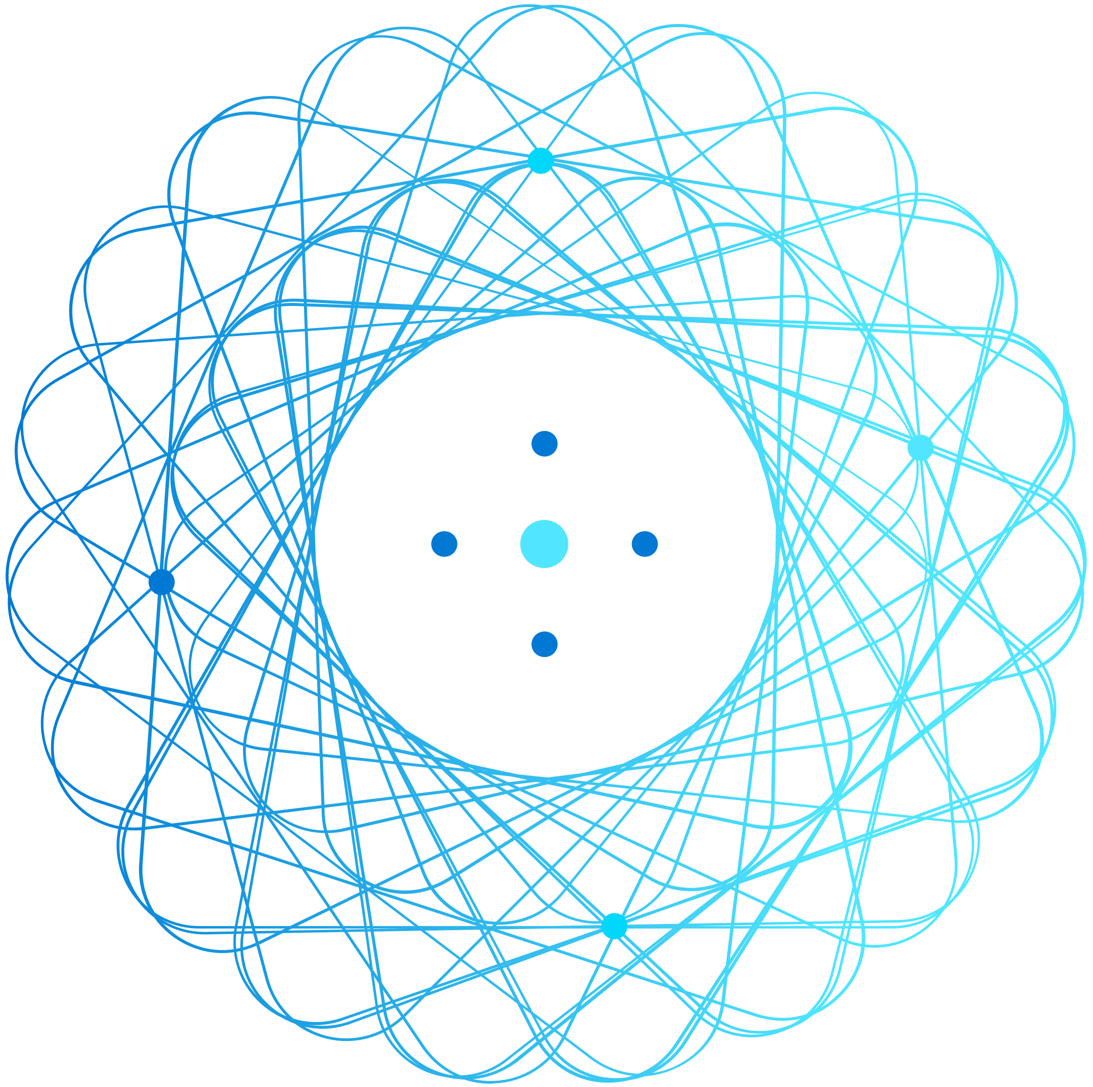
Microsoft Azure

AZ-220T01 Module 11: Develop with Azure Digital Twins



Lesson 1: Learning objectives

Module 11 – Learning objectives

Describe the working components of an Azure Digital Twins (ADT) solution

Explain how to create and configure an ADT instance

Explain how to create, query, and manage the ADT graph

Explain how to implement ADT data ingress from IoT hub and data egress from ADT for downstream business analysis

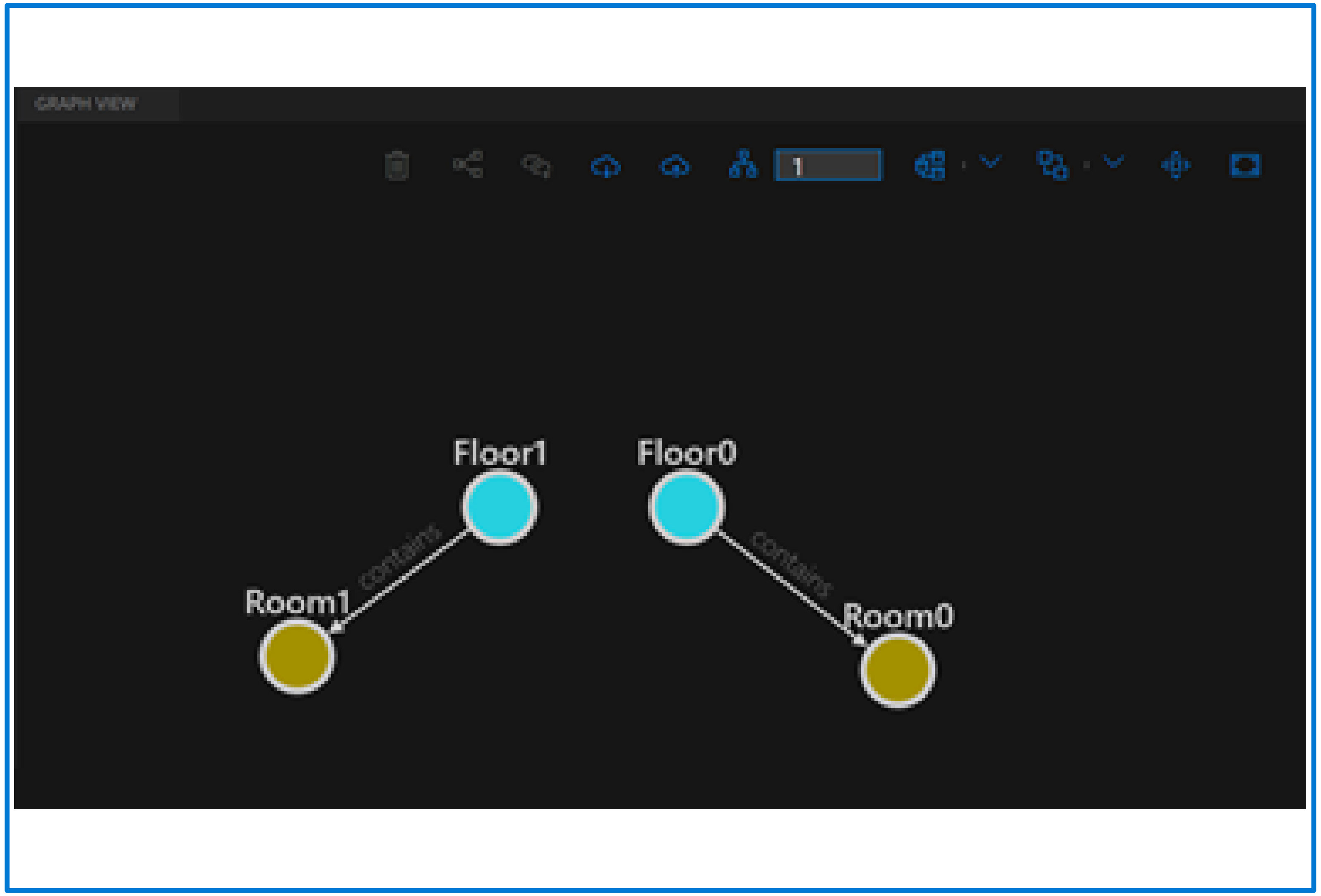
Describe how to monitor and troubleshoot ADT

