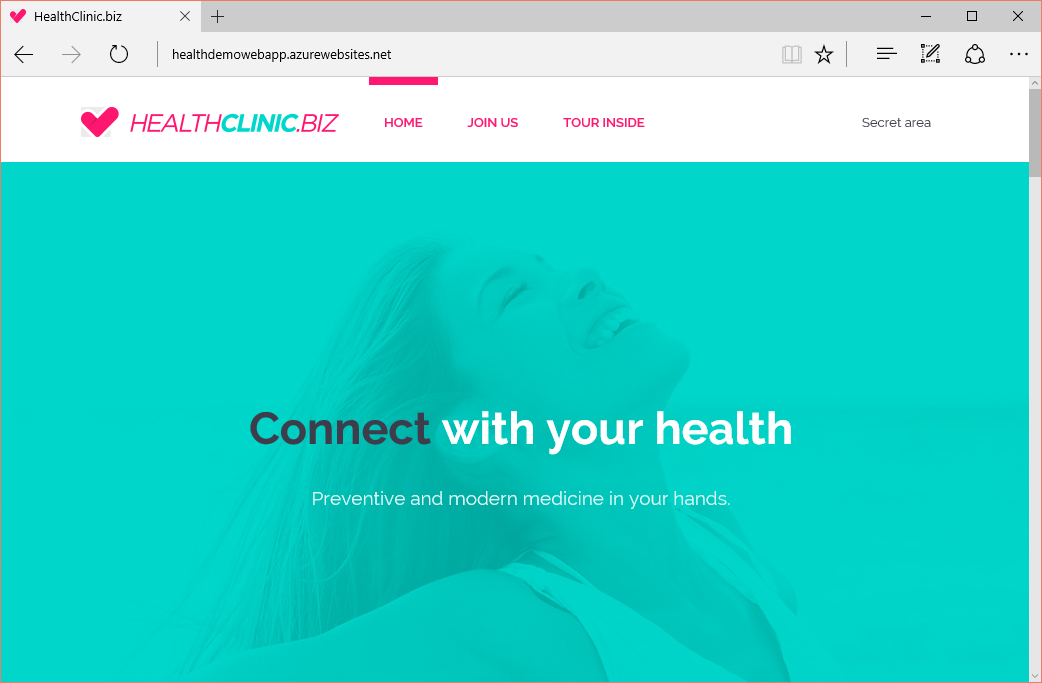
Create and deploy an ASP.NET web app in Azure App Service

Azure App Service is the only cloud service that integrates everything you need to quickly and easily build web and mobile apps for any platform and any device. Built for developers, App Service is a fully managed platform that has powerful capabilities such as built-in DevOps, continuous integration with Visual Studio Team Servicesand GitHub, staging and production support, and automatic patching.

This walkthrough shows how to deploy an ASP.NET web application to a web app in Azure App Service by using Visual Studio 2015 or Visual Studio 2013. The walkthrough assumes that you are an ASP.NET developer who has no prior experience with Azure. After you complete the tutorial, you’ll have a web app that’s deployed to Azure from Visual Studio. Samples and tutorials in this article are based on the [Connect(); //2015](https://blogs.msdn.microsoft.com/visualstudio/2015/12/08/connectdemos-2015-healthclinic-biz/) event demos.

The following illustration shows the completed application:



This article will help you learn the following:

* How to prepare your computer for Azure development by installing the [Azure SDK for .NET.](https://azure.com/tools)
* How to set up Visual Studio to create a new web app in Azure App Service and an ASP.NET MVC 5 web project.
* How to compose the Demo project that is presented in Microsoft Connect(); //2015.
* How to deploy a web app project to App Serviceby using Visual Studio.
* How to use the [Azure portal](https://azure.microsoft.com/overview/preview-portal/) to monitor and manage your web app.

# Sign up for Microsoft Azure

You need an Azure account to complete this tutorial. You can:

* [Open an Azure account for free](https://azure.microsoft.com/pricing/free-trial/?WT.mc_id=A261C142F). You get credits that you can use to try paid Azure services. Even after the credits are used up, you can keep the account and use free Azure services and features, such as the Web Apps feature of Azure App Service.
* [Activate Visual Studio subscriber benefits](https://azure.microsoft.com/pricing/member-offers/msdn-benefits-details/?WT.mc_id=A261C142F). Your Visual Studio subscription gives you credits every month that you can use for paid Azure services.
* Get credits every month by joining to [Visual Studio Dev Essentials](https://www.visualstudio.com/products/visual-studio-dev-essentials-vs).

If you want to get started with Azure App Servicebefore you sign up for an Azure account, go to [Try App Service](http://go.microsoft.com/fwlink/?linkid=523751&clcid=0x409). There, you can immediately create a short-lived starter web app in App Service without a credit card or commitments.

# Set up the development environment

To start, set up your development environment by installing the latest version of the [Azure SDK](https://www.microsoft.com/web/handlers/webpi.ashx/getinstaller/VWDOrVs2015AzurePack.appids).

[Visual Studio 2015](https://go.microsoft.com/fwlink/?linkid=746481&clcid=0x409)▶

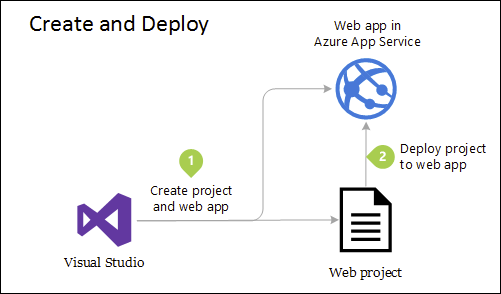
[Visual Studio 2013](https://go.microsoft.com/fwlink/?linkid=746482&clcid=0x409)▶

If you don't have Visual Studio installed, use the link for Visual Studio 2015, and Visual Studiowill be installed along with the SDK.

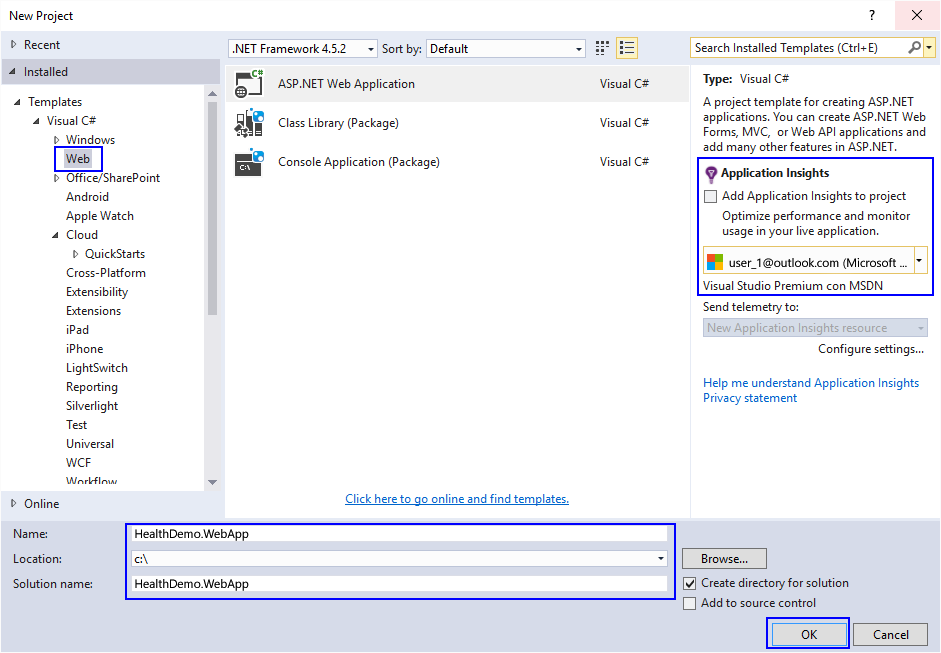
# Create a new project and a web app

Your first step is to create a web project in Visual Studio and a web app in Azure App Service. When that's done, you’ll deploy the project to the web app to make it available on the Internet.

