Demo- Web Apps Production Dubugging & Application Insights

11 minutes

Value Props

The **two things developers** should take away from this demo:

Key Message –Deploy high performance serverless apps across the globe in milliseconds

1. **.NET and Azure App Service give developers unrivaled efficiency and production runtime visibility.**
    - Developers who use .NET on Azure App Service get rich telemetry with Application Insights automatically. It enables adding end-to-end telemetry from JavaScript to ASP.NET Core MVC Controllers on the server side via the Azure portal without needing to change code, add configuration, or even re-deploy your app.
2. **Visual Studio 2017 enables production debugging without impacting production environment quality and stability.**
    - Visual Studio 2017 has updated debugging features, like Snapshot Debugging for enabling code debugging of snapshots taken of the production environment automatically by Application Insights' rich monitoring features. Developers can also set snap points from within Visual Studio 2017, so that as exceptions occur in production they're automatically downloaded and debuggable, all without causing any down time or performance issues on the production app.

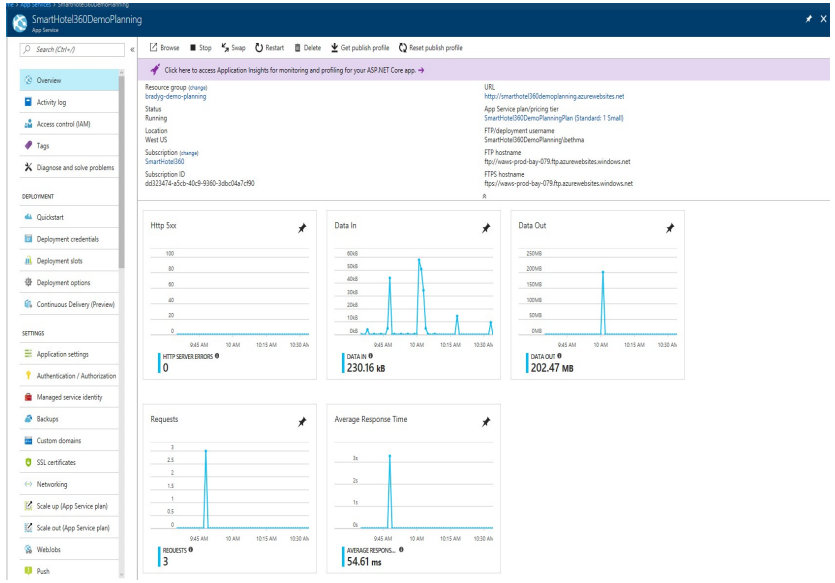
Install Pre-Requisites

- **Visual Studio 2017 15.5 or higher – Azure & .NET Core workloads selected in VS installer**
- **Snapshot debugger, Azure Functions Tools installed (use the Quick launch toolbar in Visual Studio to install)**
- **Clone repo:** https://github.com/Microsoft/SmartHotel360-public-web
- Install Node 8.9.1
- Install Python 2

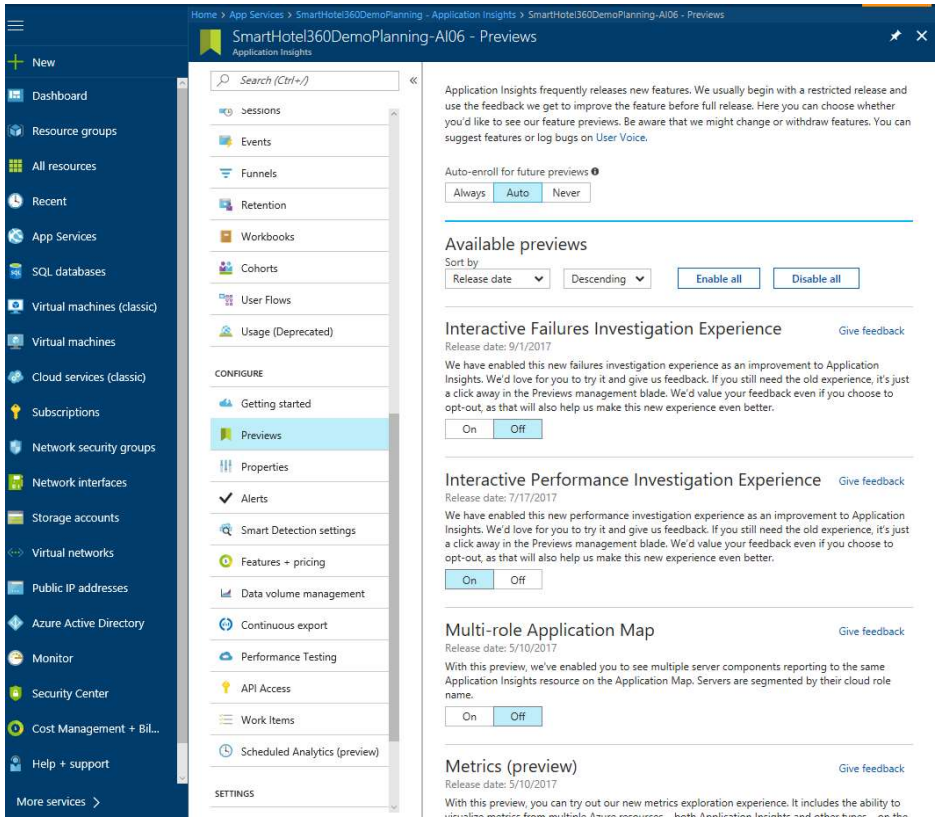
From an admin prompt, navigate to SmartHotel360.PublicWeb:

- `npm install -g windows-build-tools`
- `Npm install`
- `Npm rebuild node-sass –force`
- `Npm run dev`

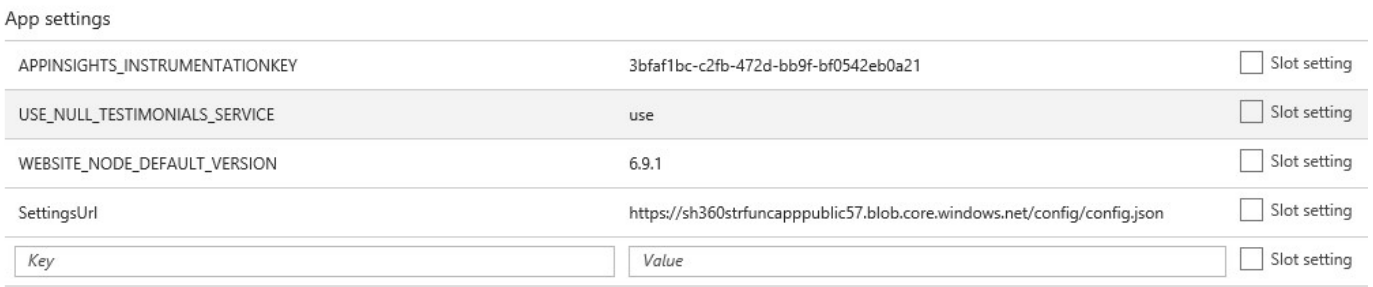
Demo Pre-Setup

- Follow the instructions for setting up the Azure services related to this demo: https://github.com/Microsoft/SmartHotel360-public-web/blob/master/doc/demo-setup.md
- Open browser tabs to:
    - The PublicWeb app deployed to your Azure app service (i.e. mysmarthotel360app.azurewebsites.net)
    - Open the portal to the App Service overview:
        - 

    - Open Portal to App Settings:
        - Scroll to the App Settings values
- Set the Application Insights → Previews -> Interactive Performance Investigation Experience to ON.



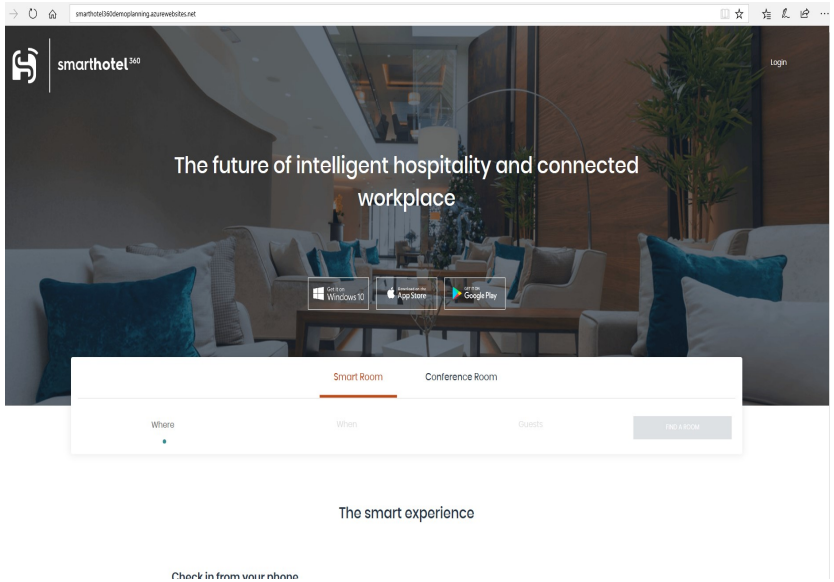
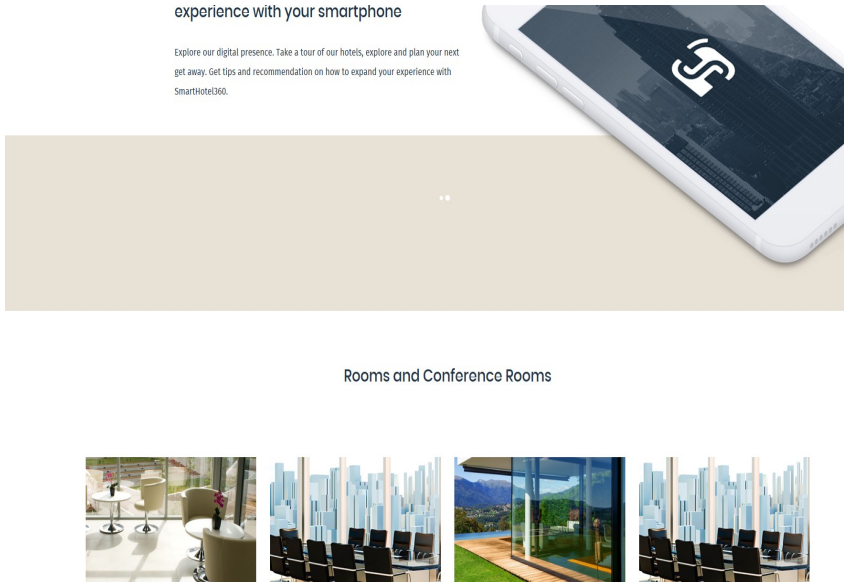
- Make sure in the App Settings, the USE_NULL_TESTIMONIAL_SERVICE is present.

