Global Microsoft 365 Developer Bootcamp

Build an application on Microsoft Identity Platform

Microsoft

Microsoft

Speaker Introduction

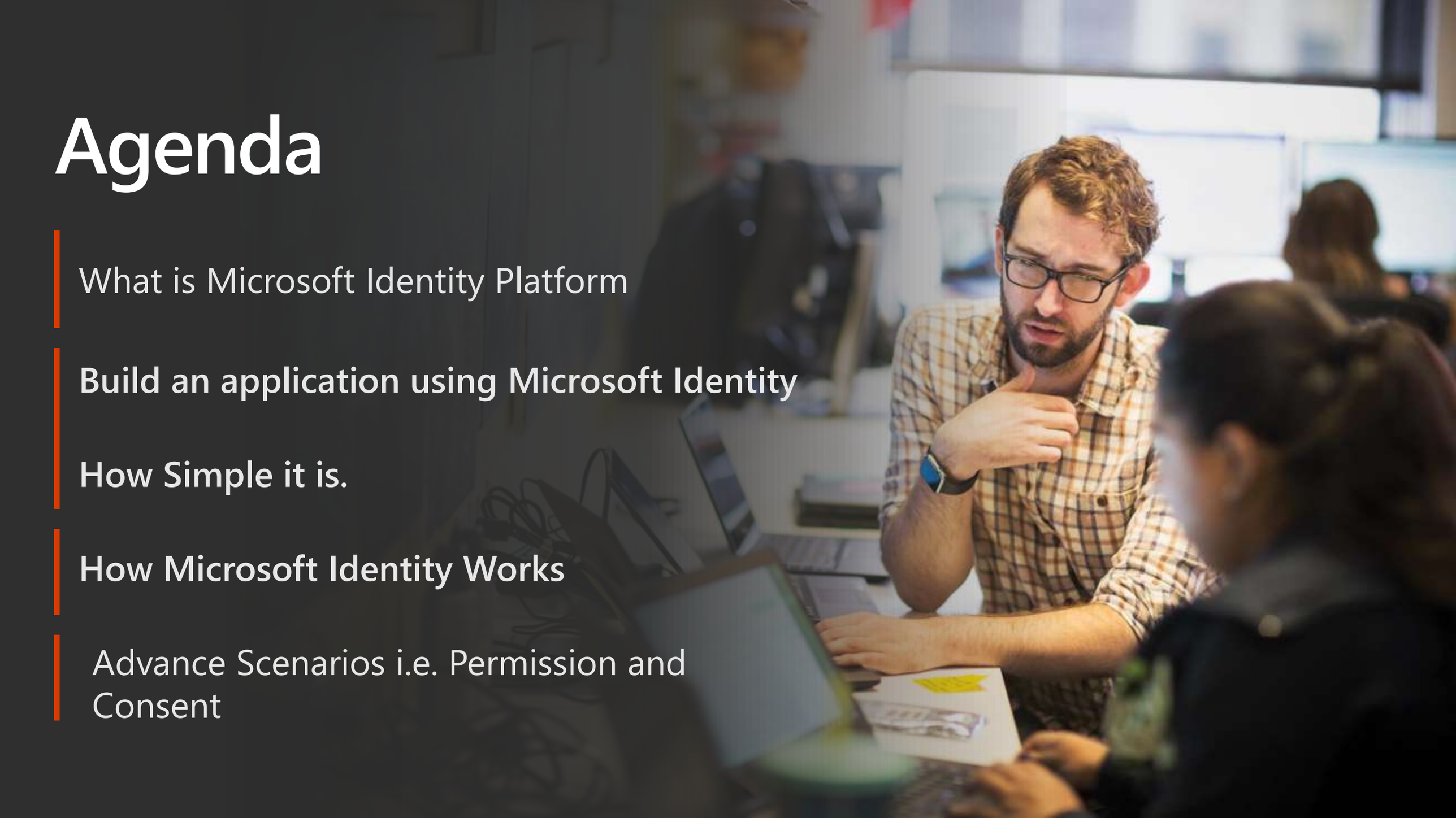
Manoj Mittal

Technical Architect
@ Mindtree Ltd

- C# Corner MVP, MCP, MCTS,MCSA, OCA
- http://manojmittalblogs.blogspot.in
- https://www.youtube.com/user/manojmcans/videos
- /Manojmcans
- In/manoj-mittal-mcp-mcts-oca-mcsa/
- https://www.c-sharpcorner.com/members/manoj-mittal