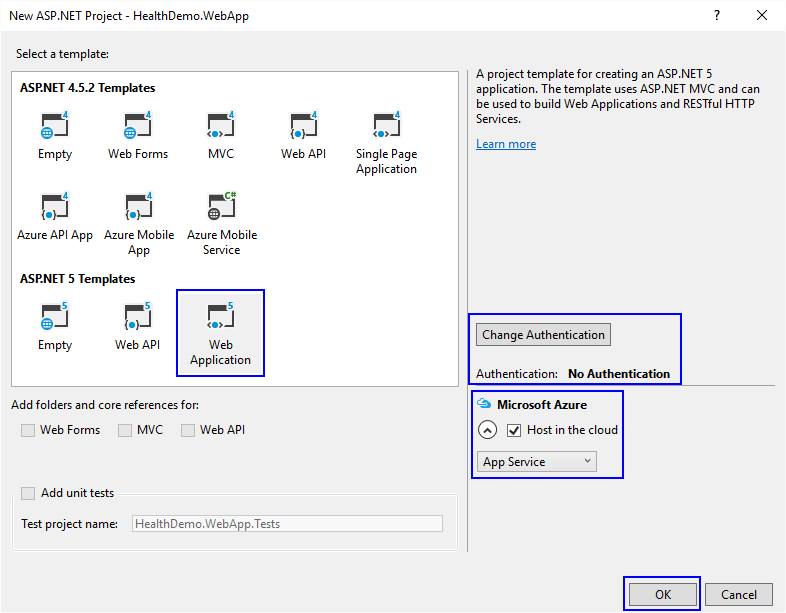
The diagram below illustrates what you're doing in the create and deploy steps.



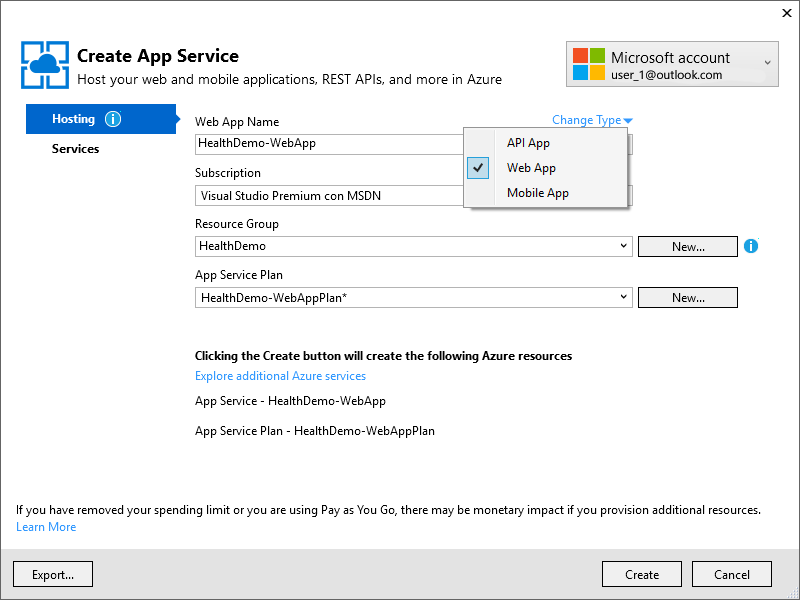
1. Open Visual Studio 2015 or Visual Studio 2013.
2. On the **File** menu, click **New** *>* **Project**.
3. In the **New Project** dialog box, click **C#** > **Web** > **ASP.NET Web Application**.
4. Choose a good name for the project, for example, HealthDemo.WebApp.
5. You can deactivate or activate **Application Insights** if you want your application to be monitored in terms of availability and performance by using [Azure Application Insights](https://azure.microsoft.com/es-es/documentation/articles/app-insights-overview/).
6. Click **OK**.



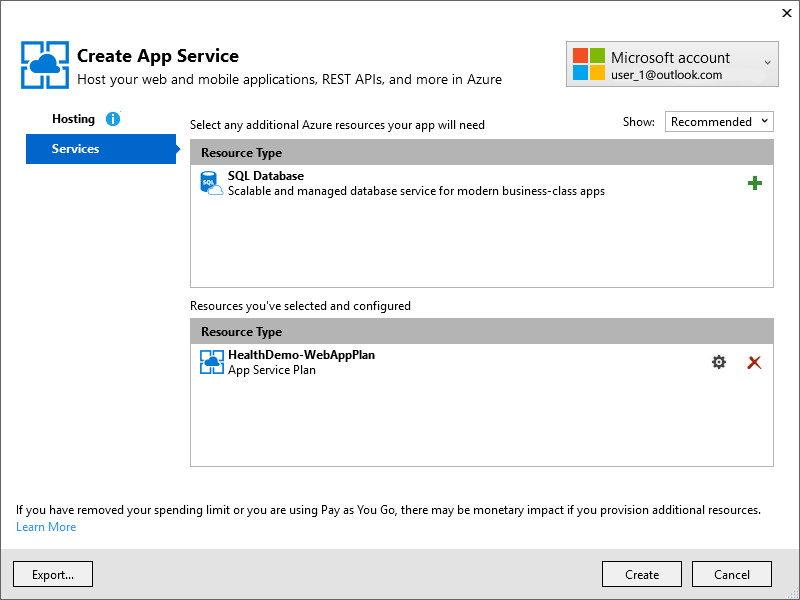
1. In the **Select a template** window, select **ASP.NET 5 Web Application**. On the other side, indicate that you want to host the **Web App** in **Azure**. Finally, indicate that you don’t want authentication for the application.

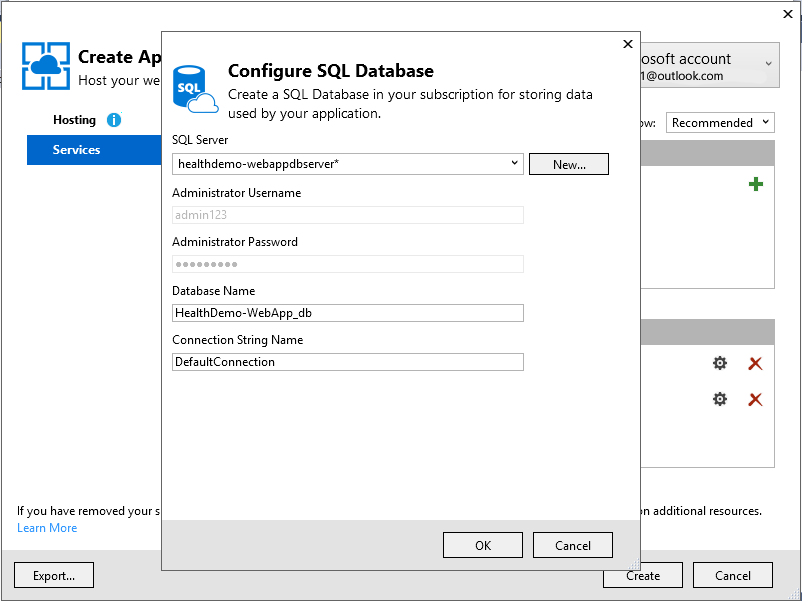


1. When you select Azure as the host, a new window will automatically open to indicate that the resources that will be created in Azure, which will host all application resources. Fill in the required information and make sure to indicate the type of deployment as **Web App**.

