Microsoft Entra ID Integration Options  
  
1. Microsoft Authentication Library (MSAL)

What it is: Microsoft's official authentication library

- MSAL.js 2.0 - For browser-based React/Next.js apps

- MSAL Node - For server-side Node.js applications

- MSAL React - React-specific wrapper

Benefits:

- ✅ Official Microsoft library with full support

- ✅ Handles token refresh automatically

- ✅ Built-in cache management

- ✅ Supports all Microsoft identity features

Use Case: When you want Microsoft's official solution with comprehensive features

2. Microsoft Graph SDK

What it is: Access Microsoft 365 data and services

- User profiles, calendars, emails

- Teams data, SharePoint files

- OneDrive integration

Integration: Usually combined with MSAL for authentication

// Example: Get user's calendar events

const events = await graphClient.me.events.get();

3. Azure AD B2C (Business to Consumer)

What it is: Consumer-facing identity management

- Social logins (Google, Facebook, Twitter)

- Custom user journeys

- Self-service password reset

- Custom branding

Use Case: Public applications with external customers

4. SAML 2.0 Integration

What it is: Enterprise SSO standard

- XML-based protocol

- Works with legacy enterprise systems

- Complex but widely supported

Use Case: When required by enterprise customers or compliance

5. WS-Federation

What it is: Microsoft's legacy federation protocol

- Older than OAuth/OIDC

- Still supported for compatibility

- Common in .NET environments

6. Azure AD Application Proxy

What it is: Secure access to on-premises applications

- Publish internal apps externally

- No VPN required

- Pre-authentication with Entra ID

7. Managed Identity

What it is: Service-to-service authentication

- No credentials in code

- Automatic token management

- For Azure services only

8. Certificate-Based Authentication

What it is: Use certificates instead of passwords

- Smart card authentication

- Mobile device certificates

- High security environments

🎯 Comparison for Your Use Case

| Integration Method | Complexity | Security | Features | Best For |

|--------------------------|------------|-----------|------------------|-------------------------------|

| OAuth 2.0/OIDC (Current) | Medium | High | Complete | Web applications |

| MSAL.js | Low | High | Comprehensive | Microsoft-first development |

| SAML 2.0 | High | High | Enterprise | Legacy enterprise integration |

| Azure AD B2C | Medium | High | Consumer-focused | Public applications |

| Certificate Auth | High | Very High | Specialized | High-security environments |

🔄 Could You Switch?

From your current OAuth/OIDC implementation:

To MSAL.js: ✅ Easy - Similar concepts, Microsoft's official library

To SAML: 🔶 Medium - Different protocol, more complex

To B2C: 🔶 Medium - Different service, consumer-focused

To Certificate Auth: 🔴 Hard - Completely different approach

💡 Recommendations

Stick with OAuth 2.0/OIDC if:

- Your current implementation works

- You want standard web authentication

- You need broad compatibility

Consider MSAL.js if:

- You want Microsoft's official library

- You need advanced Microsoft Graph integration

- You want simplified token management

Consider Azure AD B2C if:

- You need social login options

- You have external customers (not employees)

- You need custom user registration flows