Configure Instrumentation in a Web Application

Jeff Hopper
www.hoppertech.net

Overview

Configuration Modifications

Instrumentation API

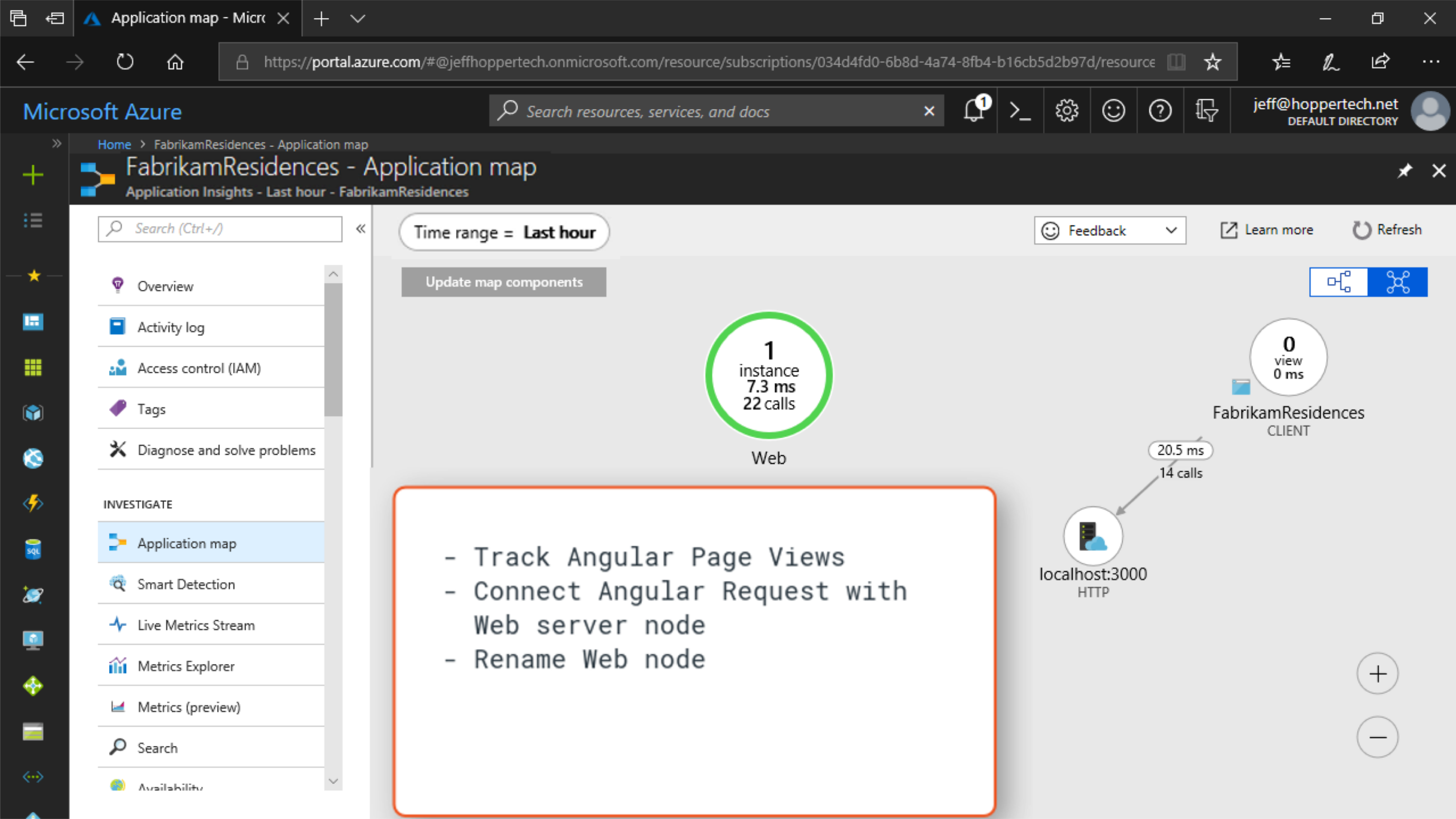
- Page Views
- User Interactions
- Trace Logs and Exceptions