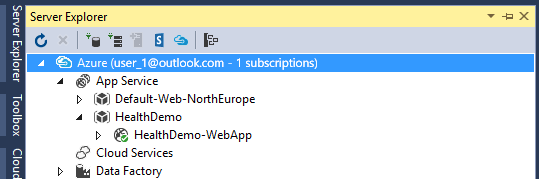


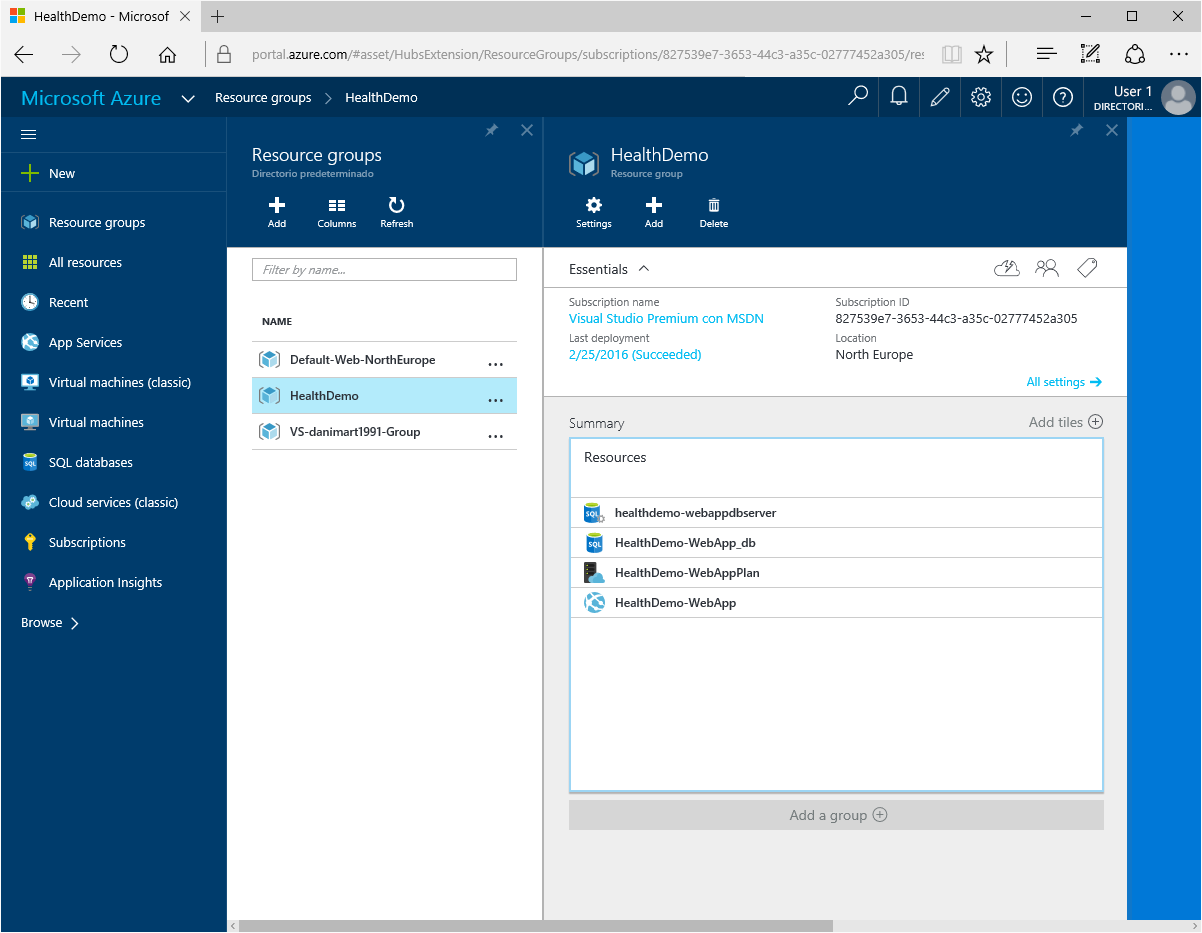
1. Because your web application will have a SQL database in the future, you can add one as a service in the configuration window.





1. When you click the **OK** button, Azure will begin to create the web app in Azure. After it is deployed, you can see all the information in the Azure portal and in Server Explorer in Visual Studio.





# Azure Resource Manager templates

You can use an Azure Resource Manager template to provision the services that the applications need. In a single template, you can deploy multiple services along with their dependencies. In that way, you can use the same template to repeatedly deploy your application in different environments during every stage of the application lifecycle.

[Azure Quickstart Templates](https://azure.microsoft.com/en-us/documentation/templates/) are popular deployments from the community that you can adapt to the needs of the developed application.

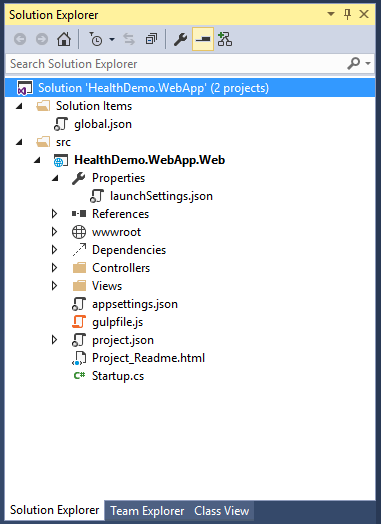
# Build the Connect(); //2015 Demo web app

In this part of the tutorial, you learn how to develop the ASP.NET 5 Demo shown at[Connect(); //2015](https://blogs.msdn.microsoft.com/visualstudio/2015/12/08/connectdemos-2015-healthclinic-biz/). In this walkthrough, we show you how to create the skeleton and the structure of the solution. The completed project is in the following location:

Download the [Demo solution](https://github.com/Microsoft/HealthClinic.biz/blob/master/01_Demos_ASPNET5.sln), which is the result of this walkthrough, to see the structure and organization.

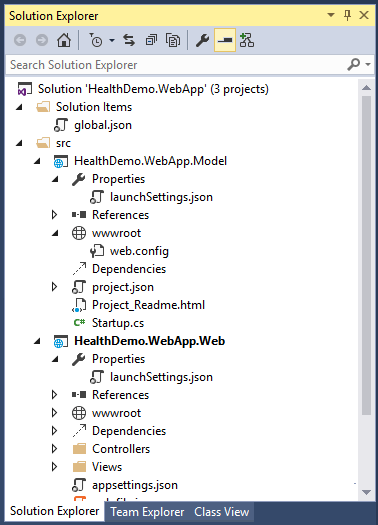
## HealthDemo.WebApp.Web

Use the solution that you created in the previous section of this article, and rename the project as HealthDemo.WebApp.Web. The project contains the website that users see, the client-side code of the web application, and the definition of the view.

Make sure that you define this as a startup project and a project that will be deployed in Azure. This project will contain references to other projects including the one that’s included in the solution. You will use the projects to add data, information, and context to the web application.

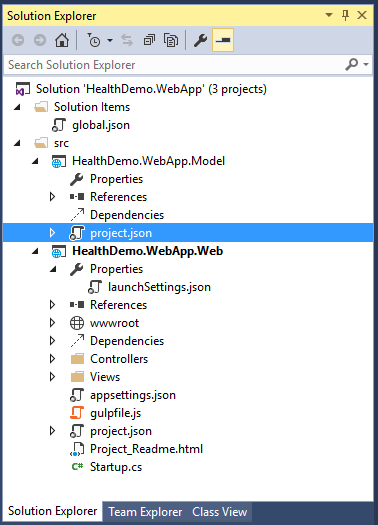
## HealthDemo.WebApp.Model

The following steps help you add an ASP.NET 5 empty project. You should call the project HealthDemo.WebApp.Model. The project contains the model files of the application. Model files are a list of classes that define the data model of the web application that the other projects will use.



The following files that will be created must be deleted because they aren’t necessary or useful:

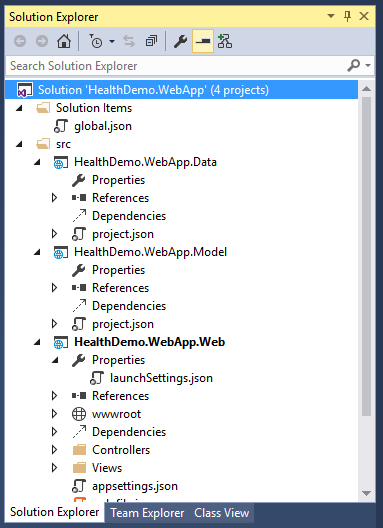
* File Properties > launchSettings.json.
* Node wwwroot.
* File Project\_Readme.html
* FileStartup.cs



Although you could create a Class Library project type , we recommend an ASP.NET project type to contain all references, dependencies, and components that a web project needs.

## HealthDemo.WebApp.Data

Create an empty ASP.NET 5 project named HealthDemo.WebApp.Data. Like the previous project, you need to delete the same files and leave an empty project. This project will contain repositories, initial application data, and the infrastructure that is used in the database to store the information about the web application.