Lesson 2: Introduction to Azure Digital Twins

Get started with Azure Digital Twins

Azure Digital Twins is a platform as a service (PaaS) offering that enables the creation of knowledge graphs

The knowledge graph is composed of interconnected digital entities that combine to represent a larger, interactive digital environment

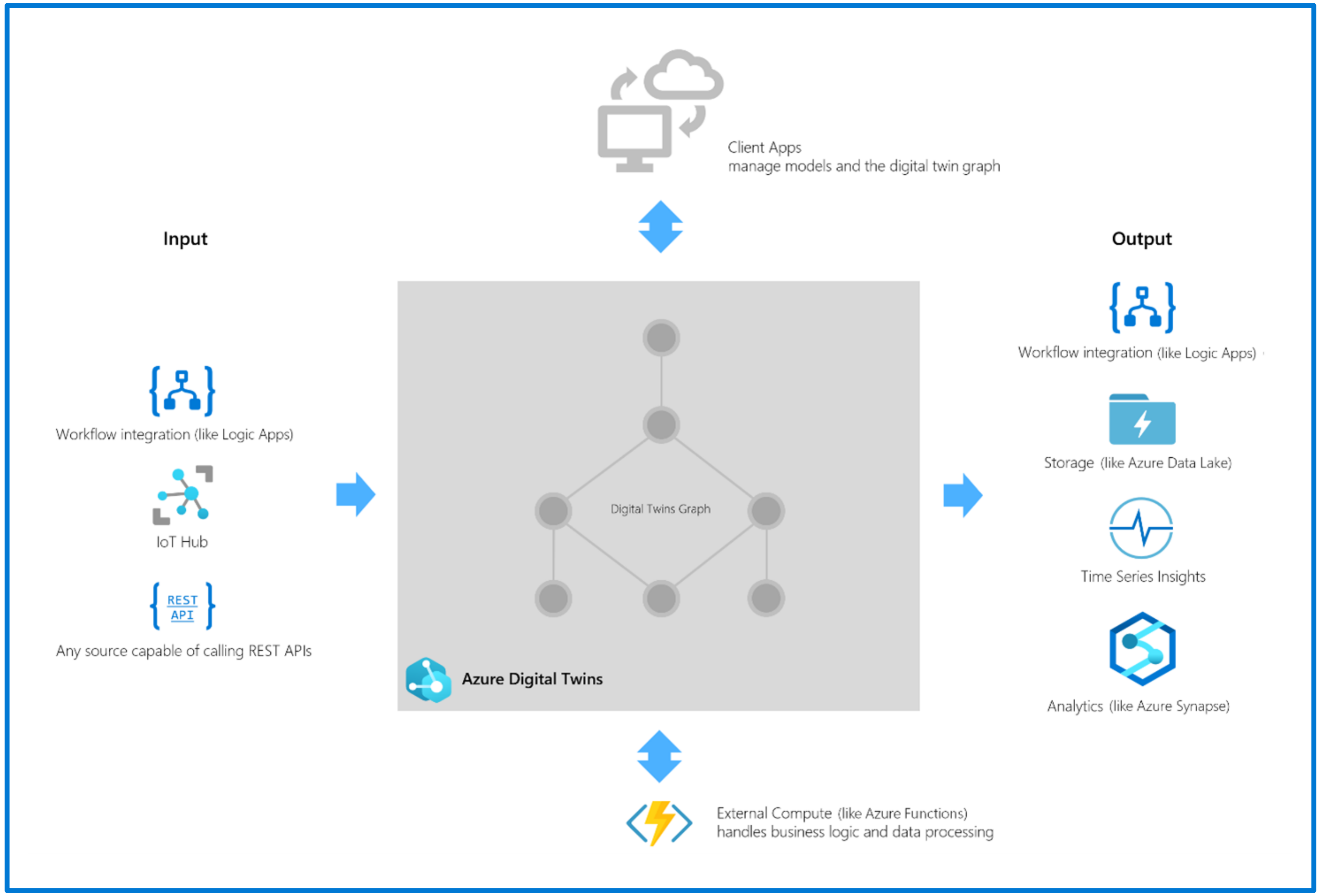


Examine an ADT solution

Azure Digital Twins is typically used together with other services to create flexible, connected solutions that use your data in a variety of workflows

Azure Digital Twins can receive data (Input) from upstream services such as IoT Hub or Logic Apps, which are used to deliver telemetry and notifications

Azure Digital Twins can also route data (Output) to downstream services, such as Time Series Insights or Azure Maps, for storage, workflow integration, analytics, and more


Client Apps
manage models and the digital twin graph
Input
Workflow integration (like Logic Apps)
IoT Hub
REST API
Any source capable of calling REST APIs
Digital Twins Graph
Azure Digital Twins
Output
Workflow integration (like Logic Apps)
Storage (like Azure Data Lake)
Time Series Insights
Analytics (like Azure Synapse)
External Compute (like Azure Functions)
handles business logic and data processing

Examine an ADT solution

Design and implementation scenario

Evaluation: Azure Digital Twins supports the ability to aggregate and combine data from multiple sources in a single, securely accessible location.

Design Flexibility: Azure Digital Twins supports any industry vertical investing in IoT and has the flexibility to connect the inputs and outputs that an individual company requires.

Implementation: Azure Digital Twins uses a robust event system to build dynamic business logic and data processing.

Results: Azure Digital Twins integration with analytics and AI services help you to track the past and predict the future.

Get started with digital twin models

The top-level of a model definition is called an Interface. The Interface encapsulates the entire model, and may contain zero, one, or many of each of the following fields:

Property - Properties are data fields that represent the state of an entity (like the properties in many object-oriented programming languages)

Telemetry - Telemetry fields represent measurements or events, and are often used to describe device sensor readings.

Component - Components allow you to build your model interface as an assembly of other interfaces.

Relationship - Relationships let you represent how a digital twin can be involved with other digital twins.

Examine the Digital Twins Definition Language

Models for Azure Digital Twins are defined using the Digital Twins Definition language (DTDL), which is based on JSON-LD. The model contains the following fields.

