Snapshot Debugger

# Overview

# The Snapshot Debugger is a new and powerful diagnostic experience coming to your .NET (Framework and Core 2.0) apps, hosted in Azure. The Snapshot Debugger is available through the Azure portal as a standalone capability; but also integrates with Visual Studio Enterprise 2017 to provide a powerful F5 like debugging experience inside Visual Studio, without impacting the production application.

# Objectives

Completing this Quick Start should make you familiar with the Snapshot Debugger capabilities and help you understand how to take advantage of these capabilities to reduce the mean-time-to resolution. You can use Snapshots of production issues to help you get directly to the root cause of problems in your .NET/Azure hosted production apps.

# Prerequisites

The Virtual Machine that accompanies this Quick Start comes with all the software configured for you to quickly complete this lab. All you will need is an Azure Subscription.

* + If you do not have an Azure Subscription, you can join our [Visual Studio Dev Essentials](https://www.visualstudio.com/dev-essentials/) program to gain free monthly Azure credits.

The username for this VM is **build** and the password is **QuickStart2017**

# **Intended Audience**

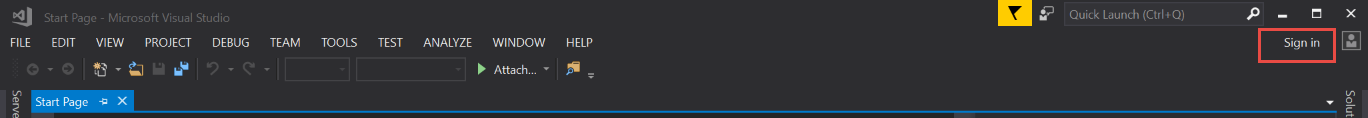
This Quick Start is intended for .NET developers who are hosting or who intend to host their applications in Azure.

# **Background**

You have an Azure web app running in production. The application allows users to enter visitor information for anyone visiting your company. You’re seeing reports from users that they cannot create new visits or even see the list of visitors and you need to investigate. We will walk through preparing an application to leverage the SnapShot Debugger, publishing that application and then using Application Insights to track down the problem.

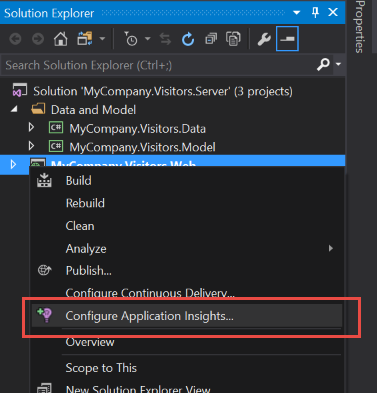
# Task 1: Open and configure the solution

1. Open Visual Studio Enterprise 2017
2. Connect Visual Studio to your Azure account

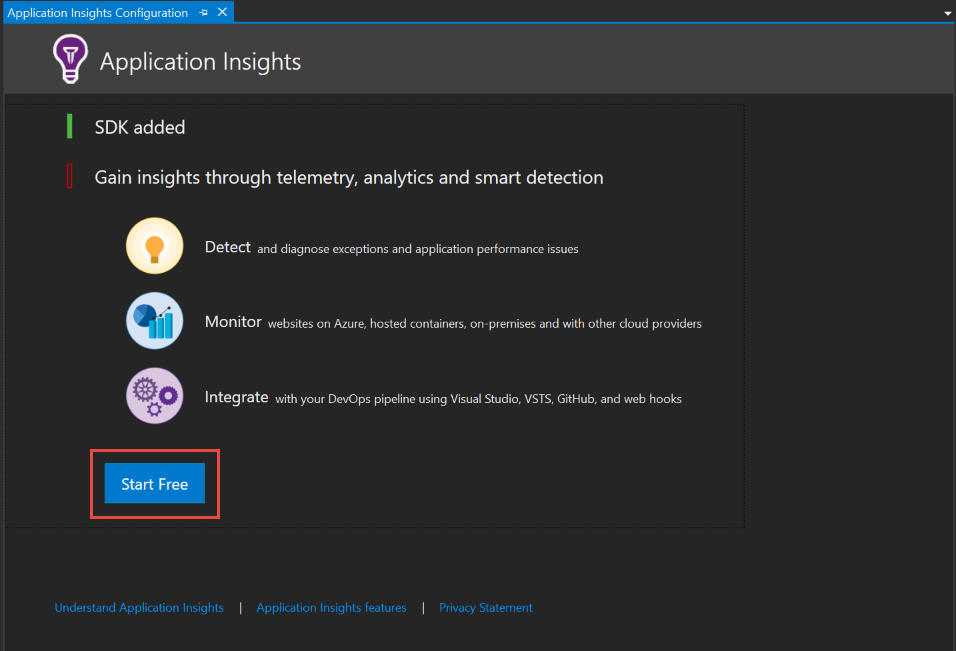
­­­­

And complete the sign in steps

1. Open the project on the Desktop, either through the File menu option or through Start Page
   1. Browse to C:\users\build\desktop\mycompany.visitors\
   2. Open the solution MyCompany.Visitors.Server.sln
2. Right click on the “MyCompany.Visitors.Web” project and select “Configure Application Insights