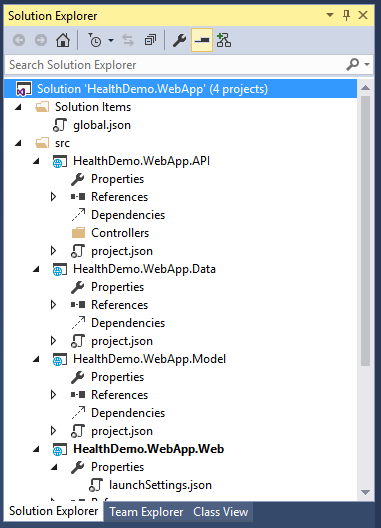


The MyHealthContext file in the project contains the database context as well as two directories:

* **Infrastructure**: Contains the data infrastructure that the web application uses, the data initialization, and the sample images that are used in the application itself.
* **Repositories**: Contains the database repositories that the web application uses. You can find the repositories that access the data in the database and that will be used by the controllers of the application.

## HealthDemo.WebApp.API

Finally, create an ASP.NET 5 empty project named HealthDemo.WebApp.API. Delete the files that are indicated in the previous steps so that the final structure of the solution looks like the following illustration:

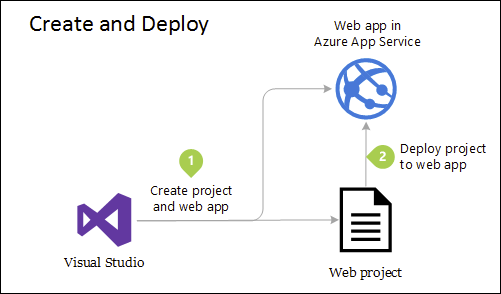


This project will contain the Web Application API. This is the web service that will provide the data to the web application. The apps have the controllers that have the calls that the application needs to access. The project also uses the repositories that are located in the Data project and the data model that’s in the Model project.

From this point, you have the basic structure of the web applicationand the availability to deploy it in Azure. Details about the content of each of the projects can be consulted in the repository of the demo shown in Microsoft Connect(); //2015.

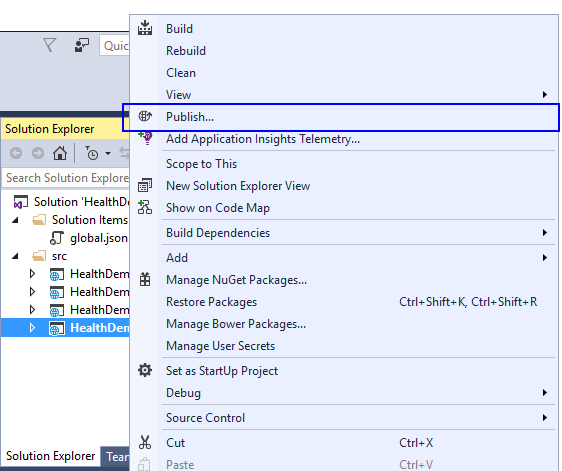
# Deploy the project to a web app in Azure

To have the web application in an App Service in Azure, just follow a few steps.

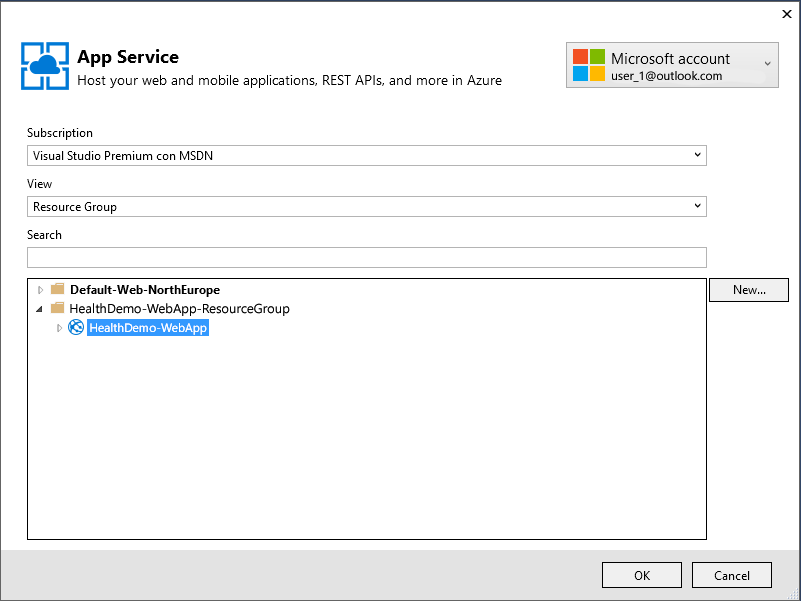


With the previous steps, you already generated the Azure resources that can help create more deployments.

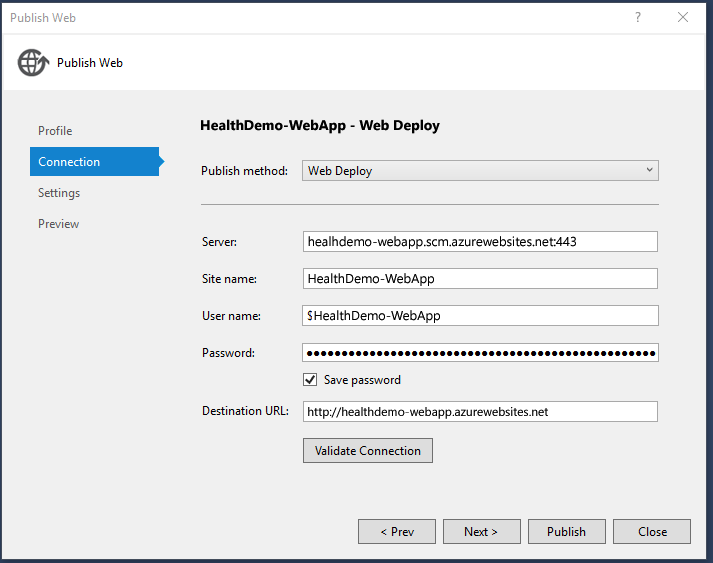
1. In Solution Explorer, right-click the HealthDemo.WebApp.Web project, and then click **Publish**.



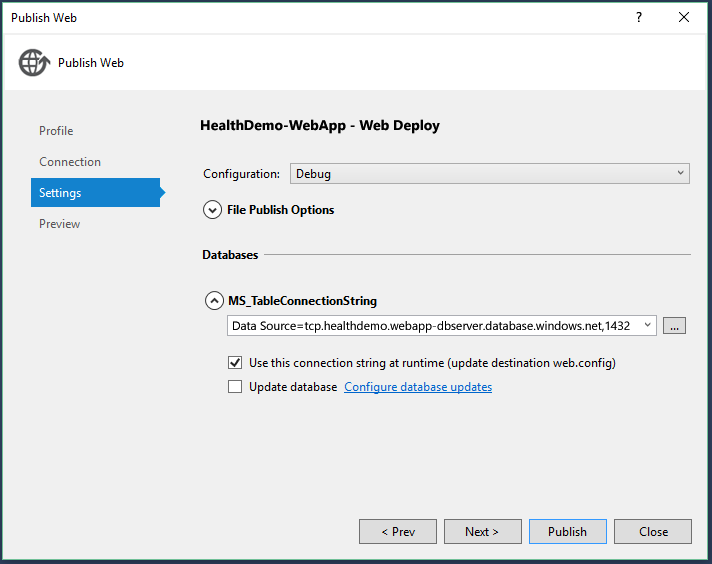
1. You will see the **Publish Web** dialog box. The wizard shows you a list of available **Publish** profiles. If you select **Microsoft Azure App Service**, you can see a list of available Azure subscriptions and the resource groups that were previously created. Among them is the resource group that you used to deploy the web application that you configured in the previous steps.



