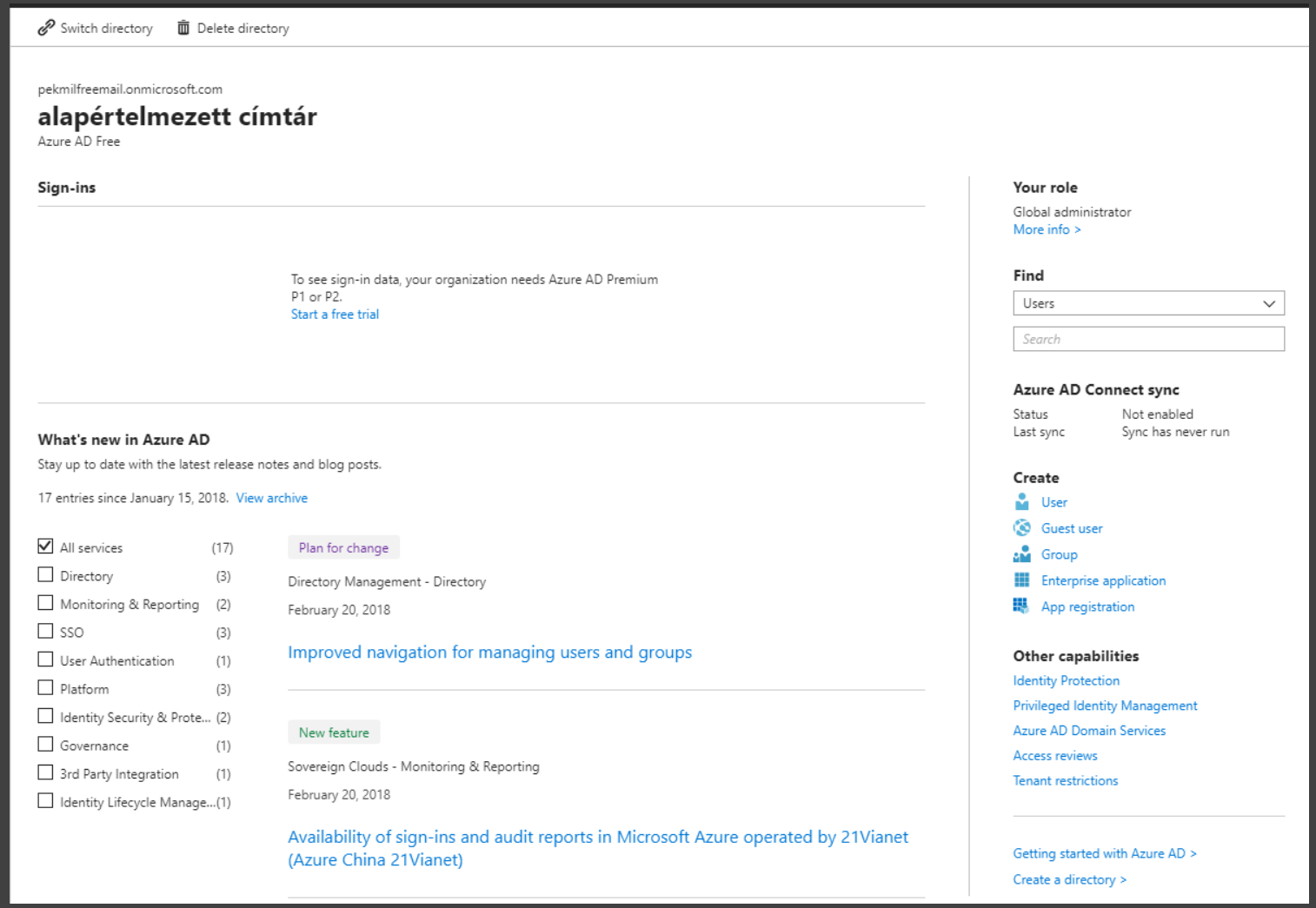
Microsoft Azure – Azure AD

Milan Pekardy

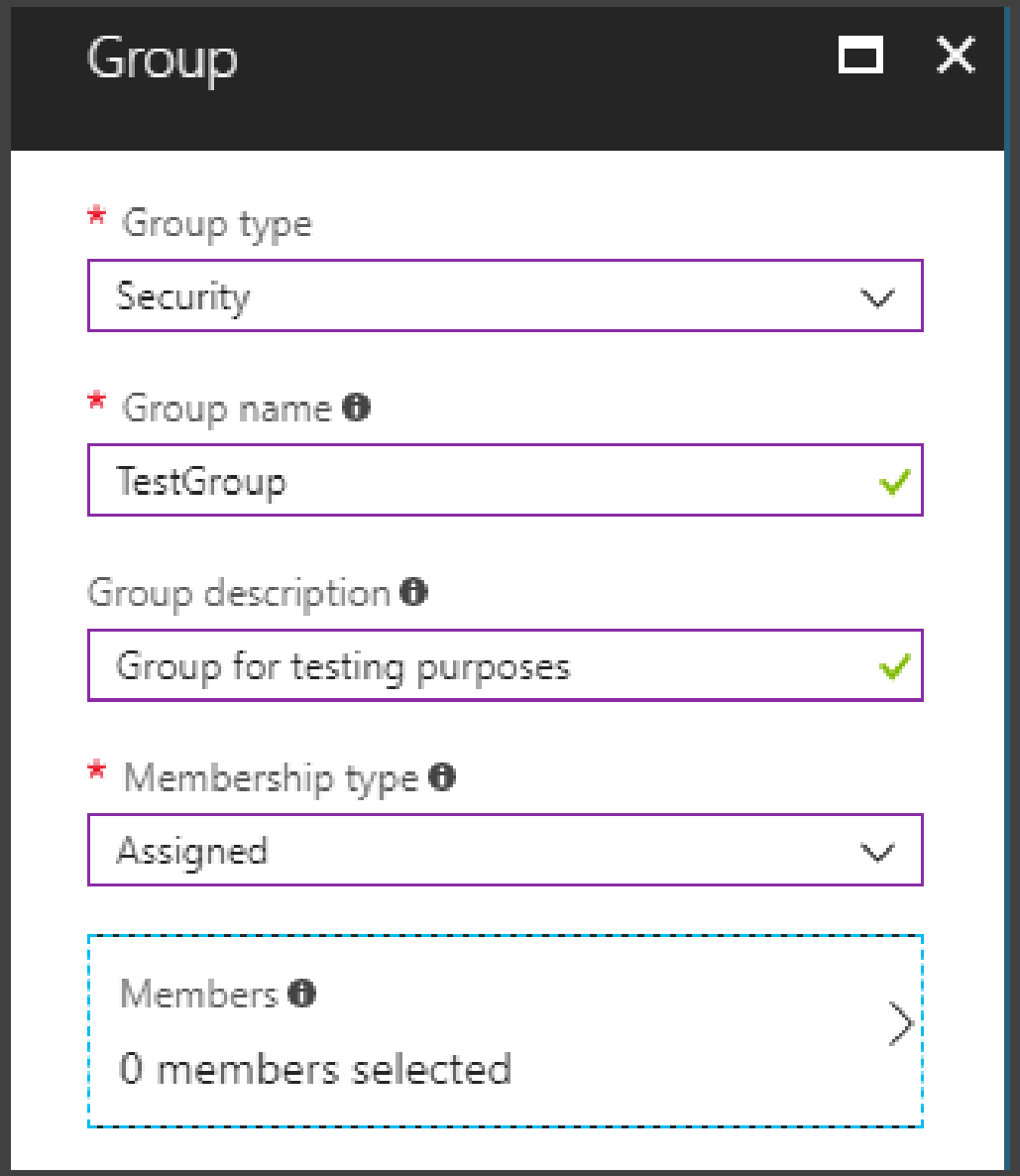
pekardy@dcs.uni-pannon.hu