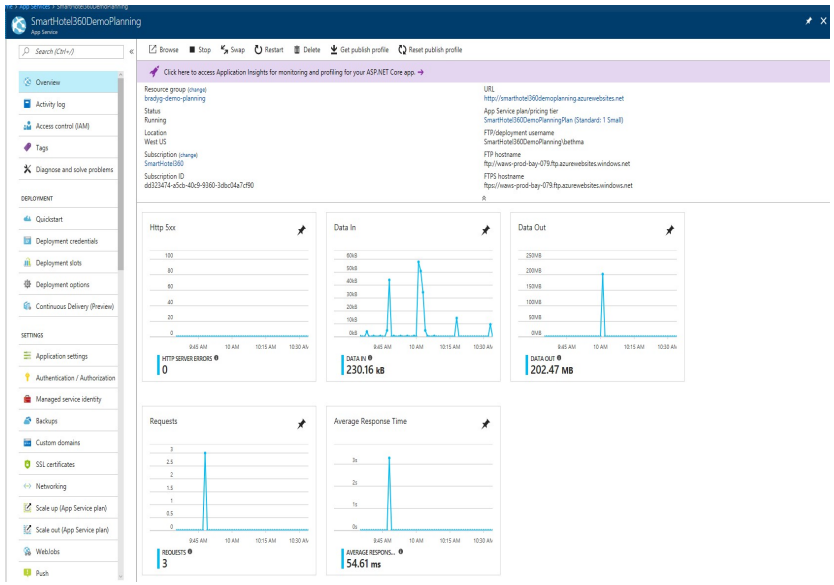


- Hit the site a bunch of times. Like, a lot of times. Generates app insights data and causes exceptions.

- Open an instance of Visual Studio 2017 15.5 (or higher) to SmartHotel360.PublicWeb.sln. This contains the PublicWeb project.

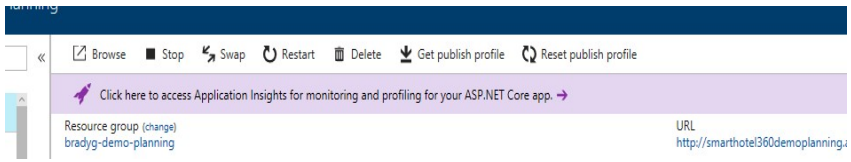
Overall Demo walk-through

Screenshots	Instructions	Narrative
	Open your public deployed website. Quickly walk through features Search for a city: New York Click “Find a room” Show a room detail	Let’s take a look at the SmartHotel360 public website. This website was **built in C#** and runs on **.NET Core** which is our **open source, cross-platform** runtime that gives us great performance and is **perfect for cloud-native apps.** As you can see this is where people get their first impression of our Smart Hotel and make reservations.
	Scroll down to the twitter quote area of the page. You will see a “loading” glyph.	But something looks odd here. It seems that this part of the site is having issues loading. I wonder what’s going on?



Open up the App Service in the Azure Portal

This website is deployed to **Azure App Service** which **is perfect for websites** like this. It provides a fully managed platform where we can **scale and monitor our apps effortlessly**.



Hover over the purple Application Insights bar.

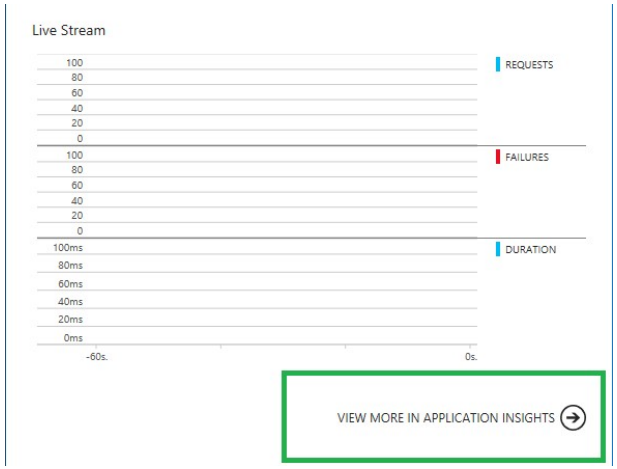
And since we're using .NET Core, we **automatically get application performance monitoring** with Application Insights right out of the box **without writing any code.**

This makes it extremely easy to pinpoint anomalies and debug production exceptions in my .NET Core apps on Azure.

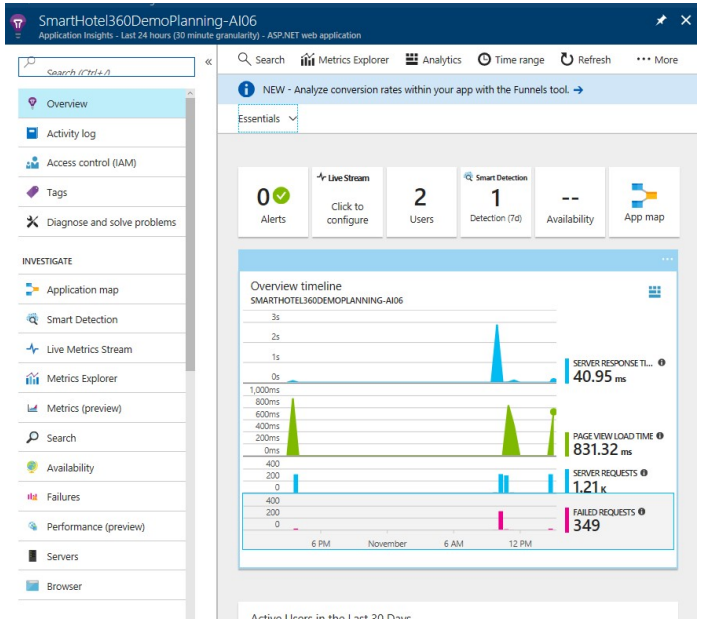
Click it

[NOTE: If the purple bar is not there, search (or scroll down) to the Application Insights in the left Nav.]