Download Exercise Files

Source Code

Resources.md

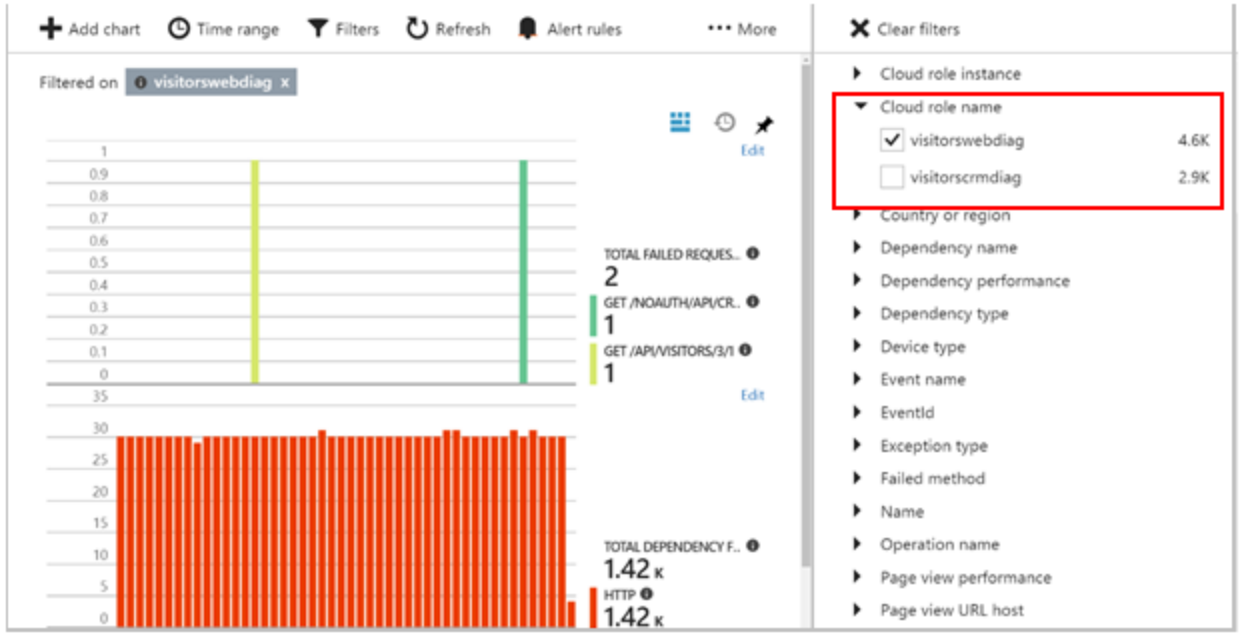
Configuration Modifications


Application map - Micr
https://portal.azure.com/#@jeffhoppertech.onmicrosoft.com/resource/subscriptions/034d4fd0-6b8d-4a74-8fb4-b16cb5d2b97d/resource
Microsoft Azure
Search resources, services, and docs
jeff@hoppertech.net
DEFAULT DIRECTORY
Home > FabrikamResidences - Application map
FabrikamResidences - Application map
Application Insights - Last hour - FabrikamResidences
Search (Ctrl+/)
Overview
Activity log
Access control (IAM)
Tags
Diagnose and solve problems
INVESTIGATE
Application map
Smart Detection
Live Metrics Stream
Metrics Explorer
Metrics (preview)
Search
Time range = Last hour
Update map components
Feedback
Learn more
Refresh
1
instance
7.3 ms
22 calls
Web
0
view
0 ms
FabrikamResidences
CLIENT
20.5 ms
14 calls
localhost:3000
HTTP
- Track Angular Page Views
- Connect Angular Request with Web server node
- Rename Web node

Use cloud_RoleName to separate components

The `cloud_RoleName` property is attached to all telemetry. It identifies the component - the role or service - that originates the telemetry. (It is not the same as cloud_RoleInstance, which separates identical roles that are running in parallel on multiple server processes or machines.)

In the portal, you can filter or segment your telemetry using this property. In this example, the Failures blade is filtered to show just information from the front-end web service, filtering out failures from the CRM API backend:



Trace operations between components

Cloud_RoleName SDK Comparison

```
// ASP.Net in Startup.cs
public void ConfigureServices(IServiceCollection services)
{
  services.AddSingleton<ITelemetryInitializer>(new ServiceNameInitializer());
}
internal class ServiceNameInitializer : ITelemetryInitializer
{
    public void Initialize(ITelemetry telemetry)
    {
        telemetry.Context.Cloud.RoleName = "RoleName";
    }
}
```

```
// Node
const appInsights = require('applicationinsights');
appInsights.setup();
const key = appInsights.defaultClient.context.keys.cloudRole;
appInsights.defaultClient.context.tags[key] = "RoleName";
appInsights.start();
```

```
// ApplicationInsights-js
AppInsights.queue.push(() => {
  AppInsights.context.addTelemetryInitializer(envelope => {
    envelope.tags['ai.cloud.role'] = 'RoleName';
  });
});
```

Telemetry correlation data model

Application Insights defines a data model for distributed telemetry correlation. To associate telemetry with the logical operation, every telemetry item has a context field called `operation_Id`. This identifier is shared by every telemetry item in the distributed trace. So even with loss of telemetry from a single layer you still can associate telemetry reported by other components.

Distributed logical operation typically consists of a set of smaller operations - requests processed by one of the components. Those operations are defined by request telemetry. Every request telemetry has its own `id` that uniquely globally identifies it. And all telemetry - traces, exceptions, etc. associated with this request should set the `operation_parentId` to the value of the request `id`.

Every outgoing operation like http call to another component represented by dependency telemetry. Dependency telemetry also defines its own `id` that is globally unique. Request telemetry, initiated by this dependency call, uses it as `operation_parentId`.

You can build the view of distributed logical operation using `operation_Id`, `operation_parentId`, and `request.id` with `dependency.id`. Those fields also define the causality order of telemetry calls.