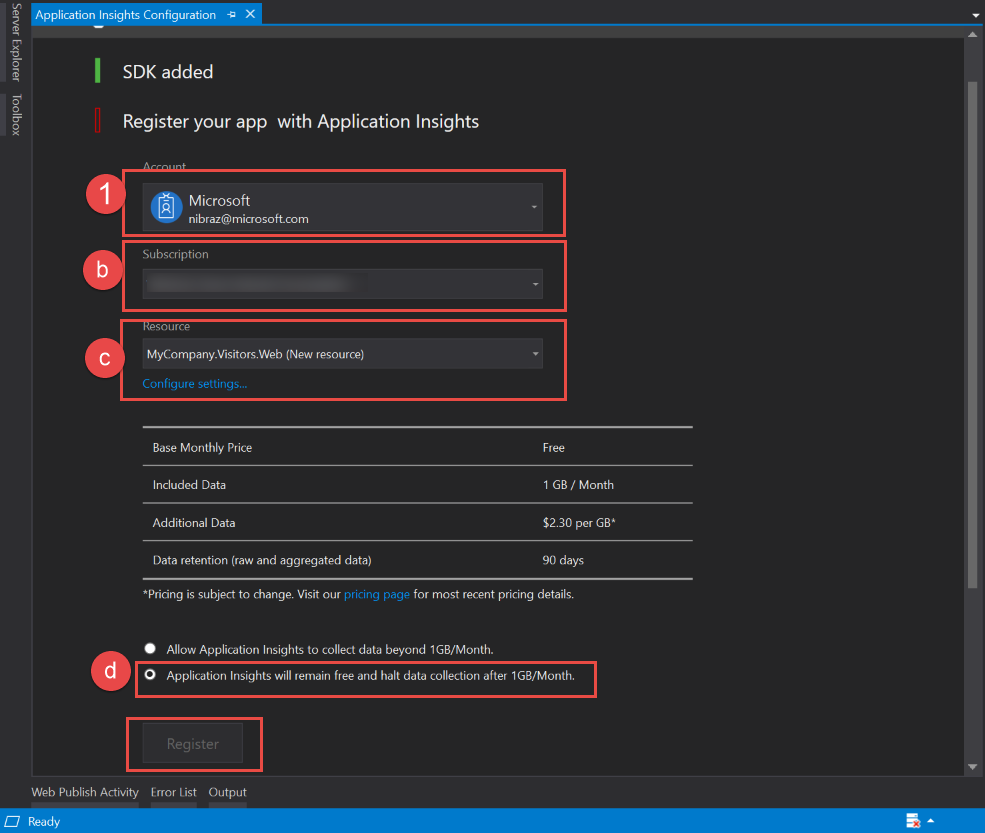


1. Click the “Start Free” button

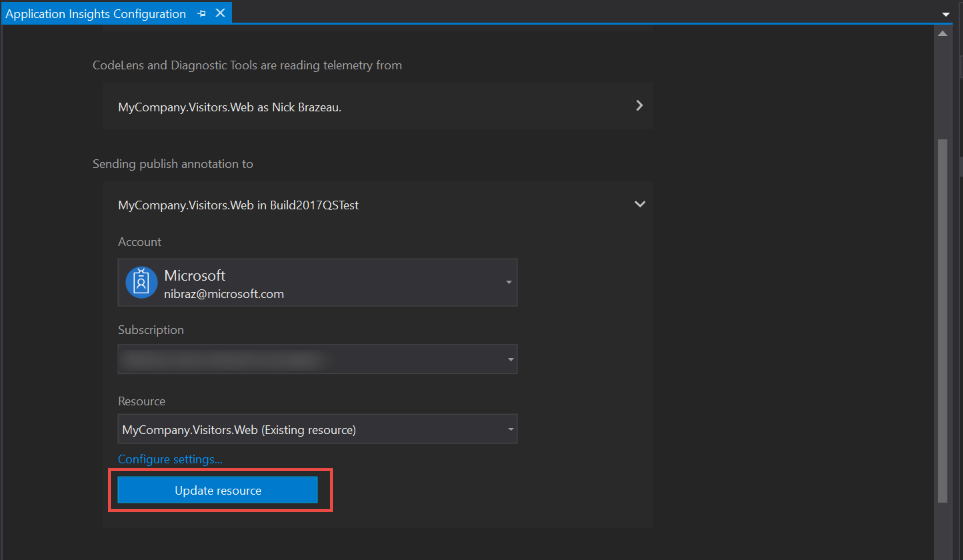


1. Configure the Application Insights resource creation by
   1. Verifying your account
   2. Selecting your subscription
   3. Configuring the resource, providing the
      1. Resource Group name: azure-snapshot-lab
      2. Application Insights Resource name: MyCompany.Visitors.Web
   4. If you have already selected a billing model for this subscription, skip the next step.
      1. Switch the billing model “Application Insights will remain free and halt data collection after 1GB/Month” for this lab. You can change this through the Azure portal later if you wish

Then click the “Register” button to trigger the resource creation in Azure

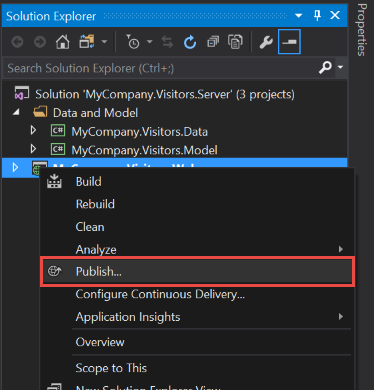


1. Once the resources are created in Azure, click the “Finish” button to dismiss the dialog
2. To complete the Application Insights configuration, click the “Update resource” button in the “Sending publish annotations to” section.