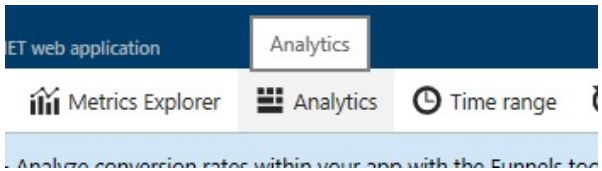
Let's see what is causing the problem on our production site.



Click the VIEW MORE IN APPLICATION INSIGHTS button.

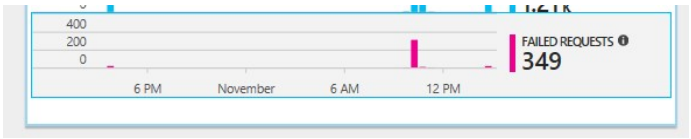


Here is where we can see an overview of the application's health like response times, page loads and server requests. This is **extremely powerful application performance management you get for free, only with Azure and .NET.** And it's also viewable directly from Visual Studio.



Hover over Analytics button

And if this isn't what you're looking for, you can also write custom queries yourself.

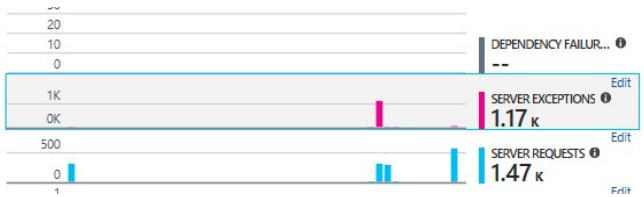


Click on Failed Requests

Let's investigate the failed requests.

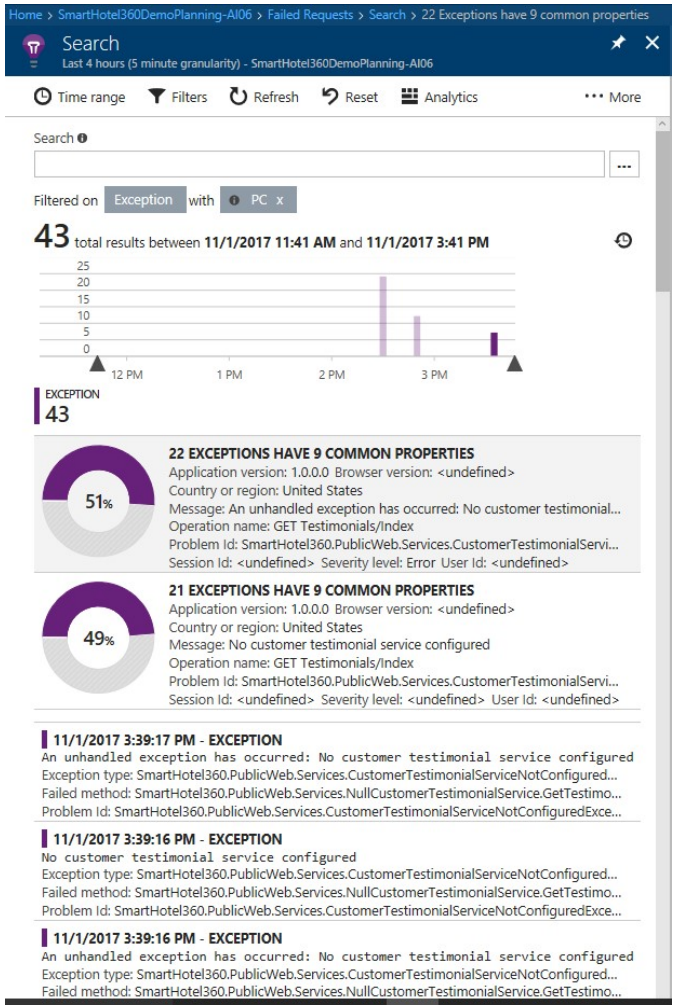
We get a lot of detailed data here on the different types of failures.

I'm a developer so I'm not happy seeing these server exceptions.



Click on Server Exceptions

Let's see what's going on.

