ADVANCED TELEMETRY WITH AZURE MONITOR APPLICATION INSIGHTS

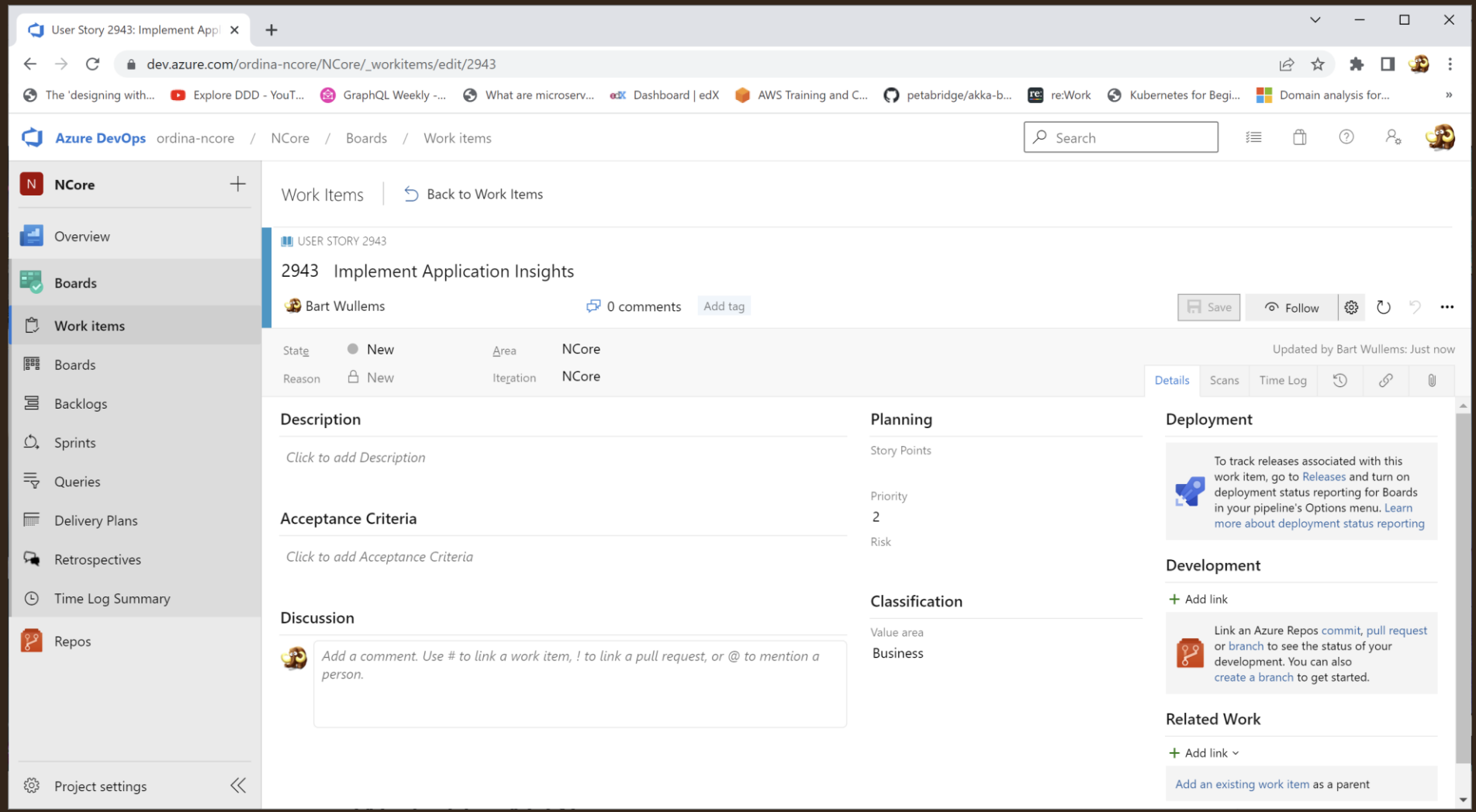
Bart Wullems

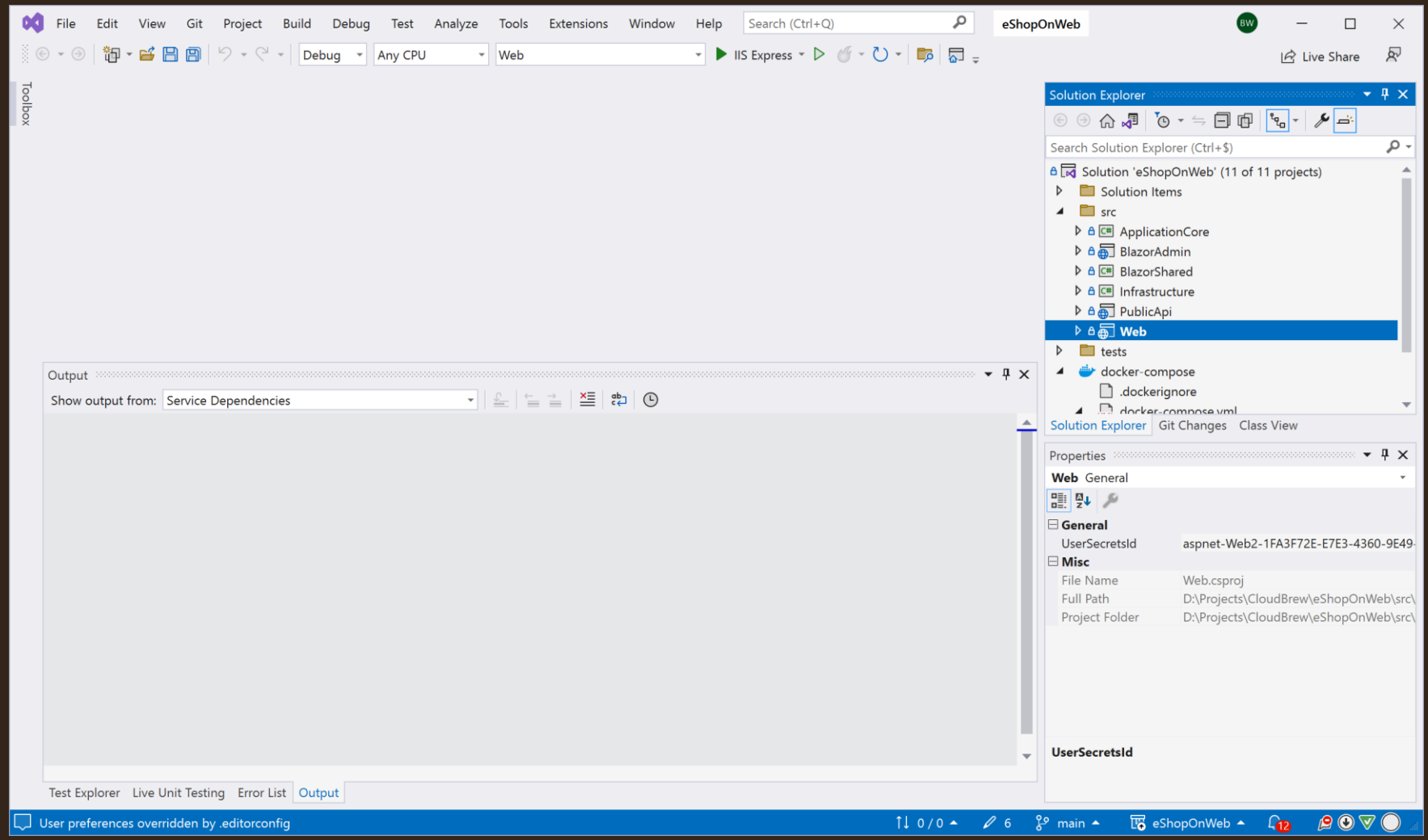
bart.wullems@ordina.be

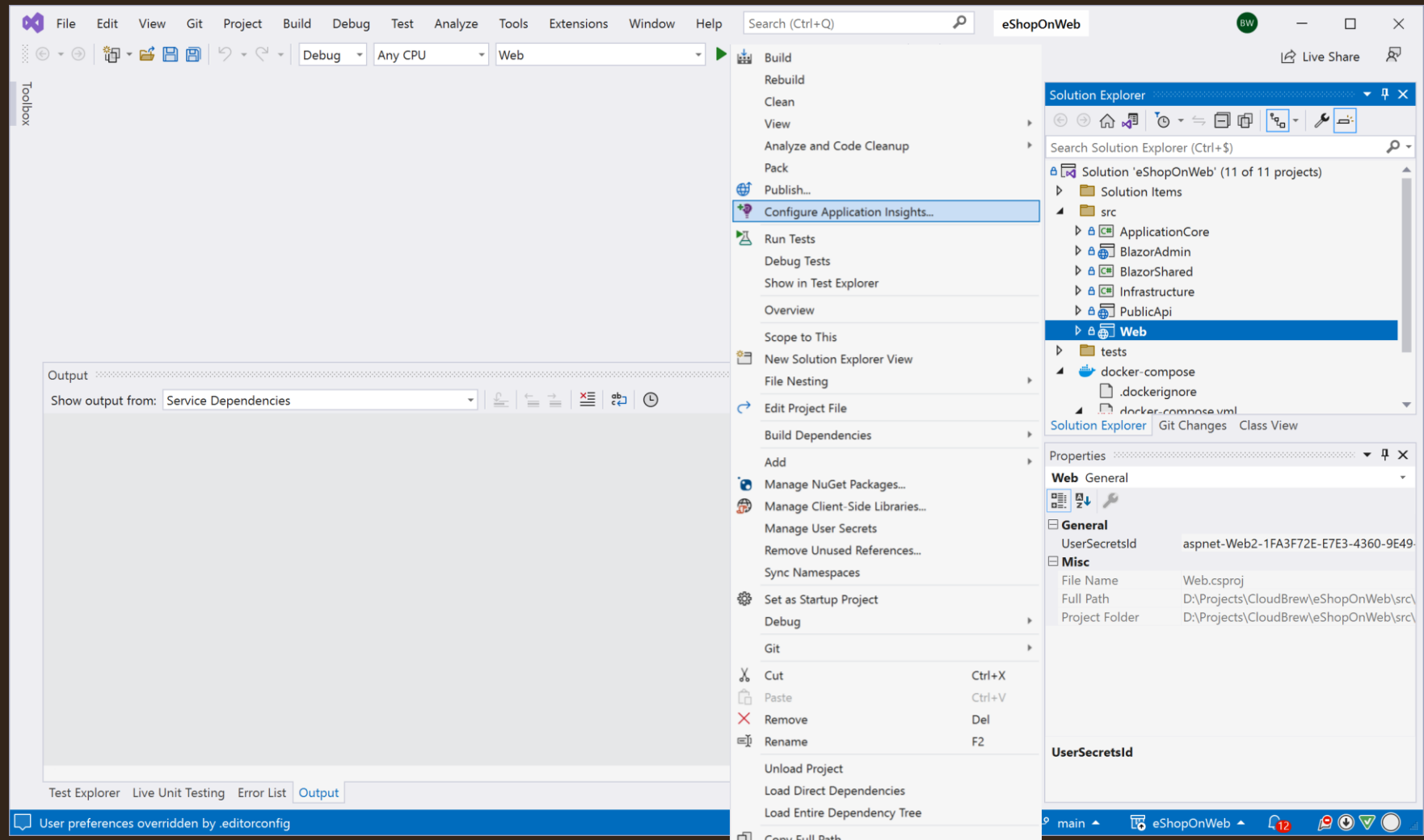
https://bartwullems.blogspot.com

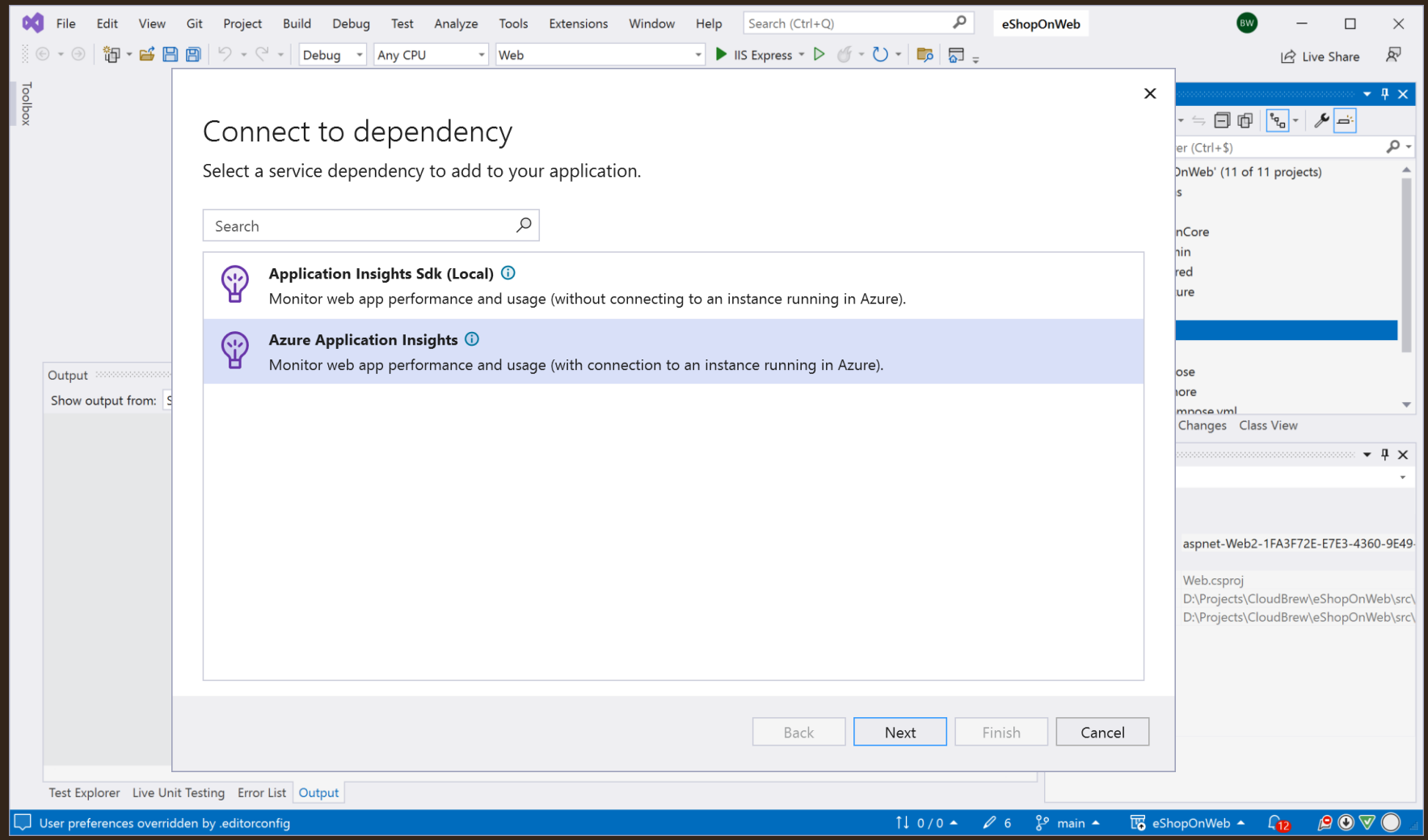

EXIT
THINKSTOCK

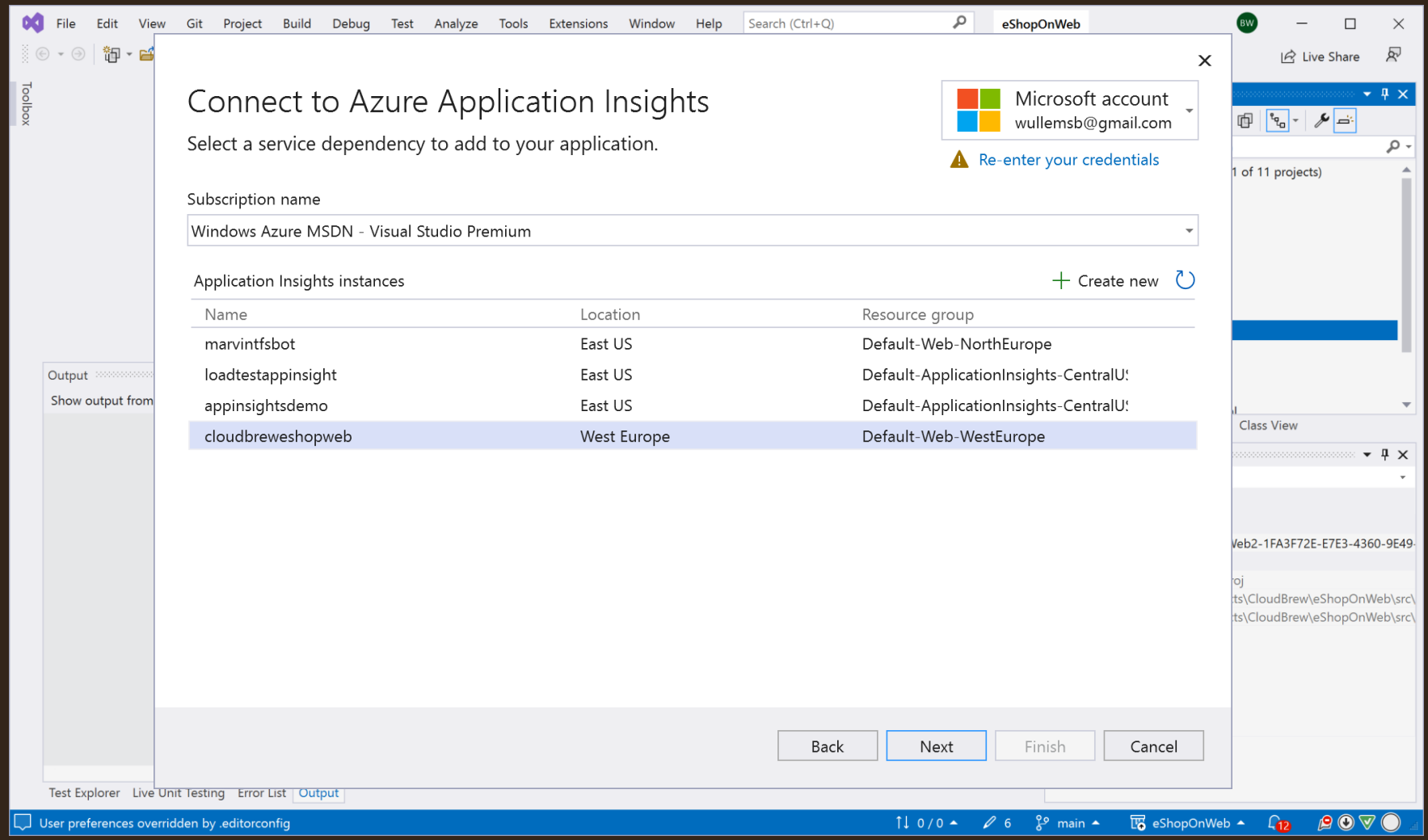
NEED MORE COFFEE

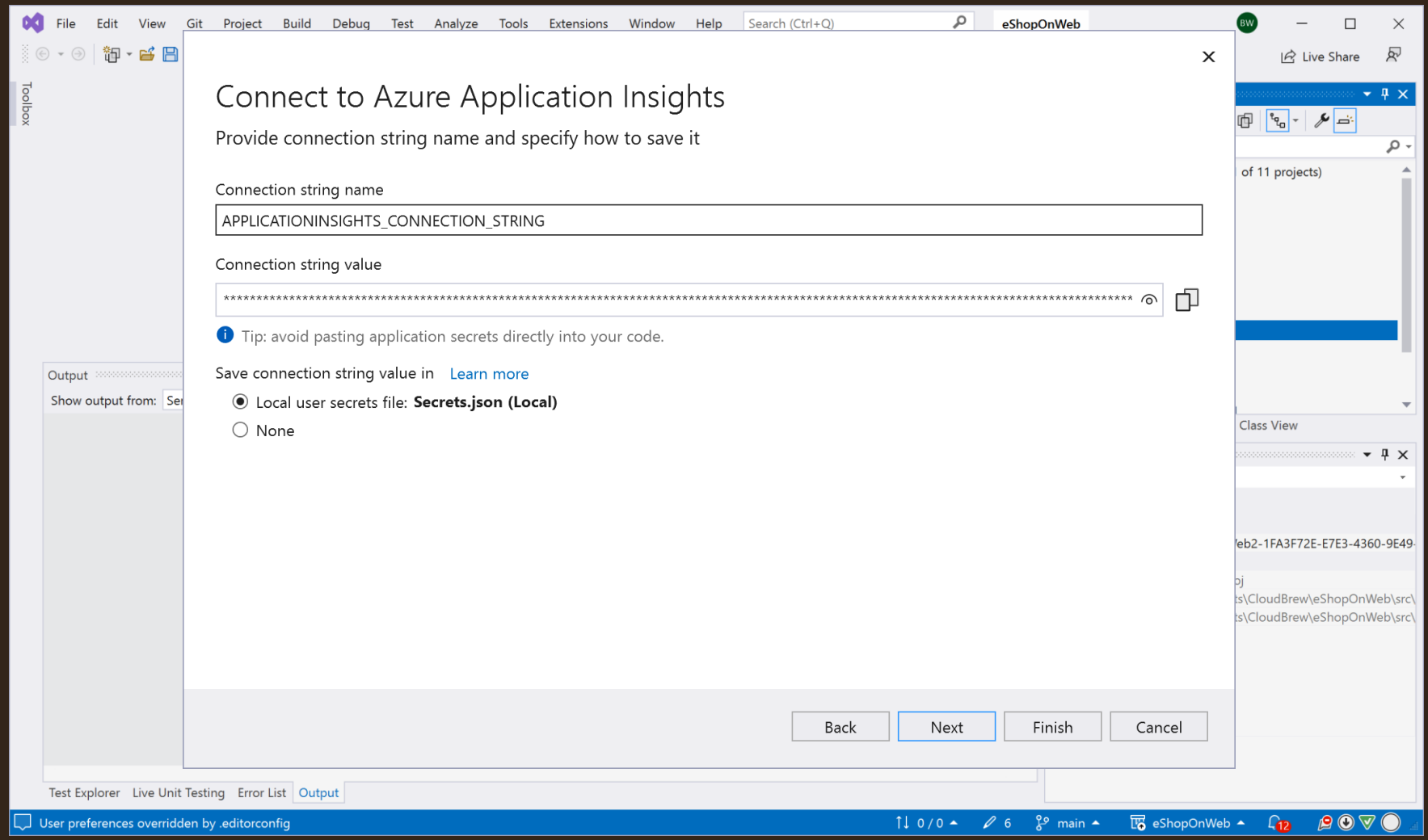

User Story 2943: Implement Appl
dev.azure.com/ordina-ncore/NCore/_workitems/edit/2943
Azure DevOps ordina-ncore / NCore / Boards / Work items
Search
NCore
Overview
Boards
Work items
Boards
Backlogs
Sprints
Queries
Delivery Plans
Retrospectives
Time Log Summary
Repos
Project settings
Work Items
Back to Work Items
USER STORY 2943
2943 Implement Application Insights
Bart Wullems
0 comments
Add tag
Save
Follow
State New
Reason New
Area NCore
Iteration NCore
Updated by Bart Wullems: Just now
Details
Scans
Time Log
Description
Click to add Description
Acceptance Criteria
Click to add Acceptance Criteria
Discussion
Add a comment. Use # to link a work item, ! to link a pull request, or @ to mention a person.
Planning
Story Points
Priority
2
Risk
Classification
Value area
Business
Deployment
To track releases associated with this work item, go to Releases and turn on deployment status reporting for Boards in your pipeline's Options menu. Learn more about deployment status reporting
Development
Add link
Link an Azure Repos commit, pull request or branch to see the status of your development. You can also create a branch to get started.
Related Work
Add link
Add an existing work item as a parent


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
eShopOnWeb
Debug
Any CPU
Web
IIS Express
Live Share
Toolbox
Solution Explorer
Search Solution Explorer (Ctrl+$)
Solution 'eShopOnWeb' (11 of 11 projects)
Solution Items
src
ApplicationCore
BlazorAdmin
BlazorShared
Infrastructure
PublicApi
Web
tests
docker-compose
.dockerignore
Solution Explorer
Git Changes
Class View
Output
Show output from: Service Dependencies
Properties
Web General
General
UserSecretsId
aspnet-Web2-1FA3F72E-E7E3-4360-9E49-
Misc
File Name
Web.csproj
Full Path
D:\Projects\CloudBrew\eShopOnWeb\src\
Project Folder
D:\Projects\CloudBrew\eShopOnWeb\src\
UserSecretsId
Test Explorer
Live Unit Testing
Error List
Output
User preferences overridden by .editorconfig
0 / 0
6
main
eShopOnWeb
12


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
eShopOnWeb
Debug
Any CPU
Web
Live Share
Toolbox
Build
Rebuild
Clean
View
Analyze and Code Cleanup
Pack
Publish...
Configure Application Insights...
Run Tests
Debug Tests
Show in Test Explorer
Overview
Scope to This
New Solution Explorer View
File Nesting
Edit Project File
Build Dependencies
Add
Manage NuGet Packages...
Manage Client-Side Libraries...
Manage User Secrets
Remove Unused References...
Sync Namespaces
Set as Startup Project
Debug
Git
Cut Ctrl+X
Paste Ctrl+V
Remove Del
Rename F2
Unload Project
Load Direct Dependencies
Load Entire Dependency Tree
Copy Full Path
Open Folder in File Explorer
Open in Terminal
Refactor
Go to File Nearby
Properties Alt+Enter
Solution Explorer
Search Solution Explorer (Ctrl+$)