Fields	Description
@id	An identifier for the model. Must be in the following format: `dtmi:<domain>:<unique model identifier>;<model version number>`
@type	Identifies the kind of information being described: Interface, Property, Telemetry, Relationship, or Component.
@context	Sets the context for the JSON document. Models should use the following: `dtmi:dtdl:context;2`
displayName	(optional) Allows you to give the model a friendly name if desired.
contents	All remaining interface (model) data is placed here, as an array of attribute definitions. Each attribute must provide an @type to identify the type of interface information it describes, and then a set of properties that define the actual attribute

Examine the Digital Twins Definition Language

The following DTDL example for a Planet includes properties and telemetry as well as a relationship and component

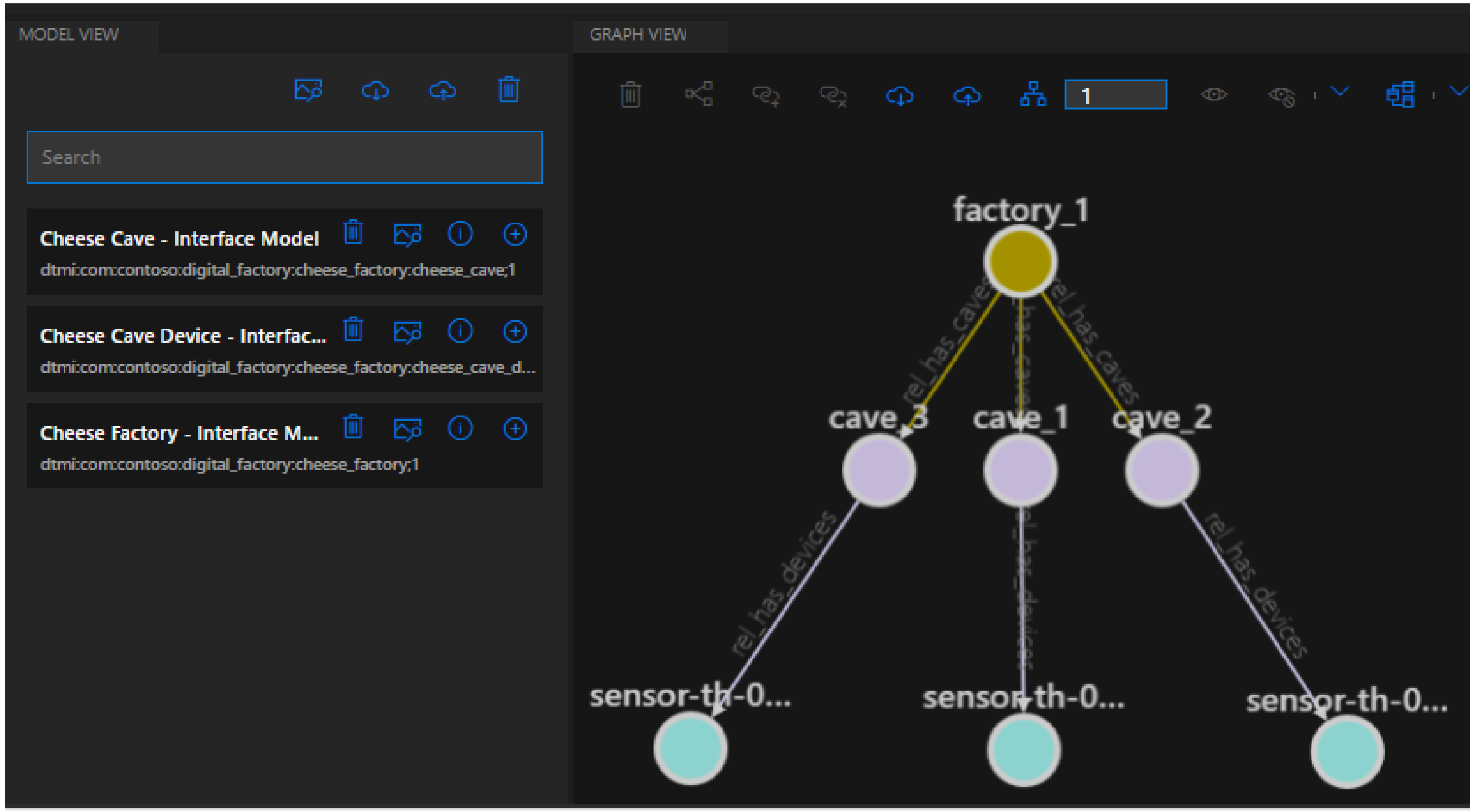
```
[
  {
    "@id": "dtmi:com:contoso:Planet;1",
    "@type": "Interface",
    "@context": "dtmi:dtdl:context;2",
    "displayName": "Planet",
    "contents": [
      {
        "@type": "Property",
        "name": "name",
        "schema": "string"
      },
      {
        "@type": "Property",
        "name": "mass",
        "schema": "double"
      },
      {
        "@type": "Telemetry",
        "name": "Temperature",
        "schema": "double"
      },
      {
        "@type": "Relationship",
        "name": "satellites",
        "target": "dtmi:com:contoso:Moon;1"
      },
      {
        "@type": "Component",
        "name": "deepestCrater",
        "schema": "dtmi:com:contoso:Crater;1"
      }
    ]
  },
  {
    "@id": "dtmi:com:contoso:Crater;1",
    "@type": "Interface",
    "@context": "dtmi:dtdl:context;2"
  },
  {
    "@id": "dtmi:com:contoso:Moon;1",
    "@type": "Interface",
    "@context": "dtmi:dtdl:context;2"
  }
]
```

Examine digital twins and graph construction

The first step in adding a digital twin to ADT is to upload a model type to your ADT instance

After creating and uploading a model, you can create one or more instances of the model type; your digital twins

Digital twins are connected into a twin graph by their relationships (which must be defined as part of the model)



Cheese Cave - Interface Model
dtmi:com:contoso:digital_factory:cheese_factory:cheese_cave;1

Cheese Cave Device - Interfac...
dtmi:com:contoso:digital_factory:cheese_factory:cheese_cave_d...

Cheese Factory - Interface M...
dtmi:com:contoso:digital_factory:cheese_factory;1

Lesson 3: Introduction to ADT solution development

Get started with the ADT service and tools

The Azure Digital Twins service comes equipped with APIs for managing your ADT instance and its elements. Multi-purpose and ADT-specific tools are available for various stages of solution develop.

ADT Solution Stage/Area	Azure Portal	Azure CLI	SDKs (VSCode)	DTDL Validator	ADT-explorer	CSV (Excel)	Azure Functions	REST APIs (Postman)
Create/Configure ADT instance	x	x	x					x
Develop DTDL Model files			x	x				
Build Graph Environment		x	x		x	x		x
Query/Manage Graph Environment		x	x		x			x
Manage Data Ingress (upstream)	x						x	
Manage Data Egress (downstream)	x						x	

Note: The list of tools above is not intended to be a complete list of the tools that can be used to develop an ADT solution.