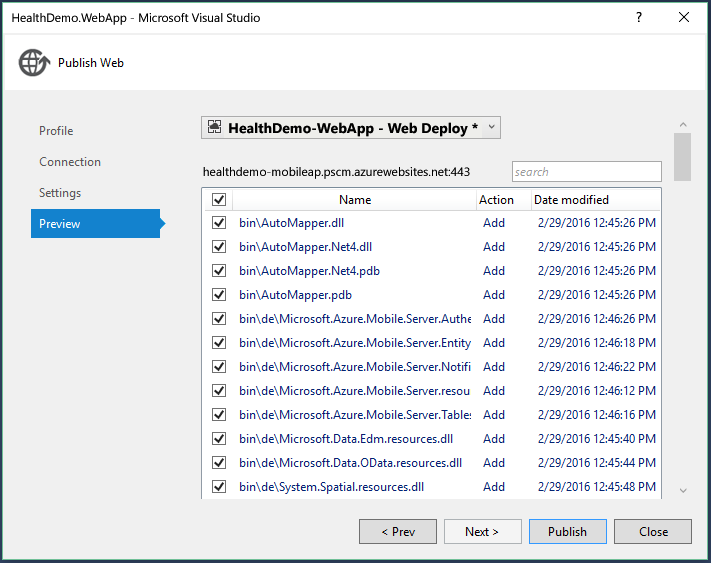
1. After you select the resource group, the page that opens shows the connection information. The default parameters will populate the fields. You can modify the fields if necessary. To check that the connection works correctly, click **Validate Connection**.



1. In the **Settings** page, you can configure the deployment type depending on whether you require a deployment in a production environment or in a debug environment. You can see the connection to the database.

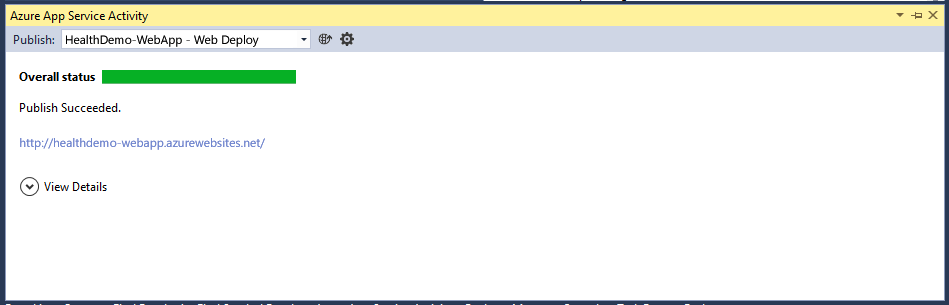


1. On the last page, **Preview**, you can determine the changes that affect the Azure environment.

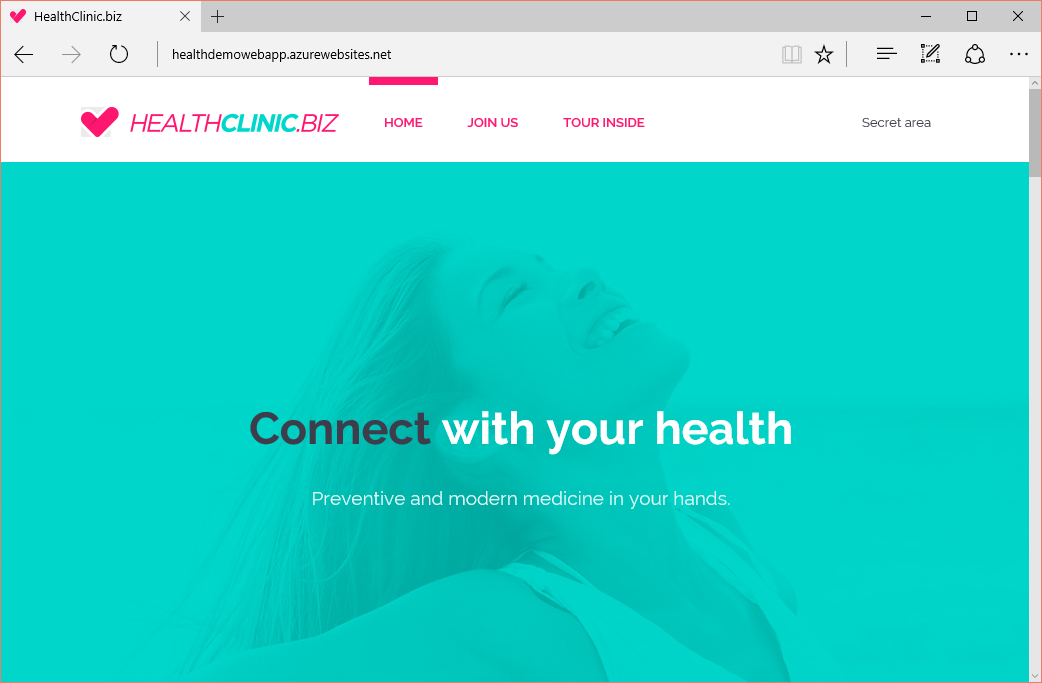


1. At last, click **Publish**.

The **Output** window displays information about the deployment. When it’s finished, a message that everything has been completed successfully is displayed.



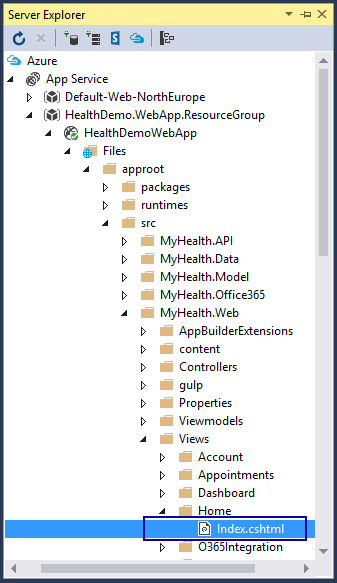
You can see the final result in the browser that will open.



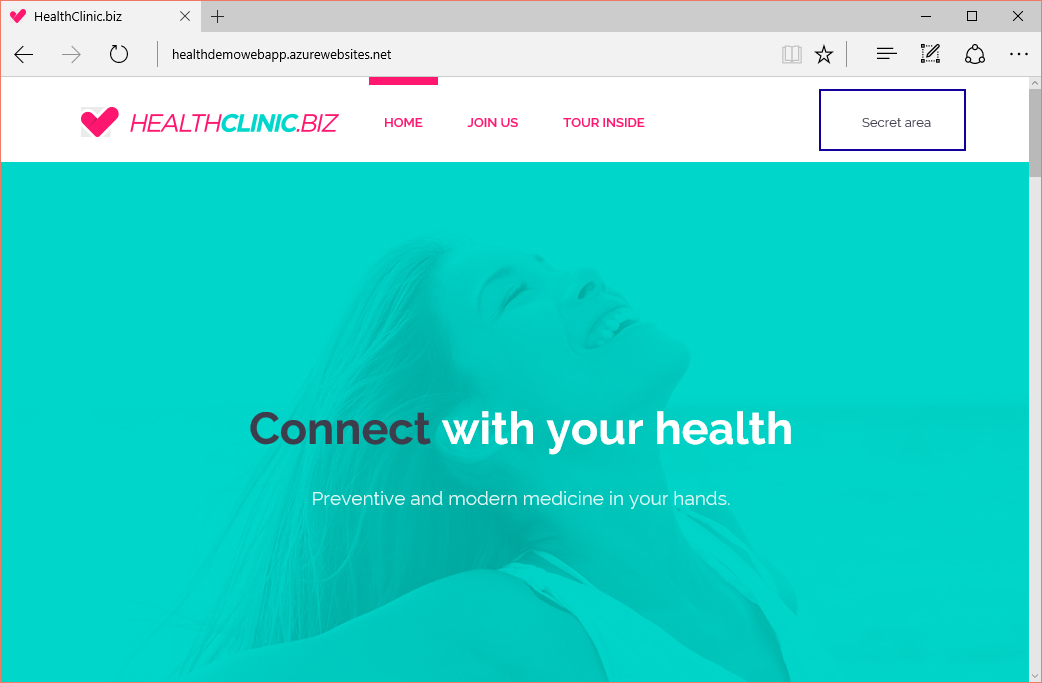
# Open remote files in Server Explorer

After the web application is deployed in Azure, you can make temporary changes to the remote site. To test and debug in a quick way, you only need to open and modify the files in Server Explorer.

1. You need to be signed in to Azure by using Server Explorer.
2. In Server Explorer, expand the node at **Azure** > **App Services** > **HealthDemo.WebApp.ResourceGroup**.
3. Expand the node **Files** > **approot** > **src** > **MyHealth.Web** > **Views** > **Home**, and open the Index.cshtml file.