Solution 'eShopOnWeb' (11 of 11 projects)
Solution Items
src
ApplicationCore
BlazorAdmin
BlazorShared
Infrastructure
PublicApi
Web
tests
docker-compose
.dockerignore
Solution Explorer Git Changes Class View
Properties
Web General
General
UserSecretsId aspnet-Web2-1FA3F72E-E7E3-4360-9E49-
Misc
File Name Web.csproj
Full Path D:\Projects\CloudBrew\eShopOnWeb\src\
Project Folder D:\Projects\CloudBrew\eShopOnWeb\src\
UserSecretsId
Output
Show output from: Service Dependencies
Test Explorer Live Unit Testing Error List Output
User preferences overridden by .editorconfig
main eShopOnWeb
Type here to search
10°C Bewolkt
8:35
11/11/2022


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
eShopOnWeb
Debug
Any CPU
Web
IIS Express
Live Share
Toolbox
Connect to dependency
Select a service dependency to add to your application.
Search
Application Insights Sdk (Local)
Monitor web app performance and usage (without connecting to an instance running in Azure).
Azure Application Insights
Monitor web app performance and usage (with connection to an instance running in Azure).
Back
Next
Finish
Cancel
Output
Show output from:
Changes
Class View
Test Explorer
Live Unit Testing
Error List
Output
User preferences overridden by .editorconfig
main
eShopOnWeb


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
eShopOnWeb
Live Share
Toolbox
Connect to Azure Application Insights
Select a service dependency to add to your application.
Microsoft account
wullemsb@gmail.com
Re-enter your credentials
Subscription name
Windows Azure MSDN - Visual Studio Premium
Application Insights instances
Create new
Name
Location
Resource group
marvintfsbot
East US
Default-Web-NorthEurope
loadtestappinsight
East US
Default-ApplicationInsights-CentralU
appinsightsdemo
East US
Default-ApplicationInsights-CentralU
cloudbreweshopweb
West Europe
Default-Web-WestEurope
Back
Next
Finish
Cancel
Output
Show output from
Class View
Test Explorer
Live Unit Testing
Error List
Output
User preferences overridden by .editorconfig
main
eShopOnWeb


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
eShopOnWeb
Live Share
Toolbox
Connect to Azure Application Insights
Provide connection string name and specify how to save it
Connection string name
APPLICATIONINSIGHTS_CONNECTION_STRING
Connection string value
Tip: avoid pasting application secrets directly into your code.
Save connection string value in Learn more
Local user secrets file: Secrets.json (Local)
None
Back
Next
Finish
Cancel
of 11 projects)
Class View
Output
Show output from:
Test Explorer
Live Unit Testing
Error List
Output
User preferences overridden by .editorconfig
main
eShopOnWeb

Microsoft® Most Valuable Professional

ADVANCED TELEMETRY WITH AZURE MONITOR APPLICATION INSIGHTS

Bart Wullems

bart.wullems@ordina.be

https://bartwullems.blogspot.com

AGENDA

Why application monitoring?

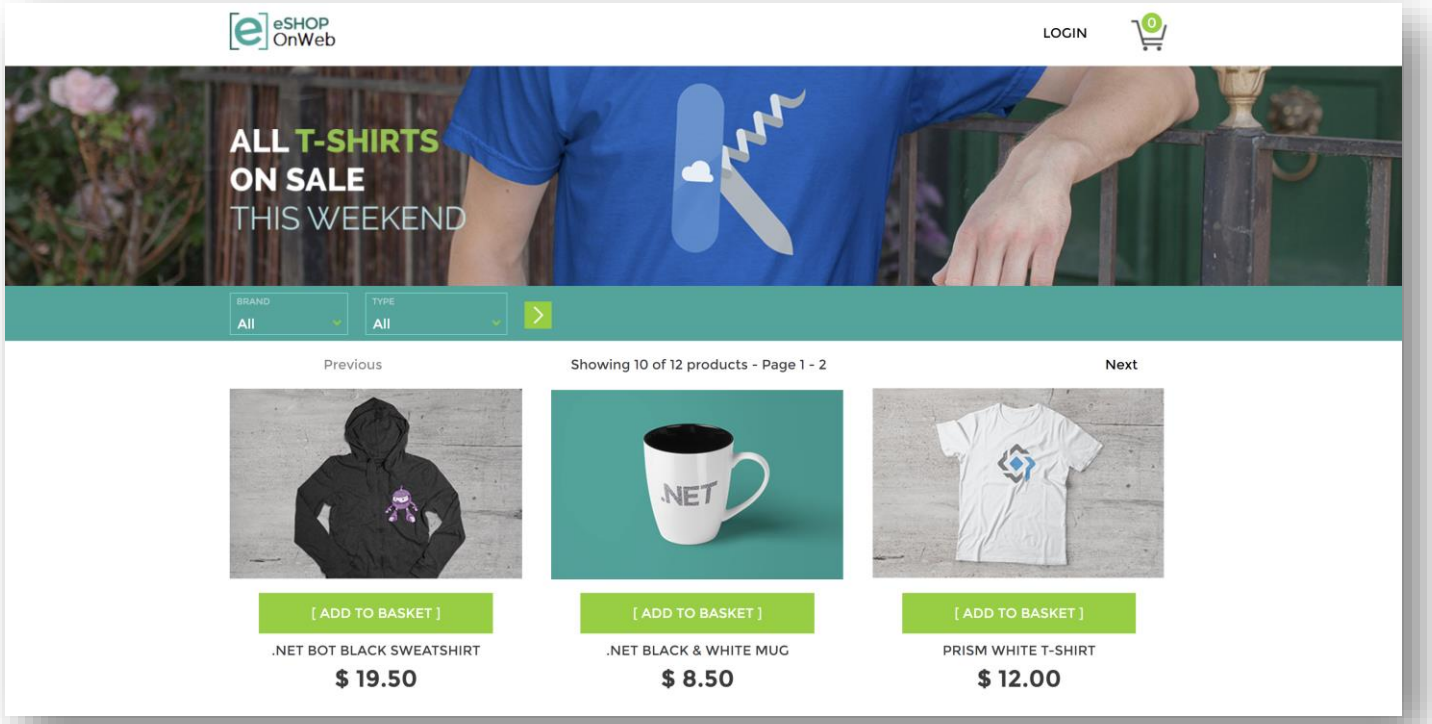
Application Insights

What is it?

Feature overview

Hands-on demos

ORDINA
Ahead of change



https://cloudbreweshopweb.azurewebsites.net

ORDINA
Ahead of change

WHY APPLICATION MONITORING?

ORDINA
Ahead of change

WHY APPLICATION MONITORING?