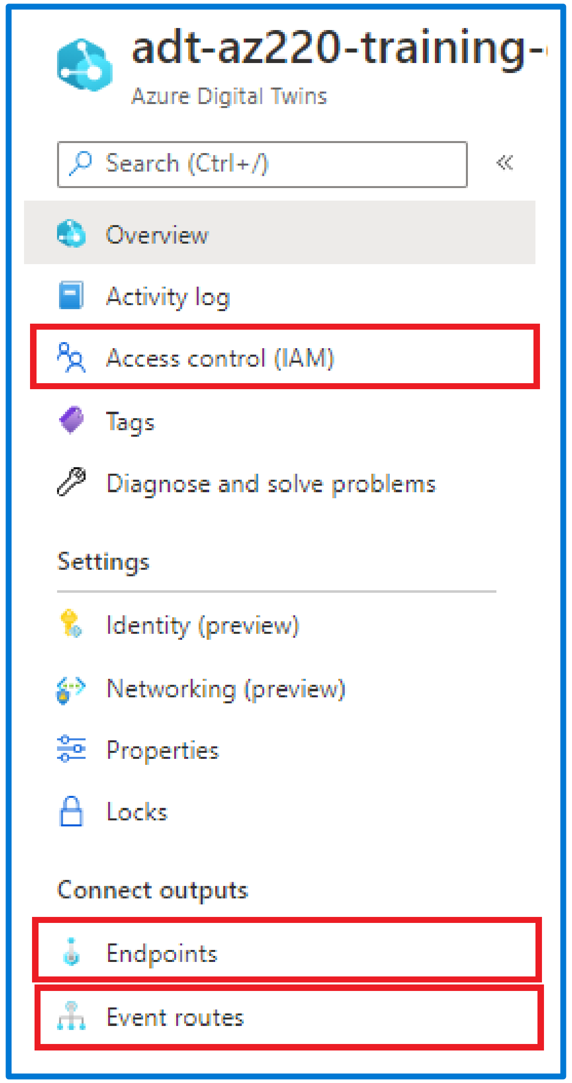
Examine ADT service configuration

To create an ADT instance, you must specify an Azure subscription, a resource group, a location, and a resource name.

Access control permissions must be configured for an ADT instance. The Azure Digital Twins Data Owner role is required for a user or app to access ADT data.

Endpoints are used to make ADT event data available to downstream services. An ADT instance supports the following Endpoint types: Event Grid, Event Hubs, and Service Bus.

Event routes specify which events generated by Azure Digital Twins are delivered to which endpoints. A routing filter is used to restrict the types of events being sent.



Get started with model management

Validate and upload - validate your models offline before uploading them

- Validation tools: DTDL Validator, DTDL Editor for Visual Studio Code
- Upload options: APIs, Azure CLI, SDKs, ADT-Explorer, ADT Model Uploader

Update and version - models uploaded to ADT cannot be edited

- You must upload a newer version to replace an older model
- The old model version(s) are used by existing digital twins until the twins are updated (patched)
- New digital twins can be created using any available version of a model

Removal – unused models can be removed (decommissioned or deleted)

- Twins that were created from a model that has been decommissioned can still be updated.
- Twins that were created from a model that has been deleted cannot be updated until they are assigned to another model. Twins can still be queried even after their model is deleted.

Explore the ADT APIs and Postman

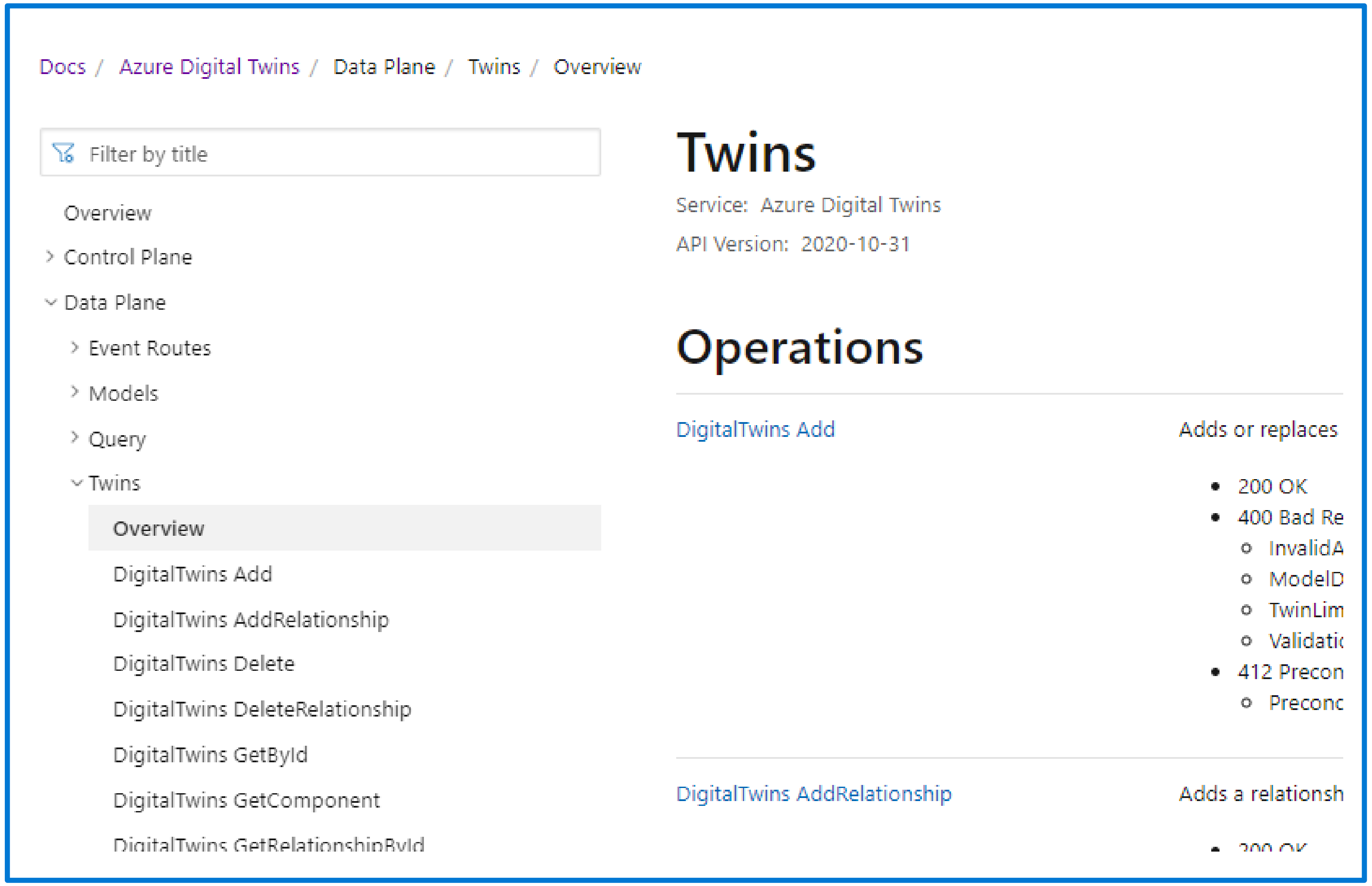
Azure Digital Twins API reference documentation

Control Plane APIs: used to manage your Azure Digital Twins instance. API categories:

- Check Name Availability
- Digital Twins Instance
- Endpoints
- Operations
- Private Endpoints

Data Plane APIs: used to manage elements of the ADT instance. API categories:

- Event Routes
- Models
- Query
- Twins



Docs / Azure Digital Twins / Data Plane / Twins / Overview

Filter by title

- Overview
- Control Plane
- Data Plane
  - Event Routes
  - Models
  - Query
  - Twins
    - Overview
    - DigitalTwins Add
    - DigitalTwins AddRelationship
    - DigitalTwins Delete
    - DigitalTwins DeleteRelationship
    - DigitalTwins GetById
    - DigitalTwins GetComponent
    - DigitalTwins GetRelationshipById

Twins

Service: Azure Digital Twins

API Version: 2020-10-31

Operations

DigitalTwins Add — Adds or replaces

- 200 OK
- 400 Bad Re
  - InvalidA
  - ModelD
  - TwinLim
  - Validatic
- 412 Precon
  - Preconc