Agenda

- What is Microsoft Identity Platform
- **Build an application using Microsoft Identity**
- **How Simple it is.**
- **How Microsoft Identity Works**
- Advance Scenarios i.e. Permission and Consent

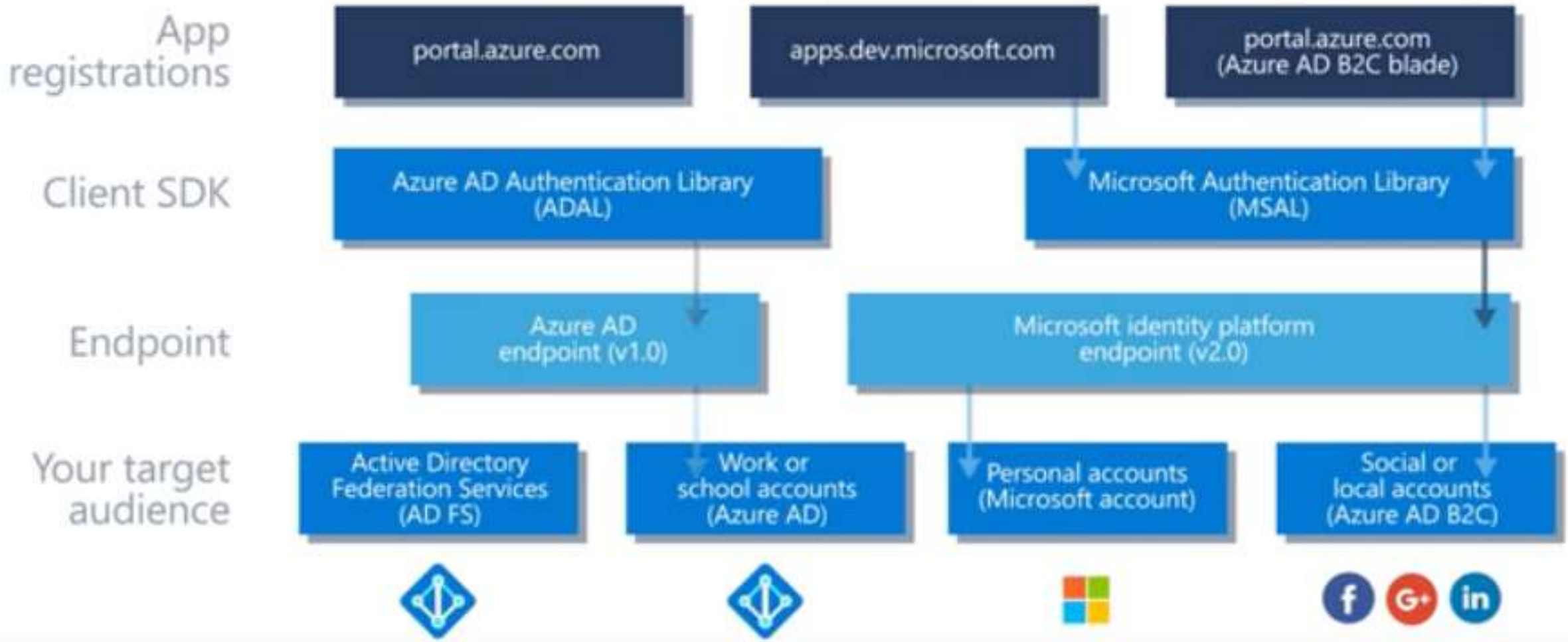


Microsoft Identity Platform

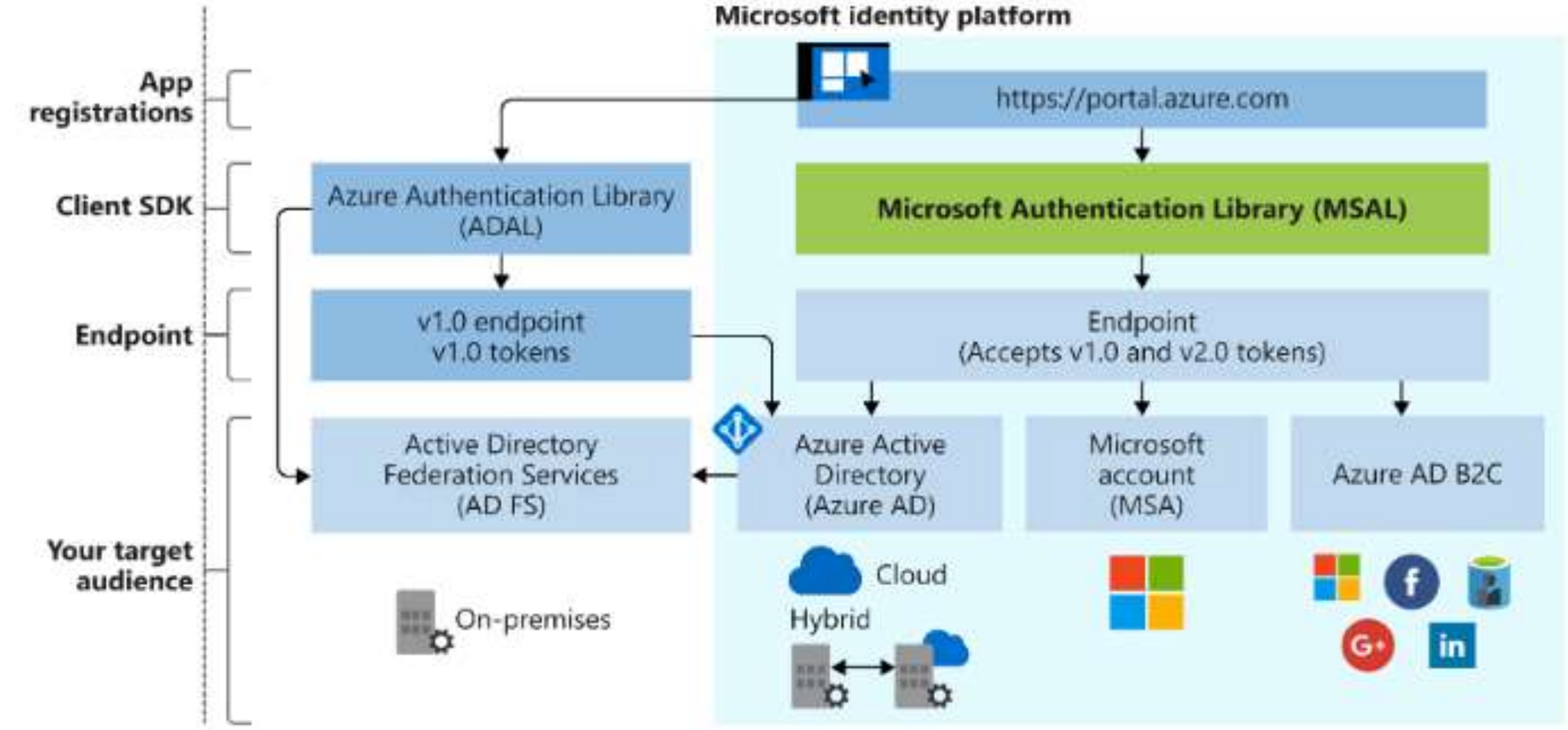
(Formerly Known as Azure Active Directory)