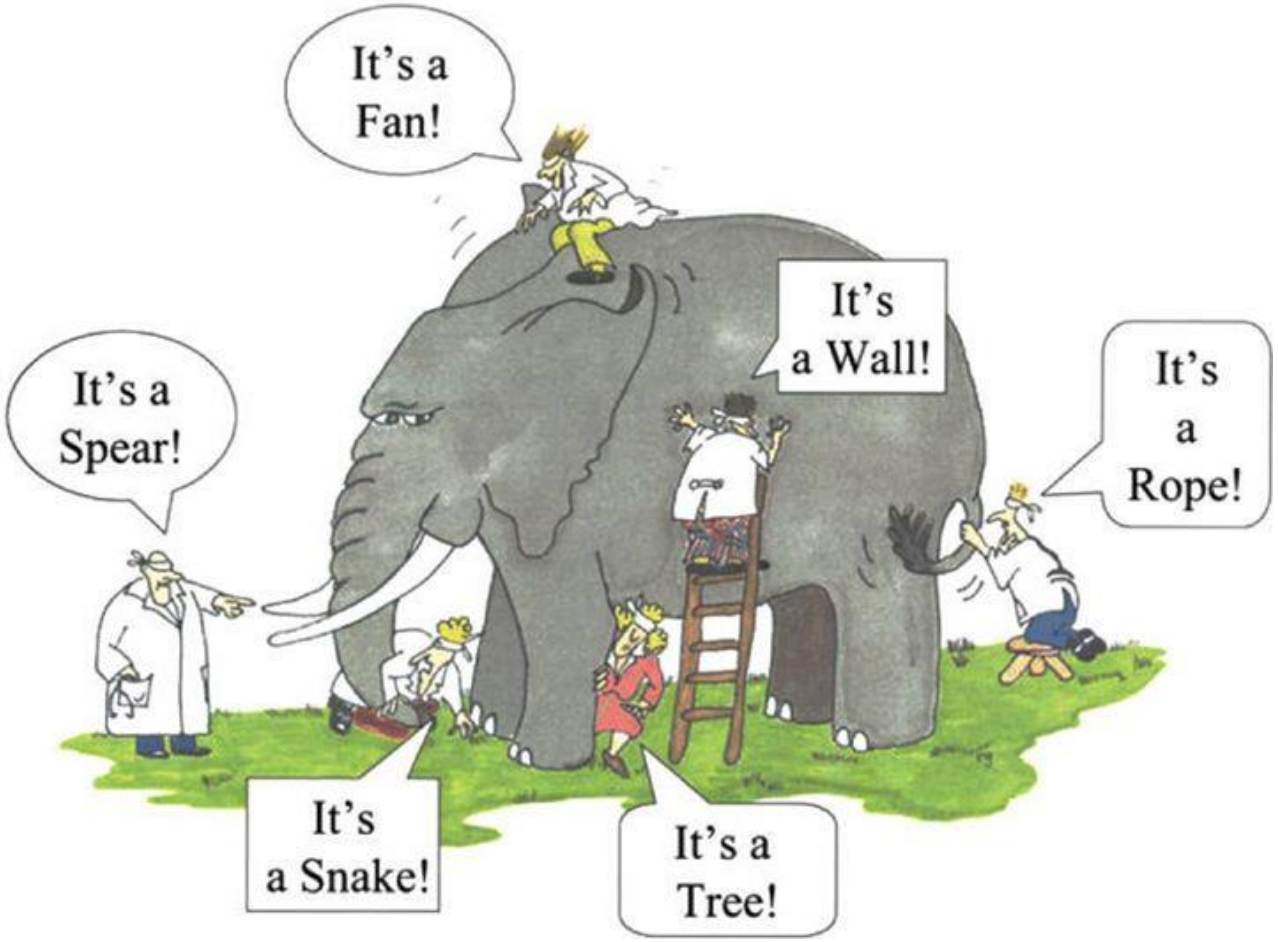
Find out:

- What’s going on inside your application service
- how your application is performing
- how your application is used
- what errors users face
- events that need your reaction

ORDINA
Ahead of change

IS IT BETTER THAN LOGGING?

- Building good logging is a challenging task
- Most of logging frameworks don’t provide analyzable logs
- Log correlation may be difficult
- Logs are not a perfect fit for all telemetry(statistics,…)



ORDINA
Ahead of change

WHY TELEMETRY DATA?

Useful for:

- Developers: find bugs and performance bottle necks
- Testers: discover areas that need more testing
- UX designers: find out how users are using applications
- Administrators: get ready for load growth
- Support: find problematic areas and be ready to support users
- Business: see how sales and business operations are going

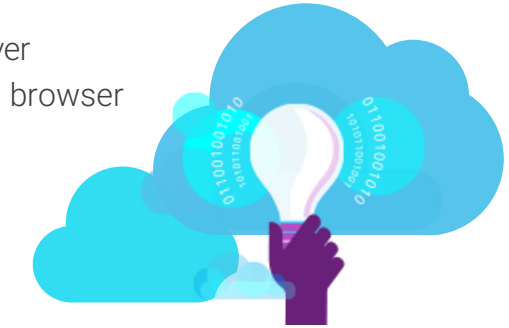
APPLICATION INSIGHTS

ORDINA
Ahead of change

WHAT IS APPLICATION INSIGHTS?

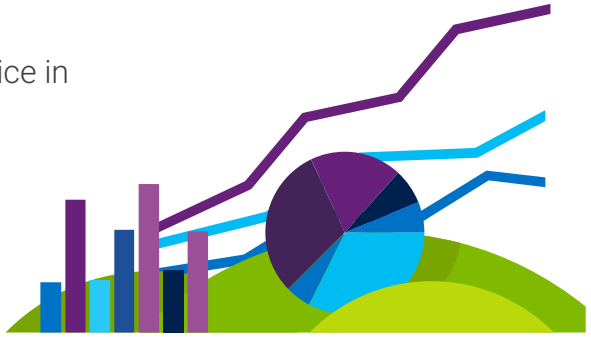
1

Telemetry is collected at each tier: mobile applications, server applications and browser



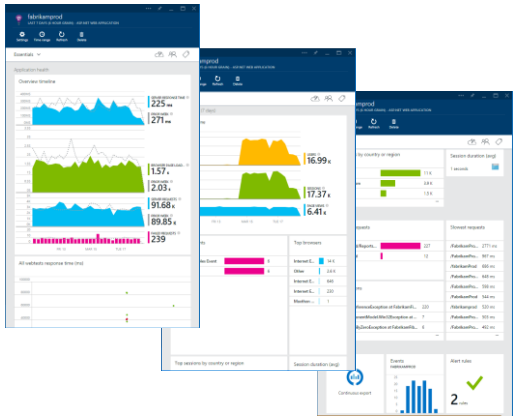
2

Telemetry arrives in the Application Insights service in the cloud where it is processed & stored



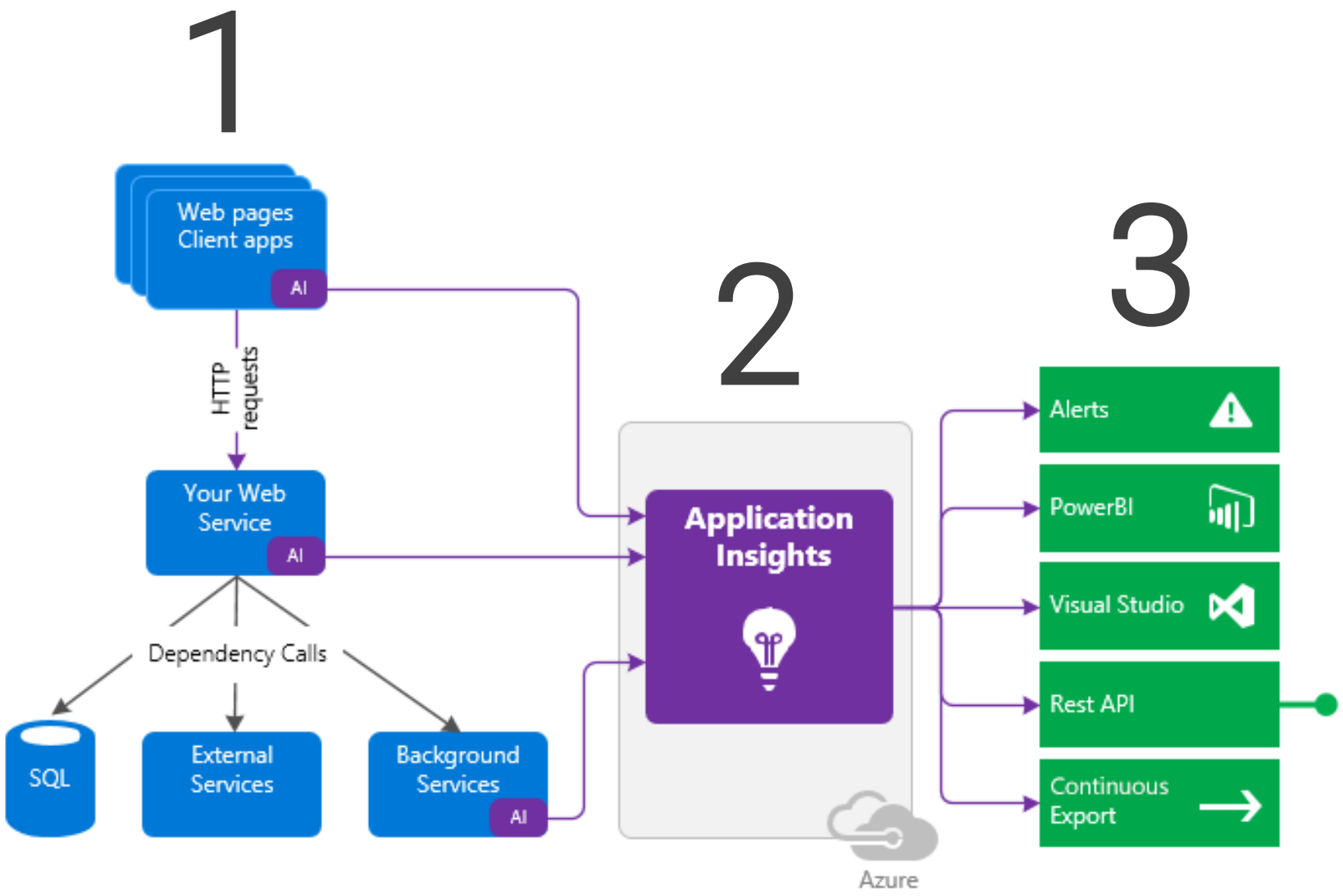
3

Get a 360° view of the application including availability, performance and usage patterns



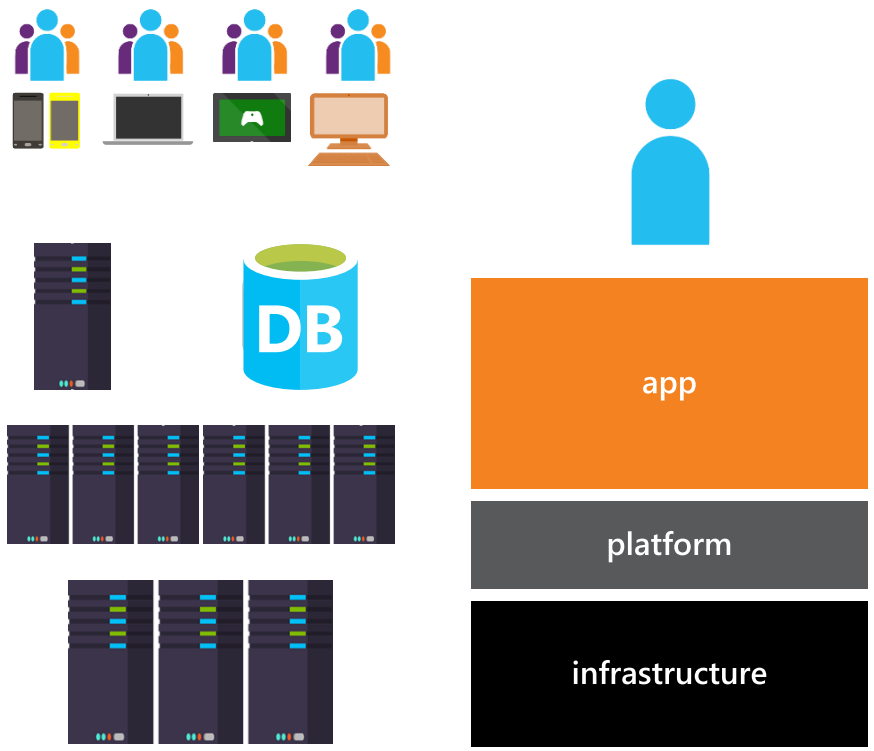
ORDINA
Ahead of change

WHAT IS APPLICATION INSIGHTS?



ORDINA
Ahead of change

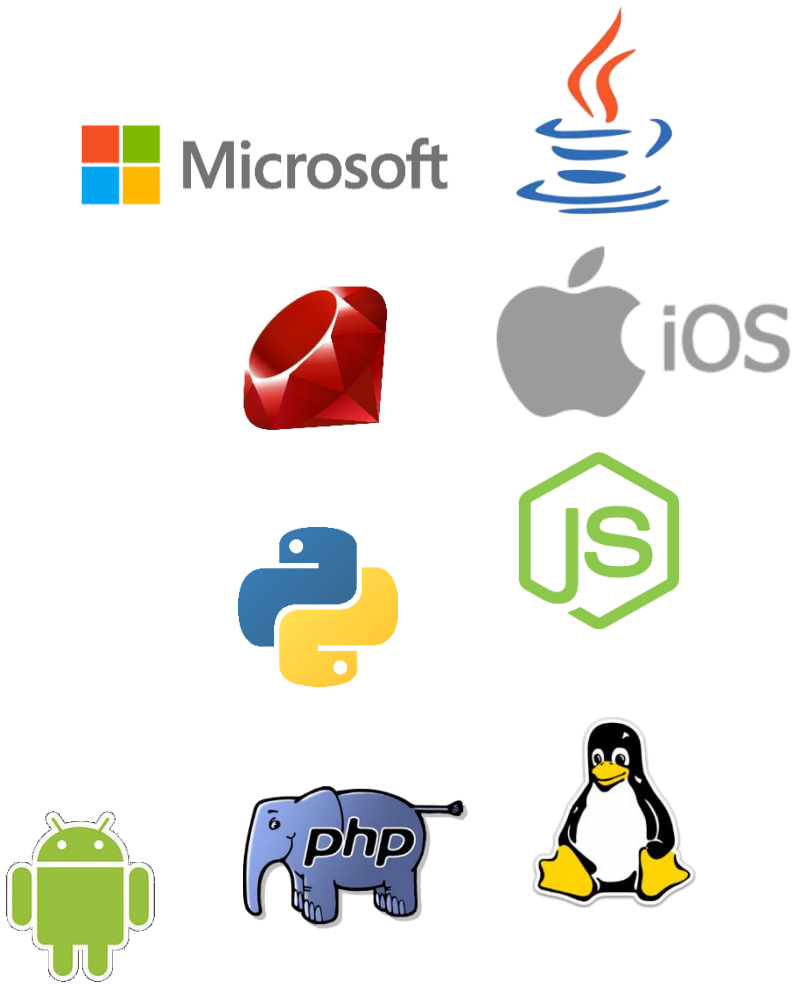
SOURCES OF TELEMETRY



1. **Outside-in monitoring**
   URL pings and web tests from 16 global points of presence
2. **Observed user behavior**
   How is the application being used?
3. **Developer traces and events**
   Whatever the developer would like to send to Application Insights
4. **Observed application behavior**
   No coding required – service dependencies, queries, response time, exceptions, logs, etc.
5. **Infrastructure performance**
   System performance counters

SUPPORTED PLATFORMS

- Any app support(mobile, desktop, SPA, middleware, …)
- Multiple platforms including: ASP.NET, Java/J2EE, iOS, Android, Windows, as well as OSS technologies such as Node.JS, PHP, Ruby, Python, etc.
- Open source SDK: https://github.com/microsoft/ApplicationInsights-dotnet
- Integrates with most logging frameworks: Log4Net, nLog, System.Diagnostics Log4J, Logback, Serilog



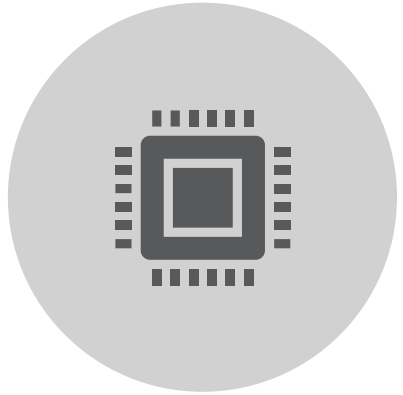
INSTALLATION & CONFIGURATION

ORDINA
Ahead of change

3 OPTIONS

BUILD TIME: ADD THE APPLICATION INSIGHTS SDK TO YOUR APP.

RUN TIME: INSTRUMENT YOUR APP ON THE SERVER, USING AZURE MONITOR APPLICATION INSIGHTS AGENT, WITHOUT REBUILDING AND REDEPLOYING THE CODE.



BOTH: BUILD THE SDK INTO YOUR APP, AND ALSO APPLY THE RUN-TIME EXTENSIONS. GET THE BEST OF BOTH OPTIONS.