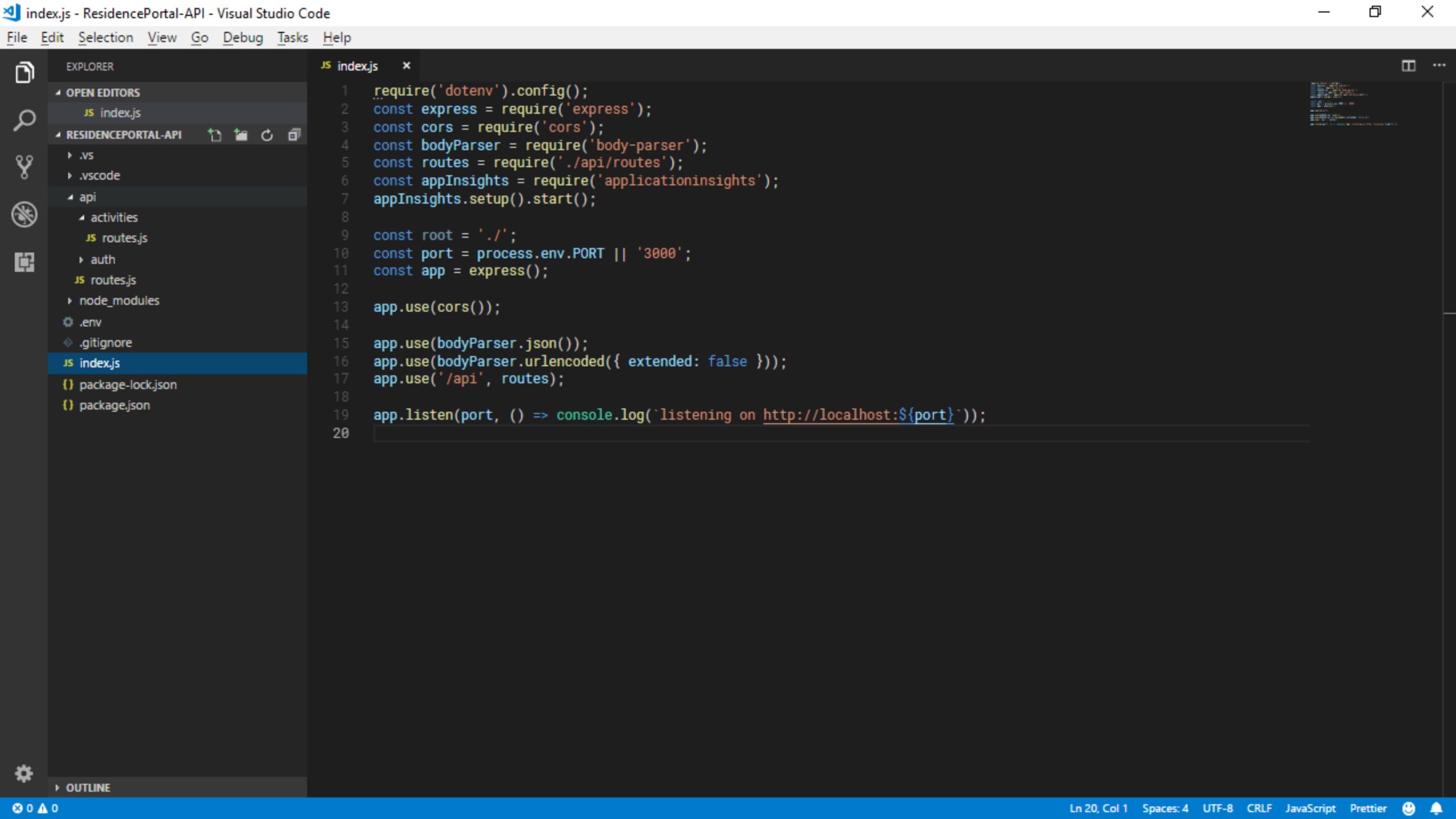
In micro services environment, traces from components may go to the different storages.

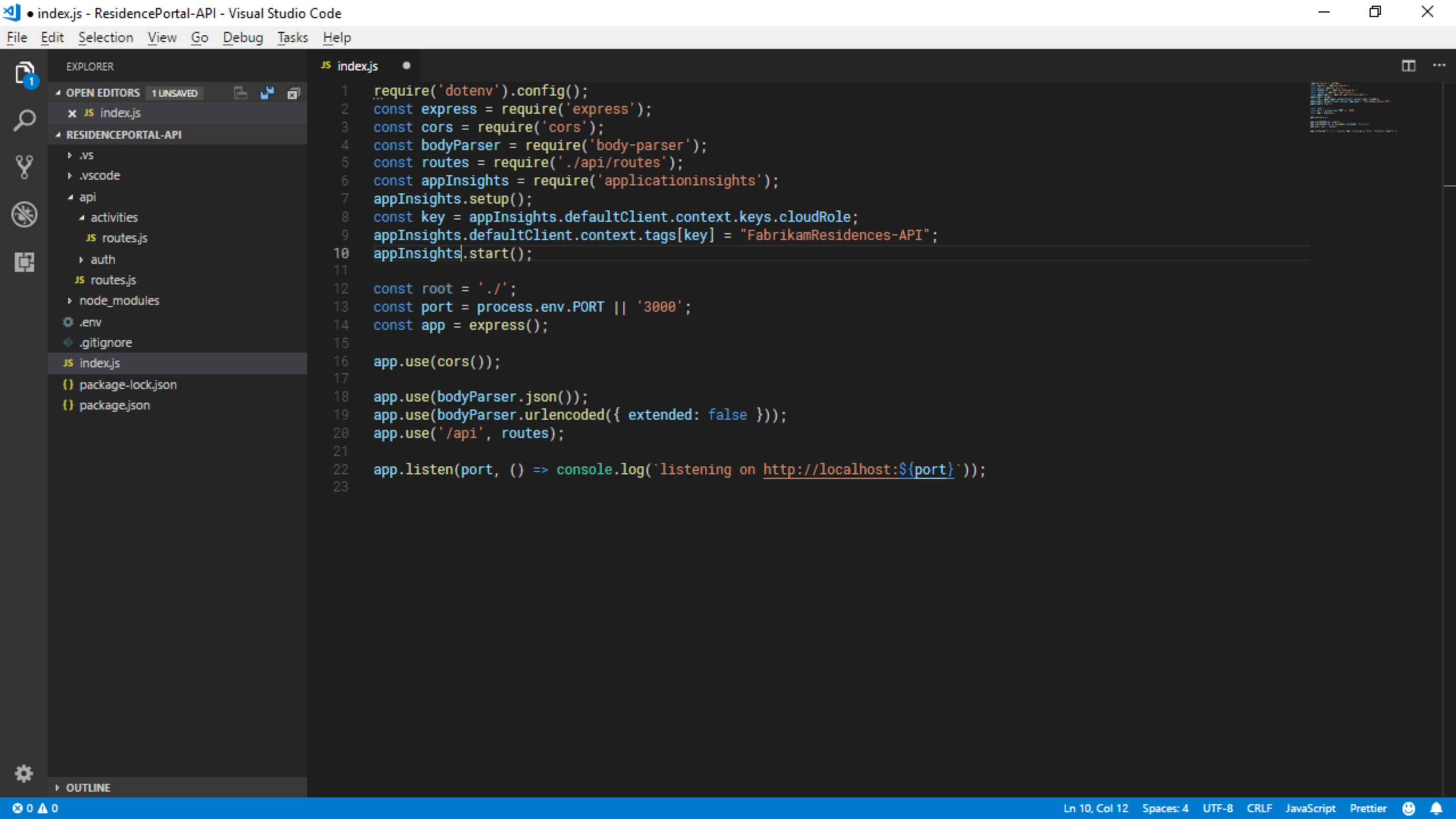
Enable Correlation

```
AppInsights.downloadAndSetup({
  instrumentationKey: environment.appInsights.instrumentationKey,
  enableCorsCorrelation: true
});
```