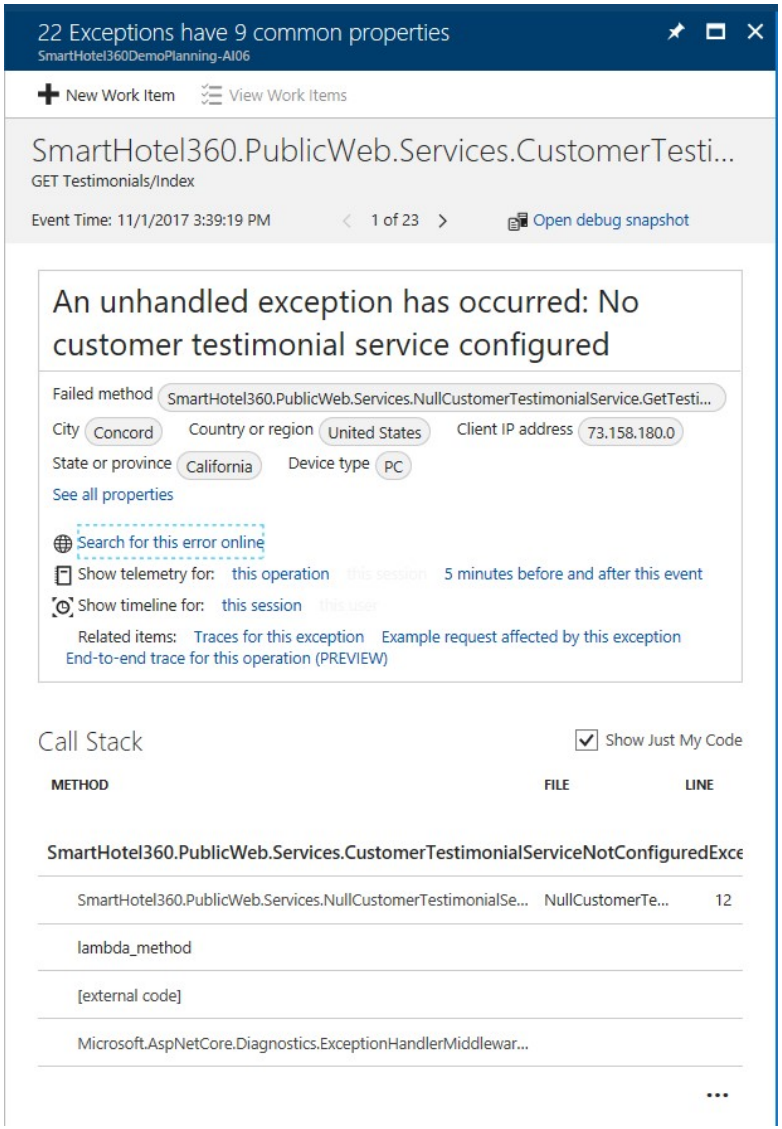


Notice here that **Application Insights is helping me make sense of all these exceptions by summarizing the huge amounts of data** and showing me common properties.

It looks like I forgot to configure the customer testimonials service and it's throwing an exception.

Click on the **Unhandled Exception**.... summary

Let's drill into this.



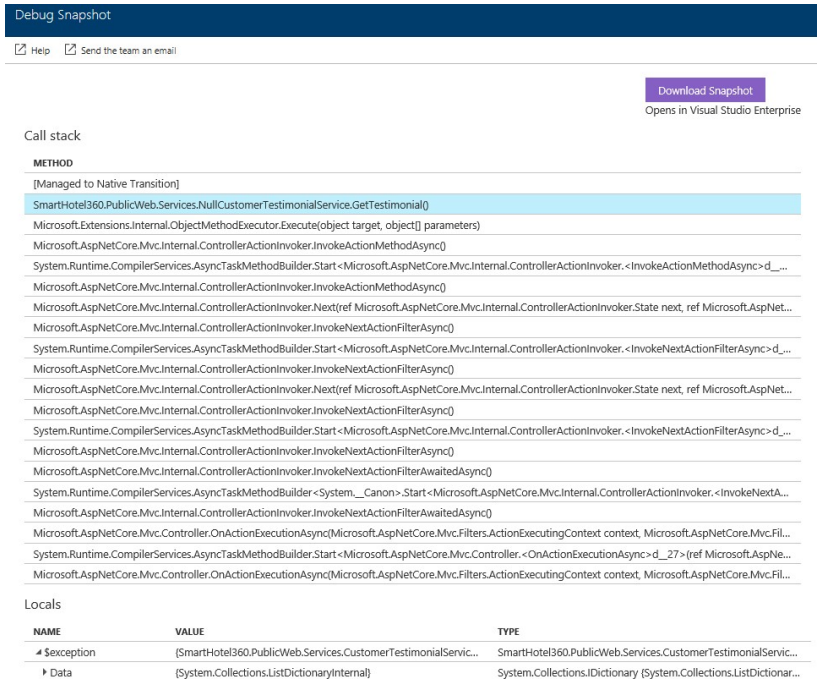
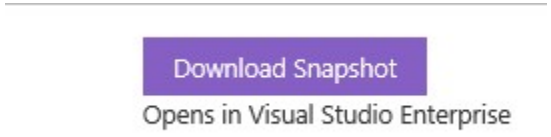
Notice all the rich diagnostics I can view here. I can drill into properties and rich telemetry of the operation or events leading up to and after this one.

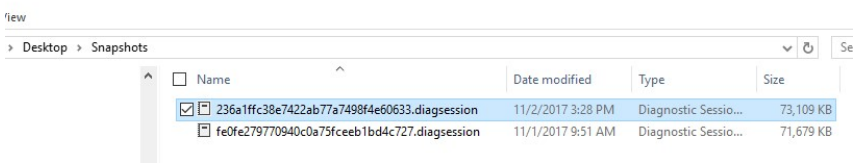
Let's open the debug snapshot.

Click Open debug snapshot

[takes a couple seconds to load. If you don't see this button, select the End-to-End trace under related items and you