ORDINA
Ahead of change

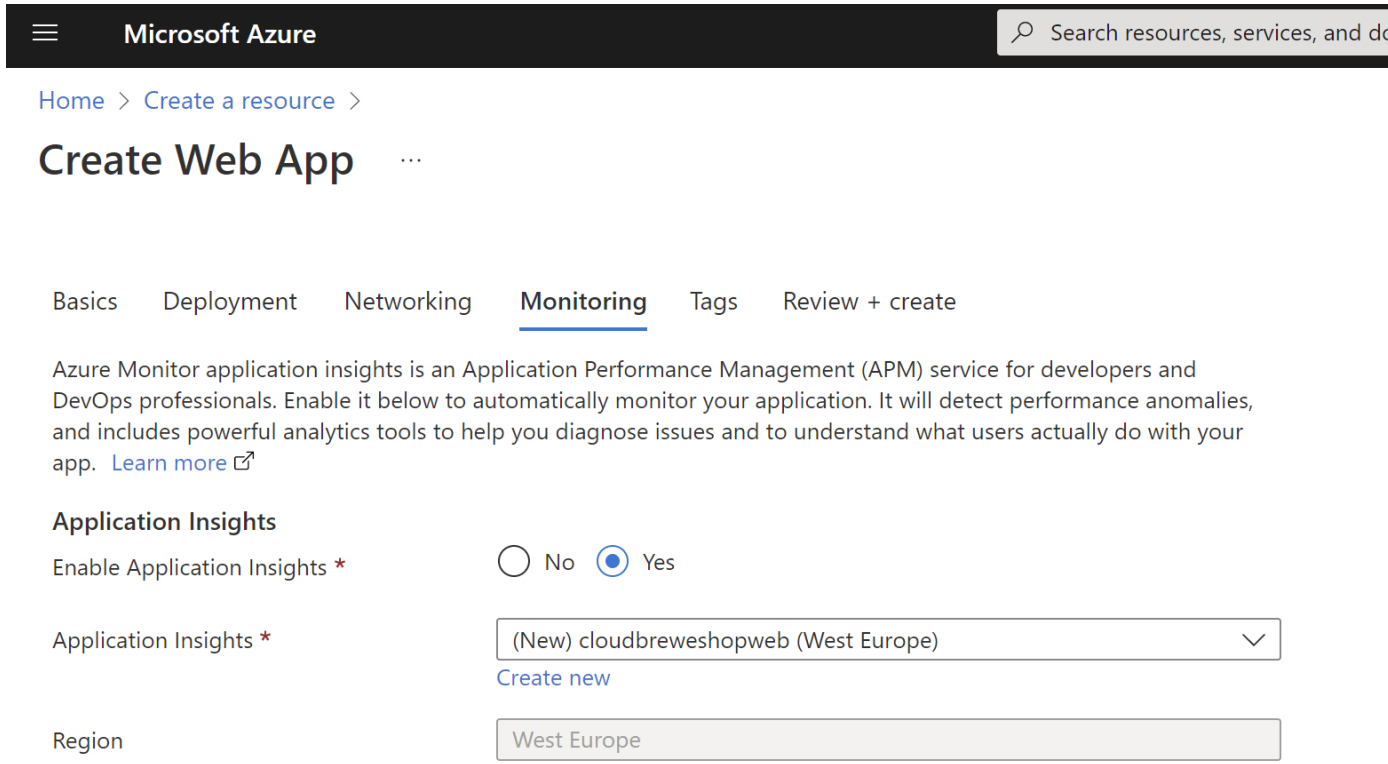
RUN TIME – AUTO-INSTRUMENTATION

Advantages

- No developer investment
- Don’t need to update the SDK
- Also works when you don’t have access to the source code

Disadvantages

- Less control



ORDINA
Ahead of change

BUILD TIME INTEGRATION

ORDINA

ASP.NET CORE

Configure Telemetry

- services.AddApplicationInsightsTelemetry();
- Uses “ApplicationInsights” section in appsettings.json by default

Logging

- By default an ILogger instance is registered
- Default log level= Warning!
- https://bartwullems.blogspot.com/2022/03/application-insights-configure-log-level.html
- When using Serilog an Application Insights sink should be configured
- https://github.com/serilog-contrib/serilog-sinks-applicationinsights

.NET CORE WORKER SERVICE

Configure Telemetry

- services. AddApplicationInsightsTelemetryWorkerService();
- Uses “ApplicationInsights” section in appsettings.json by default

Logging

- By default an ILogger instance is registered
- Default log level= Warning!
- https://bartwullems.blogspot.com/2022/03/application-insights-configure-log-level.html
- When using Serilog an Application Insights sink should be configured
- https://github.com/serilog-contrib/serilog-sinks-applicationinsights

CLIENT SIDE TELEMETRY

ORDINA

CLIENT SIDE TELEMETRY

ASP.NET Core MVC, Razor pages

- Add snippet in the head section

```
@inject Microsoft.ApplicationInsights.AspNetCore.JavaScriptSnippet JavaScriptSnippet
```

```
@Html.Raw(JavaScriptSnippet.FullScript)
</head>
```

Single Page Applications

- NPM: @microsoft/applicationinsights-web
- Angular: Create a Service and inject this in the main App-Component
- Important to track page changes and correlate requests

```
enableCorsCorrelation: true,
enableAutoRouteTracking: true
```

ANGULAR

```
import { Injectable } from '@angular/core';
import { ApplicationInsights } from '@microsoft/applicationinsights-web';
import { environment } from 'src/environments/environment';

@Injectable({
  providedIn: 'root'
})
export class AppInsightsService {

  instance: ApplicationInsights;

  constructor() {
    this.instance = new ApplicationInsights({ config: {
      instrumentationKey: environment.appInsights.instrumentationKey,
      enableCorsCorrelation: true,
      enableAutoRouteTracking: true
    } });
    this.instance.loadAppInsights();
    this.instance.trackPageView();
  }
}
```

ANGULAR

```
import { Component } from '@angular/core';
import { AuthService } from './auth/auth.service';
import { User } from './auth/user';
import { AppInsightsService } from './app-insights.service';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  title = 'Fabrikam Residences';

  get user(): User { return this.auth.getAuthenticatedUser(); }

  constructor(private auth: AuthService, private appInsights: AppInsightsService) { }
}
```

BLAZOR

Configure Telemetry

- No official support (yet)
- Use BlazorApplicationInsights Nuget
- https://github.com/IvanJosipovic/BlazorApplicationInsights
- Add call to Program.cs

```
builder.Services.AddBlazorApplicationInsights();
```

- Add component to App.razor

```
<ApplicationInsightsComponent />
```

- Add Application Insights JS to head in index.html
- Add JS Interop to the bottom of body in index.html

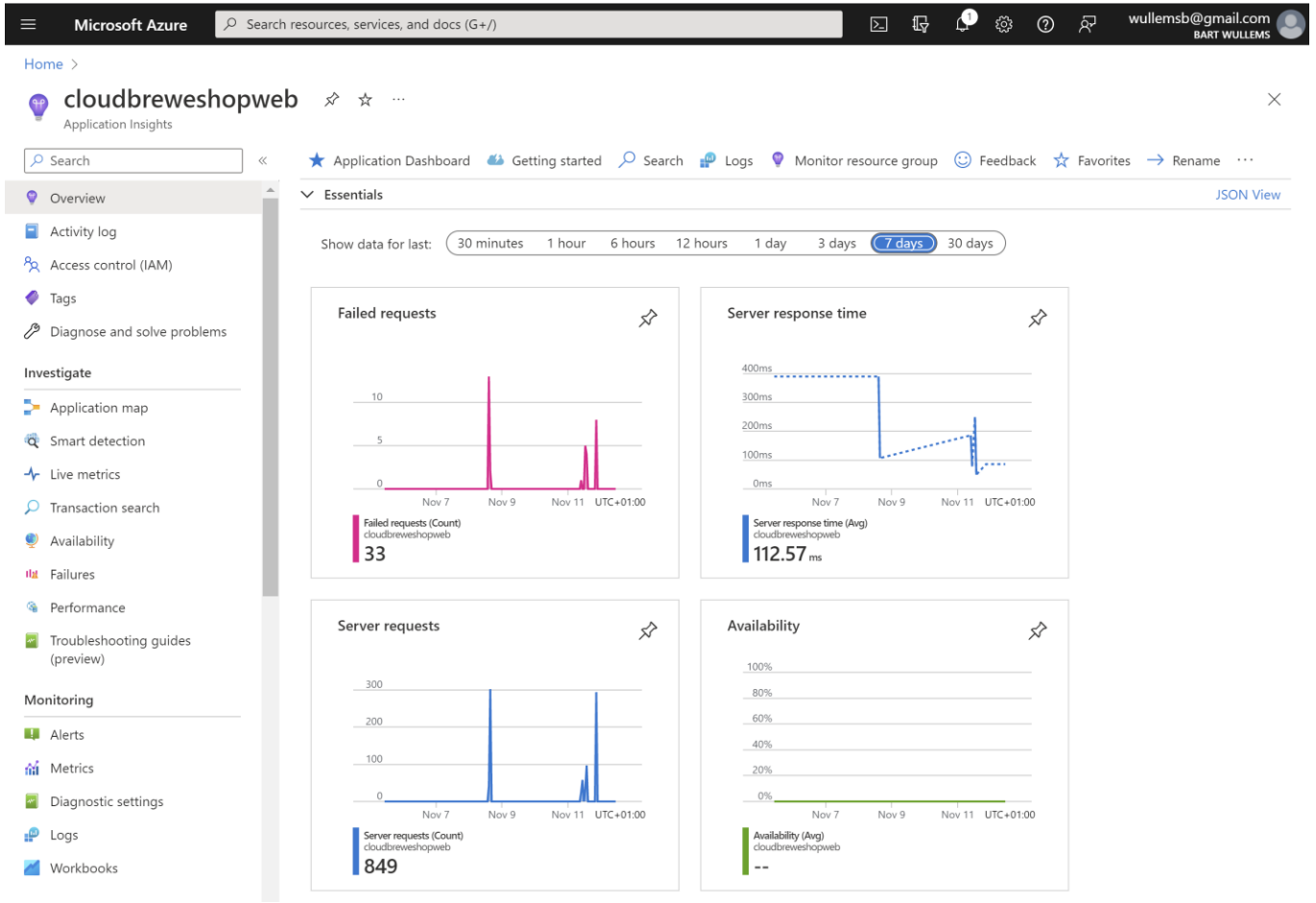
```
<script src="_content/BlazorApplicationInsights/JsInterop.js"></script>
```

CONSUMING TELEMETRY DATA

ORDINA
Ahead of change

OVERVIEW

- Provides a summary of the application's health - availability, performance and usage
- Paired timeline allows users to quickly correlate metrics and identify trends
- Drilldown into results

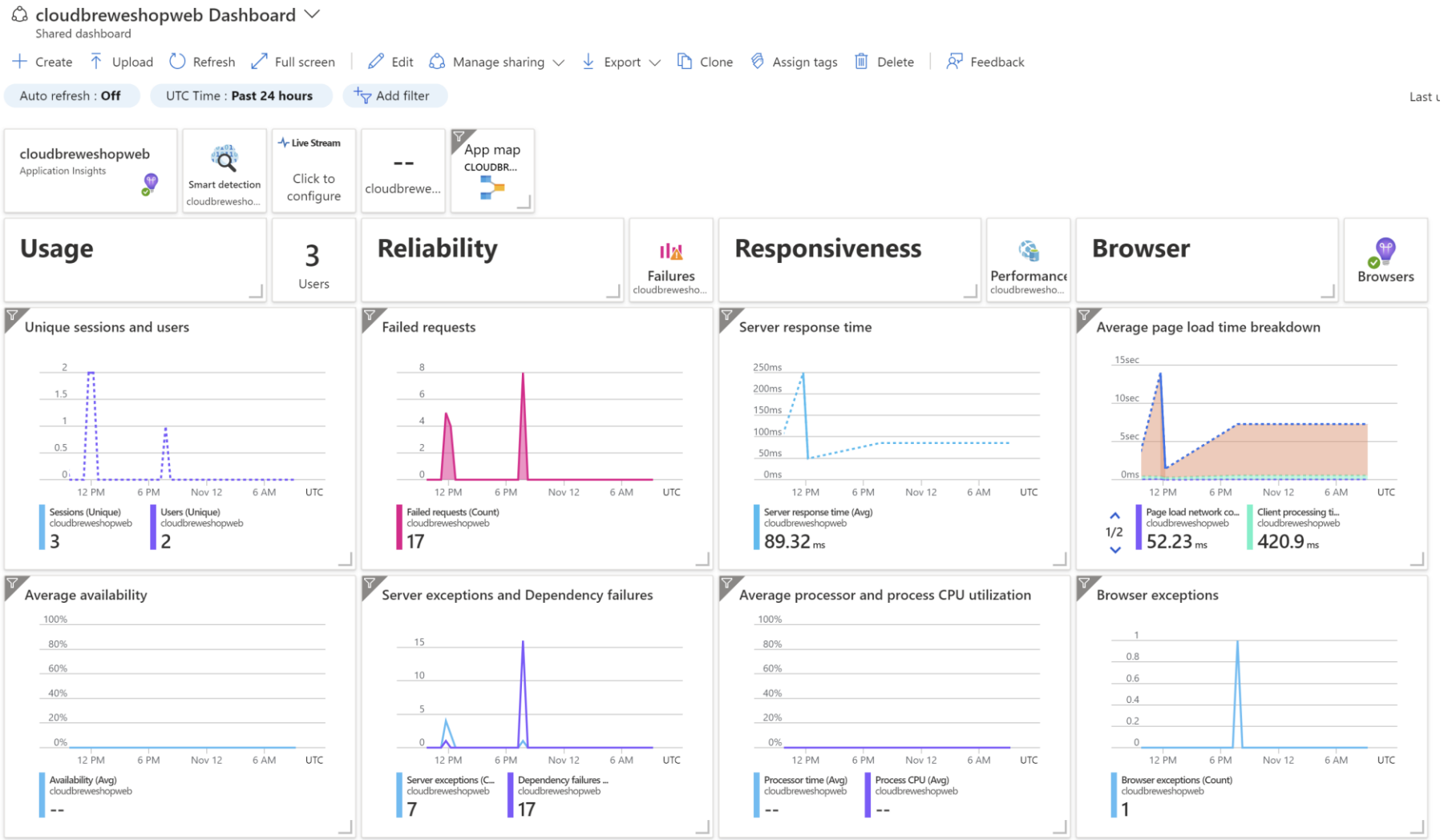


ORDINA
Ahead of change

CONSUMING TELEMETRY DATA

ORDINA

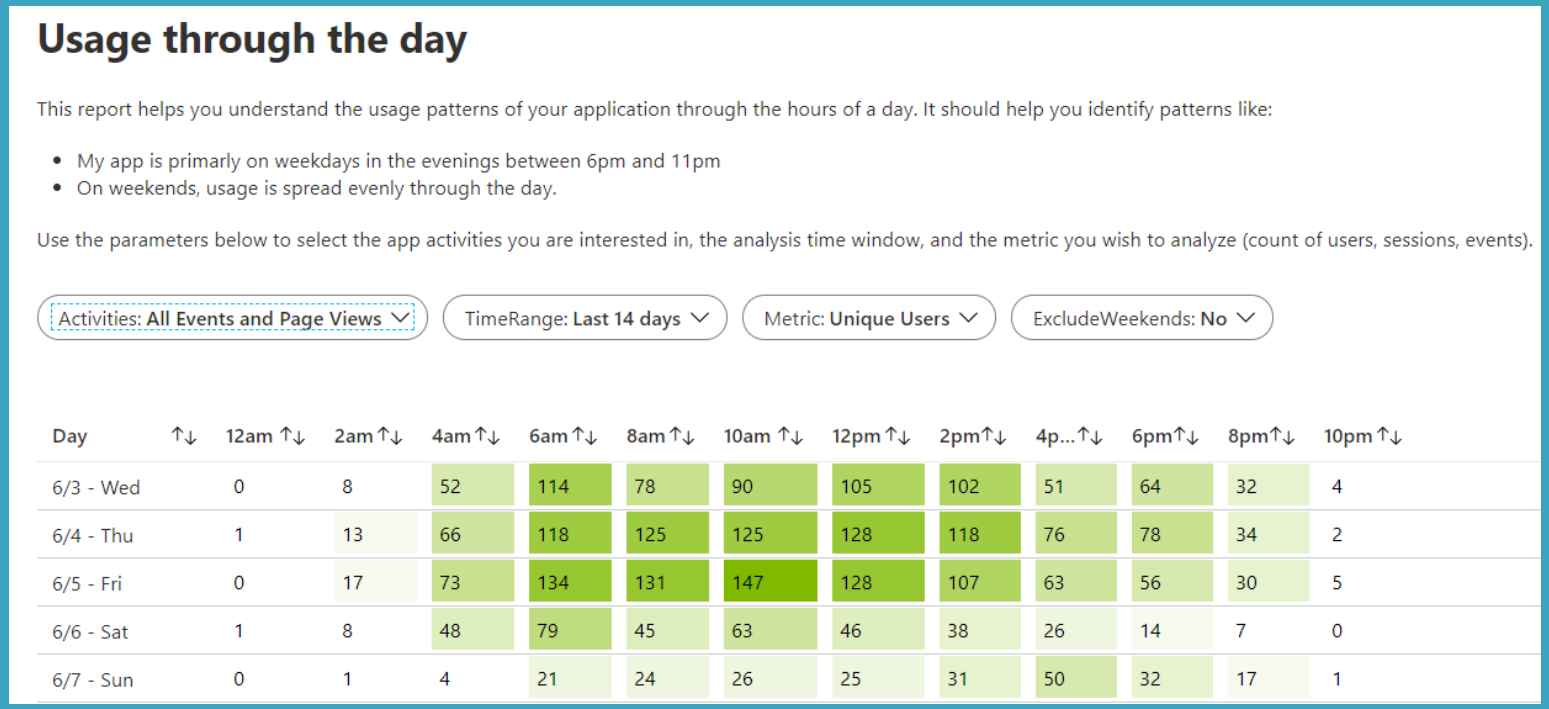
APPLICATION DASHBOARD



ORDINA
Ahead of change

WORKBOOKS

Workbooks provide a flexible canvas for data analysis and the creation of rich visual reports within the Azure portal. They allow you to tap into multiple data sources from across Azure, and combine them into unified interactive experiences.