What it is & Why

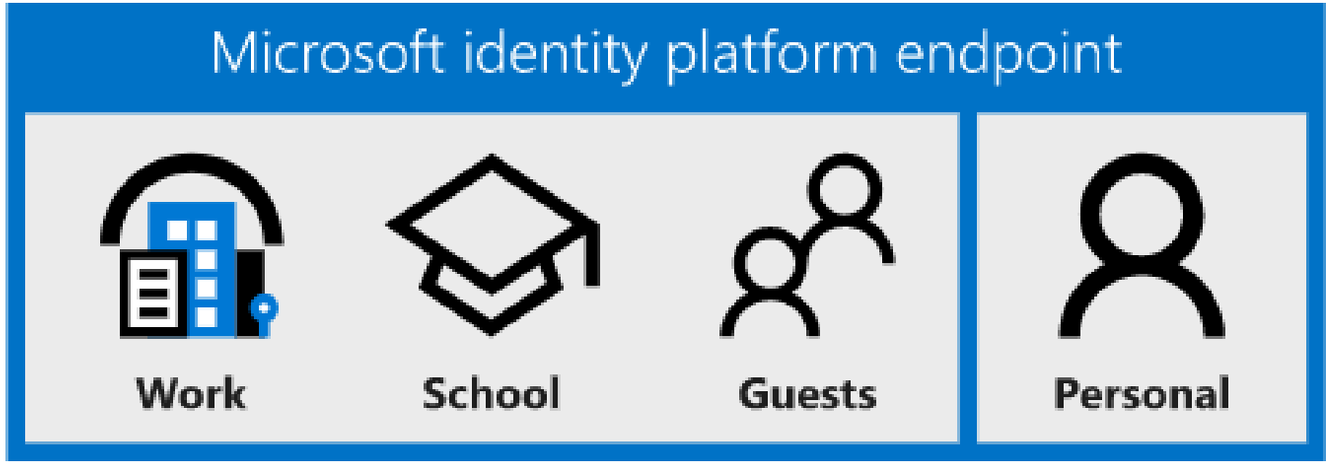
Authentication Before

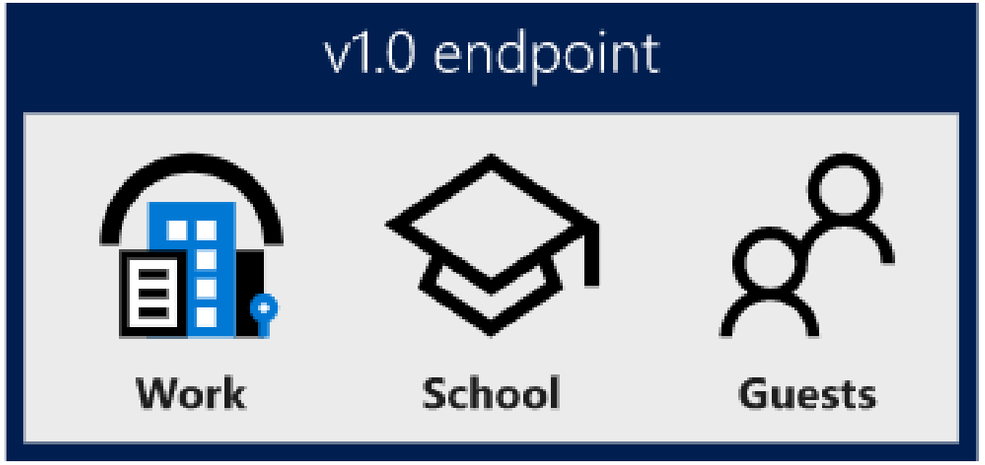


Authentication Simplified



Who can sign in





The Microsoft identity platform endpoint allows
Work
School accounts from Azure AD
Personal Microsoft accounts (MSA), such as hotmail.com, outlook.com, and msn.com, to sign in

The v1.0 endpoint allows only work and school accounts to sign in to your application (Azure AD)

MSAL JS or MSAL .NET (Microsoft Authentication Library) V2.0	ADAL JS or ADAL .NET (Azure AD Authentication Library) V1.0
Both used to authentication Azure AD entities and request token from Azure AD	
MSAL used to authenticate a broader set of Microsoft Identities • Azure AD identities • Microsoft account, social and local account through Azure B2C)	Azure AD V1.0 used to authenticate • Azure identities (work and school)
Method to renew tokens silently without prompting users is named **acquireTokenSilent** (more descriptive)	Method to renew tokens silently without prompting users is named **acquireToken**
MSAL.js API is designed around user agent client application such as **Web Browser**	ADAL.js uses AuthenticationContext as the representation of an instance of your application's connection
Method to acquire token requests can take different authority values than what is set in the UserAgentApplication	Methods to acquire tokens are associated with a single authority set in the AuthenticationContext
Authority Value: V2.0 use https://login.microsoftonline.com/common authority, will **allow users to sign in with any Azure AD organization account or a Microsoft personal account (MSA).** To restrict the sign in to only Azure AD accounts use **https://login.microsoftonline.com/organizations**	**Authority Value :** v1.0, use https://login.microsoftonline.com/common authority will allow users to sign in with any Azure AD account (for any organization)

Microsoft Identity Platform

(Formerly Known as Azure Active Directory)

- Build organizational applications using one Sign In Experience
- Securely access data in any API (i.e. Microsoft Graph)
- Evolution of Azure Active Directory for Developers

Microsoft Identity Platform

(Formerly Known as Azure Active Directory)

- Build an application using Microsoft Identity and How Simple it is.