Azure Active Directory

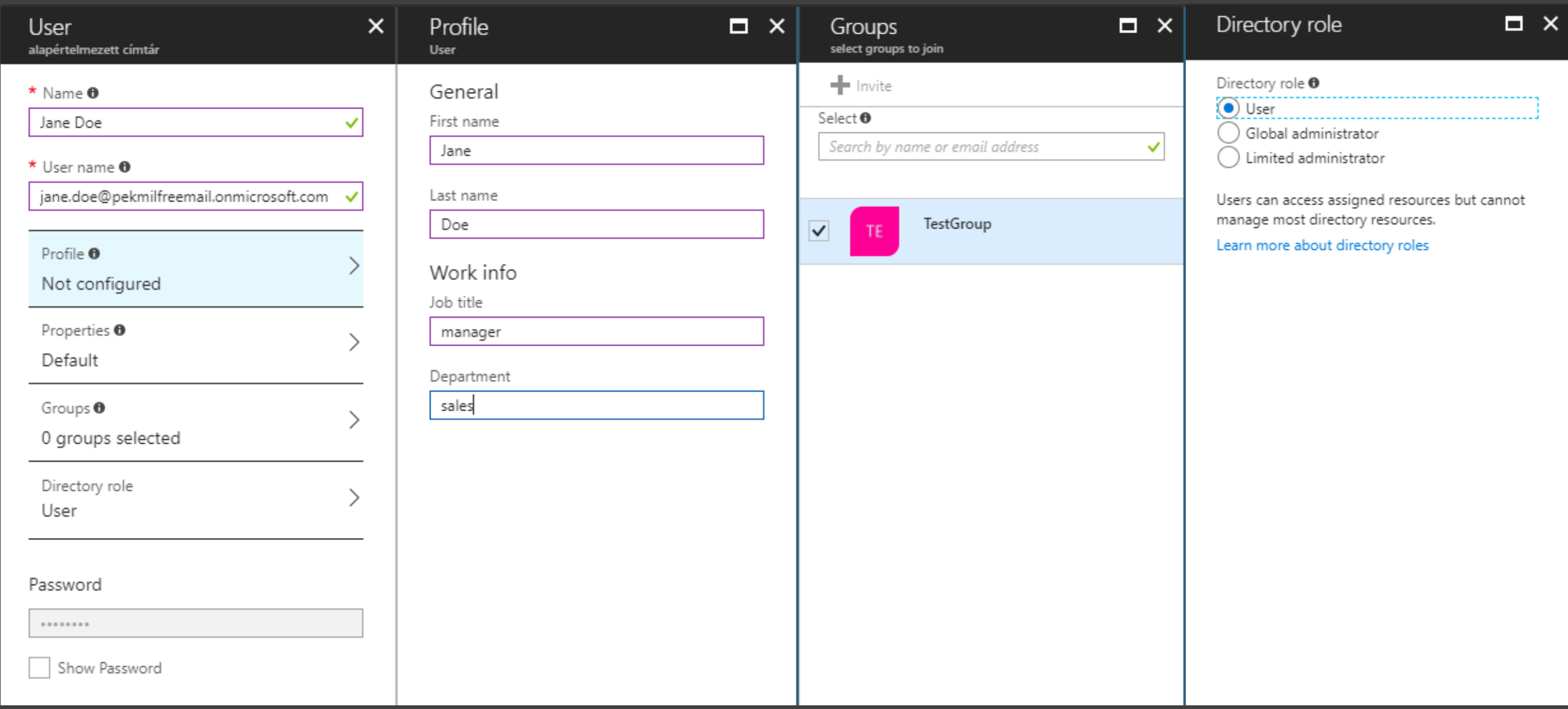
- You automatically have one Azure AD tenant for your Azure account
- You can add additional directories if necessary (eg. development, testing, etc.)
- The portal has full support for creating, deleting, and managing users within this environment