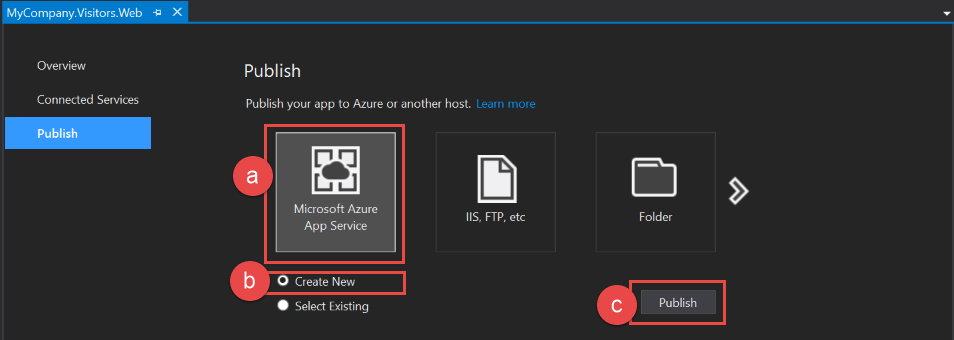


# Task 2: Publish the app

1. Right-click on the MyCompany.Visitors.Web project, and select the “Publish” option

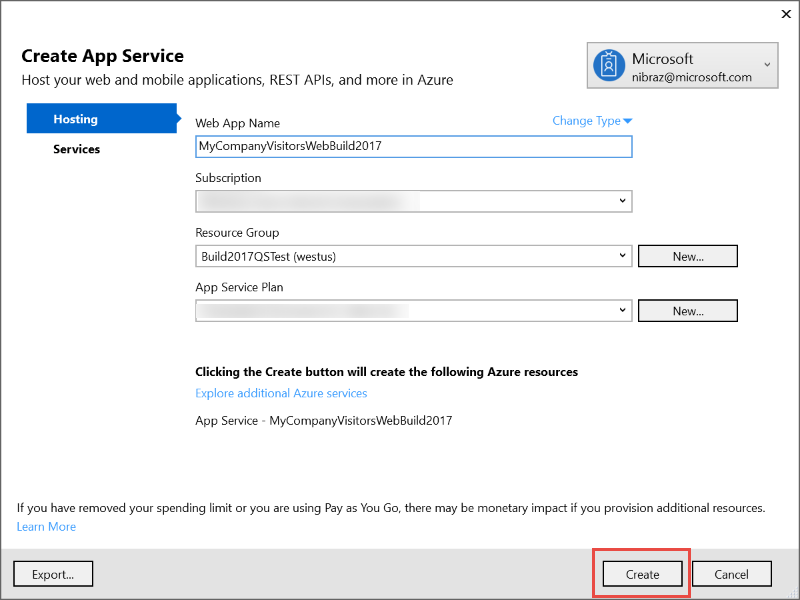


1. When you publish:
   1. Select a Microsoft Azure App service
   2. For the lab, we will create a new App Service
   3. And Publish

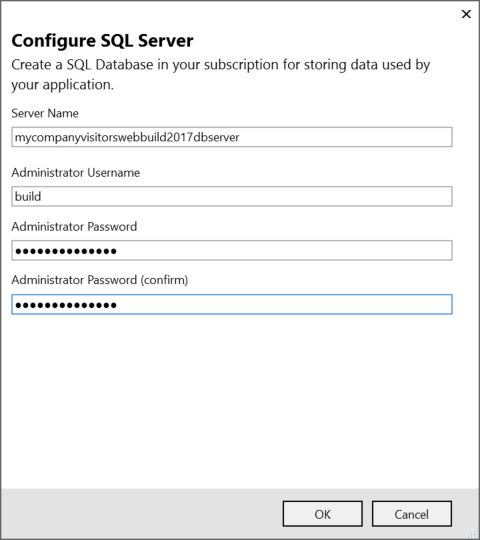


1. Configure your app service by
   1. Providing an app name
   2. Verifying you are deploying to the correct subscription
   3. Selecting the Resource Group you created when you configured Application Insights: azure-snapshot-lab
   4. Select the App Service Plan for the app

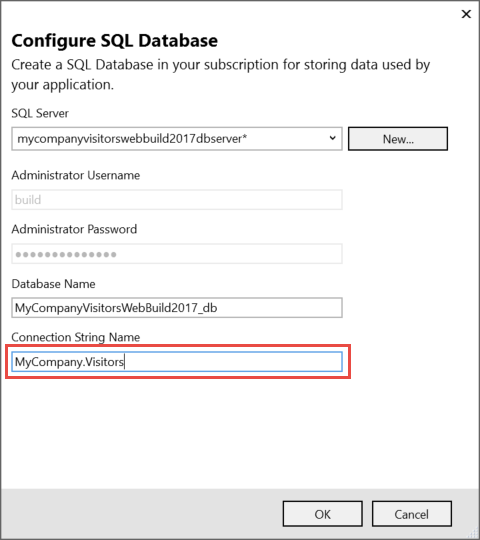
Before publishing, we need to add a database



1. To add a database
   1. Click on the Services option on the left
   2. Click the plus button on the right next to SQL Database, and configure the database and server as needed
      1. Configure the SQL Server if needed