DigitalTwins AddRelationship — Adds a relationsh

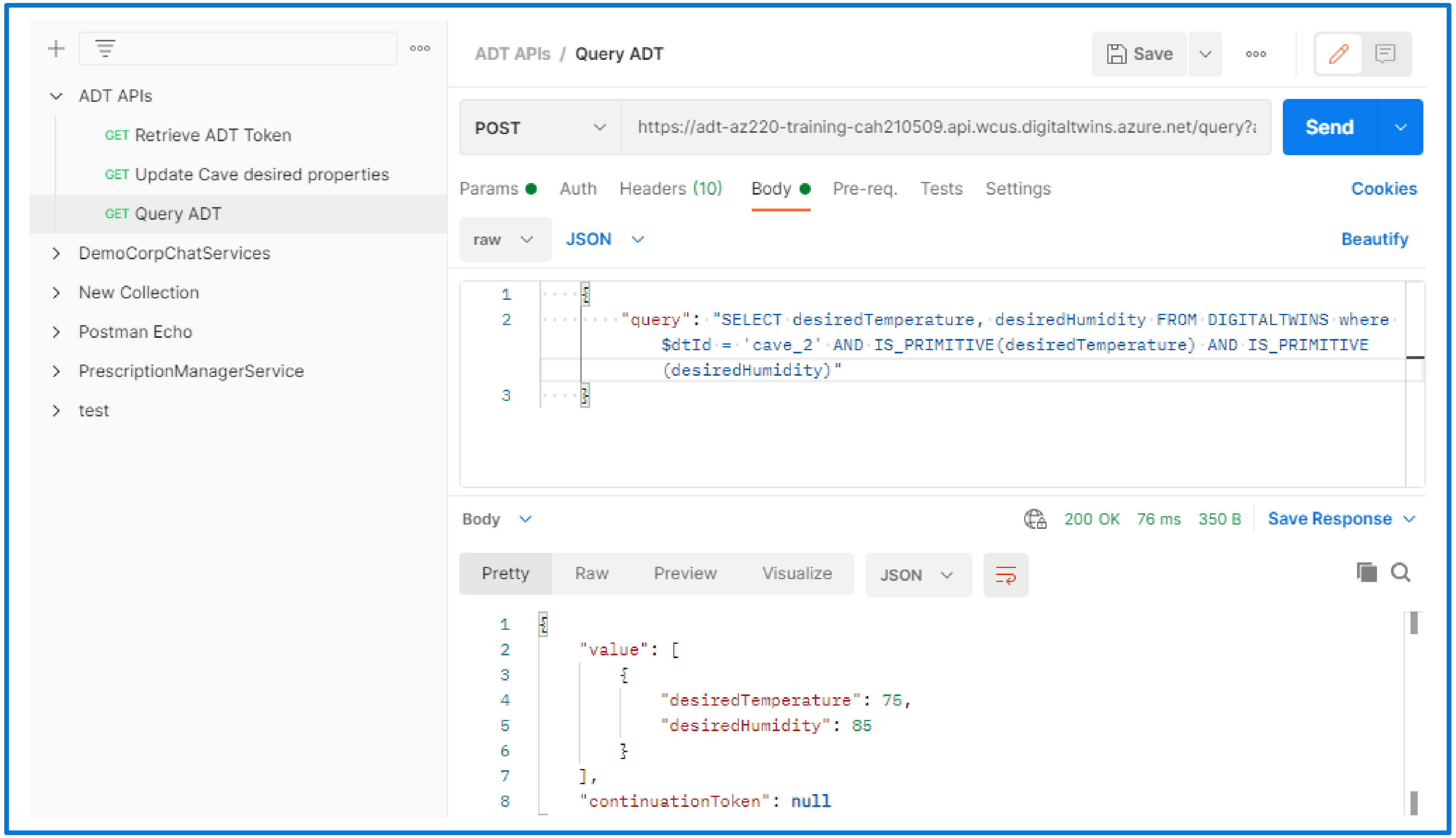
Explore the ADT APIs and Postman

Three steps to using Postman

Step 1: authorize Postman to make requests against the ADT APIs with a bearer token

Step 2: set up (or import) a collection of Postman requests for ADT

Step 3: edit the details of a request in the Postman collection run the request with the Send button



POST https://adt-az220-training-cah210509.api.wcus.digitaltwins.azure.net/query?a

```
{
    "query": "SELECT desiredTemperature, desiredHumidity FROM DIGITALTWINS where
        $dtId = 'cave_2' AND IS_PRIMITIVE(desiredTemperature) AND IS_PRIMITIVE
        (desiredHumidity)"
}
```

Body 200 OK 76 ms 350 B

```
{
    "value": [
        {
            "desiredTemperature": 75,
            "desiredHumidity": 85
        }
    ],
    "continuationToken": null
```

Get started with Azure CLI for ADT

The Azure CLI command set for ADT is part of the Azure IoT extension for Azure CLI

Azure CLI commands can be used for:

- Managing an ADT instance
- Managing models
- Managing digital twins
- Managing twin relationships
- Configuring endpoints
- Managing routes
- Configuring security via Azure role-based access control (Azure RBAC)

Azure CLI examples for ADT

Create an ADT instance

Azure CLI

```
az dt create -n {instance_name} -g {resouce_group}
```

Create a digital twin

Azure CLI

```
az dt twin create -n {instance_or_hostname} --dtmi "dtmi:com:example:Room;1" --twin-id {twin_id}
```

Delete a digital twin

Azure CLI

```
az dt twin delete -n {instance_or_hostname} --twin-id {twin_id}
```