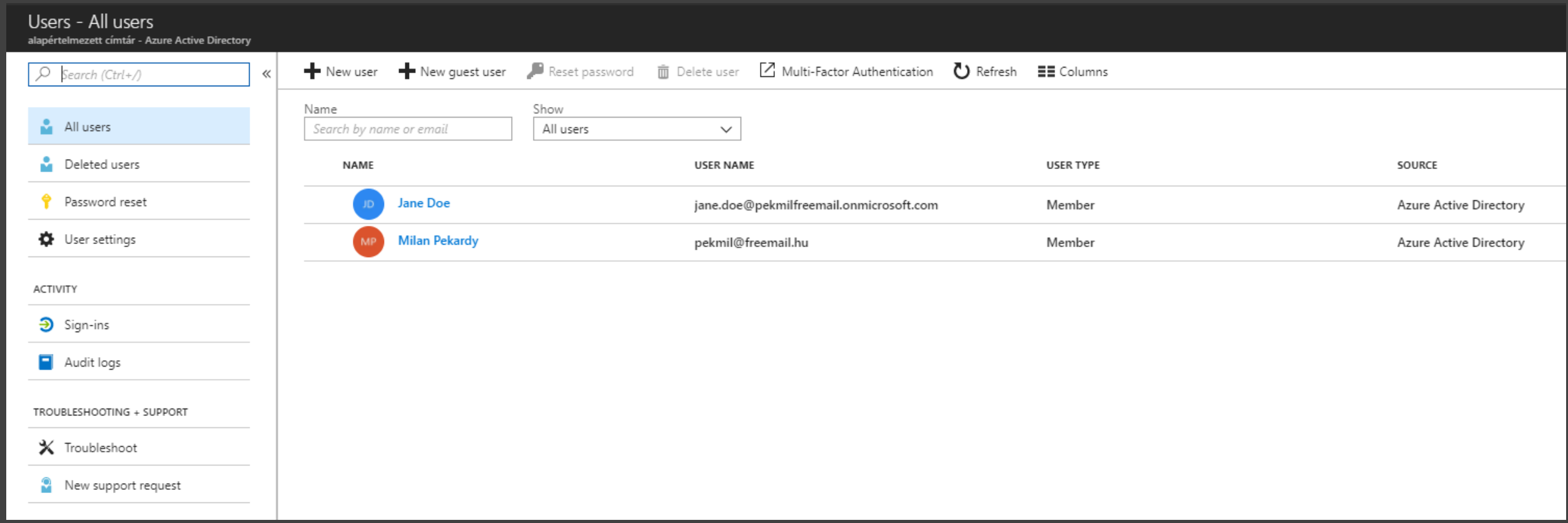
Azure Active Directory – Create New Group