Usage through the day

This report helps you understand the usage patterns of your application through the hours of a day. It should help you identify patterns like:

- My app is primarly on weekdays in the evenings between 6pm and 11pm
- On weekends, usage is spread evenly through the day.

Use the parameters below to select the app activities you are interested in, the analysis time window, and the metric you wish to analyze (count of users, sessions, events).

Activities: **All Events and Page Views** | TimeRange: **Last 14 days** | Metric: **Unique Users** | ExcludeWeekends: **No**

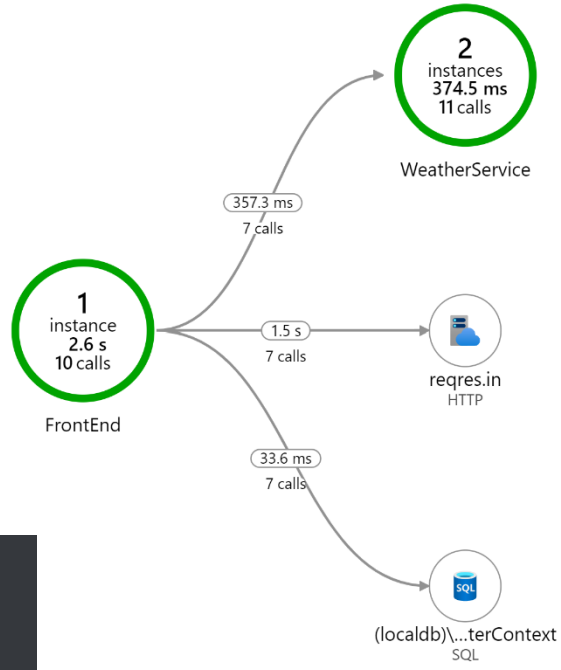
Day	12am	2am	4am	6am	8am	10am	12pm	2pm	4p...	6pm	8pm	10pm
6/3 - Wed	0	8	52	114	78	90	105	102	51	64	32	4
6/4 - Thu	1	13	66	118	125	125	128	118	76	78	34	2
6/5 - Fri	0	17	73	134	131	147	128	107	63	56	30	5
6/6 - Sat	1	8	48	79	45	63	46	38	26	14	7	0
6/7 - Sun	0	1	4	21	24	26	25	31	50	32	17	1

ORDINA
Ahead of change

APPLICATION MAP

ORDINA
Ahead of change

APPLICATION MAP



- Use the same Application Insights Resource for your complete solution
- Update the RoleName to better identify the different components

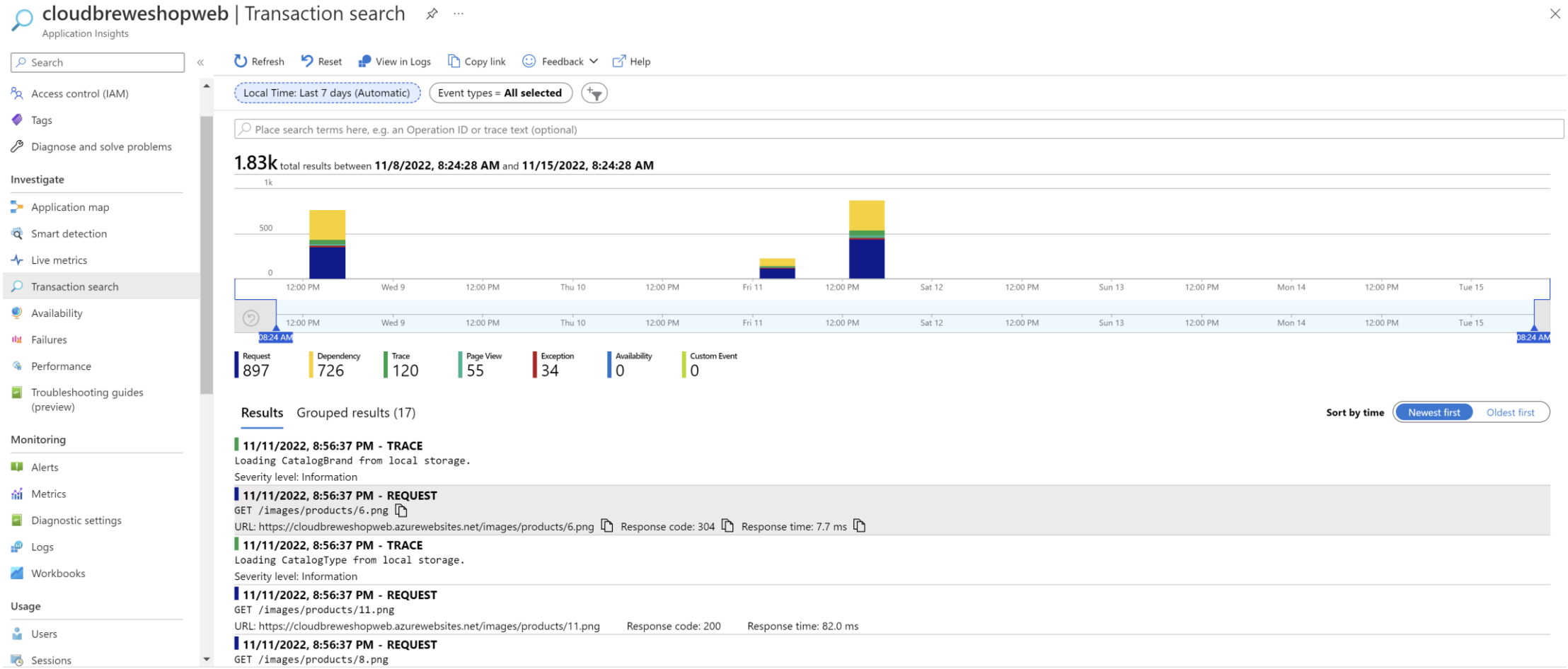
```
public class CloudRoleNameTelemetryInitializer : ITelemetryInitializer
{
    public void Initialize(ITelemetry telemetry)
    {
      // set custom role name here
      telemetry.Context.Cloud.RoleName = "Custom RoleName";
    }
}
```

```
services.AddSingleton<ITelemetryInitializer, CloudRoleNameTelemetryInitializer>();
```

ORDINA
Ahead of change

https://bartwullems.blogspot.com/2022/06/azure-application-insightsset-cloud.html
https://bartwullems.blogspot.com/2018/10/application-insightschange-application.html

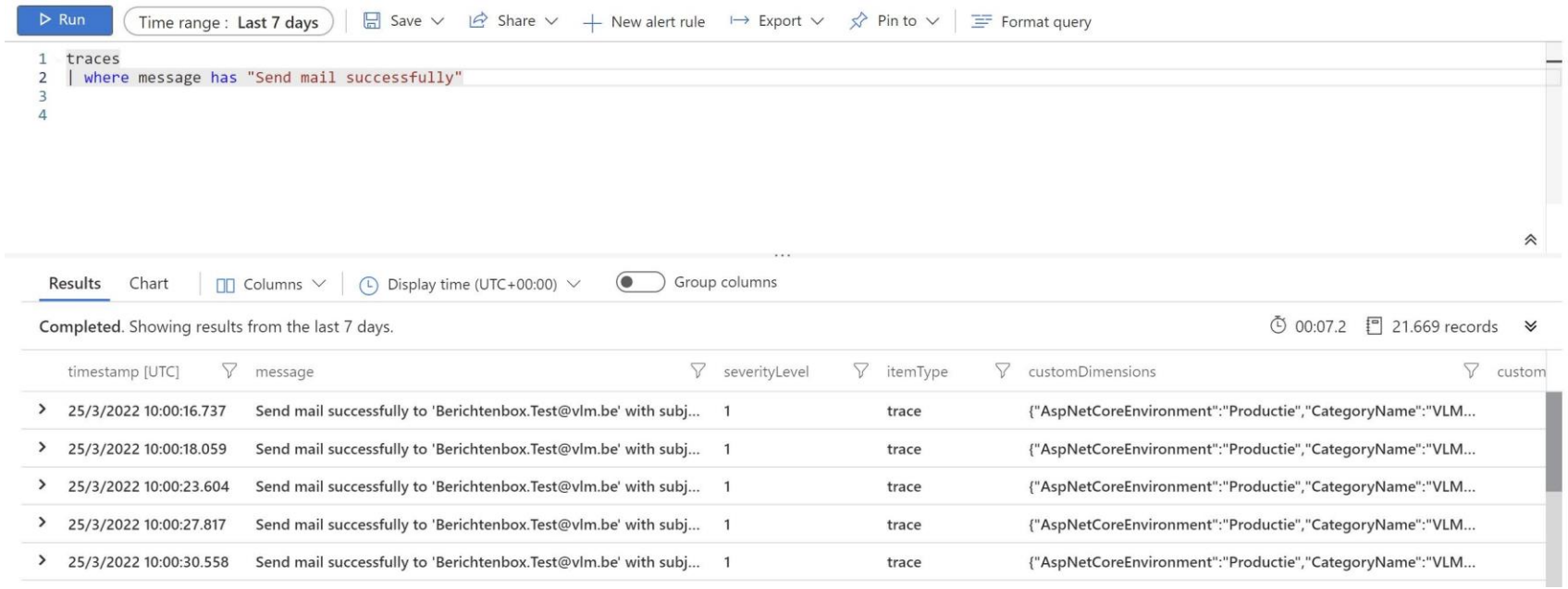
TRANSACTION SEARCH



ORDINA
Ahead of change

LOGS

- Learn KQL:
  - Kusto Query Language is a powerful tool to explore your data and discover patterns, identify anomalies and outliers, create statistical modeling, and more.
  - It uses schema entities that are organized in a hierarchy similar to SQL's: databases, tables, and columns.



AZURE MANAGED GRAFANA (PREVIEW)

ORDINA
Ahead of change

AZURE MANAGED GRAFANA

ORDINA

PERFORMANCE COUNTERS

ORDINA
Ahead of change

PERFORMANCE COUNTERS

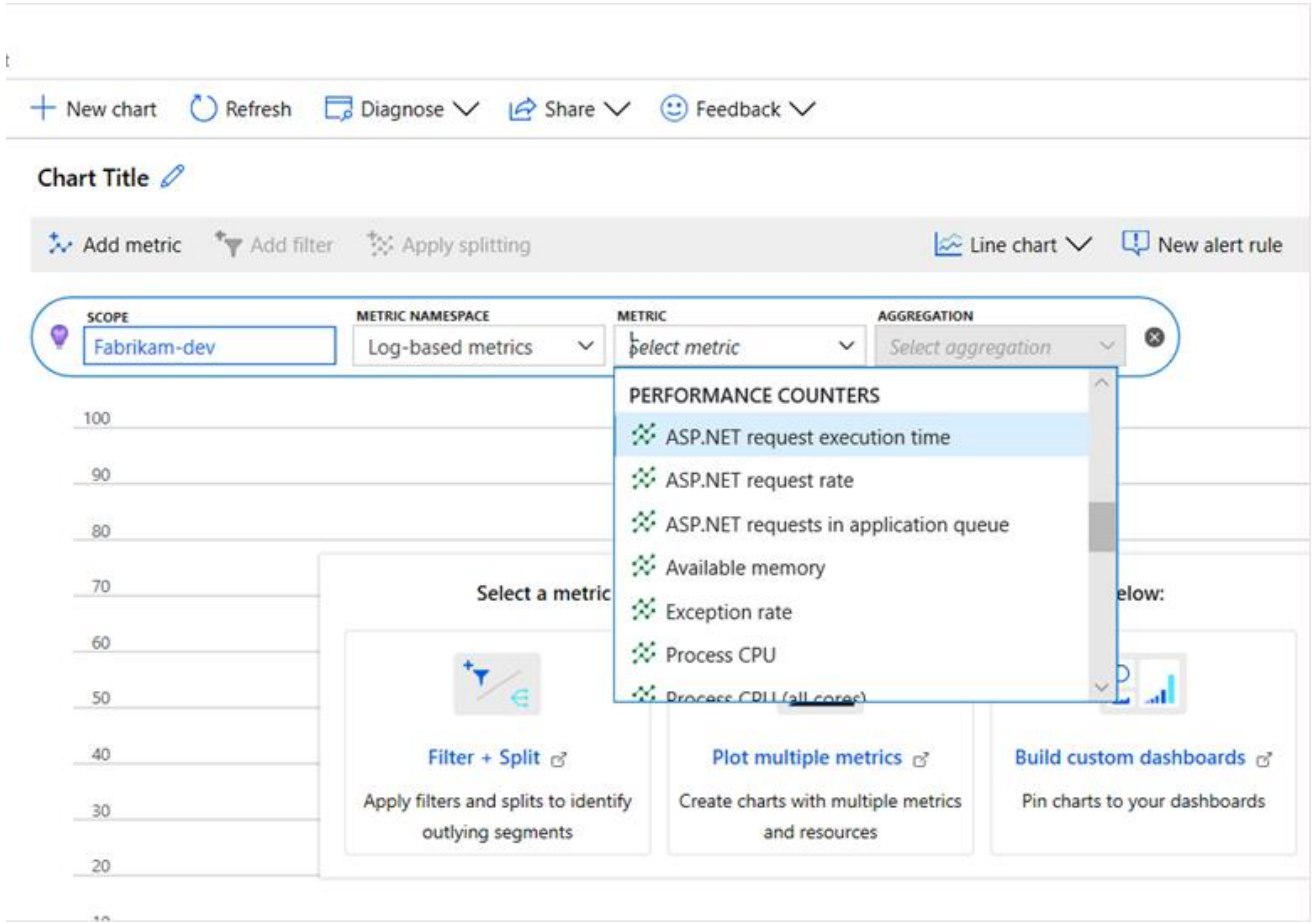
Set of default counters are collected by default

- Memory
- CPU
- # of requests
- …

Replaced by *EventCounters* to enable x-platform compatibility

- Disabled by default

Tip: If running on-premise; add AppPool user to the correct groups!



ORDINA
Ahead of change

PERFORMANCE COUNTERS

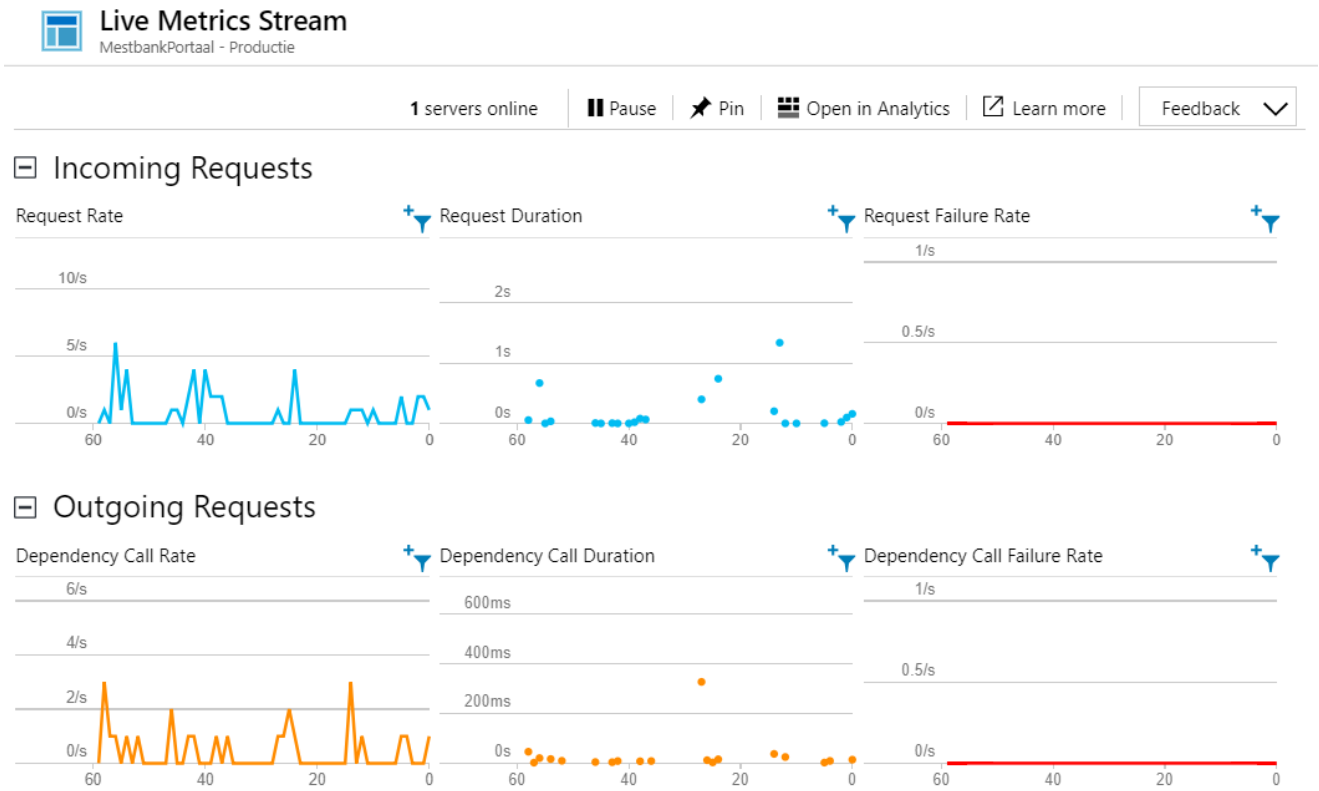
ORDINA

LIVE METRICS

ORDINA
Ahead of change

LIVE METRICS

- Enabled by default
- Watch the effect of test loads, and diagnose issues live
- Get exception traces as they happen
- Experiment with filters to find the most relevant KPIs
- Monitor any Windows performance counter live