1. Replace *<li><a asp-controller="Account" asp-action="Login">Private area</a></li>* with *<li><a asp-controller="Account" asp-action="Login">Secret area</a></li>*
2. Save the file.
3. Open a browser and open the web app in Azure App Service that you previously created.



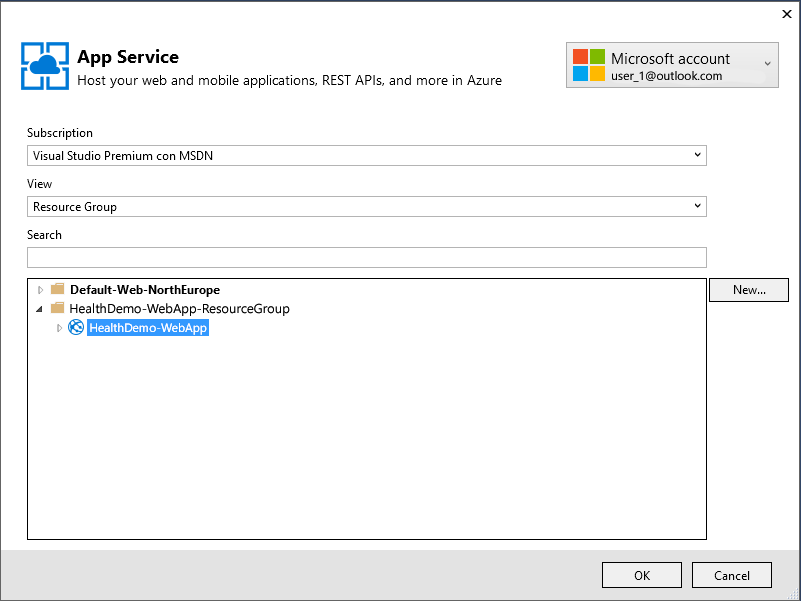
When you see a change in the web, the change is implemented on the website in Azure but not in the local project. So, if you redeploy the project in Azure, these changes will be reversed.

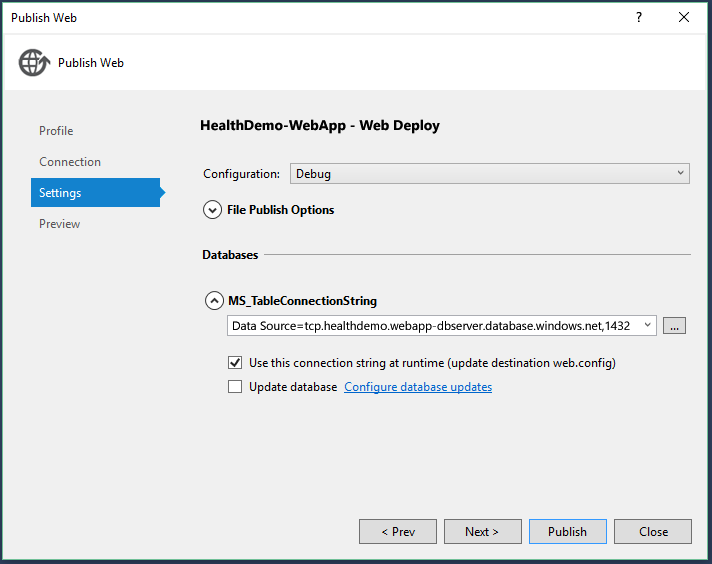
It is also possible to access the configuration of the web application from Visual Studio, start a remote debugging session, and view logs on the application in real time.

# Remote debugging

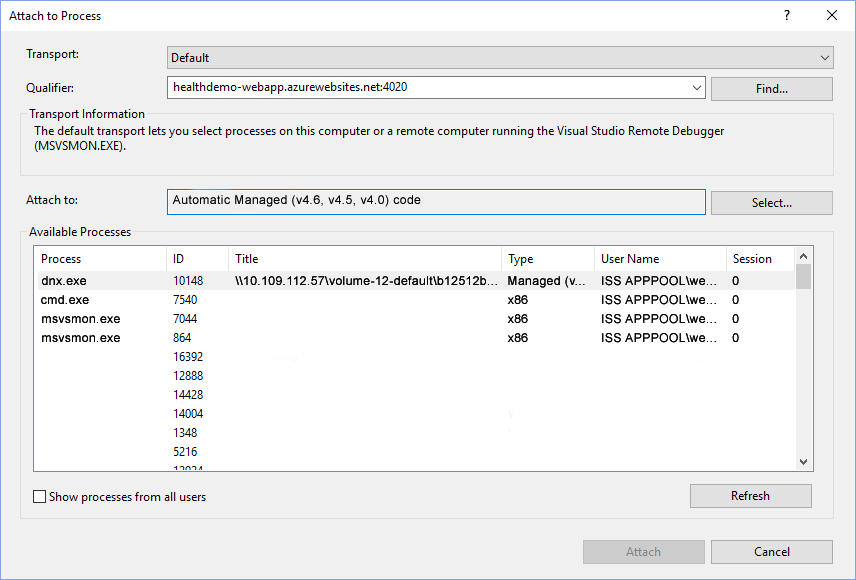
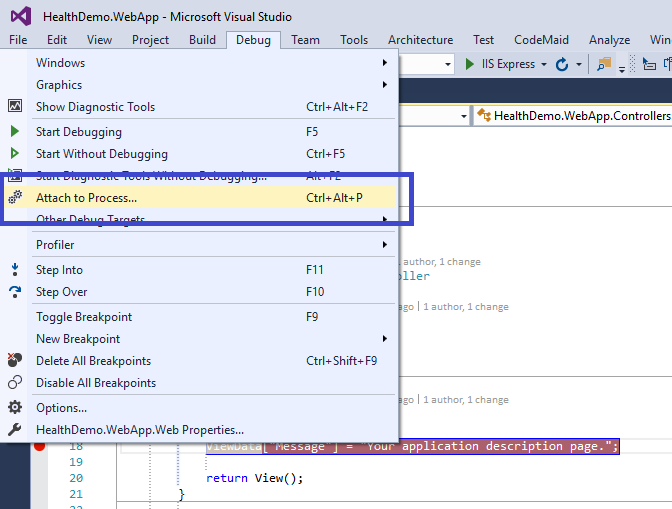
You can place a breakpoint on your solution and debug your application remotely from Azure in Visual Studio*.*

Right-click **Publish** again and publish debug files/symbols so that you can show remote debugging.

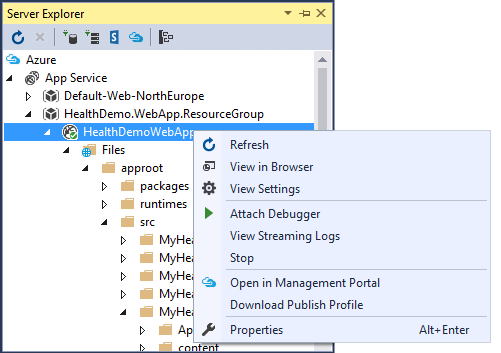




After publishing is finished, on the **Debug** menu, click **Attach to Process**, enter the azurewebsites URL along with its port, and select the **dnx.exe** process.



Finally, from **Server Explorer** -> **App Service**, right-click the **Application Service** node, and attach the debugger. As soon as you run through the breakpoint, the application will stop to be debugged.