Demo

Adding Configuration Changes


index.js - ResidencePortal-API - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS
index.js
RESIDENCEPORTAL-API
.vs
.vscode
api
activities
routes.js
auth
routes.js
node_modules
.env
.gitignore
index.js
package-lock.json
package.json
OUTLINE
index.js
require('dotenv').config();
const express = require('express');
const cors = require('cors');
const bodyParser = require('body-parser');
const routes = require('./api/routes');
const appInsights = require('applicationinsights');
appInsights.setup().start();
const root = './';
const port = process.env.PORT || '3000';
const app = express();
app.use(cors());
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use('/api', routes);
app.listen(port, () => console.log(`listening on http://localhost:${port}`));
Ln 20, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript
Prettier


● index.js - ResidencePortal-API - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS 1 UNSAVED
index.js
RESIDENCEPORTAL-API
.vs
.vscode
api
activities
routes.js
auth
routes.js
node_modules
.env
.gitignore
index.js
package-lock.json
package.json
OUTLINE
index.js
require('dotenv').config();
const express = require('express');
const cors = require('cors');
const bodyParser = require('body-parser');
const routes = require('./api/routes');
const appInsights = require('applicationinsights');
appInsights.setup();
const key = appInsights.defaultClient.context.keys.cloudRole;
appInsights.defaultClient.context.tags[key] = "FabrikamResidences-API";
appInsights.start();
const root = './';
const port = process.env.PORT || '3000';
const app = express();
app.use(cors());
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use('/api', routes);
app.listen(port, () => console.log(`listening on http://localhost:${port}`));
0 0
Ln 10, Col 12 Spaces: 4 UTF-8 CRLF JavaScript Prettier

app.component.ts - ResidencePortal-UI - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS
- app.component.ts src\app

RESIDENCEPORTAL-UI
- interceptors
- login
- app-routing.module.ts
- app.component.css
- app.component.html
- app.component.spec.ts
- app.component.ts
- app.module.ts
- assets
- environments
  - environment.prod.ts
  - environment.ts
- browserslist
- favicon.ico
- index.html
- karma.conf.js
- main.ts
- polyfills.ts
- styles.css
- test.ts
- tsconfig.app.json
- tsconfig.spec.json
- tslint.json
- .editorconfig
- .gitignore
- angular.json
- package-lock.json
- package.json
- proxy.conf.json
- README.md

OUTLINE

app.component.ts

```
import { Component } from '@angular/core';
import { AuthService } from './auth/auth.service';
import { User } from './auth/user';
import { AppInsights } from 'applicationinsights-js';
import { environment } from '../environments/environment';

@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) {
    if (!AppInsights.config) {
      AppInsights.downloadAndSetup({instrumentationKey: environment.appInsights.instrumentationKey});
    }
  }
}
```

0 0 Ln 25, Col 1 Spaces: 2 UTF-8 LF TypeScript 2.9.2 Prettier: ✓

• app.component.ts - ResidencePortal-UI - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS 1 UNSAVED
app.component.ts src\app
RESIDENCEPORTAL-UI
interceptors
login
app-routing.module.ts
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
assets
environments
environment.prod.ts
environment.ts
browserslist
favicon.ico
index.html
karma.conf.js
main.ts
polyfills.ts
styles.css
test.ts
tsconfig.app.json
tsconfig.spec.json
tslint.json
.editorconfig
.gitignore
angular.json
package-lock.json
package.json
proxy.conf.json
README.md
OUTLINE
app.component.ts
import { Component } from '@angular/core';
import { AuthService } from './auth/auth.service';
import { User } from './auth/user';
import { AppInsights } from 'applicationinsights-js';
import { environment } from '../environments/environment';
@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) {
if (!AppInsights.config) {
AppInsights.downloadAndSetup({
instrumentationKey: environment.appInsights.instrumentationKey,
enableCorsCorrelation: true
});
}
}
}
Ln 24, Col 10 Spaces: 2 UTF-8 LF TypeScript 2.9.2 Prettier:

Application map - Micr

https://portal.azure.com/#@jeffhoppertech.onmicrosoft.com/resource/subscriptions/034d4fd0-6b8d-4a74-8fb4-b16cb5d2b97d/resource

Microsoft Azure

Search resources, services, and docs

jeff@hoppertech.net
DEFAULT DIRECTORY

Home > FabrikamResidences - Application map

FabrikamResidences - Application map

Application Insights - Last hour - FabrikamResidences

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

INVESTIGATE

- Application map
- Smart Detection
- Live Metrics Stream
- Metrics Explorer
- Metrics (preview)
- Search
- Availability

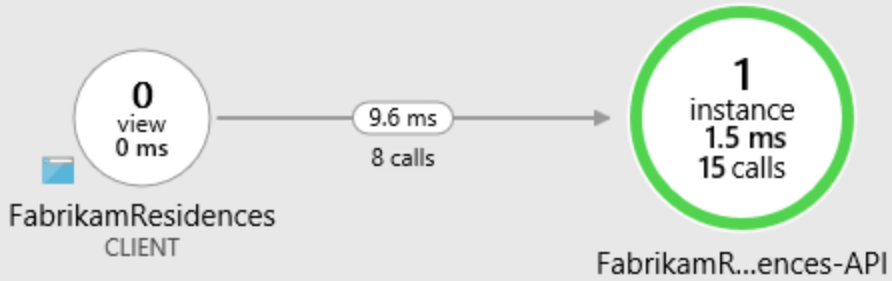
Time range = **Last hour**

Feedback

Learn more

Refresh

Update map components



Instrumentation API

API summary

The API is uniform across all platforms, apart from a few small variations.