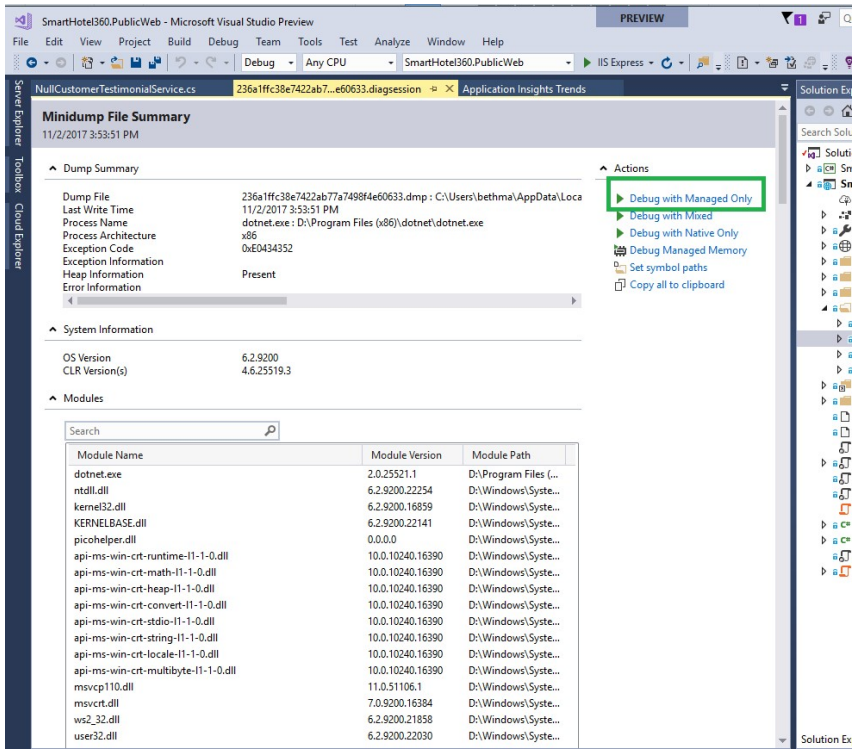
Snapshots are taken of the production environment automatically when exceptions

	can open the snapshot from the next screen]	**happen by Application Insights' rich monitoring features.** This will show the entire call stack. And as we click through it, we can investigate the locals as well.
		
	Hover over "Download snapshot" [it takes 40 seconds to download sometimes, you may want to save on locally]	That's all great, but what's even better is I can download the entire snapshot minidump and debug this directly in Visual Studio.



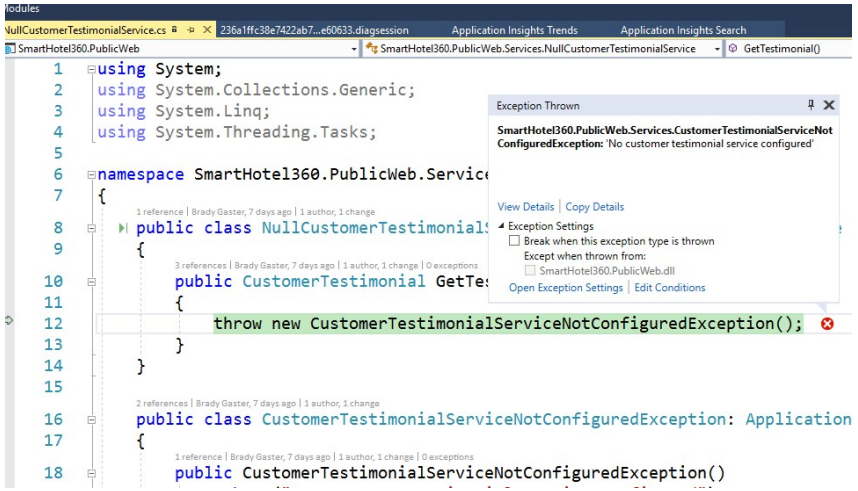
Open Windows Explorer, double-click on the .diagsession file.

It takes about 30 seconds to download these so let's take a look at one I've already got here.



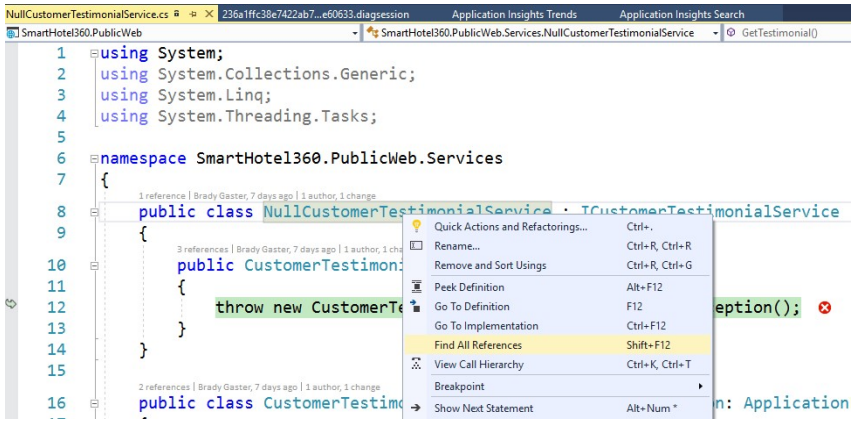
Click "Debug with Managed Only"

Now we can fire up the debugger to step into the exact line of code that's causing us trouble here in production.



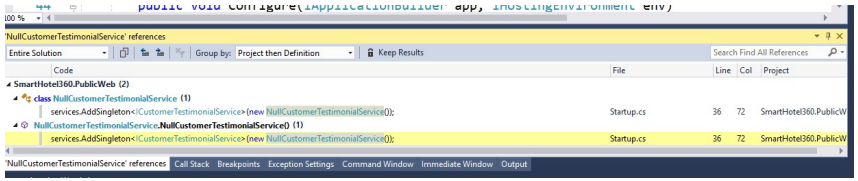
Close the exception helper.

Here's the exception being thrown.