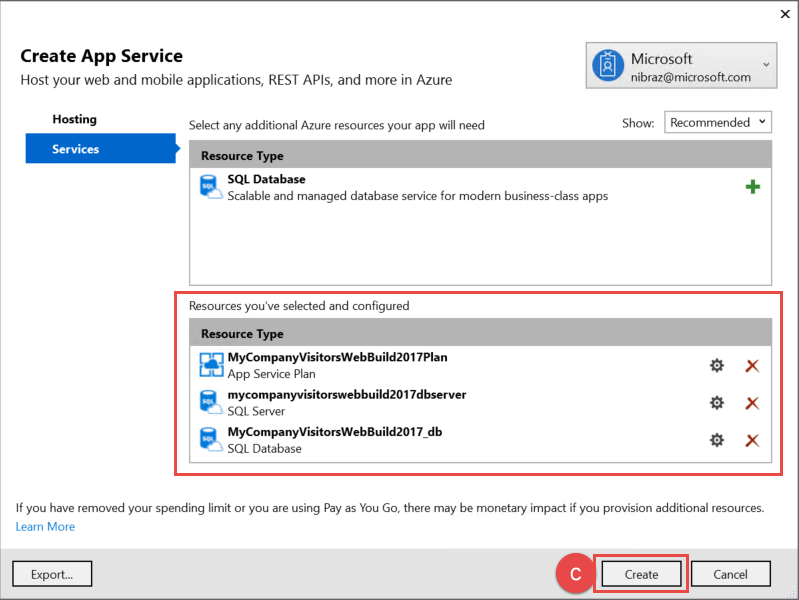


* + 1. Configure the SQL Database, and be sure to set the Connection String Name to MyCompany.Visitors to update the connection string on publish

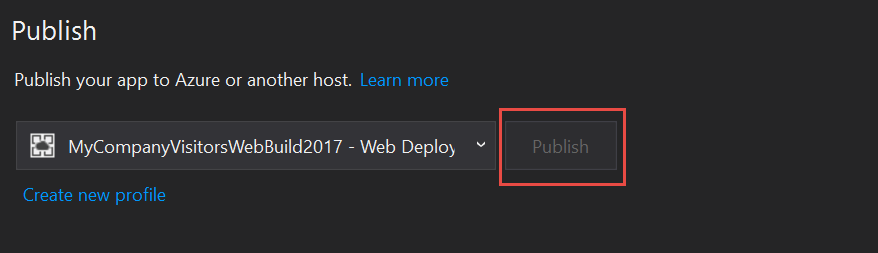


Then click OK to setup the creation of the Server and Database on publish

* 1. Then click Create to create the App Service Plan, App Service, Database Server and Database; and publish the app to Azure



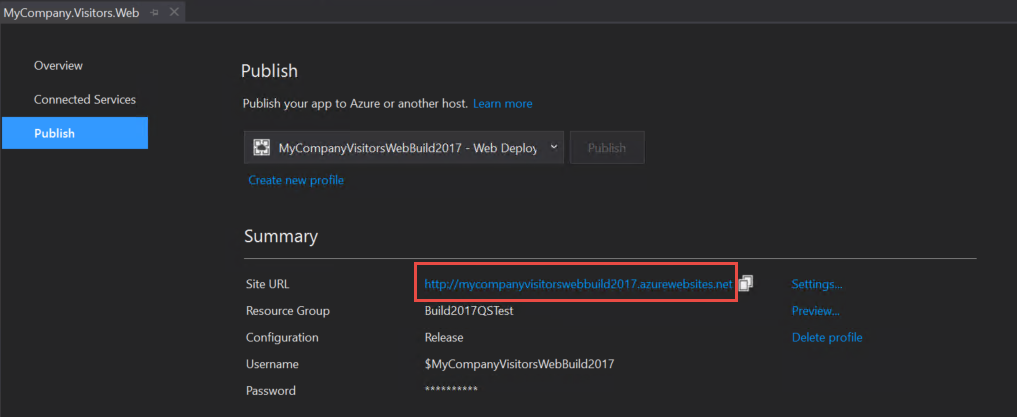
* 1. If this is the first time that you’ve published you app, it will be automatically published after your resources are created. Otherwise, click Publish to update your web app.



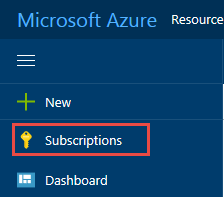
1. Wait for the resources to be created and the app to publish, and then we can move on to triggering SnapShots in a live app in Azure.

# Task 3: Debugging a running app with SnapShots

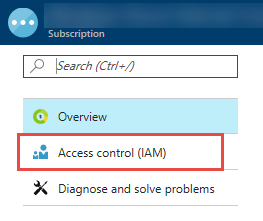
1. Once the application is deployed, we need to browse to the web app so we can begin to trigger some errors. The site should open in a web browser automatically, but if that does not happen you can use the URL shown in Visual Studio to easily navigate to the website:



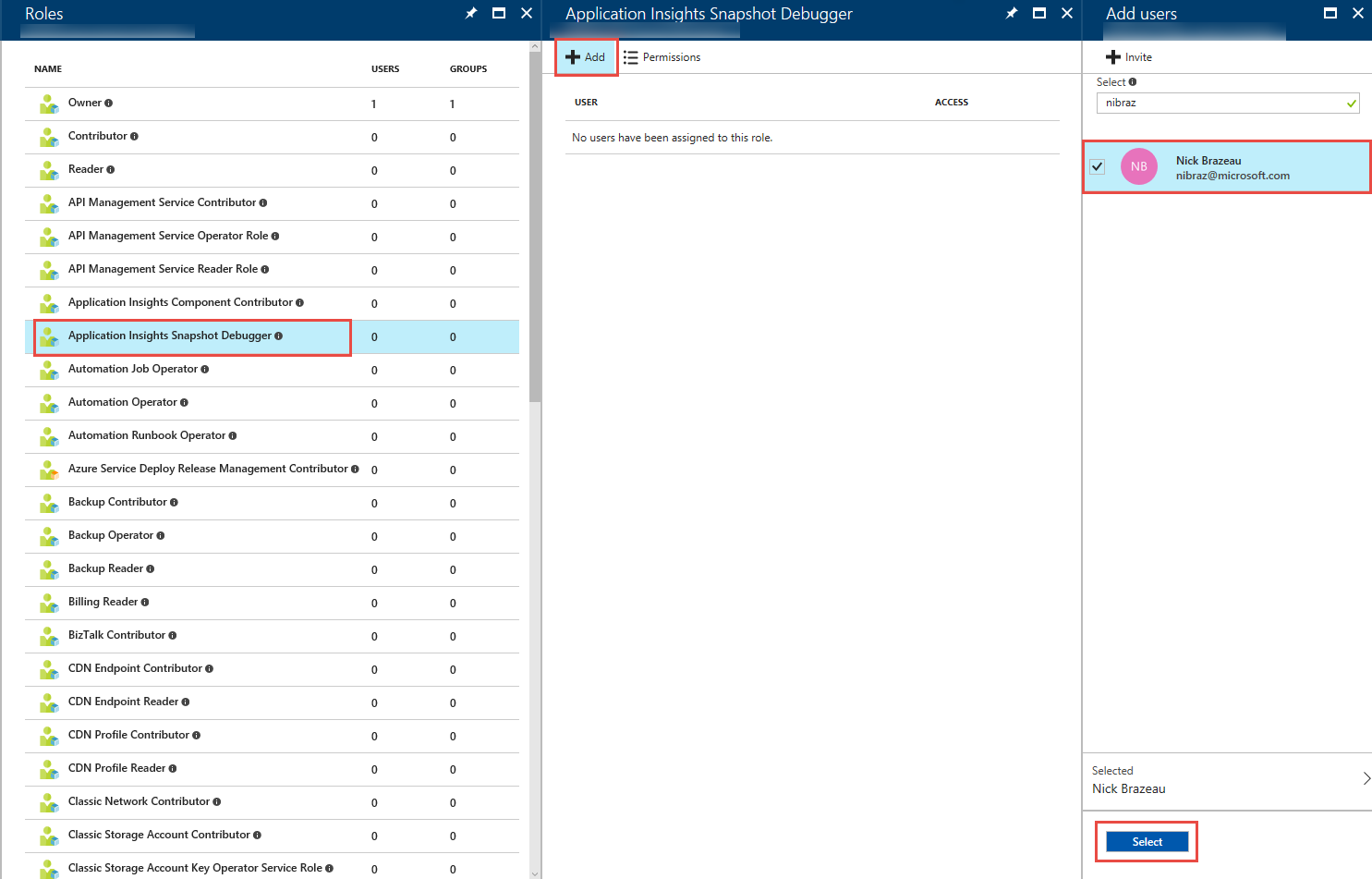
1. Next, we need to trigger some exceptions. Once the website loads, click the “Visitors” tab of the website 5 or more times to simulate a recurring problem.
   1. Note that automatic SnapShots are generated in the background for problems that occur frequently. A problem that occurs only once or rarely will not trigger an automatic SnapShot by default.
2. Now that we have triggered an error to be collected, we need to ensure you have permissions to view the SnapShots. SnapShots can contain sensitive information as they are the full environmental and application state when the exception occurs, so Microsoft requires that explicit permissions be granted for viewing SnapShots.
   1. In the Azure portal, Use the left-hand navigation bar to navigate to the subscriptions blade and select the subscription where your Application Insights resource is running.



* + 1. If you do not see Subscriptions in the left-hand navigation, you can find it through More Services at the bottom of that navigation bar
  1. Select the “Access Control (IAM)” blade and select “Roles” at the top



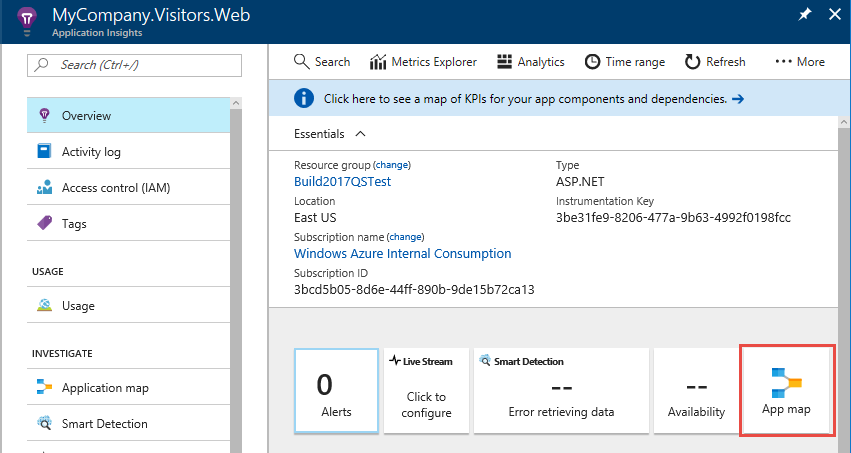
* 1. Switch the view to see all roles available in the subscription:
  2. Find the role “Application Insights Snapshot Debugger” role and
     1. Click on the role to open the blade to manage membership in the role
     2. Click Add
     3. Select your name from the list of users or simply start typing if you don’t see your name
     4. Click Select add yourself to the role and enable your account to view the SnapShots