Azure Active Directory – Create New User

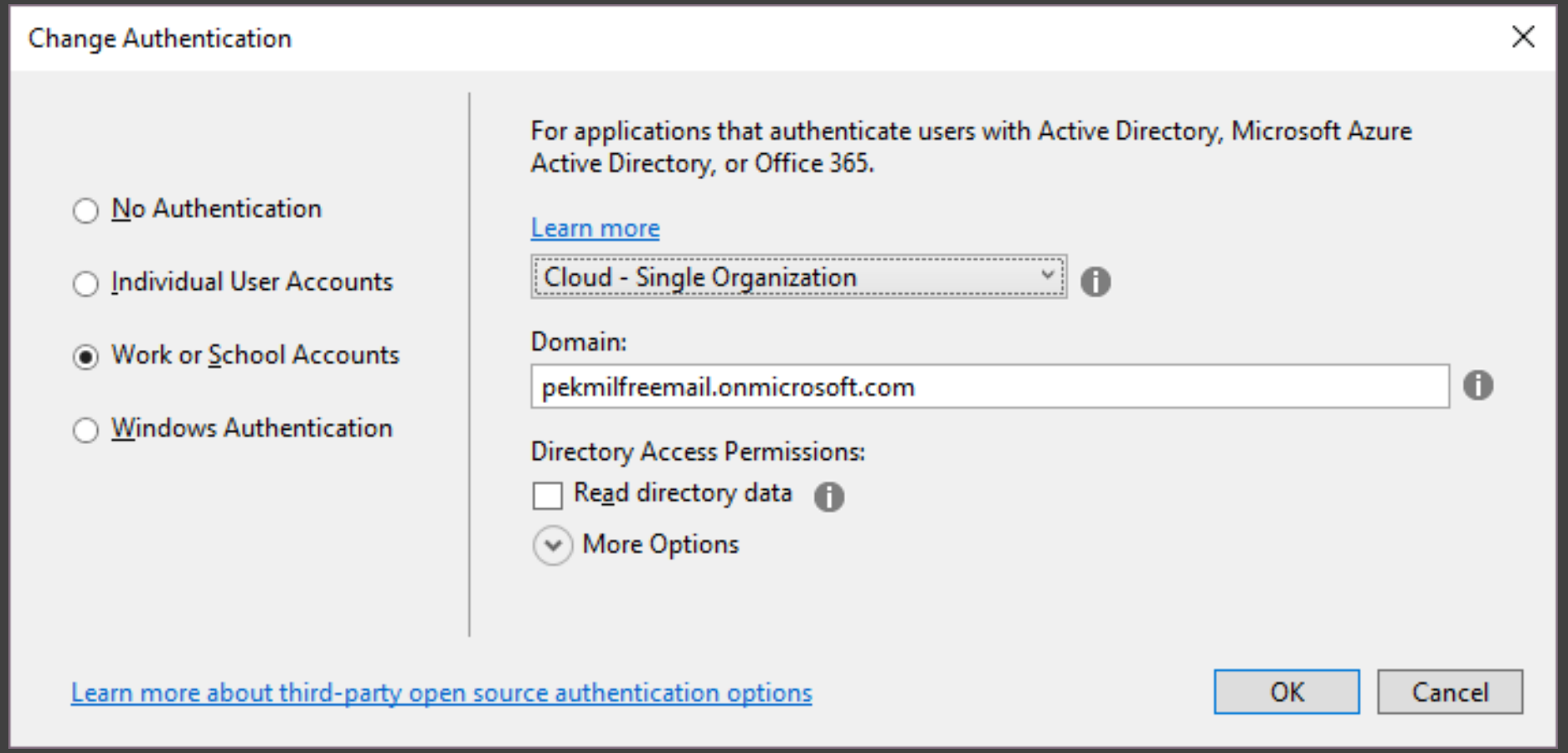


Azure Active Directory – Users



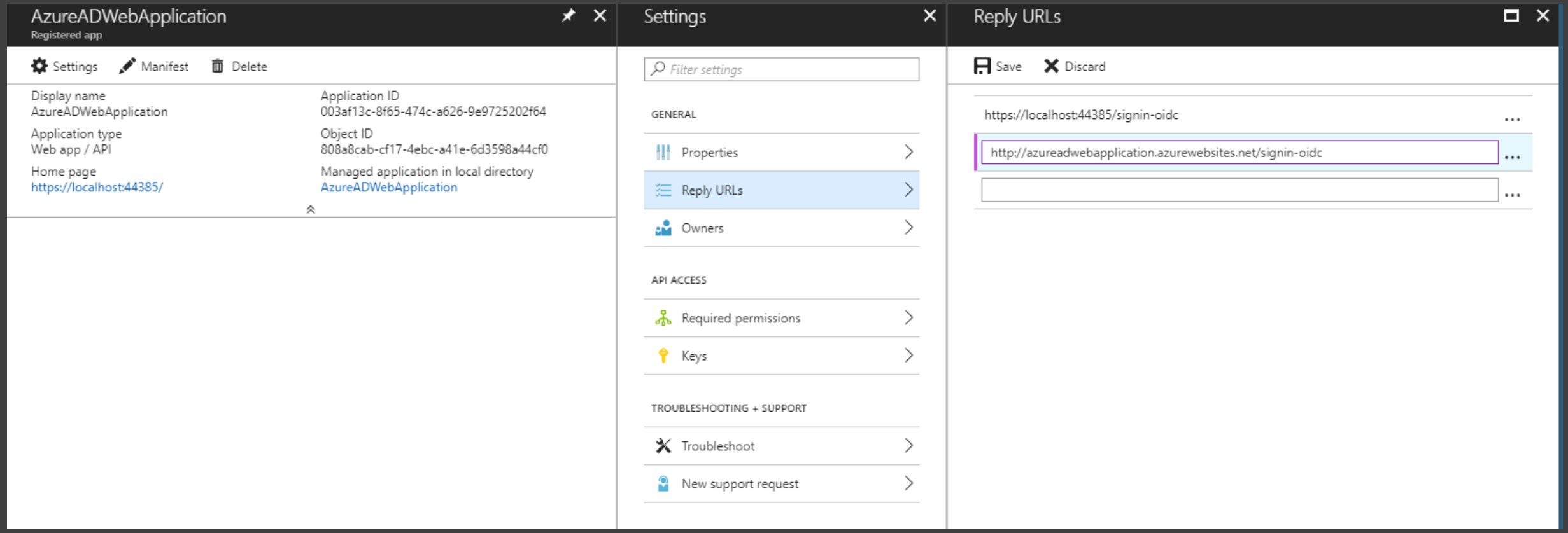
Azure Active Directory – Sample Application

- Visual Studio > Create New ASP.NET Core Web Application
- Change Authentication to Work or School Accounts