ORDINA
Ahead of change

LIVE METRICS

STRUCTURED LOGGING

ORDINA
Ahead of change

STRUCTURED LOGGING


```
{
    "dt": "2020-04-07T12:23:34Z",
    "level": "info",
    "message": "Completed 200 OK in 79ms (Views: 78.8ms | ActiveRecord: 0.0ms)",
    "context": {
        "host": "34.225.155.83",
        "user_id": 5,
        "path": "/welcome",
        "method": "GET"
    },
    "http_response_sent": {
        "status": 200,
        "duration_ms": 79.0,
        "view_ms": 78.8,
        "active_record_ms": 0.0
    }
}
```

STRUCTURED LOGGING

ORDINA

STRUCTURE

- Supported by most logg
  - Serilog, Nlog, L
- Also works with standa

```
_logger.LogInformation(message:
```

TRACE
Warning

Custom Properties

SpanId	6b7bfc088498d924
ParentId	0000000000000000
{OriginalFormat}	Fetching catalog items - Brand: {ProductBrand} - Type: {ProductType}
RequestPath	/
ActionName	/Index
ProductType	All
TraceId	606055cfeb1cfa8e61247e363353e8b6
ProductBrand	All
CategoryName	Microsoft.eShopWeb.Web.Pages.IndexModel
ActionId	d0b325fd-900c-406d-bf11-36b3eaa07b99
RequestId	80000bdf-0000-d100-b63f-84710c7967bb
AspNetCoreEnvironment	Production

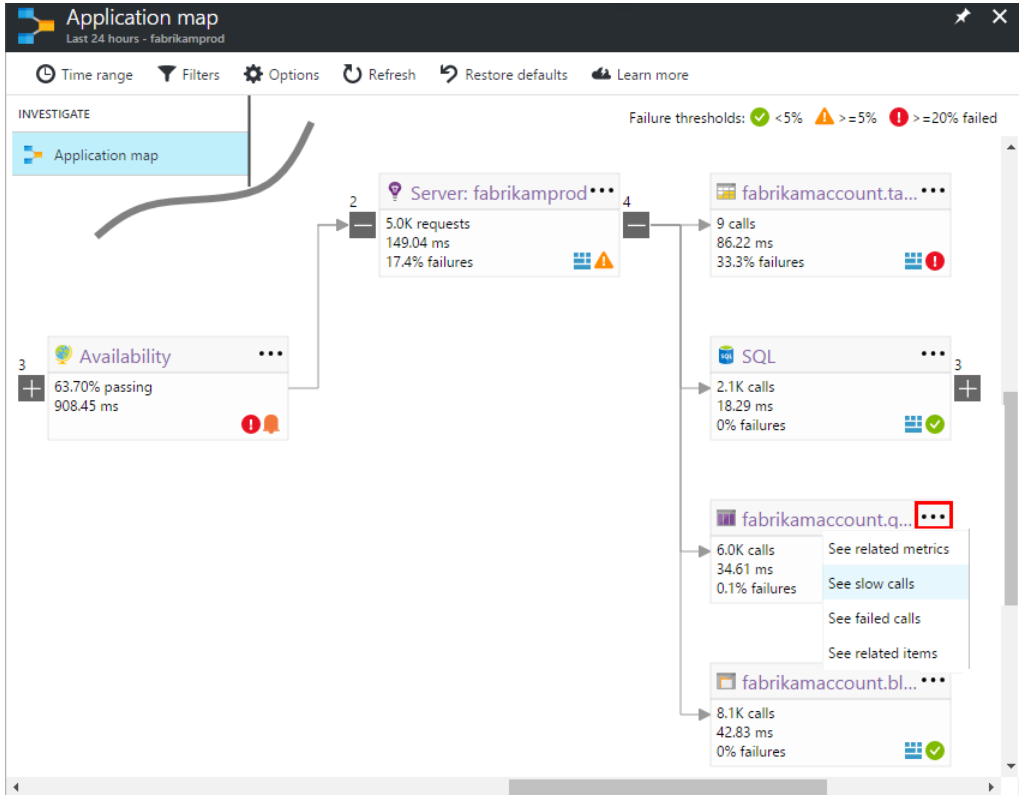
```
:Type}",params args:brand,type);
```

DEPENDENCY TRACKING

ORDINA
Ahead of change

DEPENDENCY TRACKING

- Dependencies are calls to external systems and services
- Some dependencies are tracked automatically
- We can also monitor dependencies performance
- Tip: Set the cloud role name correct for every component



ORDINA
Ahead of change

DEPENDENCY TRACKING

- Following dependencies are tracked automatically

Dependencies	Details
Http/Https	Local or Remote http/https calls
WCF calls	Only tracked automatically if Http-based bindings are used.
SQL	Calls made with `SqlClient`. See this for capturing SQL query.
Azure storage (Blob, Table, Queue)	Calls made with Azure Storage Client.
EventHub Client SDK	Version 1.1.0 and above.
ServiceBus Client SDK	Version 3.0.0 and above.
Azure Cosmos DB	Only tracked automatically if HTTP/HTTPS is used. TCP mode won't be captured by Application Insights.

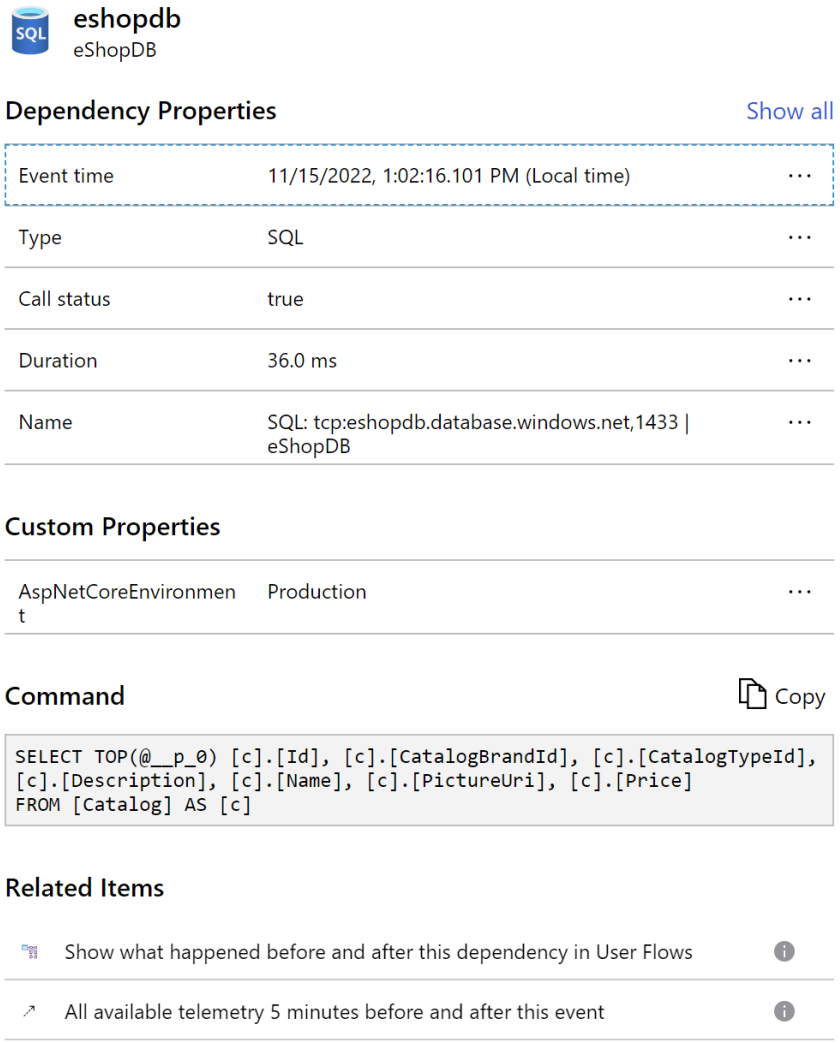
DEPENDENCY TRACKING

ORDINA

SQL SERVER

- SQL Command logging is disabled by default
- Enable through:

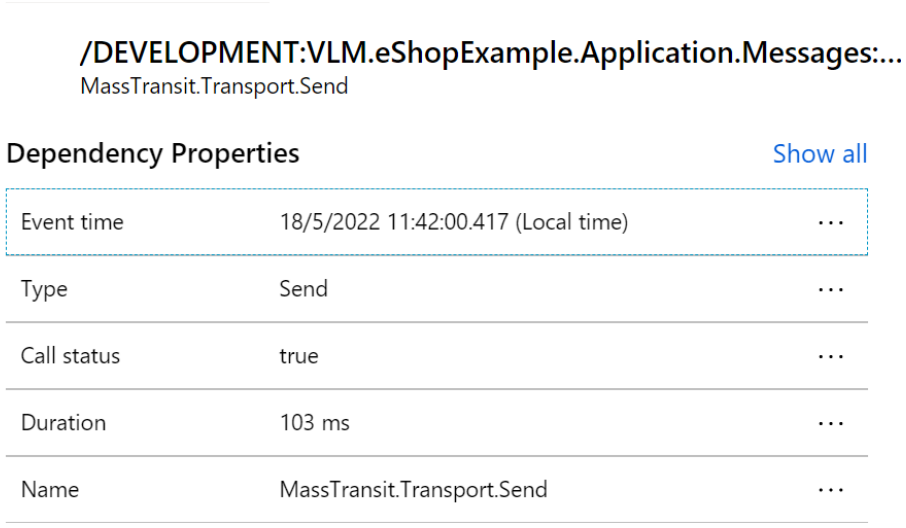
```
services.ConfigureTelemetryModule<DependencyTrackingTelemetry
Module>((module, o)
                => {
module.EnableSqlCommandTextInstrumentation = true; });
```



MASSTRANSIT

- You can add other dependencies (e.g. MassTransit)

```
services.ConfigureTelemetryModule<DependencyTrackingTelemetryModule>((module, o)
    => {
        module.EnableSqlCommandTextInstrumentation = true;
        module.IncludeDiagnosticSourceActivities.Add("MassTransit");
    });
```



ENHANCE TELEMETRY

ORDINA
Ahead of change

ENHANCE TELEMETRY

Telemetry initializers

- Sets context properties that are sent along with every item of telemetry.
- Lots of built-in initializers.
- Extra initializers exist; e.g. Kubernetes
- Easy to write your own e.g. CloudRoleNameInitializer.

Telemetry modules

- Collects a specific type of data and uses the core API to send it.
- Lots of built-in modules
- 3th party modules exist; e.g. NServiceBus

Telemetry processors

- Can filter and modify each telemetry item before it's sent to the portal.
- E.g. filter out all 400 responses

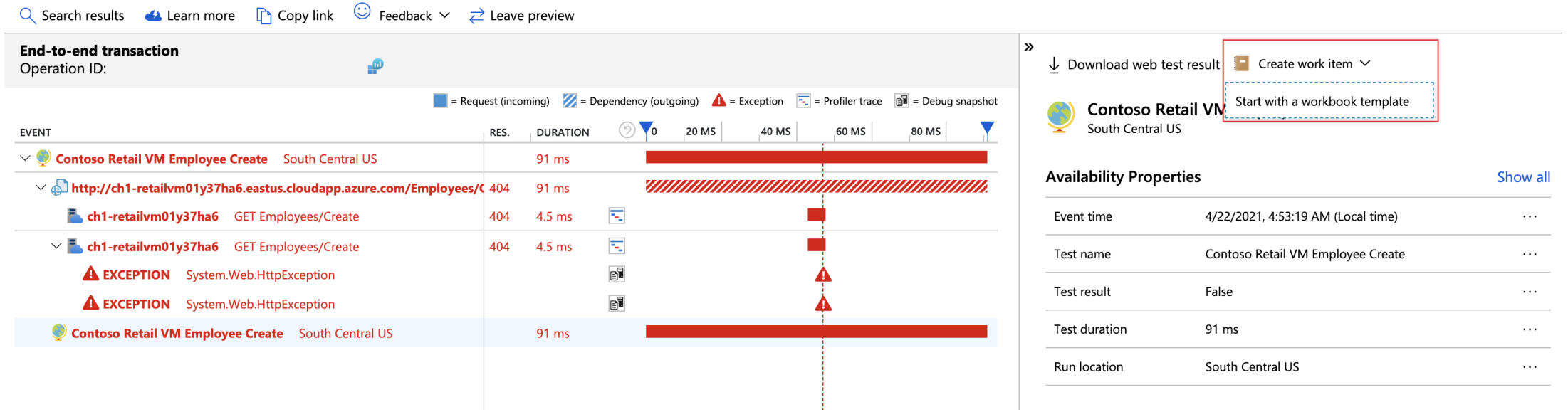
https://learn.microsoft.com/en-us/azure/azure-monitor/app/configuration-with-applicationinsights-config#telemetry-modules-aspnet

AZURE DEVOPS/ GITHUB INTEGRATION

ORDINA
Ahead of change

WORK ITEM INTEGRATION

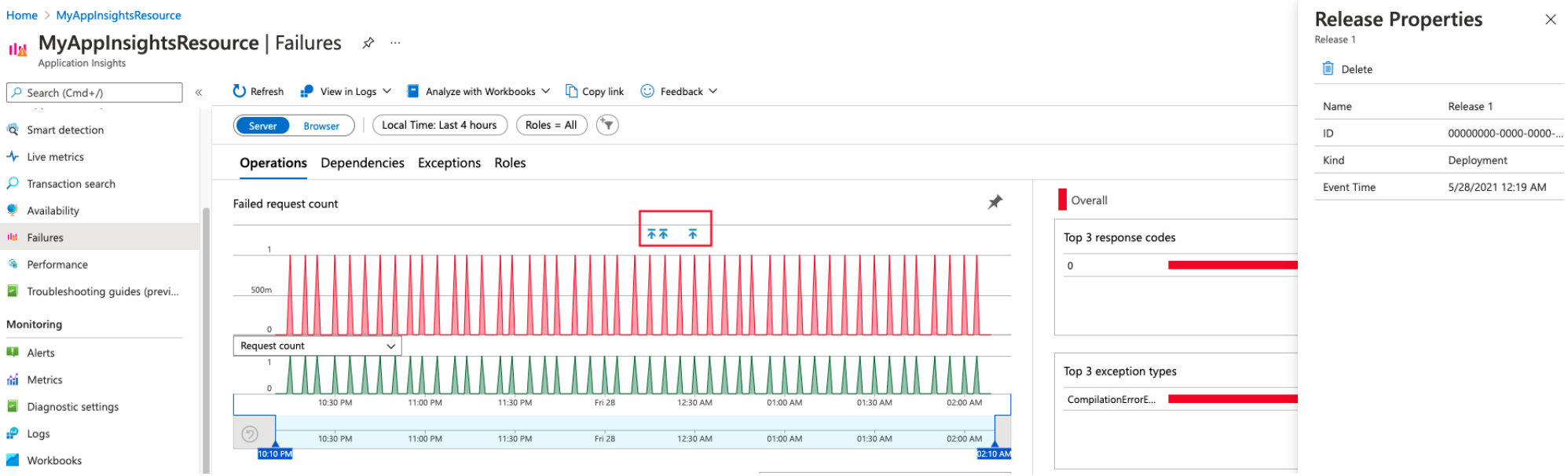
- Allows you to easily create work items in GitHub or Azure DevOps that have relevant Application Insights data embedded in them.
- Works for both Azure DevOps and Github



ORDINA
Ahead of change

RELEASE ANNOTATIONS

- Annotations show where you deployed a new build, or other significant events.
- Annotations make it easy to see whether your changes had any effect on your application's performance.
- Use the ReleaseAnnotation.ps1 powershell script in your release pipeline