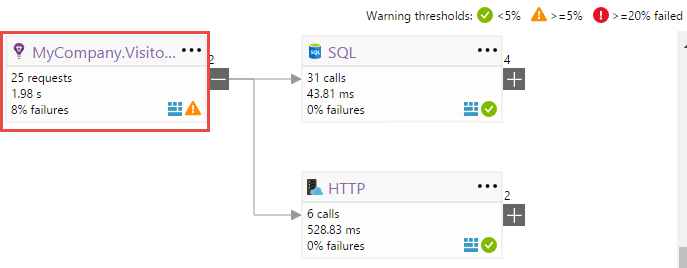
# Task 4: Debugging a running app with SnapShots

*Note: Depending on how quickly you have completed these steps the data may not yet be populated in your new Application Insights resource. If this happens just wait a few minutes for the data to appear.*

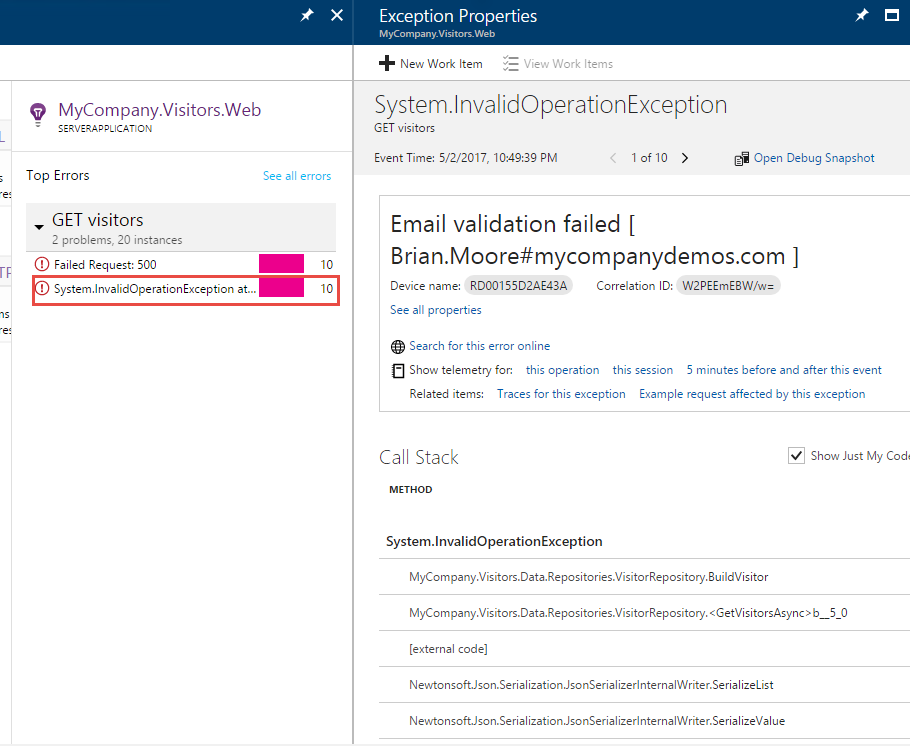
1. One of the best ways to see at a glance how your application is behaving is the App Map. To go to the App Map, open the Application Insights resource where your application is sending its telemetry, if you used the default naming it will be named MyCompany.Visitors.Web. select the App Map option



1. Notice that there are some exceptions on the MyCompany.Visitors.Web component in the App Map. Select the server node to see the information about the node, including the failures.

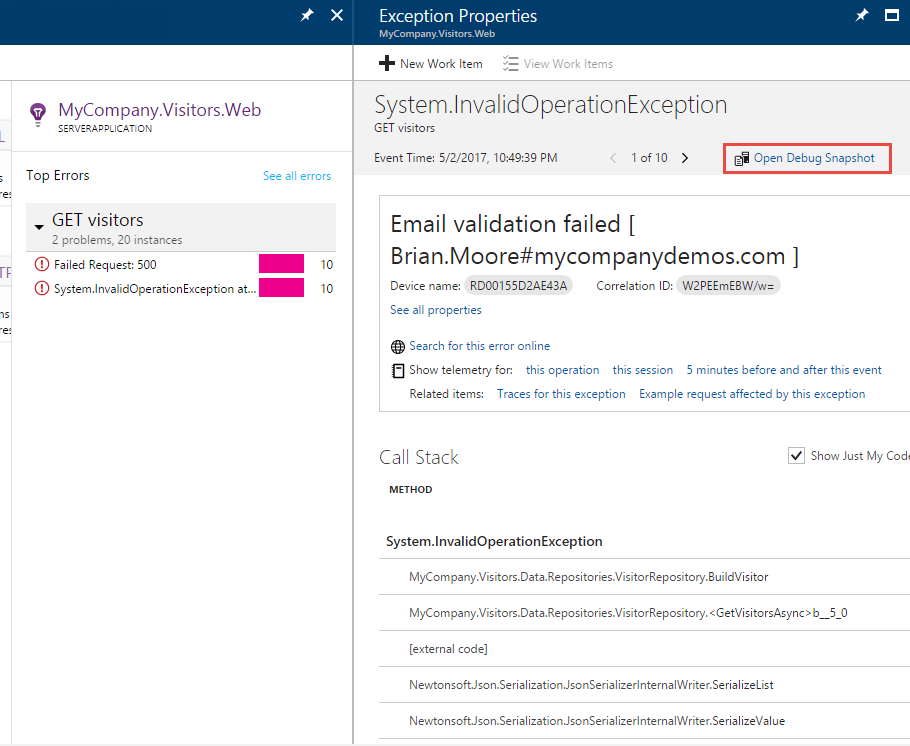


1. Notice that we have a System.InvalidOperationException – let’s explore that exception by clicking on it and opening the Exception Properties blade. You can immediately see a significant amount of detail about the exception.