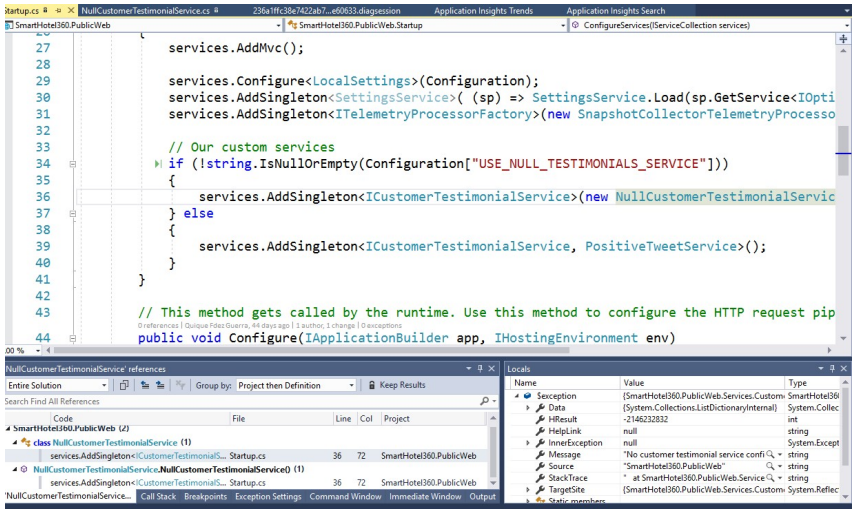
Right-click line 8 to "Find all references"

Let's see where we are calling this method.



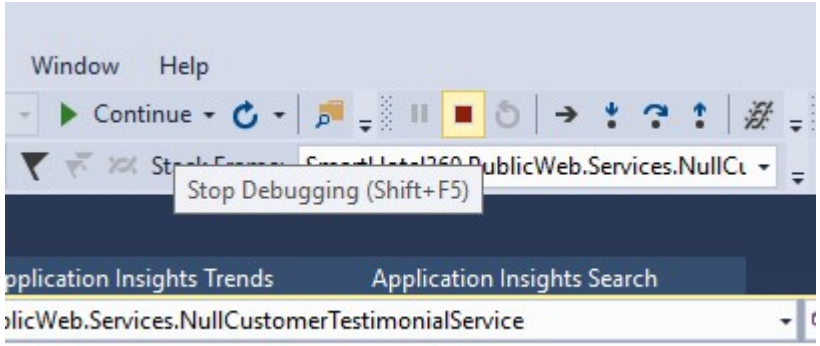
Click one of the references to open the startup.cs

Looks like it's just called from our Startup.



And now I can easily tell that I forgot to set a configuration setting on the production site.

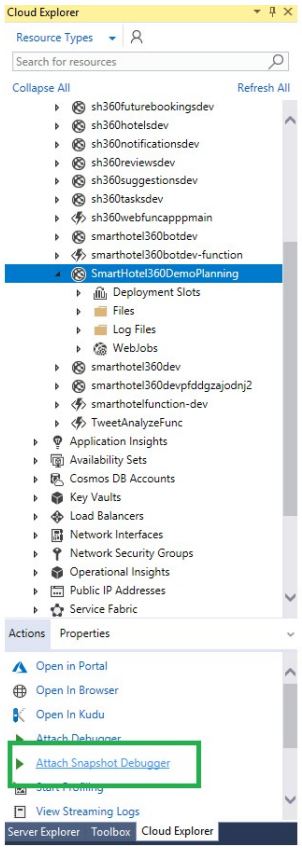
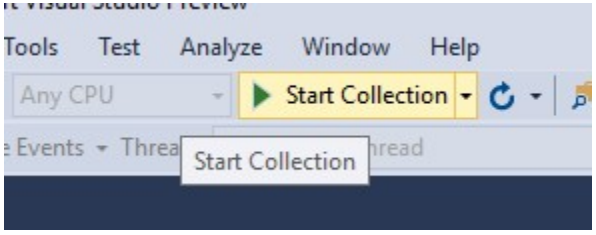
Let's go back and fix that!

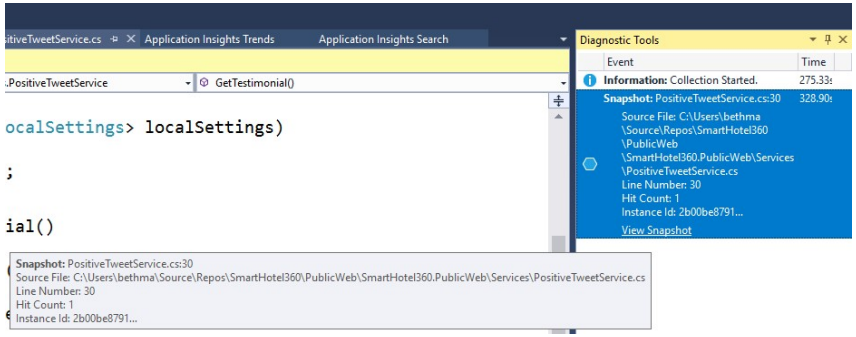
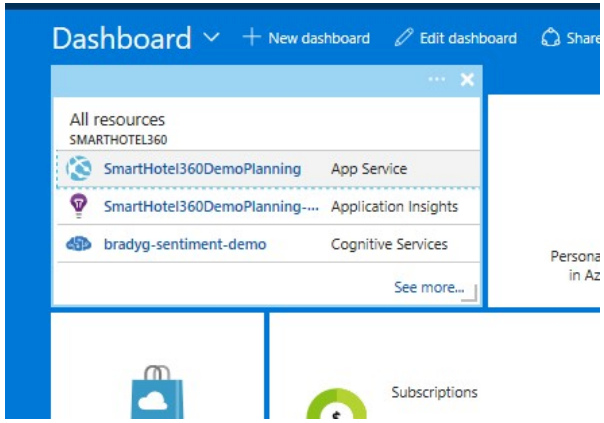


STOP THE DEBUGGER

It's really nice that **App Insights is always monitoring and collecting snap shots for exceptions automatically**, however what if we had a more subtle problem that we needed to debug, something that wasn't causing an exception but was producing the wrong behavior perhaps?