ORDINA
Ahead of change

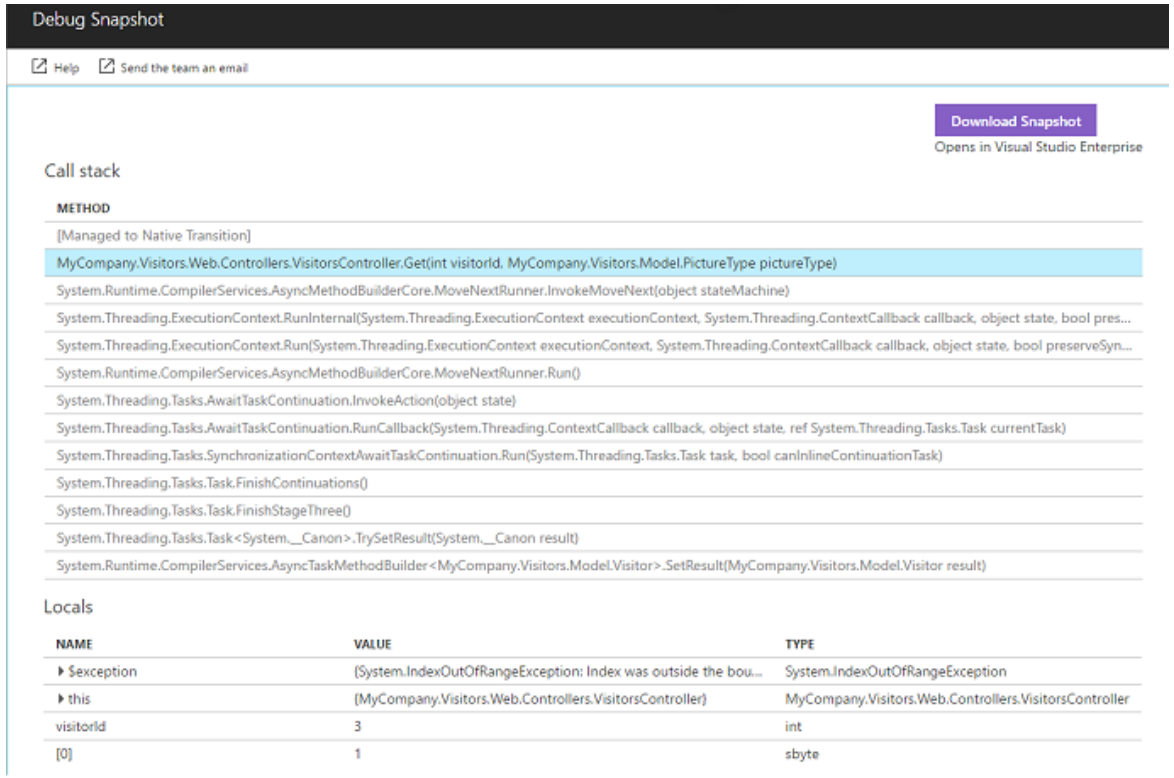
SNAPSHOT DEBUGGER

ORDINA
Ahead of change

SNAPSHOT DEBUGGER

- Collect a debug snapshot from your live application when an exception occurs.
- The snapshot shows the state of source code and variables at the moment the exception was thrown.
- Tricky to get up and running ☺

- Only on Windows Service Plan(Azure)



ORDINA
Ahead of change

ENABLE SNAPSHOT DEBUGGER

Enable Application Insights Snapshot Debugger for your application

Snapshot collection is available for:

- .NET Framework and ASP.NET applications running .NET Framework LTS or later.
- .NET Core and ASP.NET Core applications running .NET Core LTS on Windows.
- .NET LTS applications on Windows.

We don't recommend using .NET Core versions prior to LTS since they're out of support.

The following environments are supported:

- Azure App Service
- Azure Function
- Azure Cloud Services running OS family 4 or later
- Azure Service Fabric services running on Windows Server 2012 R2 or later
- Azure Virtual Machines and virtual machine scale sets running Windows Server 2012 R2 or later
- On-premises virtual or physical machines running Windows Server 2012 R2 or later or Windows 8.1 or later

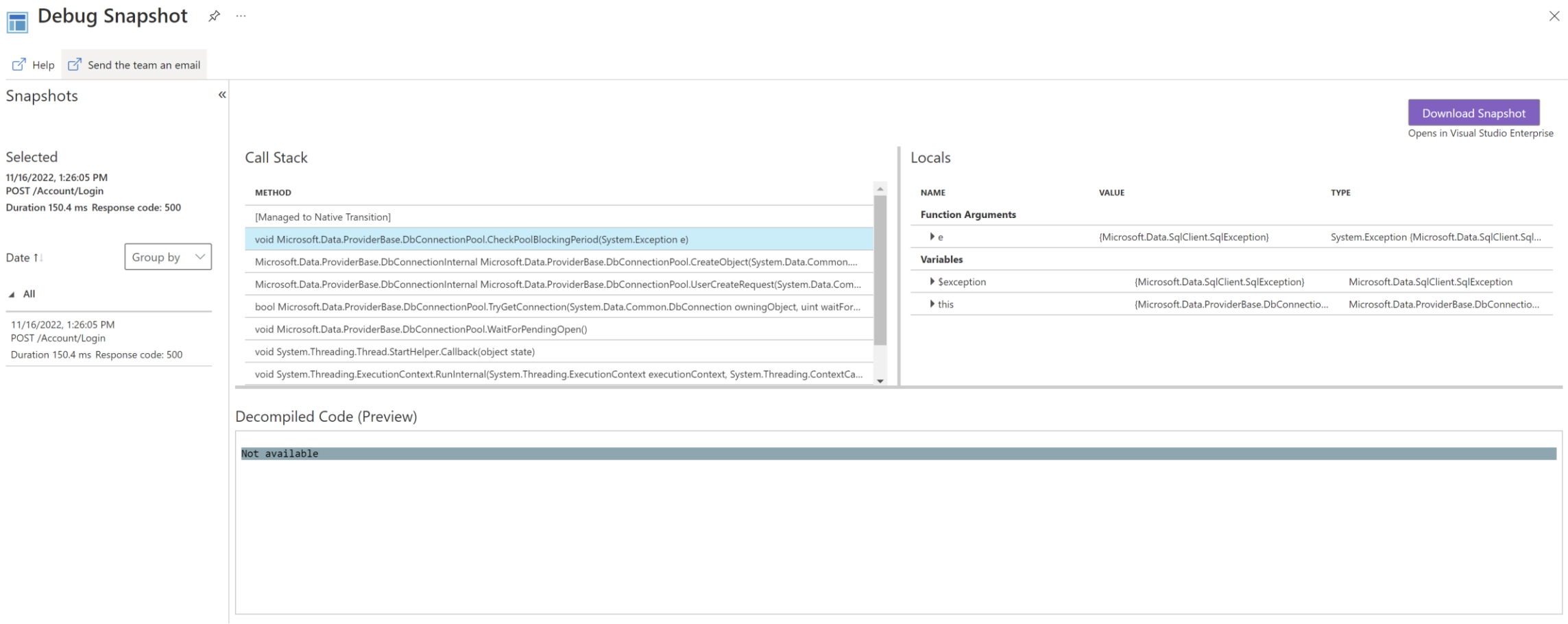
> **Note**
>
> Client applications (for example, WPF, Windows Forms or UWP) aren't supported.

If you've enabled Snapshot Debugger but aren't seeing snapshots, check our Troubleshooting guide.

SNAPSHOT DEBUGGER

ORDINA

SNAPSHOT DEBUGGER



PERFORMANCE PROFILING

ORDINA
Ahead of change

PERFORMANCE PROFILING

Through Application Insights Profiler

Capture and view performance traces

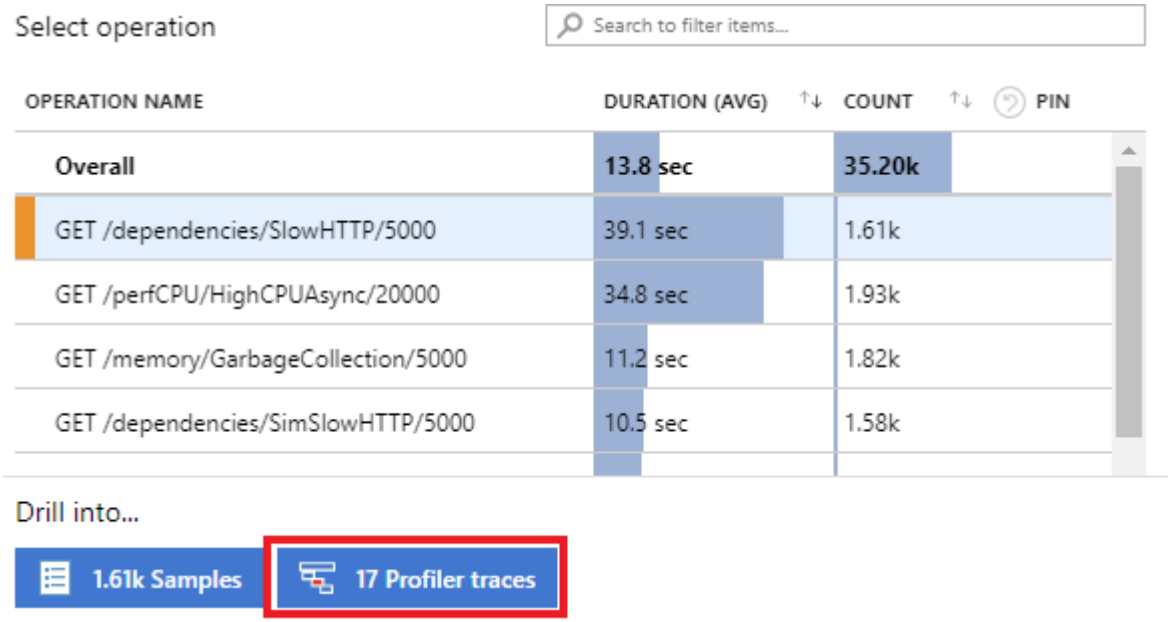
- Captures the following information:
- The median, fastest, and slowest response times
- The “hot” code path spending the most time handling a particular web request.

Trigger based

- Sampling trigger: random profiling once an hour for 2min
- CPU trigger: starts profiling when CPU usage is above 80 %
- Memory trigger: starts profiling when memory usage is above 80 %

Minimal overhead

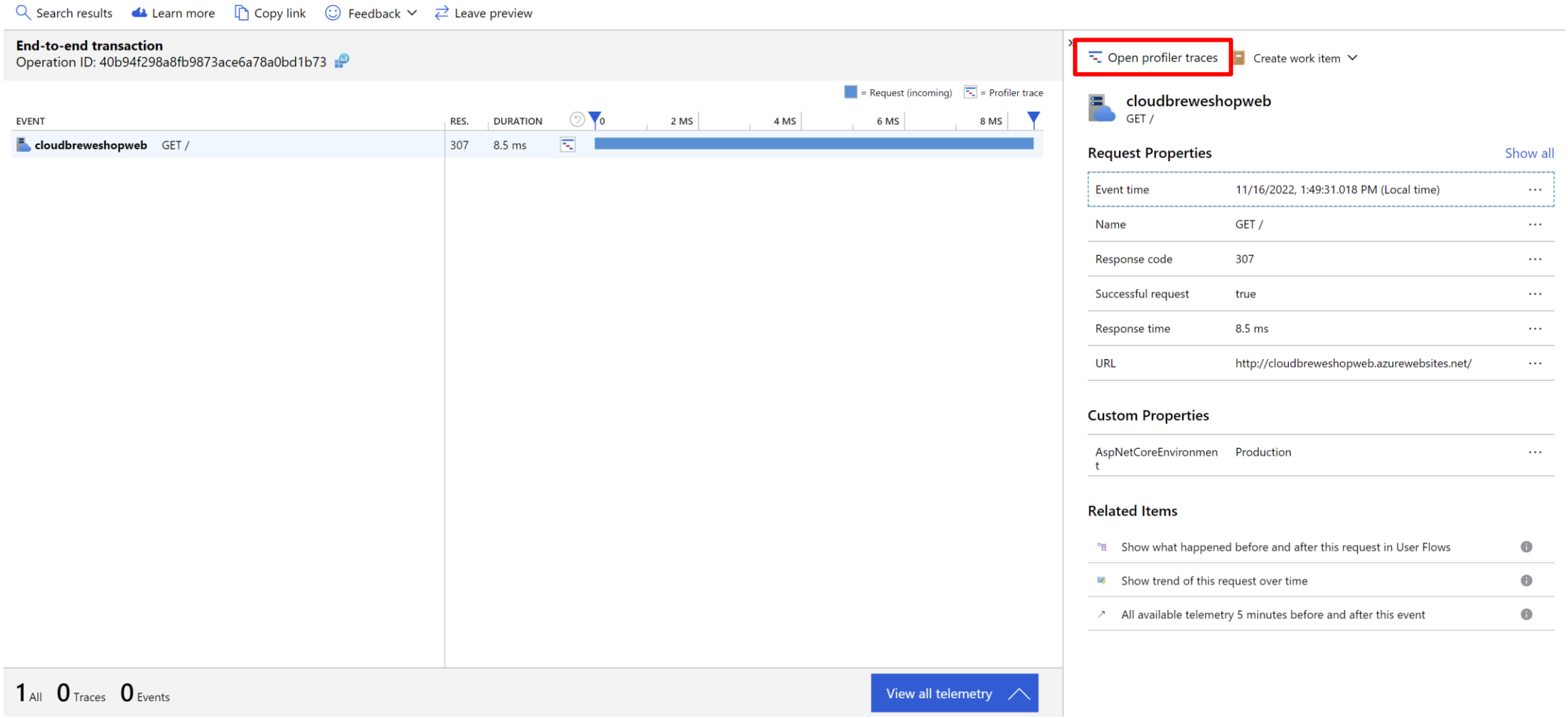
- 5-15% CPU overhead



ORDINA
Ahead of change

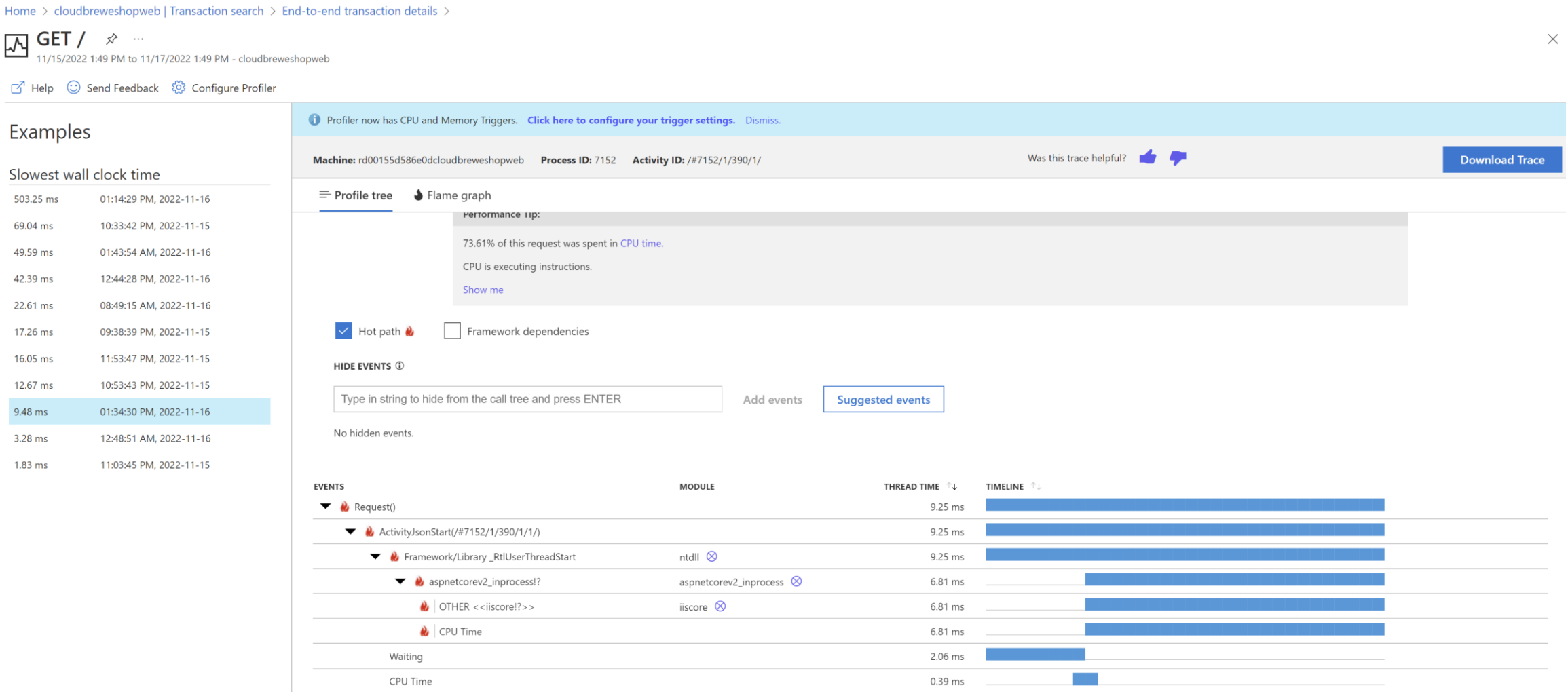
PERFORMANCE PROFILING

PERFORMANCE PROFILING



ORDINA
Ahead of change

PERFORMANCE PROFILING



ORDINA
Ahead of change

OPEN TELEMETRY

ORDINA
Ahead of change

OPEN TELEMETRY

Open Telemetry provides you with:

- A single, vendor-agnostic instrumentation library
- An end-to-end implementation to generate, emit, collect, process and export telemetry data.
- Full control of your data with the ability to send data to multiple destinations in parallel through configuration.
- Open-standard semantic conventions to ensure vendor-agnostic data collection

OPEN TELEMETRY APPLICATION INSIGHTS SUPPORT

- Supported through the Azure Monitor OpenTelemetry Exporter
- !!Still in preview!!