Method	Used for
`TrackPageView`	Pages, screens, blades, or forms.
`TrackEvent`	User actions and other events. Used to track user behavior or to monitor performance.
`TrackMetric`	Performance measurements such as queue lengths not related to specific events.
`TrackException`	Logging exceptions for diagnosis. Trace where they occur in relation to other events and examine stack traces.
`TrackRequest`	Logging the frequency and duration of server requests for performance analysis.
`TrackTrace`	Diagnostic log messages. You can also capture third-party logs.
`TrackDependency`	Logging the duration and frequency of calls to external components that your app depends on.

You can attach properties and metrics to most of these telemetry calls.

Instrumenting: Page Views

Get started

To use this SDK, you'll need a subscription to Microsoft Azure. Application Insights has a free subscription option. In the Azure Preview Portal, create new or open an existing Application Insights resource.

Initializing Application Insights JS SDK script

There are several ways to initialize Aplication Insights.

	Dynamic loading. JS script tag is inserted in the head of the page. This is the recommended approach as our CDN is getting frequent updates.	**Static loading. You are responsible for including JS script tag or bundling the script with your other scripts.**
Using initialization `snippet`	Dynamic loading with snippet This is default approach used in a new ASP.NET application created in Visual Studio. Use this for MVC applications.	Host AI JS SDK and initialize statically. Cordova applications where you would like to embed scripts into your application for faster loading is an example of when you would use this approach.
Using module import	Dynamic loading using module import. This is the recommended approach for modern modular applications.	TBD

Use JS `snippet` and initialize dynamically (download full Application Insights script from CDN)

Use this method for an MVC application. Get "code to monitor my web pages" from the Quick Start page, and insert it in the head of your web pages. Application Insights script will be downloaded from CDN or you can override the script hosting

Use JS `snippet` and initialize dynamically (download full Application Insights script from CDN)

Use this method for an MVC application. Get "code to monitor my web pages" from the Quick Start page, and insert it in the head of your web pages. Application Insights script will be downloaded from CDN or you can override the script hosting location by specifying `url` parameter in the config.

```
<script type="text/javascript">
    var appInsights=window.appInsights||function(a){
        function b(a){c[a]=function(){var b=arguments;c.queue.push(function(){c[a].apply(c,b)})}}var c={config:a},d=c
    }({
        instrumentationKey: "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxx"
    });

    window.appInsights=appInsights,appInsights.queue&&0===appInsights.queue.length&&appInsights.trackPageView();
</script>
```

Learn more.

Import as a module and initialize dynamically (download full Application Insights script from CDN)

Use this method for a modern JS application that is using modules. Just like in `snippet` scenario the full script will be downloaded from CDN.

- Obtain instrumentation key from your Application Insights resource
- Install applicationinsights-js with npm
  `npm install applicationinsights-js`

Single-Page Application (SPA)

A web application or web site that interacts with the user by dynamically rewriting the current page rather than loading entire new pages from a server.

https://en.wikipedia.org/wiki/Single-page_application

Demo

Instrumenting: Page Views in Angular

app-routing.module.ts - ResidencePortal-UI - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS
- app.component.ts src\app
- app-routing.module.ts src\app

RESIDENCEPORTAL-UI
- .vscode
- @types
- dist
- e2e
- node_modules
- src
  - app
    - activities
      - activities.component.css
      - activities.component.html
      - activities.component.spec.ts
      - activities.component.ts
      - activity.service.spec.ts
      - activity.service.ts
      - activity.ts
    - auth
    - home
    - login
    - app-routing.module.ts
    - app.component.css
    - app.component.html
    - app.component.spec.ts
    - app.component.ts
    - app.module.ts
  - assets
  - environments
    - environment.prod.ts
    - environment.ts
- browserslist

OUTLINE

app.component.ts | app-routing.module.ts

```ts
import { NgModule, Injectable } from '@angular/core';
import {
  Routes,
  RouterModule,
  CanActivate,
  ActivatedRouteSnapshot,
  RouterStateSnapshot
} from '@angular/router';

import { HomeComponent } from './home/home.component';
import { ActivitiesComponent } from './activities/activities.component';
import { LoginComponent } from './login/login.component';

import { AuthGuardService } from './auth/app-guard.service';

const routes: Routes = [
  { path: '', redirectTo: '/home', pathMatch: 'full' },
  { path: 'home', component: HomeComponent },
  { path: 'activities', component: ActivitiesComponent, canActivate: [AuthGuardService] },
  { path: 'login', component: LoginComponent }
];

@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule],
  providers: [AuthGuardService]
})
export class AppRoutingModule {}

```

0 0 | Debug APP (ResidencePortal-UI) | Ln 1, Col 1 | Spaces: 2 | UTF-8 | LF | TypeScript | 2.9.2 | Prettier: ✓

activities.component.ts - ResidencePortal-UI - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS
- app.component.ts src\app
- activities.component.ts src\app\activit...