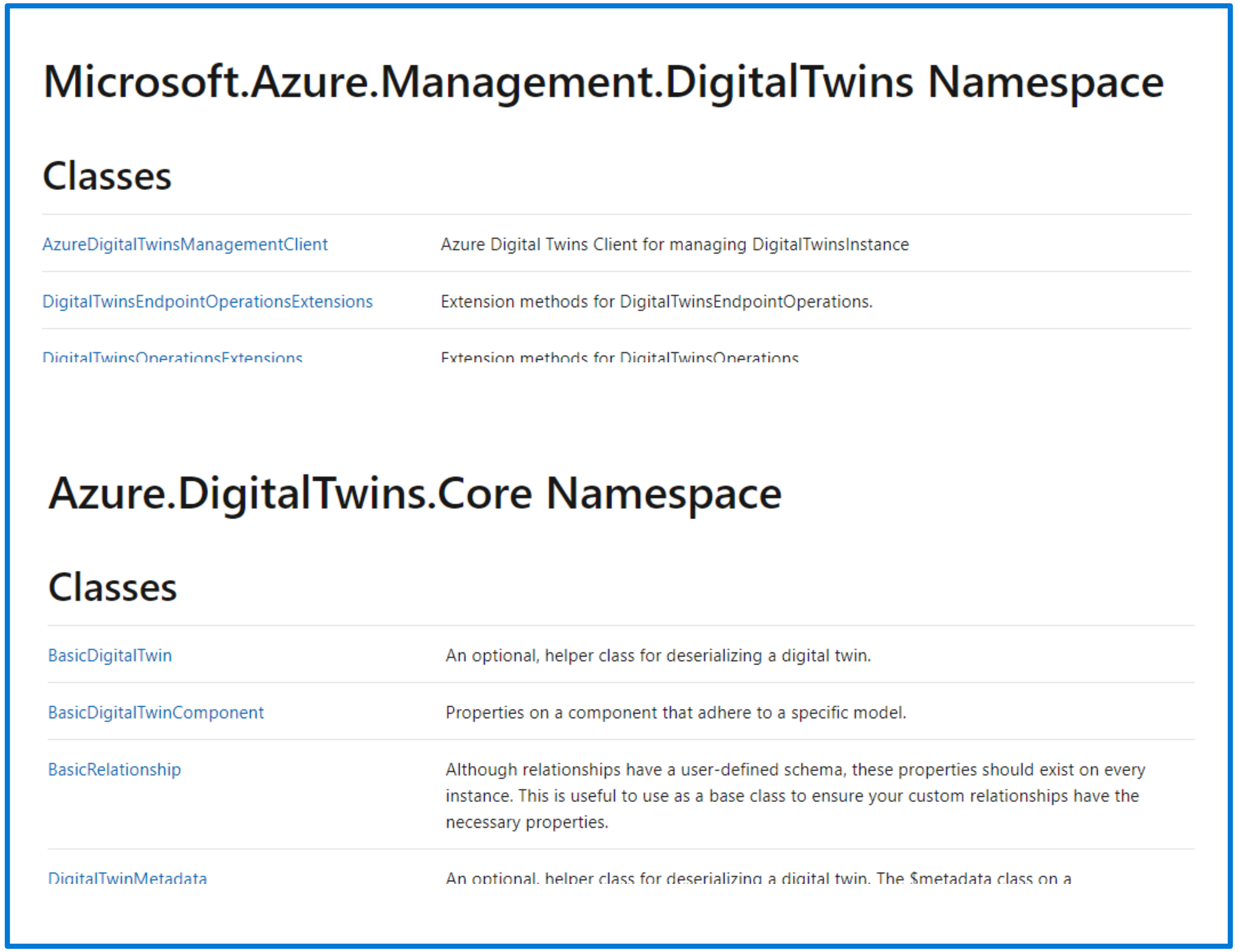
Examine the ADT SDKs

ADT SDKs cover Control plane and Data plane APIs with language support as follows:

- Control plane: .NET (C#), Java, JavaScript, Python, Go
- Data plane: .NET (C#), Java, JavaScript, Python

ADT SDKs can be used to:

- Instantiate the client
- Create, get, and remove models
- Create, query, and delete a digital twin
- Get and update components for a digital twin
- Create, get, and delete digital twin relationships
- Create, get, and delete event routes
- Publish telemetry messages to a digital twin and digital twin component



Microsoft.Azure.Management.DigitalTwins Namespace

Classes

AzureDigitalTwinsManagementClient	Azure Digital Twins Client for managing DigitalTwinsInstance
DigitalTwinsEndpointOperationsExtensions	Extension methods for DigitalTwinsEndpointOperations.
DigitalTwinsOperationsExtensions	Extension methods for DigitalTwinsOperations

Azure.DigitalTwins.Core Namespace

Classes

BasicDigitalTwin	An optional, helper class for deserializing a digital twin.
BasicDigitalTwinComponent	Properties on a component that adhere to a specific model.
BasicRelationship	Although relationships have a user-defined schema, these properties should exist on every instance. This is useful to use as a base class to ensure your custom relationships have the necessary properties.
DigitalTwinMetadata	An optional, helper class for deserializing a digital twin. The $metadata class on a

Manage digital twins in the graph

You can manage twins in code using system client methods and helper functions

To create a twin, use the CreateOrReplaceDigitalTwinAsync() method:

```
await client.CreateOrReplaceDigitalTwinAsync<BasicDigitalTwin>(twinId, initData);
```

To access twin data, use the GetDigitalTwin() method:

```
Response<BasicDigitalTwin> twinResponse = await client.GetDigitalTwinAsync<BasicDigitalTwin>(twinId);
twin = twinResponse.Value;
```

To update a digital twin, pass a JSON Patch document into an UpdateDigitalTwin() method:

```
await client.UpdateDigitalTwinAsync(twinId, updateTwinData);
```

To update a digital twin's model, apply a patch using the UpdateDigitalTwin() method:

```
await client.UpdateDigitalTwinAsync(twinId, updateTwinData);
```

To delete a digital twin, use the `DeleteDigitalTwin()` method:

```
await client.DeleteDigitalTwinAsync(twinId);
```

Manage digital twin relationships in the graph

To create a relationship, use the CreateOrReplaceRelationshipAsync() method:

```
await client.CreateOrReplaceRelationshipAsync<BasicRelationship>(srcId, relId, relationship);
```

To list relationships, use GetRelationshipsAsync() or GetIncomingRelationshipsAsync():

```
AsyncPageable<BasicRelationship> rels = client.GetRelationshipsAsync<BasicRelationship>(dtId);
AsyncPageable<IncomingRelationship> incomingRels = client.GetIncomingRelationshipsAsync(dtId);
```

To update a relationship, use the UpdateRelationship() method:

```
await client.UpdateRelationshipAsync(srcId, relId, patchDocument);
```

To delete a relationship, use :

```
await client.DeleteRelationshipAsync(srcId, relId);
```

Get started with ADT queries

ADT query language:
- a custom SQL-like query language
- similar to the IoT hub query language

Queries can be based on:
- properties of the twin
- models
- relationships
- properties of a relationship

Query limitations include:
- Up to 10 second delay between graph updates and query results
- No subqueries within FROM
- No support for OUTER JOIN
- No more than 5 JOIN levels
- Relationships can't be queried as independent entities

Example Queries

- List of all digital twins in the instance:

```
SELECT * FROM DIGITALTWINS
```

- Get digital twins by properties

```
SELECT * FROM DIGITALTWINS T WHERE T.Temperature = 70
```

- Query by model

```
SELECT * FROM DIGITALTWINS WHERE
IS_OF_MODEL('dtmi:example:thing;1’)
```

- Query by relationship

```
SELECT T, CT FROM DIGITALTWINS T
JOIN CT RELATED T.contains WHERE T.$dtId = 'ABC'
```

Get started with Azure functions for ADT

Use the following process to implement an Azure function

Create an Azure Functions project

Write the function code:
- Add authentication code to the function (for accessing ADT)
- Add code that will interact with ADT (and other Azure resources)