Limitations of the preview release

.NET | Node.js | Python

Consider whether this preview is right for you. It *enables distributed tracing, metrics* and *excludes*:

- Live Metrics
- Logging API (like console logs and logging libraries)
- Profiler
- Snapshot Debugger
- Azure Active Directory authentication
- Autopopulation of Cloud Role Name and Cloud Role Instance in Azure environments
- Autopopulation of User ID and Authenticated User ID when you use the Application Insights JavaScript SDK
- Autopopulation of User IP (to determine location attributes)
- Ability to override Operation Name
- Ability to manually set User ID or Authenticated User ID
- Propagating Operation Name to Dependency Telemetry
- Instrumentation libraries support on Azure Functions
- Status supports statuscode(unset,ok,error) and status-description. "Status Description" is ignored by Azure Monitor Exporters.

If you require a full-feature experience, use the existing Application Insights ASP.NET, or ASP.NET Core SDK until the OpenTelemetry-based offering matures.

OPEN TELEMETRY

ORDINA

SMART DETECTION

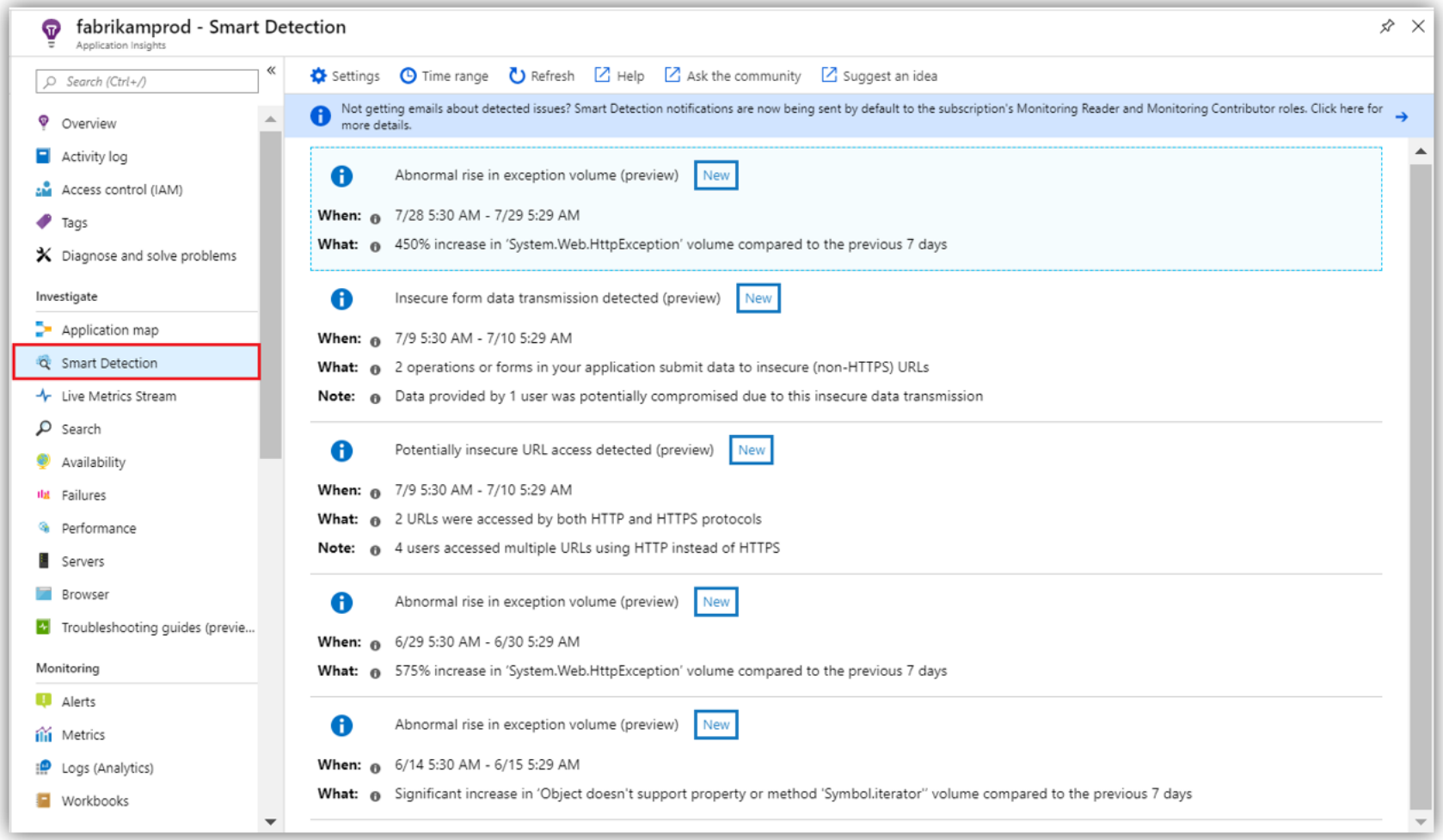
ORDINA
Ahead of change

SMART DETECTION

Smart detection detects and notifies about various issues:

- Failure Anomalies. Notifies if the failure rate goes outside the expected envelope.
- Performance Anomalies. Notifies if response time of an operation or dependency duration is slowing down, compared to historical baseline.
- General degradations and issues, like Trace degradation, Memory leak, Abnormal rise in Exception volume and Security anti-patterns.

SMART DETECTION



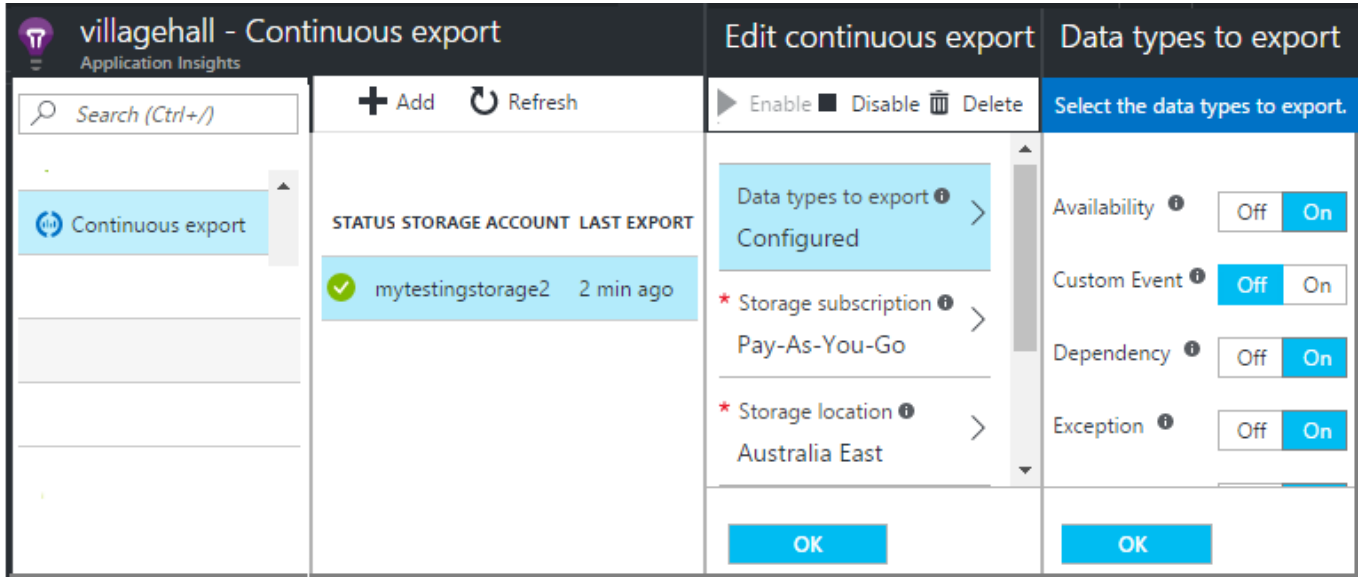
SMART DETECTION

CONTINUOUS EXPORT

ORDINA
Ahead of change

CONTINUOUS EXPORT

- Metrics data is available for 90 days in Application Insights
- Sends telemetry data to Azure Blob Storage
- Files in JSON format



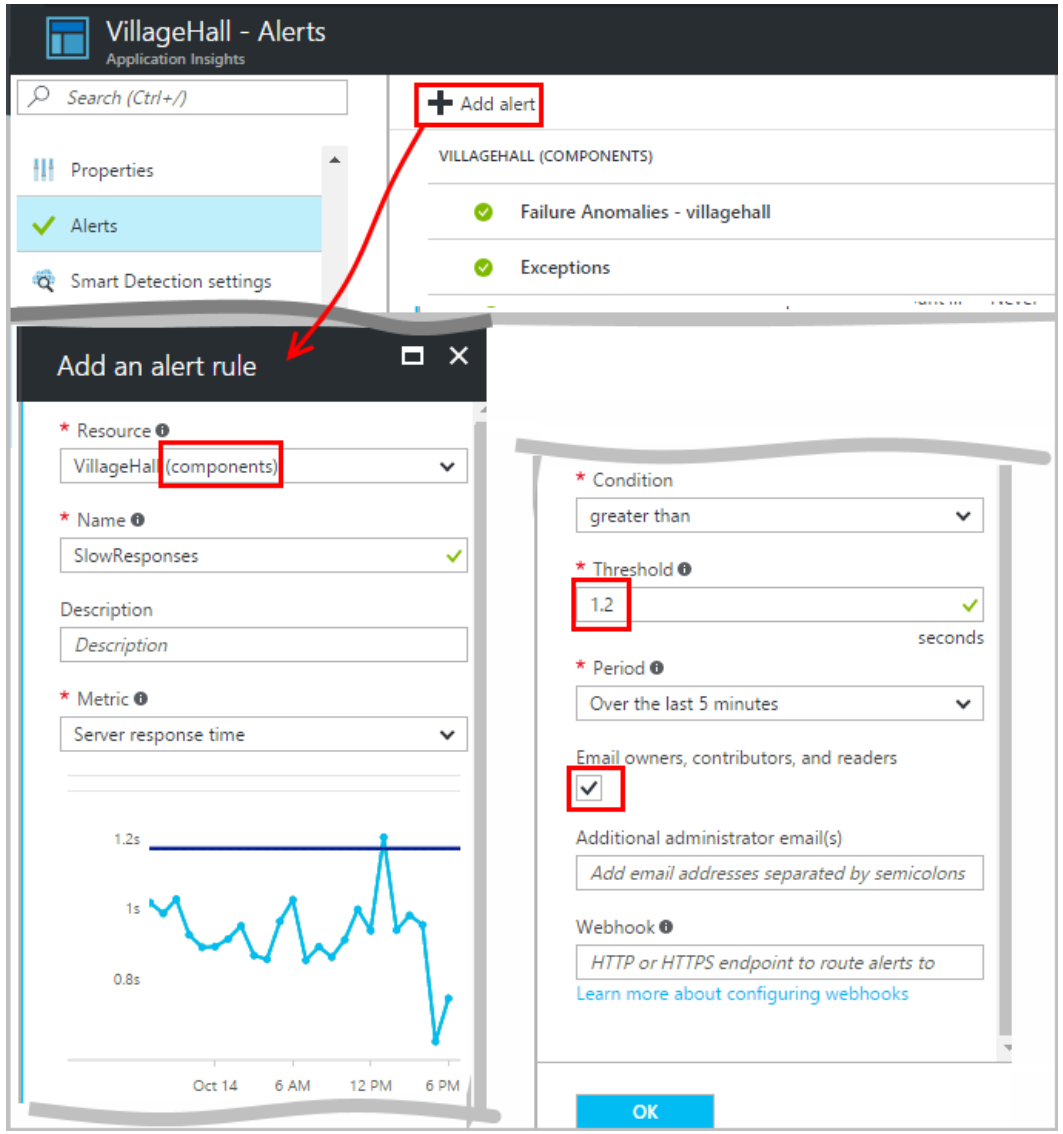
ORDINA
Ahead of change

ALERTS

ORDINA
Ahead of change

ALERTS

- Define conditions on metrics
- Set problem interval
- Set e-mail address



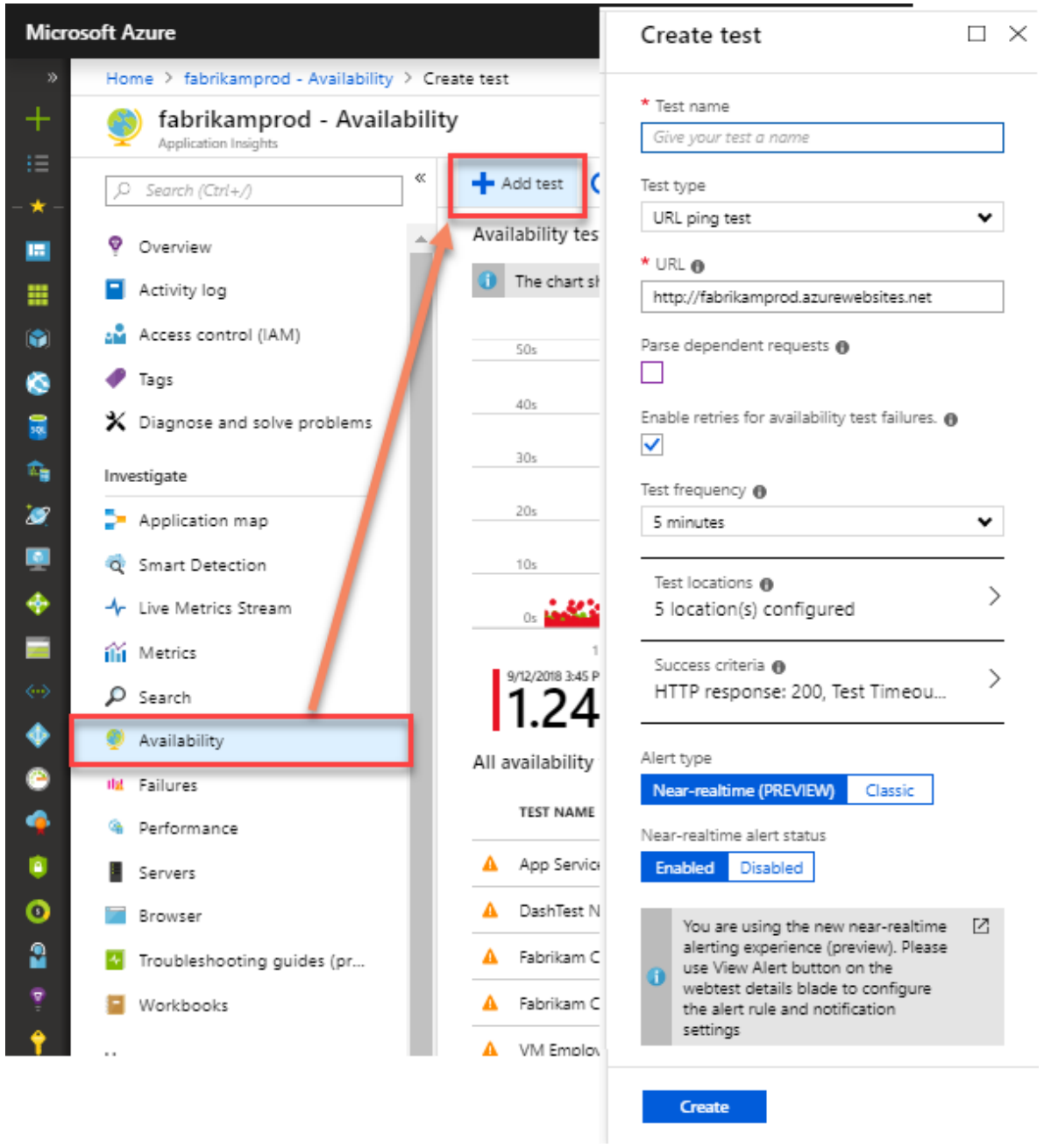
ORDINA
Ahead of change

SITE AVAILABILITY

ORDINA
Ahead of change

SITE AVAILABILITY

- Define web tests to ping
  - Visual Studio web tests are also supported
- Ping from different countries
- Get alerts when unavailable



ORDINA
Ahead of change

CONCLUSION

ORDINA
Ahead of change

WRAPPING UP

- Application Insights is an easy-to-use telemetry service.
- Gives you a lot of useful information about your applications.
- Lots of features! Start using them to your advantage.

https://github.com/wullemsb/presentations

ORDINA
Ahead of change

ORDINA

Ahead of change