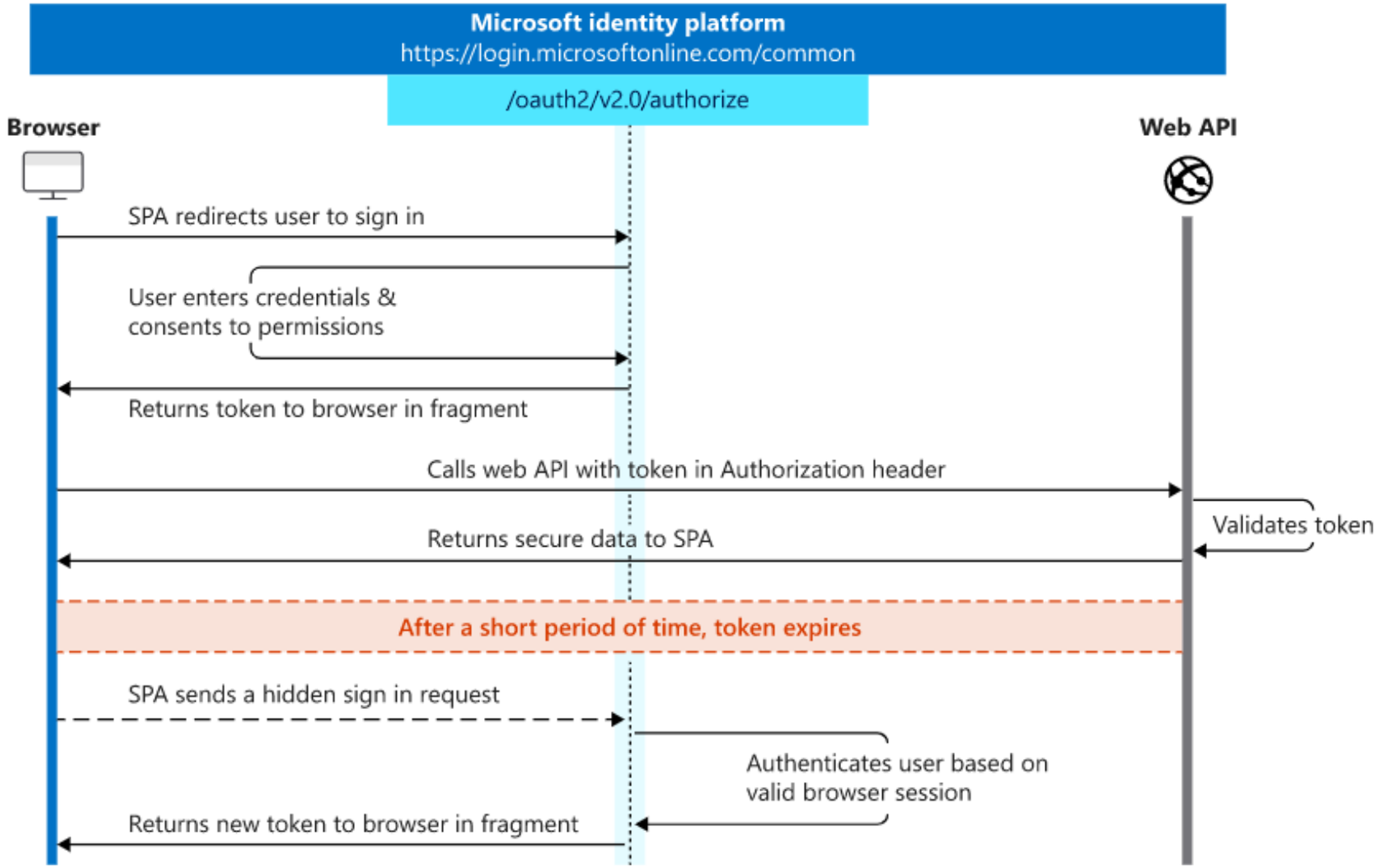
Building an application using Microsoft Identity