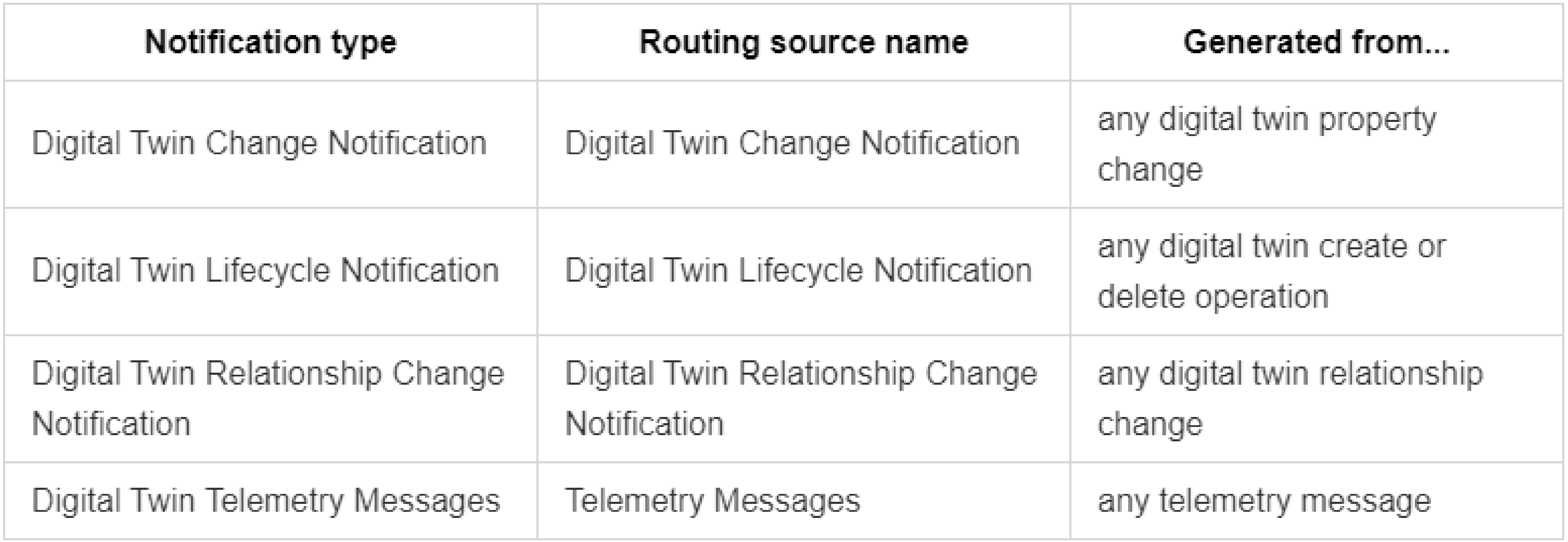
Publish the function app to Azure

Configure security access for the function app in Azure

The Azure function will interact with either upstream or downstream Azure services. Azure resource configuration before and/or after creating the Azure function will be required.

Examine ADT event data

Event notification types



Notification type	Routing source name	Generated from...
Digital Twin Change Notification	Digital Twin Change Notification	any digital twin property change
Digital Twin Lifecycle Notification	Digital Twin Lifecycle Notification	any digital twin create or delete operation
Digital Twin Relationship Change Notification	Digital Twin Relationship Change Notification	any digital twin relationship change
Digital Twin Telemetry Messages	Telemetry Messages	any telemetry message

In general, notifications are made up of two parts: the header and the body.

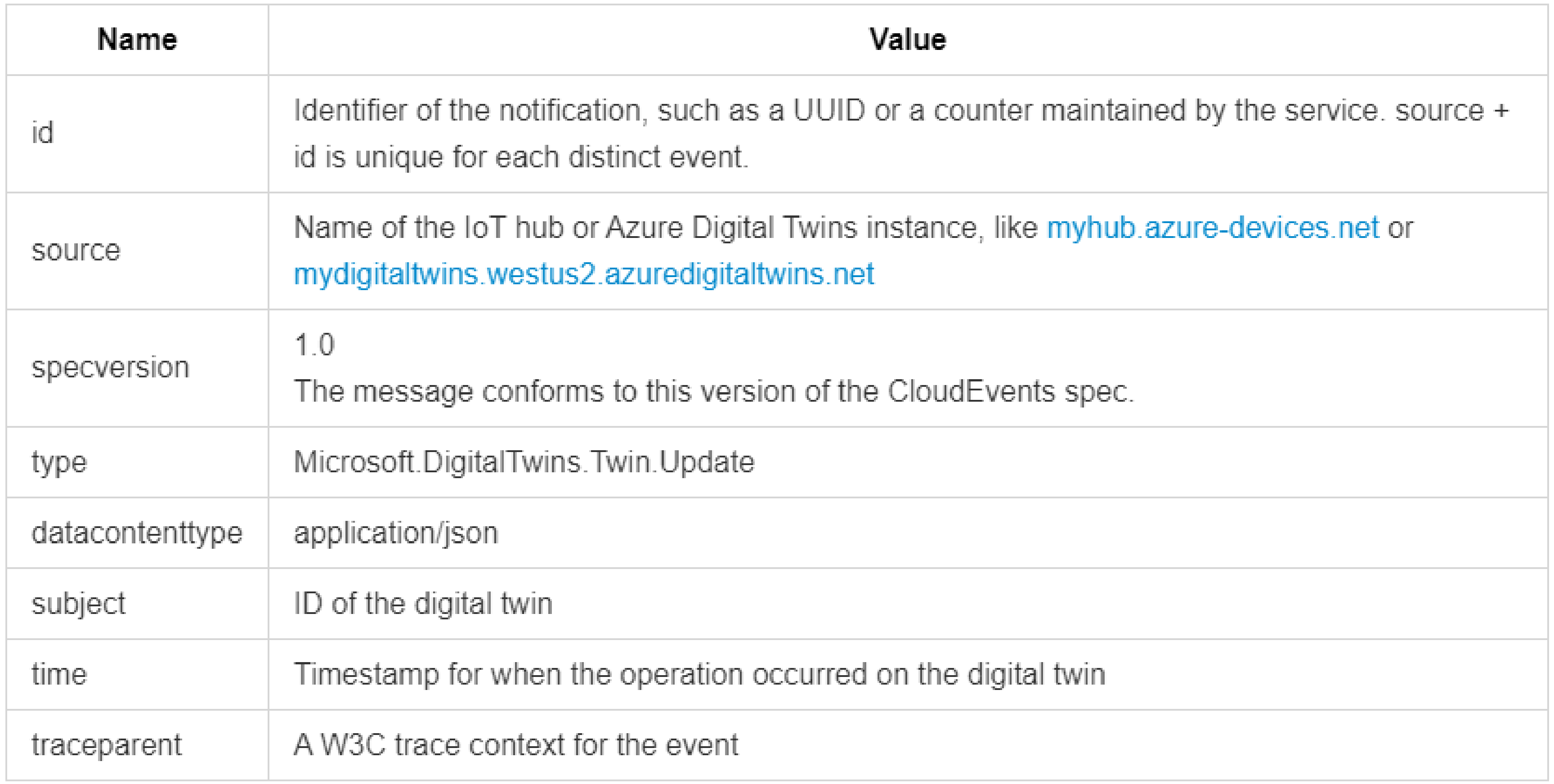
Examine ADT event data

Digital twin change notifications

Digital twin change notifications are triggered when a digital twin is updated:

- When property values or metadata changes
- When digital twin or component metadata changes

The body for the Twin.Update notification is a JSON Patch document containing the update to the digital twin



Name	Value
id	Identifier of the notification, such as a UUID or a counter maintained by the service. source + id is unique for each distinct event.
source	Name of the IoT hub or Azure Digital Twins instance, like myhub.azure-devices.net or mydigitaltwins.westus2.azuredigitaltwins.net
specversion	1.0 The message conforms to this version of the CloudEvents spec.
type	Microsoft.DigitalTwins.Twin.Update
datacontenttype	application/json
subject	ID of the digital twin
time	Timestamp for when the operation occurred on the digital twin
traceparent	A W3C trace context for the event