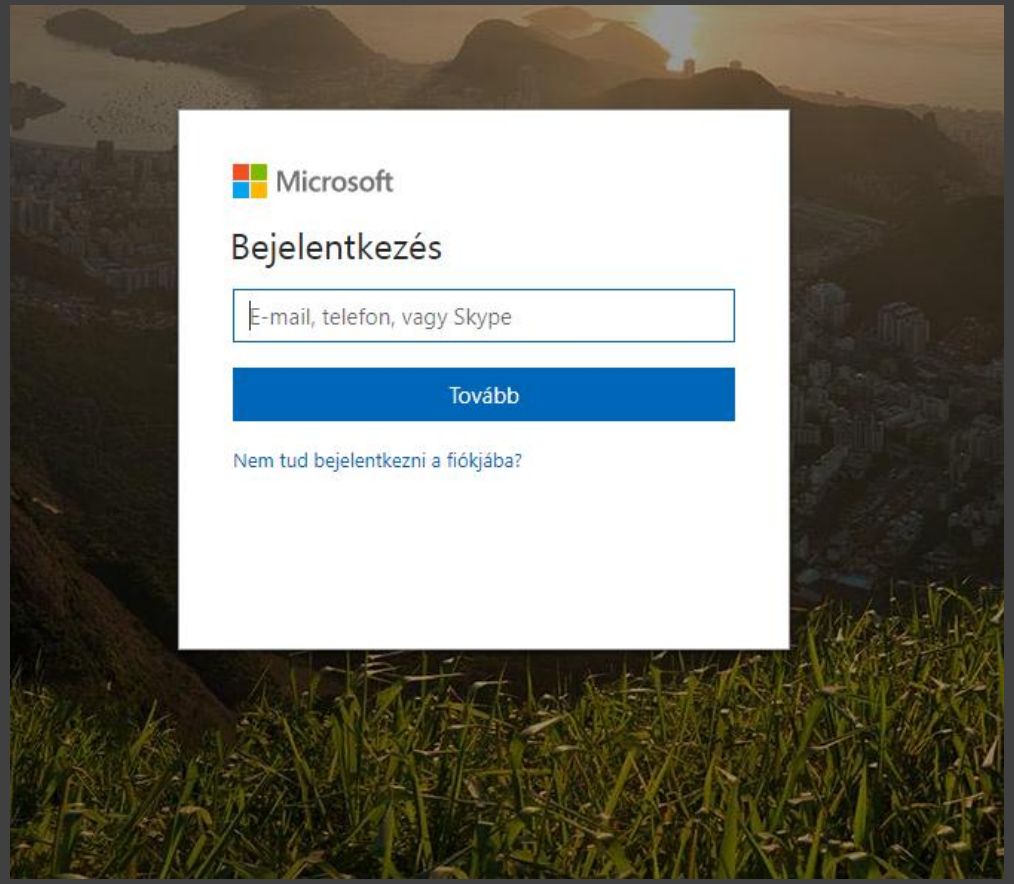
Azure Active Directory – Sample Application

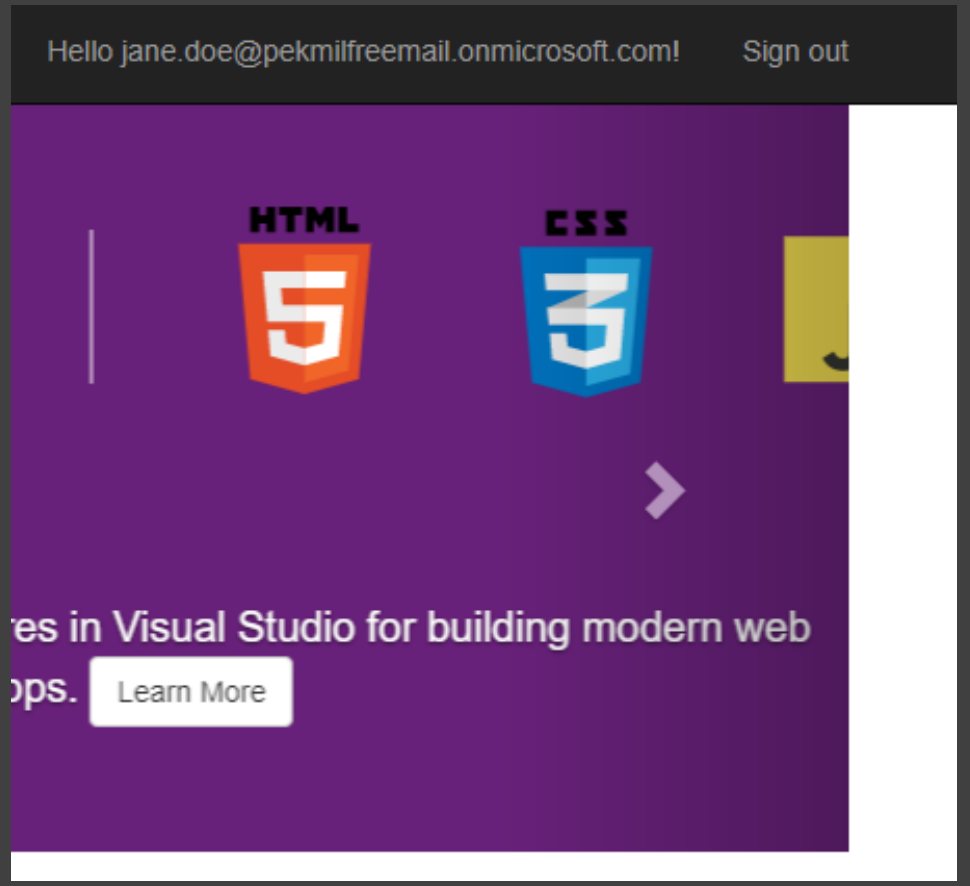
- Register the correct reply URL: Azure Active Directory > App registrations > All apps > AzureADWebApplication > Settings > Reply URLs