Protocol Diagram : Implicit Flow

Diagram shows what the entire implicit sign-in flow looks like and the sections that follow describe each step in more detail

Primary Benefits:

- It allow to get token from Microsoft Identity.
- It's allow app to sign-in the users
- Maintain the session.
- Get Token to Other WebAPI



Re-Cap

- It quite easy and share lot of stuff without any extra code.
- Single Sign On
- Token Management i.e. acquireTokenSilent

  (It acquire token from signed in user from cache)

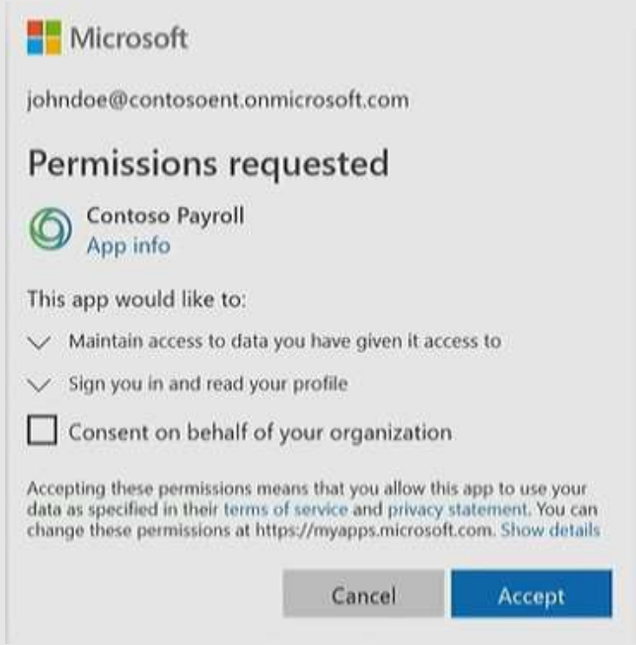
  aquireTokenSilent may not required due :-
  - Login Required
  - Multi Factor Authentication
  - Password Reset

Advance Scenario

- Customize your App Registration
- Permission and Consent Model

Permission and Consent Model

Scope define your app's range of operation against particular set of Cloud APIs



Consent is a data access contract

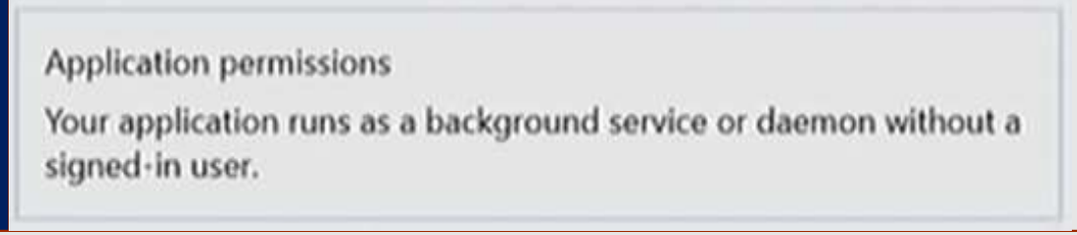
Permissions Type

Delegate Permission

- Sign In Required
- Scope: User.Read

Application Permission

- Background Jobs
- Scope: People.Read



Best Practice for Permission & Consent

- Define at least minimum permission to run your application else it will cause to malicious actions
- Use delegate permission for where user need sign in.
- Consider incremental consent for scope require the user’s approval

Building an application with Incremental Consent

Best Practice for Single Sign On

- Always try the silent token acquisition before attempting interactive token acquisition

```
try
{
    result = await app.AcquireTokenSilent(scopes, account).ExecuteAsync();
}
catch(MsalUiRequiredException ex)
{
    result = await app.AcquireToken(scopes, account).ExecuteAsync();
}
```

- This Simple pattern enable SSO.

Call to action

- Share your feedback on Bootcamp at https://aka.ms/Microsoft365DevBootcampSurvey2019
- Join Office 365 developer program https://dev.office.com/devprogram to leverage all resources for Office 365 development learning
- Build applications on Office 365 platform
- Attend MVP led local community events to continue learning on Office 365 development