





Manage identity in multitenant applications

07/21/2017 • 3 minutes to read • Contributors     

In this article

[Introduction](#)

[What is multitenancy?](#)

[Identity in a multitenant app](#)

This series of articles describes best practices for multitenancy, when using Azure AD for authentication and identity management.



[Sample code](#)

When you're building a multitenant application, one of the first challenges is managing user identities, because now every user belongs to a tenant. For example:

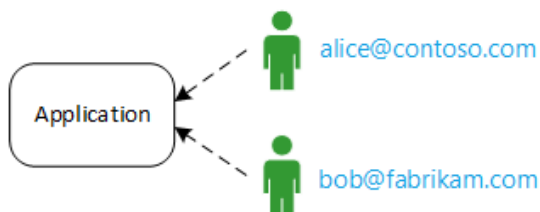
- Users sign in with their organizational credentials.
- Users should have access to their organization's data, but not data that belongs to other tenants.
- An organization can sign up for the application, and then assign application roles to its members.

Azure Active Directory (Azure AD) has some great features that support all of these scenarios.

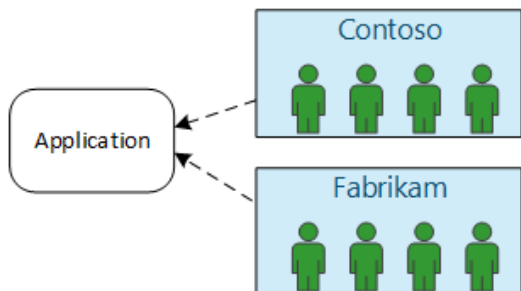
To accompany this series of articles, we created a complete [end-to-end implementation](#) of a multitenant application. The articles reflect what we learned in the process of building the application. To get started with the application, see [Run the Surveys application](#).

Introduction

Let's say you're writing an enterprise SaaS application to be hosted in the cloud. Of course, the application will have users:



But those users belong to organizations:



Example: Tailspin sells subscriptions to its SaaS application. Contoso and Fabrikam sign up for the app. When Alice (alice@contoso) signs in, the application should know that Alice is part of Contoso.

- Alice *should* have access to Contoso data.

- Alice *should not* have access to Fabrikam data.

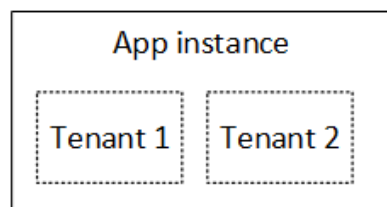
This guidance will show you how to manage user identities in a multitenant application, using [Azure Active Directory](#) ([Azure AD](#)) to handle sign-in and authentication.

What is multitenancy?

A *tenant* is a group of users. In a SaaS application, the tenant is a subscriber or customer of the application.

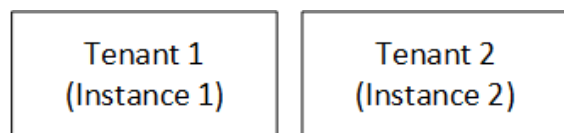
Multitenancy is an architecture where multiple tenants share the same physical instance of the app. Although tenants share physical resources (such as VMs or storage), each tenant gets its own logical instance of the app.

Typically, application data is shared among the users within a tenant, but not with other tenants.



Multitenant application

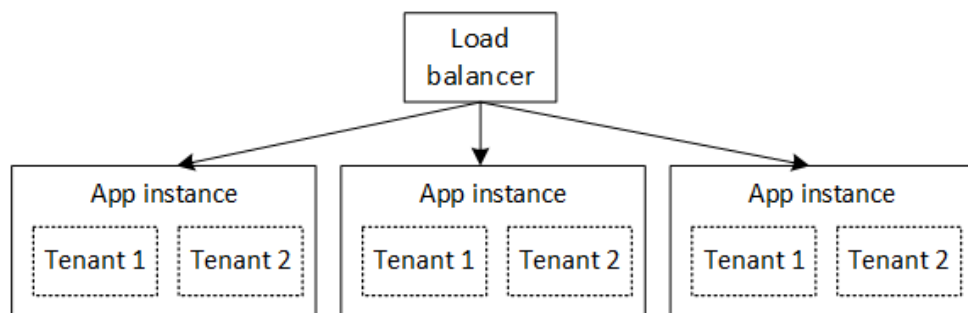
Compare this architecture with a single-tenant architecture, where each tenant has a dedicated physical instance. In a single-tenant architecture, you add tenants by spinning up new instances of the app.



Single tenant application

Multitenancy and horizontal scaling

To achieve scale in the cloud, it's common to add more physical instances. This is known as *horizontal scaling* or *scaling out*. Consider a web app. To handle more traffic, you can add more server VMs and put them behind a load balancer. Each VM runs a separate physical instance of the web app.



Any request can be routed to any instance. Together, the system functions as a single logical instance. You can tear down a VM or spin up a new VM, without affecting users. In this architecture, each physical instance is multitenant, and you scale by adding more instances. If one instance goes down, it should not affect any tenant.

Identity in a multitenant app

In a multitenant app, you must consider users in the context of tenants.

Authentication

- Users sign into the app with their organization credentials. They don't have to create new user profiles for the app.
- Users within the same organization are part of the same tenant.
- When a user signs in, the application knows which tenant the user belongs to.

Authorization

- When authorizing a user's actions (say, viewing a resource), the app must take into account the user's tenant.
- Users might be assigned roles within the application, such as "Admin" or "Standard User". Role assignments should be managed by the customer, not by the SaaS provider.

Example. Alice, an employee at Contoso, navigates to the application in her browser and clicks the "Log in" button. She is redirected to a sign-in screen where she enters her corporate credentials (username and password). At this point, she is logged into the app as `alice@contoso.com`. The application also knows that Alice is an admin user for this application. Because she is an admin, she can see a list of all the resources that belong to Contoso. However, she cannot view Fabrikam's resources, because she is an admin only within her tenant.

In this guidance, we'll look specifically at using Azure AD for identity management.

- We assume the customer stores their user profiles in Azure AD (including Office365 and Dynamics CRM tenants)
- Customers with on-premises Active Directory can use [Azure AD Connect](#) to sync their on-premises Active Directory with Azure AD. If a customer with on-premises Active Directory cannot use Azure AD Connect (due to corporate IT policy or other reasons), the SaaS provider can federate with the customer's directory through Active Directory Federation Services (AD FS). This option is described in [Federating with a customer's AD FS](#).

This guidance does not consider other aspects of multitenancy such as data partitioning, per-tenant configuration, and so forth.

[Next](#)