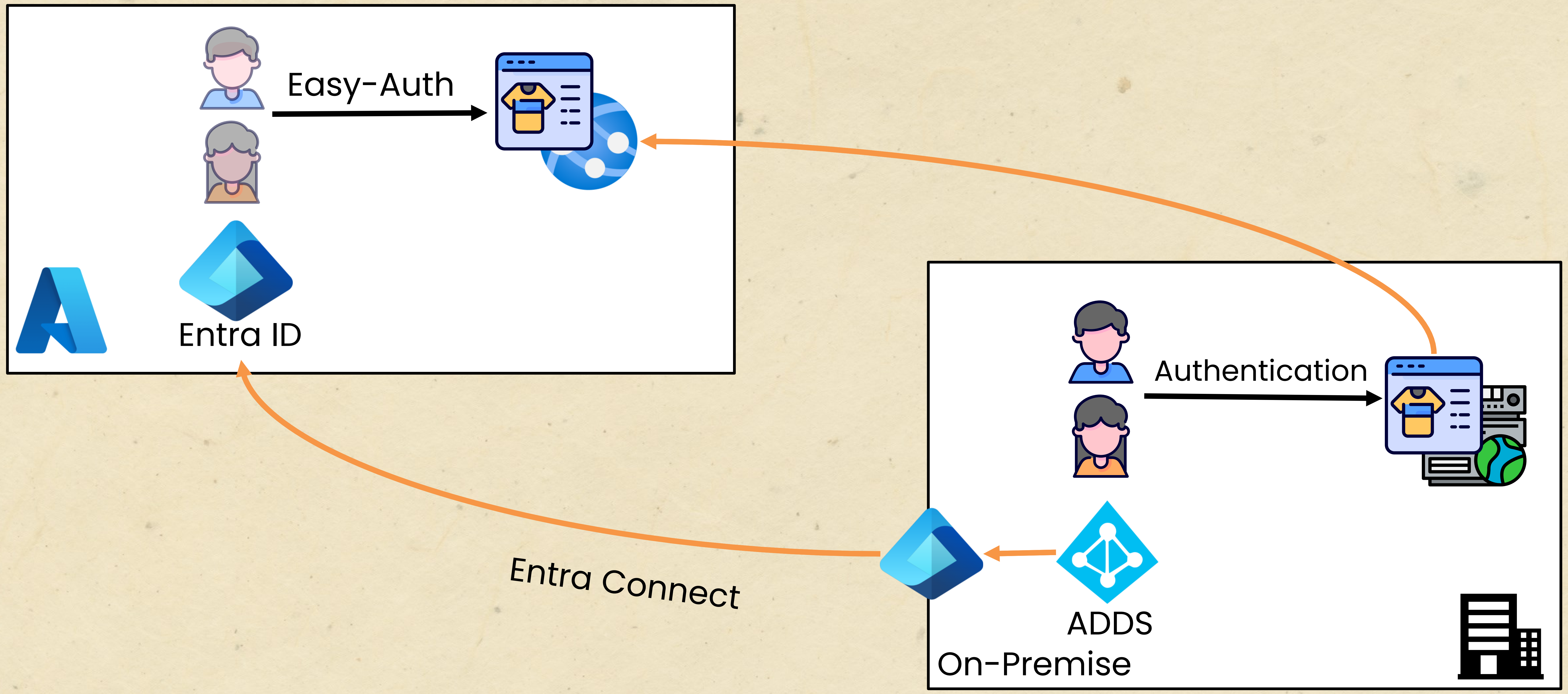
The ID tokens, access tokens, and refresh tokens are cached for the authenticated session, and they're accessible **only** by the associated user.

From your client code, send an **HTTP GET** request to `/.auth/me`. The returned **JSON** has the provider-specific tokens.

Easy-Auth scenarios

- You want less code to own and manage.
- Your app's language and SDKs don't provide user sign-in or authorization.
- You don't have the ability to modify your app code (for example, when migrating legacy apps).
- You need to handle authentication through configuration and not code.
- You need to sign in external or social users.

Migrate legacy app



Easy-Auth Samples

massimobonanni/EasyAuthSamples

Templates and other amenities

Bicep

```
resource appService 'Microsoft.Web/sites@2021-01-15' existing = {
  name: 'your-app-service-name'
}

resource appServiceAuthSettings 'Microsoft.Web/sites/config@2021-01-15' = {
  name: '${appService.name}/authsettings'
  properties: {
    enabled: true
    unauthenticatedClientAction: 'RedirectToLoginPage'
    tokenStoreEnabled: true
    defaultProvider: 'AzureActiveDirectory'
    clientId: 'your-client-id'
    clientSecretSettingName: 'your-client-secret-setting-name'
    issuer: 'https://login.microsoftonline.com/your-tenant-id'
  }
}
```

AZ CLI

```
az webapp auth update \
    --name 'your-app-service-name' \
    --resource-group 'your-resource-group' \
    --enabled true \
    --action 'LoginWithGoogle' \
    --google-client-id 'your-google-client-id' \
    --google-client-secret 'your-google-client-secret' \
    --token-store true
```

Entra ID choices?

Azure App Service built-in authentication

Allows you to sign users in and access data by writing **minimal or no code** in your web app, RESTful API, or mobile back end.

It's built directly into the platform and **doesn't require any particular language**, library, security expertise, or even any code to use.

Microsoft Authentication Library (MSAL)

Enables developers to acquire security tokens from the Microsoft identity platform to authenticate users and access secured web APIs.

Available for **multiple supported platforms and frameworks**, these are general purpose libraries that can be used in various hosted environments.

Developers can also **integrate with multiple sign-in providers**, like Microsoft Entra ID, Facebook, Google, Twitter.

`Microsoft.Identity.Web`

A higher-level library wrapping MSAL.NET.

It provides a single-surface API convenience layer that ties together **ASP.NET Core**, its authentication middleware, and MSAL.NET.

This library can be used in apps in various hosted environments.

You can integrate with **multiple sign-in providers**, like Microsoft Entra ID, Facebook, Google, Twitter.

Pros and Cons

- ✓ Simplified Authentication Process
- ✓ Support for Multiple Identity Providers
- ✓ Seamless Integration with Azure Services
- ✓ Enhanced Security

- ✓ Limited Customization
- ✓ Dependency on Azure Infrastructure
- ✓ Potential for Vendor Lock-in

improove

WEB DAY

27 MARZO 2024 **7A EDIZIONE**

Massimo Bonanni

Microsoft Technical Trainer @ Microsoft

massimo.bonanni@microsoft.com
@massimobonanni

aka.ms/maxlinkedin

SLIDE + VIDEO

HTTPS://WWW.IMPROOVE.TECH/APP

References

- Authentication and authorization in Azure App Service and Azure Functions
- Authentication scenarios and recommendations
- Implement user authentication and authorization
- massimobonanni/EasyAuthSamples