RESIDENCEPORTAL-UI
- .vscode
- @types
- dist
- e2e
- node_modules
- src
  - app
    - activities
      - activities.component.css
      - activities.component.html
      - activities.component.spec.ts
      - activities.component.ts
      - activity.service.spec.ts
      - activity.service.ts
      - activity.ts
    - auth
    - home
    - login
  - app-routing.module.ts
  - app.component.css
  - app.component.html
  - app.component.spec.ts
  - app.component.ts
  - app.module.ts
  - assets
  - environments
    - environment.prod.ts
    - environment.ts
- browserslist

OUTLINE

app.component.ts | activities.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Activity } from './activity';
import { ActivityService } from './activity.service';
import { User } from '../auth/user';
import { AuthService } from '../auth/auth.service';
import { AppInsights } from 'applicationinsights-js';

@Component({
  selector: 'app-activities',
  templateUrl: './activities.component.html',
  styleUrls: ['./activities.component.css']
})
export class ActivitiesComponent implements OnInit {
  addingActivity = false;
  activities: any = [];
  selectedActivity: Activity;

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

  constructor(
    private activityService: ActivityService,
    private auth: AuthService
  ) { }

  ngOnInit() {
    this.getActivities();
  }

  cancel() {
    this.addingActivity = false;
    this.selectedActivity = null;
  }

  deleteActivity(activity: Activity) {
    this.activityService.deleteActivity(activity).subscribe(res => {
      this.getActivities();
      if (this.selectedActivity === activity) {
        this.selectedActivity = null;
```

0 0 Debug APP (ResidencePortal-UI) Ln 30, Col 1 Spaces: 2 UTF-8 LF TypeScript 2.9.2 Prettier:

• activities.component.ts - ResidencePortal-UI - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS 1 UNSAVED
app.component.ts src\app
activities.component.ts src\app\activit...
RESIDENCEPORTAL-UI
.vscode
@types
dist
e2e
node_modules
src
app
activities
activities.component.css
activities.component.html
activities.component.spec.ts
activities.component.ts
activity.service.spec.ts
activity.service.ts
activity.ts
auth
home
login
app-routing.module.ts
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
assets
environments
environment.prod.ts
environment.ts
browserslist
OUTLINE
app.component.ts
activities.component.ts
import { Component, OnInit } from '@angular/core';
import { Activity } from './activity';
import { ActivityService } from './activity.service';
import { User } from '../auth/user';
import { AuthService } from '../auth/auth.service';
import { AppInsights } from 'applicationinsights-js';
@Component({
selector: 'app-activities',
templateUrl: './activities.component.html',
styleUrls: ['./activities.component.css']
})
export class ActivitiesComponent implements OnInit {
addingActivity = false;
activities: any = [];
selectedActivity: Activity;
get user(): User {
return this.auth.getAuthenticatedUser();
}
constructor(
private activityService: ActivityService,
private auth: AuthService
) {
AppInsights.trackPageView('Activities', window.location.href);
}
ngOnInit() {
this.getActivities();
}
cancel() {
this.addingActivity = false;
this.selectedActivity = null;
}
deleteActivity(activity: Activity) {
this.activityService.deleteActivity(activity).subscribe(res => {
this.getActivities();
0 0 Debug APP (ResidencePortal-UI)
Ln 26, Col 67 Spaces: 2 UTF-8 LF TypeScript 2.9.2 Prettier:

Application map - Micr

https://portal.azure.com/#@jeffhoppertech.onmicrosoft.com/resource/subscriptions/034d4fd0-6b8d-4a74-8fb4-b16cb5d2b97d/resource

Microsoft Azure

Search resources, services, and docs

jeff@hoppertech.net
DEFAULT DIRECTORY

Home > FabrikamResidences - Application map

FabrikamResidences - Application map

Application Insights - Last 30 minutes - FabrikamResidences

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

INVESTIGATE

- Application map
- Smart Detection
- Live Metrics Stream
- Metrics Explorer
- Metrics (preview)
- Search
- Availability

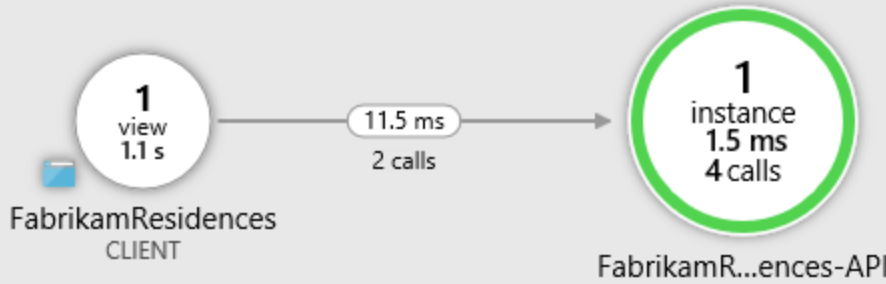
Time range = **Last 30 minutes**

Feedback

Learn more

Refresh

Update map components



Search - Microsoft Azur

https://portal.azure.com/#@jeffhoppertech.onmicrosoft.com/resource/subscriptions/034d4fd0-6b8d-4a74-8fb4-b16cb5d2b97d/resource

Microsoft Azure

Search resources, services, and docs

jeff@hoppertech.net
DEFAULT DIRECTORY

Home > FabrikamResidences - Search

FabrikamResidences - Search

Application Insights

Search (Ctrl+/)

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems

INVESTIGATE

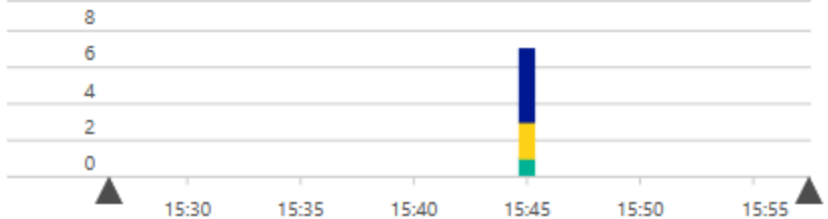
- Application map
- Smart Detection
- Live Metrics Stream
- Metrics Explorer
- Metrics (preview)
- Search
- Availability

Time range | Filters | Refresh | Reset | Analytics | More

Search

Filtered on: Trace x | Request x | Page View x | Custom Event x | Exception x | Dependency x | Availability x

7 total results between **07/24/2018 15:27** and **07/24/2018 15:57**



REQUEST	DEPENDENCY	VIEW	EXCEPTION	TRACE	EVENT	AVAIL
4	2	1	0	0	0	0

Results | Grouped results

07/24/2018 15:45:24 - PAGE VIEW
Activities
Page view URL: http://localhost:4200/activities Page view load time: 1.08 s
Browser version: Edge 17.17134

07/24/2018 15:45:24 - DEPENDENCY (AJAX)
localhost:3000
Operation name: /home Dependency duration: 11 ms Successful call: true
Dependency name: GET /api/activities

Instrumenting: User Interactions

setAuthenticatedUserContext