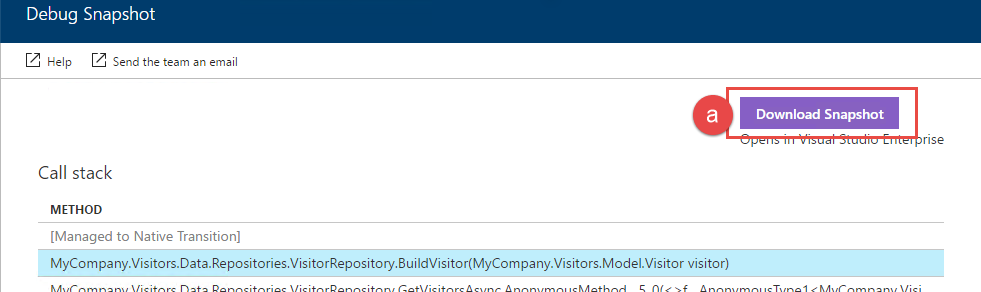
1. To open the Snapshot click on the Open Debug Snapshot link in the upper right to see a more detailed view of the exception, and the state of the application and environment when the exception occurred.

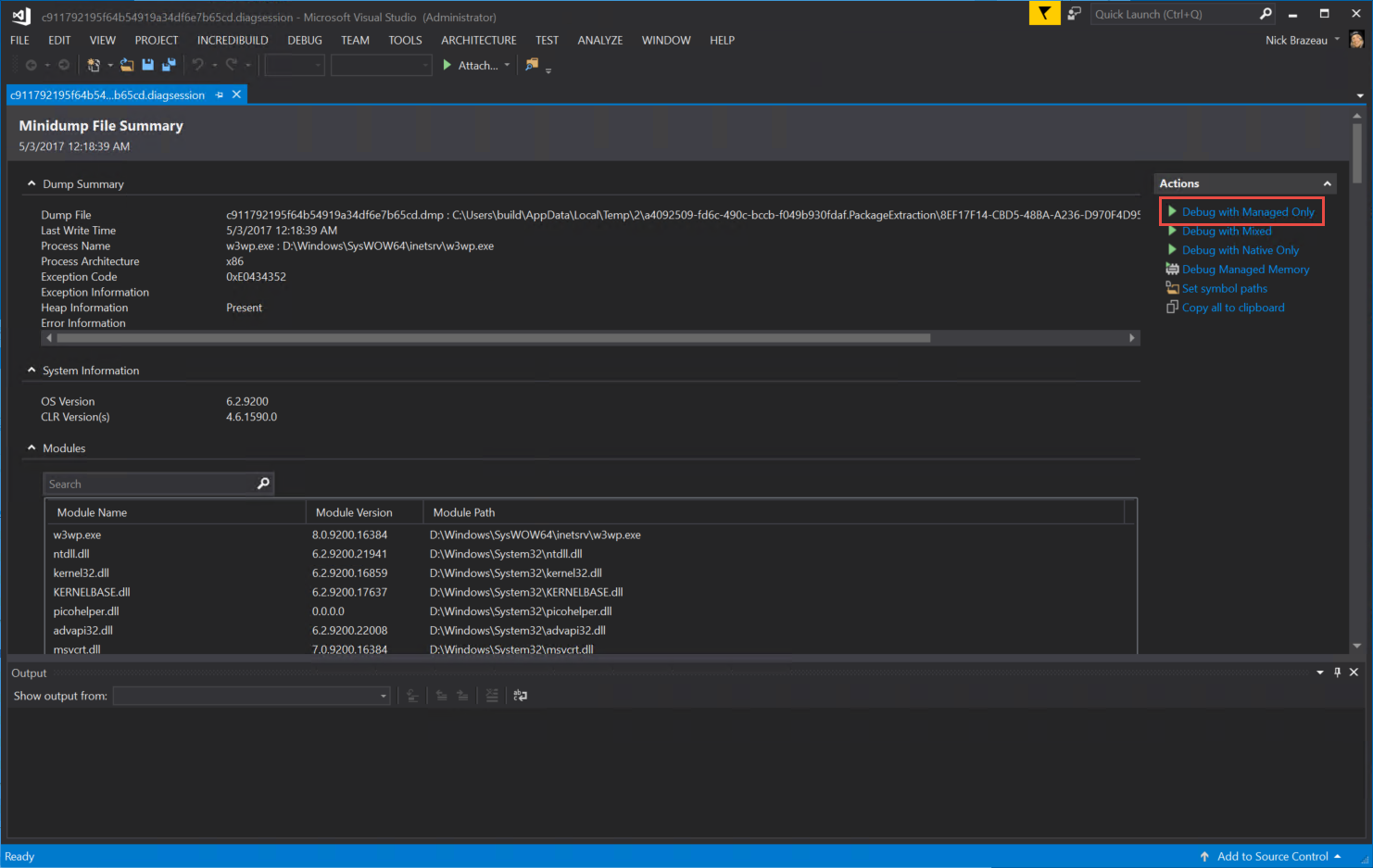
*If the exception you select does not contain a Snapshot, cycle to the next exception until you find one that does contain a Snapshot. Recall that there is a configurable threshold that must be met before a snapshot will be generated. Additionally, because snapshots are uploaded in the background to minimize impact on your side, it may take a few minutes for snapshots to appear.*