Azure Active Directory – Sample Application

- You can now log into the app with your account





Azure Active Directory – Role based authorization

- Configure Azure AD to send back claims representing a user's group membership
- App registrations > All apps > AzureADWebApplication > Manifest

```
{
  "appId": "003af13c-8f65-474c-a626-9e9725202f64",
  "appRoles": [],
  "availableToOtherTenants": false,
  "displayName": "AzureADWebApplication",
  "errorUrl": null,
  "groupMembershipClaims": "SecurityGroup",
  "optionalClaims": null,
  "acceptMappedClaims": null,
  "homepage": "https://localhost:44385/",
  "informationalUrls": {
    "privacy": null,
    "termsOfService": null
  },
```

Azure Active Directory – Role based authorization

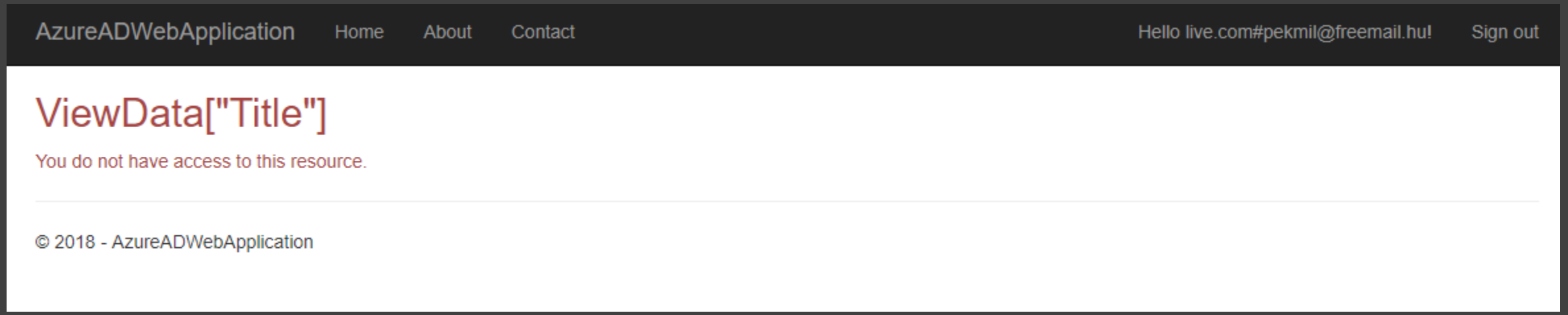
- Definig authorization policy:
  - The authorization primitives in ASP.NET Core are **claims** and **policies**.
  - Claims hold information about a user,
  - Policies encapsulate simple logic to evaluate the current user against the current context and return true to authorize a user.

```
services.AddAuthorization(options =>
{
    options.AddPolicy("TestGroupPolicy", builder => builder.RequireClaim("groups", "4097c541-9b33-4a84-b409-7030bc525438"));
});
```

```
[Authorize("TestGroupPolicy")]
0 references | 1 request | 0 exceptions
public IActionResult About()
{
    ViewData["Message"] = "Your application description page.";

    return View();
}
```

Azure Active Directory – Role based authorization



Azure Active Directory – Role based authorization