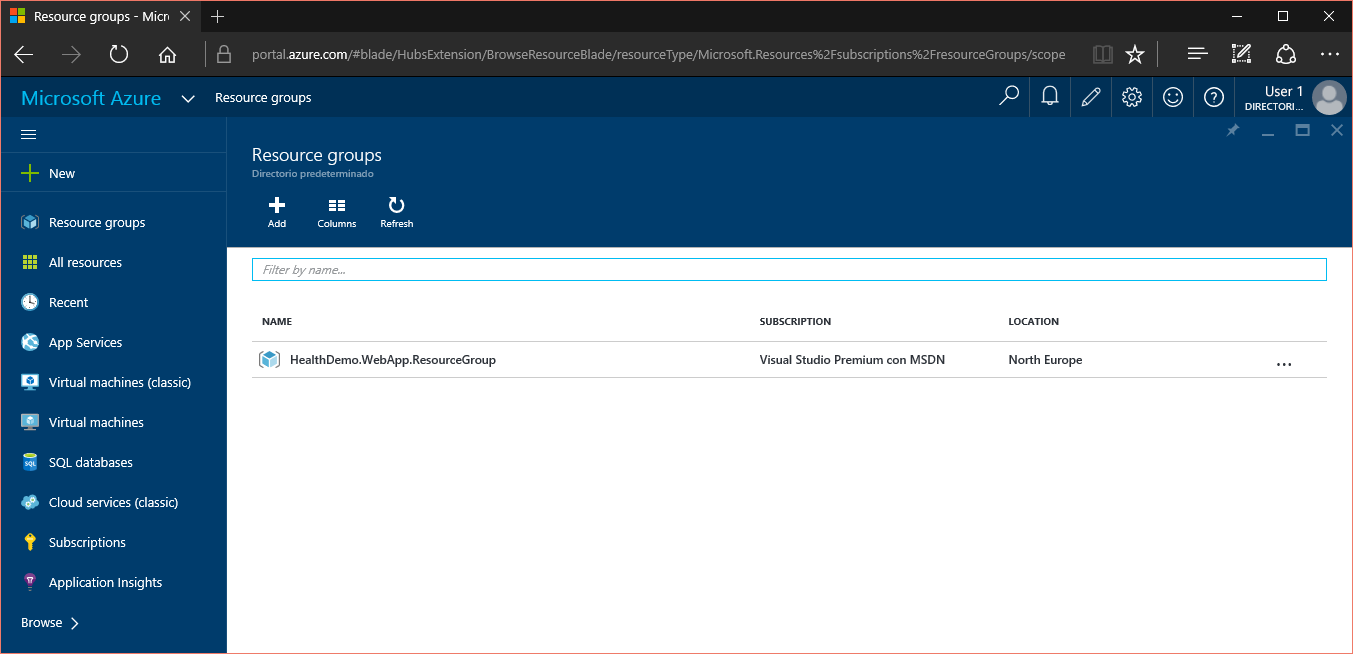


To learn more about remote debugging, see the [Remote Debug ASP.NET Core RC1 on Azure App Service](https://blogs.msdn.microsoft.com/webdev/2016/03/21/remote-debug-aspnet-core-on-azure/) blog post.

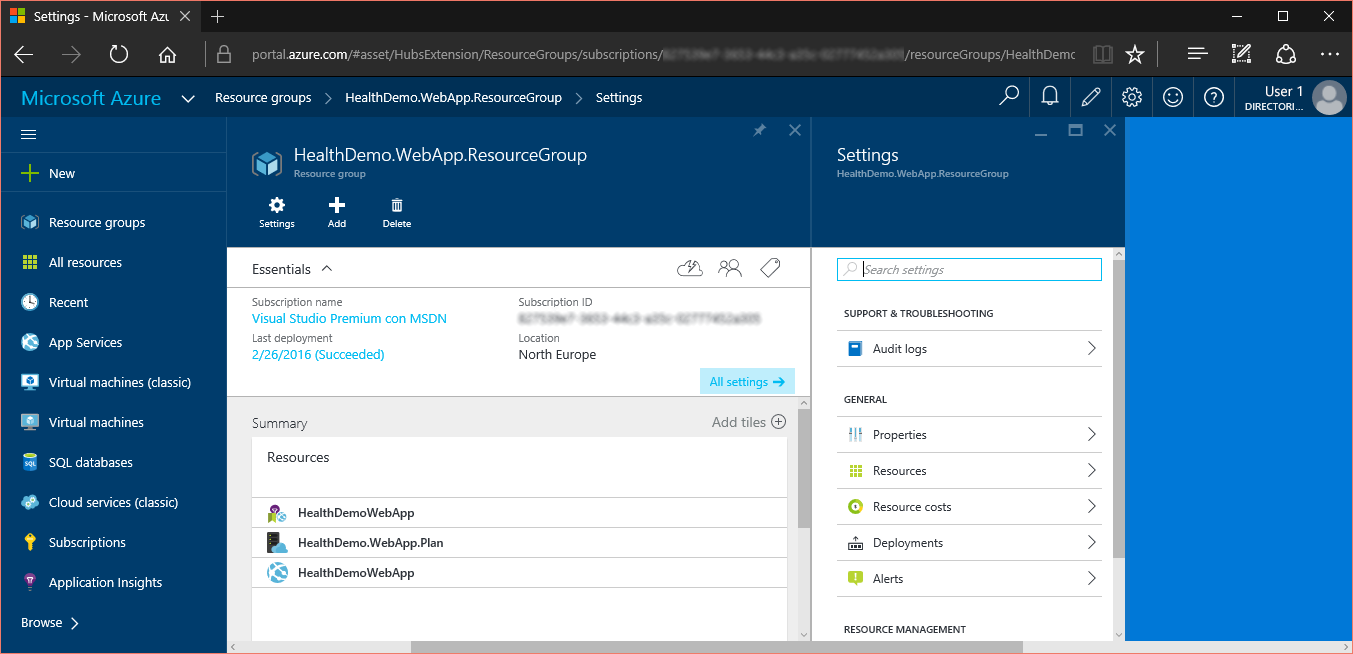
# Monitor and manage the web app in the Azure portal

You can manage and monitor the services in Azure from the [Azure portal](https://portal.azure.com/). This includes the web application that you just developed.

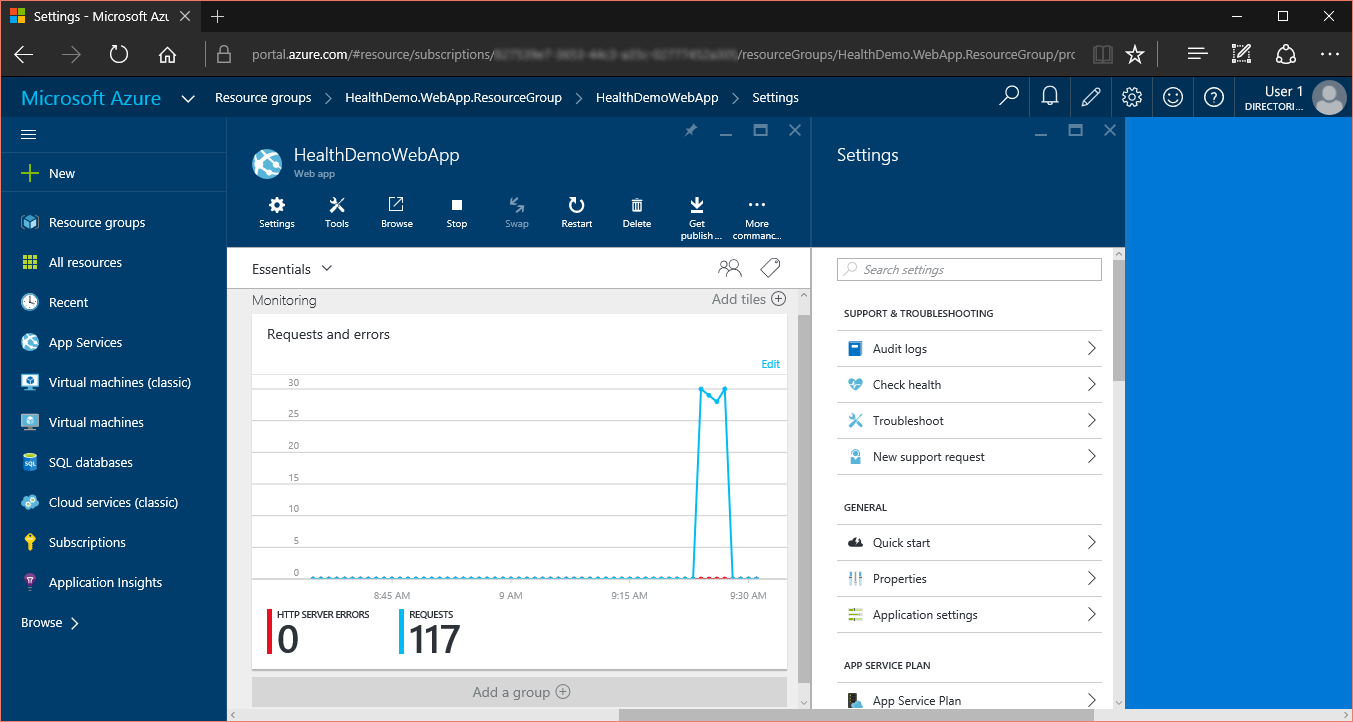
1. In a web browser of your choice, go to the [Azure portal](https://portal.azure.com/) and sign in with your Azure credentials.
2. Click **Resource groups**. Here you can see resource groups for each of the parts that make up the HealthClinic demo that you created in this tutorial: HealthDemo.WebApp.ResourceGroup.