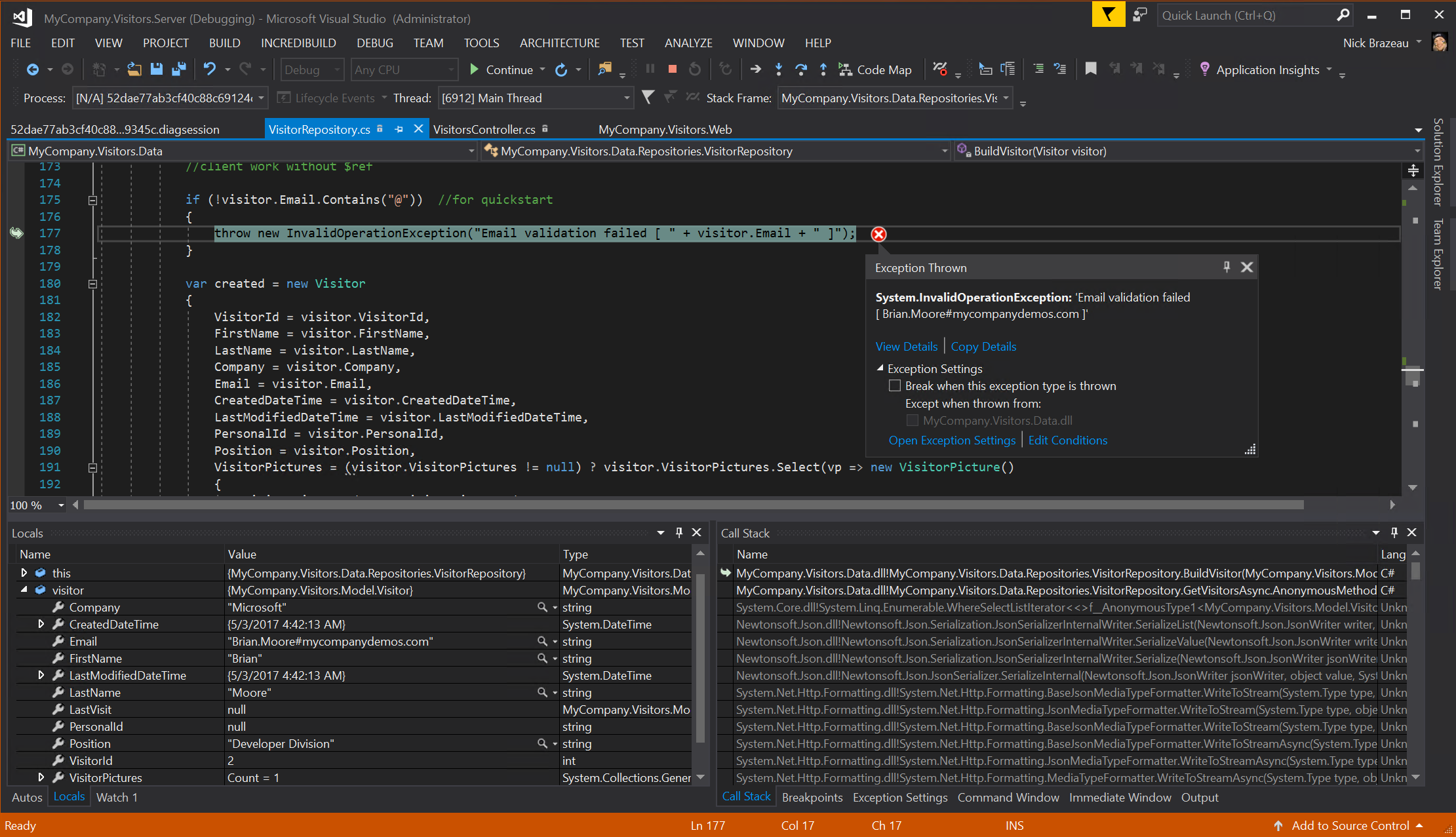


1. To get an even more in depth look at how the exception occurred, download the Snapshot and open it with Visual Studio Enterprise 2017 to walk through the application execution leading up to the exception
   1. Download the Snapshot.

If you are using Microsoft Edge or Internet Explorer, these tools change the extension automatically to a zip file, you will need to change it to a .diagsession extension manually so the file will open in Visual Studio automatically.



1. Once the Snapshot opens in Visual Studio, you can choose how to review the Snapshot. In this case, select “Debug with Managed Only” 
2. You are now debugging a Snapshot from production that was provided for you by Application Insights, because a series of exceptions occurred. You are taken to the exact line of code that caused the exception, with a Call Stack and Locals.



1. Before continuing to the next step, see if you can determine the problem given the information you have about line of code where the exception occurred, the outerMessage and the locals at the time of the exception.
2. Based on the information we have, we can see that we have a malformed email address (the email address contains # not @) – someone must have edited the production database directly, since our app has validation for these types of errors. From here you could decide to fix the database as well as modify your code so that a single bad entry won’t prevent the application from working completely.
3. Congratulations – you have quickly debugged an issue in production! Please take some time to complete the survey and let us know what you think.