```
setAuthenticatedUserContext(authenticatedUserId: string, accountId?: string, storeInCookie = false)
```

Set the authenticated user id and the account id. Use this when you have identified a specific signed-in user. Parameters must not contain spaces or ,;=|

The method will only set the `authenticatedUserId` and `accountId` for all events in the current page view. To set them for all events within the whole session, you should either call this method on every page view or set `storeInCookie = true`.

Parameter	Description
`authenticatedUserId`	An id that uniquely identifies a user of your app. No spaces, comma, semicolon, equals or vertical bar.
`accountId`	An optional account id, if your app groups users into accounts. No spaces, comma, semicolon, equals or vertical bar.

In the portal, this will add to the count of authenticated users. Authenticated users provide a more reliable count of the number of real users than the count of anonymous users.

The authenticated user id will be available as part of the context of the telemetry sent to the portal, so that you can filter and search on it. It will also be saved as a cookie and sent to the server, where the server SDK (if installed) will attach it to server telemetry.

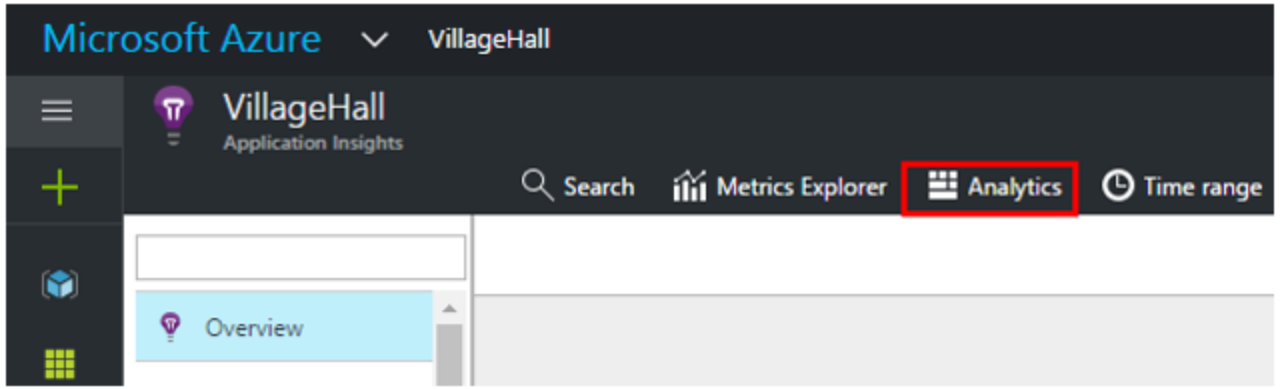
clearAuthenticatedUserContext