1. When you click a resource group, you will see a summary of the resources that are part of the group, as well as the information about the group and its configuration.

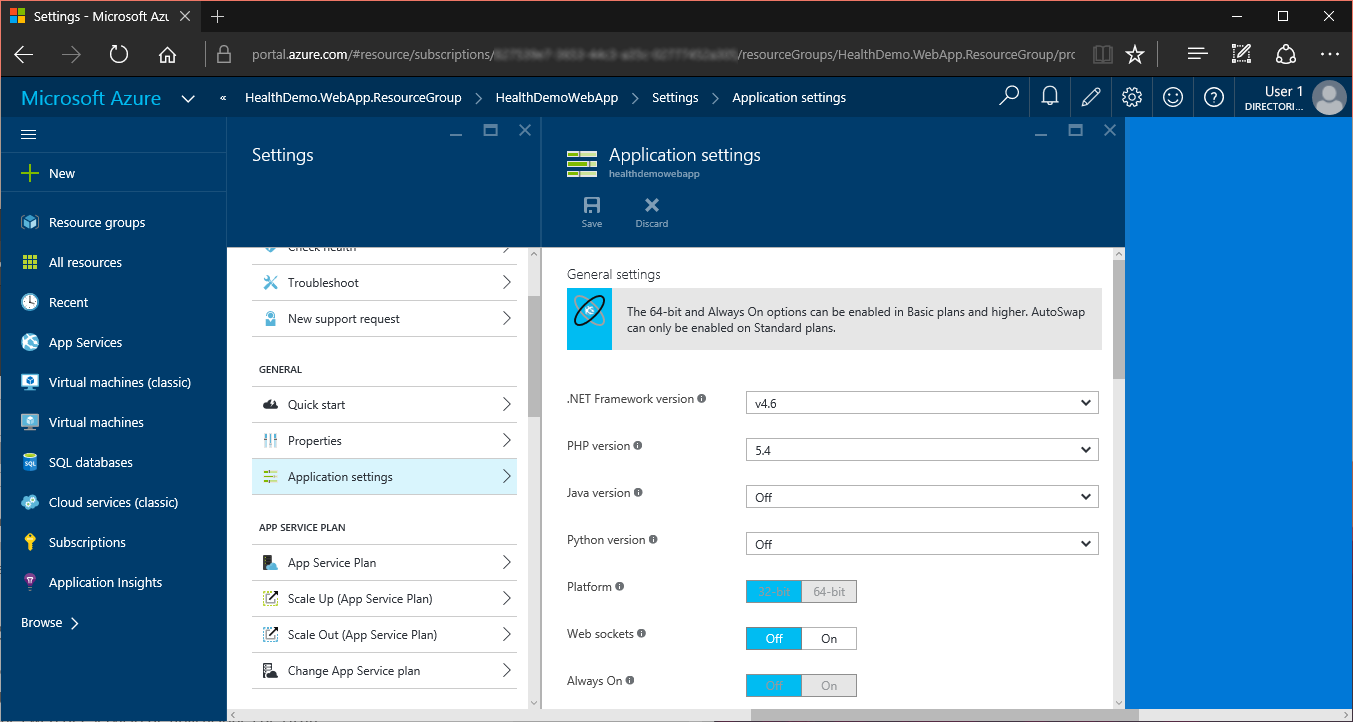


1. You can click the web application that is contained in the group, HealthDemoWebApp, to view general resource information, statistics, and content about the configuration.



It is possible that statistics are not real or that statistics are not displayed. Explore the application to start to show statistics in the portal.

1. In the list of configuration types, you can explore different configurations that are available in your application. One is **Application settings**. From here, you can modify some important aspects of the application.



From the portal, you can perform many actions that are related to applications and other Azure services, such as managing databases and virtual machines.

# Azure Key Vault

Azure Key Vault helps developers implement cryptographic functionality in an application by managing passwords and keys. The service can also store secrets (small pieces of encrypted data).

* Azure Key Vault can be used to encrypt and decrypt keys and secrets by using keys that are stored within the Azure Key Vaultservice and that are protected by Hardware Security Modules (HMS)*.*
* The stored keys can be used directly to encrypt and decrypt sensitive data.

If you want to learn more about Azure Key Vault, see the following resources:

* [What is Azure Key Vault?](https://azure.microsoft.com/en-us/documentation/articles/key-vault-whatis/)
* [Get started with Azure Key Vault](https://azure.microsoft.com/en-us/documentation/articles/key-vault-get-started/)
* [Use Azure Key Vault from a web application](https://azure.microsoft.com/en-us/documentation/articles/key-vault-use-from-web-application/)

# Other suggested topics to explore

* [What are Azure App Services?](https://azure.microsoft.com/en-us/documentation/articles/app-service-value-prop-what-is/" \l "get-started-with-app-service)
* Other ways to deploy a web project

For information about other ways to deploy web application projects to web apps by using Visual Studio or by [automating deployment](http://www.asp.net/aspnet/overview/developing-apps-with-windows-azure/building-real-world-cloud-apps-with-windows-azure/continuous-integration-and-continuous-delivery) from a [source control system](http://www.asp.net/aspnet/overview/developing-apps-with-windows-azure/building-real-world-cloud-apps-with-windows-azure/source-control), see [How to deploy an Azure web app](https://azure.microsoft.com/en-us/documentation/articles/web-sites-deploy/).

Visual Studio can also generate Windows PowerShell scripts that you can use to automate deployment. For more information, see [Automate everything (Building real-world cloud apps with Azure)](http://www.asp.net/aspnet/overview/developing-apps-with-windows-azure/building-real-world-cloud-apps-with-windows-azure/automate-everything).

* How to add a custom domain name and SSL

For information about how to use SSL and your own domain (for example, www.healthdemo-webapp.com instead of healthdemo-webapp.azurewebsites.net), see the following resources:

* + [Configure a custom domain name in *Azure* App Service](https://azure.microsoft.com/en-us/documentation/articles/web-sites-custom-domain-name/)
  + [Enable *HTTPS* for an Azure web app](https://azure.microsoft.com/en-us/documentation/articles/web-sites-configure-ssl-certificate/)
* How to add real-time features such as chat

If your web app will include real-time features (such as a chat service, a game, or a stock ticker), you can get the best performance by using [ASP.NET SignalR](http://www.asp.net/signalr) with the [WebSockets](https://azure.microsoft.com/blog/2013/11/14/introduction-to-websockets-on-windows-azure-web-sites/) transport method. For more information, see [Using SignalR with Azure web apps](http://www.asp.net/signalr/overview/signalr-20/getting-started-with-signalr-20/using-signalr-with-windows-azure-web-sites).

* How to choose between App Service, Azure Cloud Services, and Azure virtual machines for web applications

In Azure, you can run web applications in App Service, as shown in this walkthrough, in cloud services, or on virtual machines. For more information, see [Azure web apps, cloud services, and *VMs*: When to use which?](https://azure.microsoft.com/manage/services/web-sites/choose-web-app-service/).

* [How to choose or create an App Service plan](https://azure.microsoft.com/en-us/documentation/articles/azure-web-sites-web-hosting-plans-in-depth-overview/)
* [How to choose or create a resource group](https://azure.microsoft.com/en-us/documentation/articles/resource-group-portal/)
* [Azure Developer Tools](http://azure.com/tools) – Learn more about the tools and find the recently released installers.