We can also attach the snapshot debugger to our running app service on demand and use snap points (like break points without the breaking) to see values realtime.

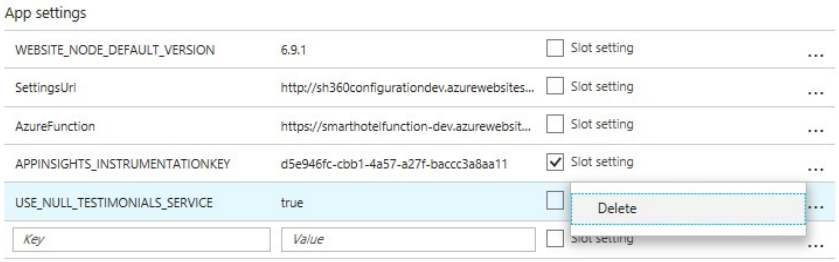
	Open the Cloud Explorer to App Service Click Attach Snapshot Debugger	Let's debug this in production using the Cloud Explorer. This will create a Shapshot for me that we can debug. **Snapshots have close to zero impact on the production performance and support debugging highly distributed apps on multiple servers.**
	Set a Snappoint (same gesture as a breakpoint) on line 27 of Startup.cs	Let's set a Snappoint.
	Click Start Collection	**Snappoints bring breakpoint-style local debugging to a cloud scale.** Unlike breakpoints, they do not halt your app: instead, they take a snapshot of the info we need to debug this, and continue execution.

	Refresh the Website, hit it a few times to cause the error.	
	Click “View Snapshot” in the Diagnostics Tools window	So now we have an up to the second snapshot that we can use to debug the value of this model real-time.
	See the snappoint get hit	As you can see we can step through the code and see the production values of variables, call stack, etc. Let’s fix this settings problem!
	Head back to the portal, click the Home breadcrumb. Click the App Service Select Application Settings (via search or scroll on the left Nav bar)	Let’s go into our App Service to see what our settings are.

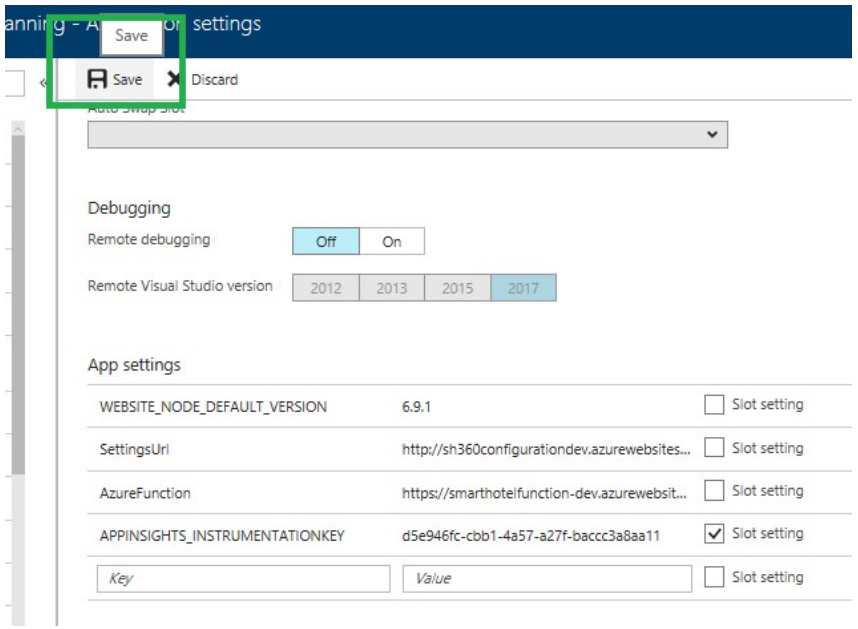
Scroll down to the App Settings

Highlight USE_NULL_TESTIMONIALS_SERVICE

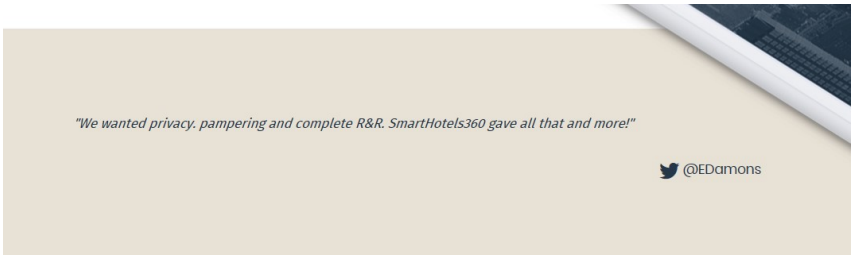
Yep, it looks like we have the wrong setting. Let's remove this one because we want to use the real service.



Click the ellipsis and delete the setting



Click Save in upper left



Tab to Public website, hit refresh. *It should pause a second and you'll know it's picked up the new config.*

Scroll down to testimonials, show the tweet working.

And now it's working!

Snapshot debugging is a production debugging feature that comes with Application Insights automatically for your .NET apps deployed to App Service in Azure. And with Visual Studio we can set snappoints to immediately debug our apps in production without affecting the performance. No other cloud gives you the power and productivity like this!

Thank you!

-