```
clearAuthenticatedUserContext()
```

Analytics in Application Insights

02/08/2018 • 2 minutes to read • Contributors all

Analytics is the powerful search and query tool of Application Insights. Analytics is a web tool so no setup is required. If you've already configured Application Insights for one of your apps then you can analyze your app's data by opening Analytics from your app's overview blade.



You can also use the Analytics playground which is a free demo environment with a lot of sample data.

Demo

Instrumenting: User Interactions

login.component.ts - ResidencePortal-UI - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS
login.component.ts src\app\login
RESIDENCEPORTAL-UI
.vscode
@types
dist
e2e
node_modules
src
app
activities
auth
app-guard.service.ts
auth.service.ts
user.ts
home
login
login.component.css
login.component.html
login.component.spec.ts
login.component.ts
app-routing.module.ts
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
assets
environments
browserslist
favicon.ico
index.html
karma.conf.js
OUTLINE
login.component.ts
1 import { Component, OnInit, Output } from '@angular/core';
2 import { Router } from '@angular/router';
3 import { AuthService } from '../auth/auth.service';
4 import { User } from '../auth/user';
5
6
7 @Component({
8 selector: 'app-login',
9 templateUrl: './login.component.html',
10 styleUrls: ['./login.component.css']
11 })
12 export class LoginComponent implements OnInit {
13 username: string;
14 password: string;
15
16 get user(): User { return this.auth.getAuthenticatedUser(); }
17
18 constructor(public auth: AuthService, private router: Router) { }
19
20 ngOnInit() {}
21
22 login() {
23 this.auth.login(this.username, this.password);
24 }
25
26 logout() {
27 this.auth.logout();
28 }
29 }
30
0 0 Debug APP (ResidencePortal-UI)
Ln 1, Col 1 Spaces: 2 UTF-8 LF TypeScript 2.9.2 Prettier:

• activities.component.ts - ResidencePortal-UI - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS 1 UNSAVED
auth.service.ts src\app\auth
activities.component.html src\app\ac...
activities.component.ts src\app\activit...
RESIDENCEPORTAL-UI
.vscode
@types
dist
e2e
node_modules
src
app
activities
activities.component.css
activities.component.html
activities.component.spec.ts
activities.component.ts
activity.service.spec.ts
activity.service.ts
activity.ts
auth
app-guard.service.ts
auth.service.ts
user.ts
home
login
login.component.css
login.component.html
login.component.spec.ts
login.component.ts
app-routing.module.ts
app.component.css
app.component.html
OUTLINE
auth.service.ts
activities.component.html
activities.component.ts
) {
AppInsights.trackPageView('Activities', window.location.href);
}
ngOnInit() {
this.getActivities();
}
cancel() {
this.addingActivity = false;
this.selectedActivity = null;
}
deleteActivity(activity: Activity) {
this.activityService.deleteActivity(activity).subscribe(res => {
this.getActivities();
if (this.selectedActivity === activity) {
this.selectedActivity = null;
}
});
AppInsights.trackEvent('EmployeeMadeEdit');
}
getActivities() {
return this.activityService.getActivities().subscribe(activities => {
this.activities = activities;
});
}
enableAddMode() {
this.addingActivity = true;
this.selectedActivity = new Activity();
}
onSelect(activity: Activity) {
this.addingActivity = false;
this.selectedActivity = activity;
}
save() {
0 0
Debug APP (ResidencePortal-UI)
Ln 45, Col 48
Spaces: 2
UTF-8
LF
TypeScript
2.9.2
Prettier:

Instrumenting: Trace Logs and Exceptions

Logging Schools of Thought

Too Many Logs

Too many log messages can also lead to difficulty in reading a log file and identifying the relevant information when a problem does occur.
https://stackify.com/9-logging-sins-java

Resist the tendency to log everything.
https://blog.codinghorror.com/the-problem-with-logging

Log Everything

Log Everything First.
https://blog.logdna.com/2016/06/22/logeverything

Log Everything All the Time.
http://highscalability.com/log-everything-all-time

Application Insights is an extensible Application Performance Management (APM) service for web developers building and managing apps on multiple platforms.

Region: East US

Currency: US Dollar ($)

Pricing details

Application Insights is billed based on the volume of telemetry data that your application sends and the number of web tests that you choose to run. The telemetry data is billed per Azure Log Analytics data ingestion rates.

FEATURE	DETAILS
Data Ingestion	See Azure Log Analytics pricing
Data Retention	90 days

Log Analytics enables you to collect and analyze monitoring data generated from multiple sources in your cloud and on-premises environments.

Region: East US

Currency: US Dollar ($)

Pricing details

Log analytics is billed per gigabyte (GB) of data ingested into the service.

FEATURE	FREE UNITS INCLUDED	PRICE
Data Ingestion	5 GB/month[1]	$2.30 per GB
Data Retention	31 days[2]	$0.10/GB/month

[1]The first 5 GB of data ingested to the Azure Log Analytics service every month is offered free.

API summary

The API is uniform across all platforms, apart from a few small variations.

Method	Used for
`TrackPageView`	Pages, screens, blades, or forms.
`TrackEvent`	User actions and other events. Used to track user behavior or to monitor performance.
`TrackMetric`	Performance measurements such as queue lengths not related to specific events.
`TrackException`	Logging exceptions for diagnosis. Trace where they occur in relation to other events and examine stack traces.
`TrackRequest`	Logging the frequency and duration of server requests for performance analysis.
`TrackTrace`	Diagnostic log messages. You can also capture third-party logs.
`TrackDependency`	Logging the duration and frequency of calls to external components that your app depends on.

You can attach properties and metrics to most of these telemetry calls.

Get a TelemetryClient instance

Get an instance of `TelemetryClient` (except in JavaScript in webpages):

C#

Copy

```
private TelemetryClient telemetry = new TelemetryClient();
```

Visual Basic

Copy

```
Private Dim telemetry As New TelemetryClient
```

Java

Copy

```
private TelemetryClient telemetry = new TelemetryClient();
```

Node.js

Copy

```
var telemetry = applicationInsights.defaultClient;
```

Filter by title

- Configure
- Analyze
- Automate
- Develop
  - API for custom events and metrics
  - Track custom operations in .NET SDK
  - Filtering and preprocessing telemetry
  - Sampling
- Manage
- Export
- Secure
- Troubleshoot
- Samples
- Reference
- Resources

Download PDF

```
} catch (Exception ex) {
    telemetry.trackException(ex);
}
```

JavaScript

Copy

```
try
{
   ...
}
catch (ex)
{
   appInsights.trackException(ex);
}
```

Node.js

Copy

```
try
{
   ...
}
catch (ex)
{
   telemetry.trackException({exception: ex});
}
```

TrackTrace

Use TrackTrace to help diagnose problems by sending a "breadcrumb trail" to Application Insights. You can send chunks of diagnostic data and inspect them in Diagnostic Search.

In .NET Log adapters use this API to send third-party logs to the portal.

In Java for Standard loggers like Log4J, Logback use Application Insights Log4j or Logback Appenders to send third-party logs to the portal.

C#

Copy

```
telemetry.TrackTrace(message, SeverityLevel.Warning, properties);
```

Java

Copy

```
telemetry.trackTrace(message, SeverityLevel.Warning, properties);
```

Node.js

Copy

```
telemetry.trackTrace({message: message, severity:applicationInsights.Contracts.Se
```

TrackTrace

Use TrackTrace to help diagnose problems by sending a "breadcrumb trail" to Application Insights. You can send chunks of diagnostic data and inspect them in Diagnostic Search.

In .NET Log adapters use this API to send third-party logs to the portal.

In Java for Standard loggers like Log4J, Logback use Application Insights Log4j or Logback Appenders to send third-party logs to the portal.

C#

Copy

```
telemetry.TrackTrace(message, SeverityLevel.Warning, properties);
```

Java

Copy

```
telemetry.trackTrace(message, SeverityLevel.Warning, properties);
```

Node.js

Copy

```
({message: message, severity:applicationInsights.Contracts.SeverityLevel.Warning,
```

Demo

Instrumenting: Trace and Exceptions

Summary

Make Configuration Changes

Instrumentation API

- Page Views
- User Interactions
- Logging