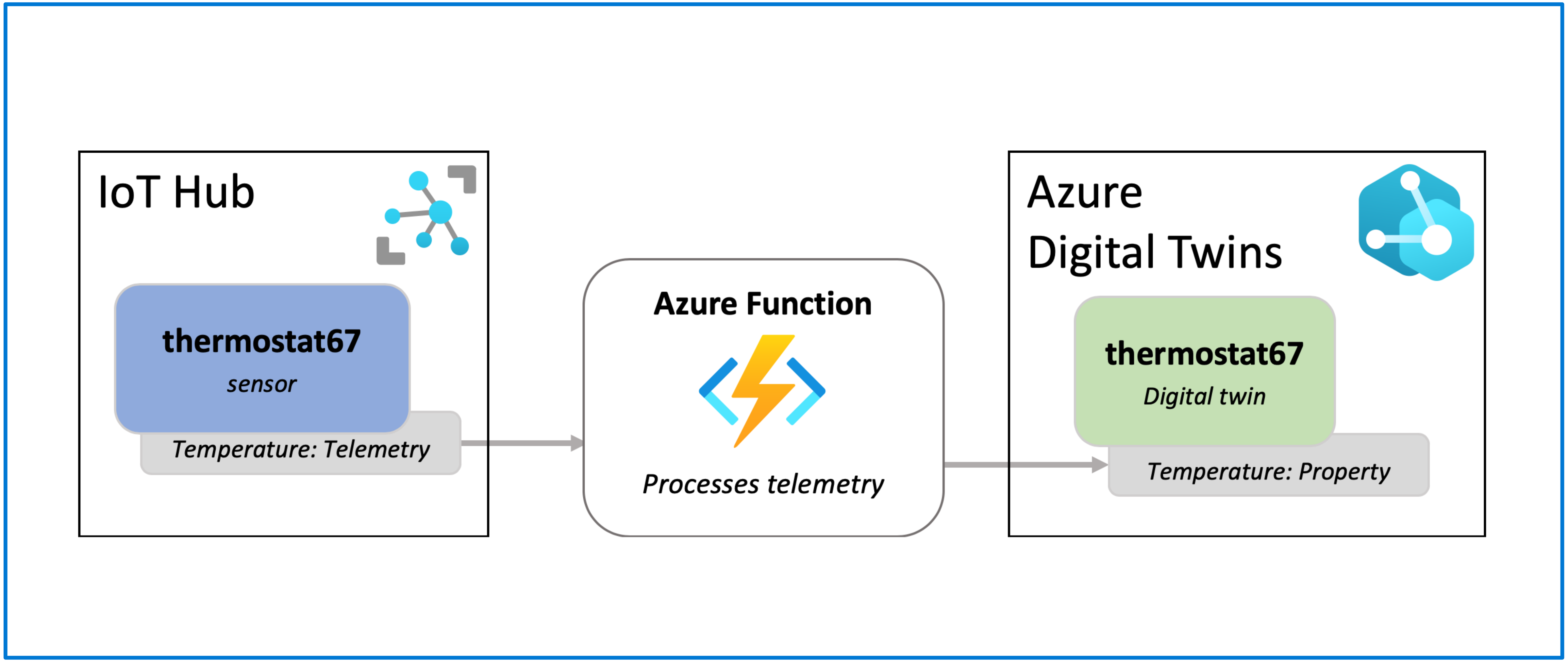
Examine data ingress and egress processes

An Azure Digital Twins solution relies on external resources for data inputs as well as analysis and storage of data outputs. ADT workflows fall into three main categories:

- Data ingress – ingesting data from upstream resources
- Data egress – using ADT data for in-service updates
- Data egress – providing ADT data to downstream resources

Examine data ingress and egress processes

Data Ingress (data ingestion from an upstream resource)



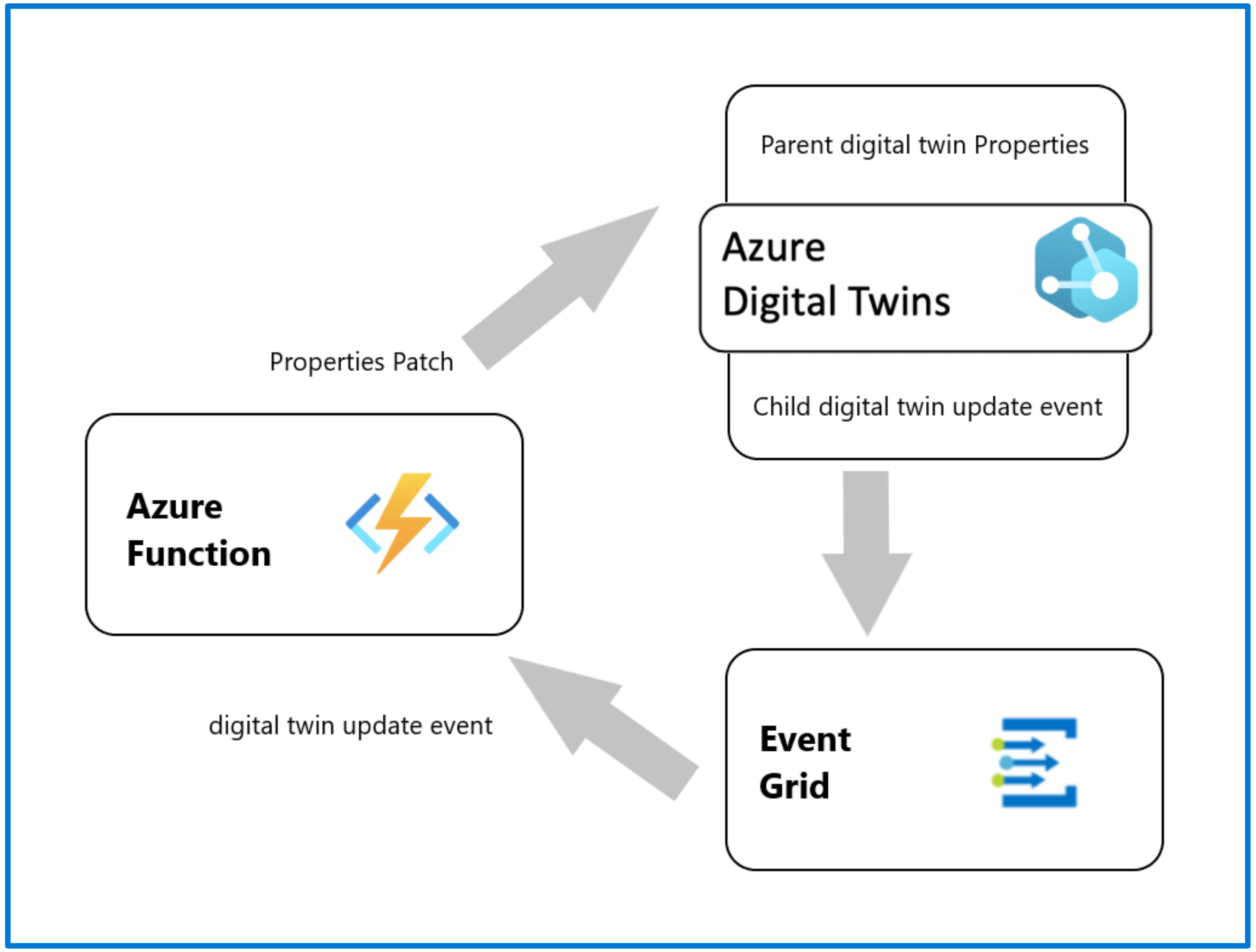
Examine data ingress and egress processes

Data Egress (in-service update – update parent twin property)

ADT uses digital twin change notification events as a trigger to route data to an ADT endpoint.

An Event Grid endpoint is used to transfer the digital twin change notification message, such as a twin update, to an Azure Function.

An Azure Function extracts data from the notification header and body and uses that data to get additional information from ADT, such as finding a parent digital twin. The function then performs the required action.


Parent digital twin Properties
Azure Digital Twins
Child digital twin update event
Properties Patch
Azure Function
digital twin update event
Event Grid

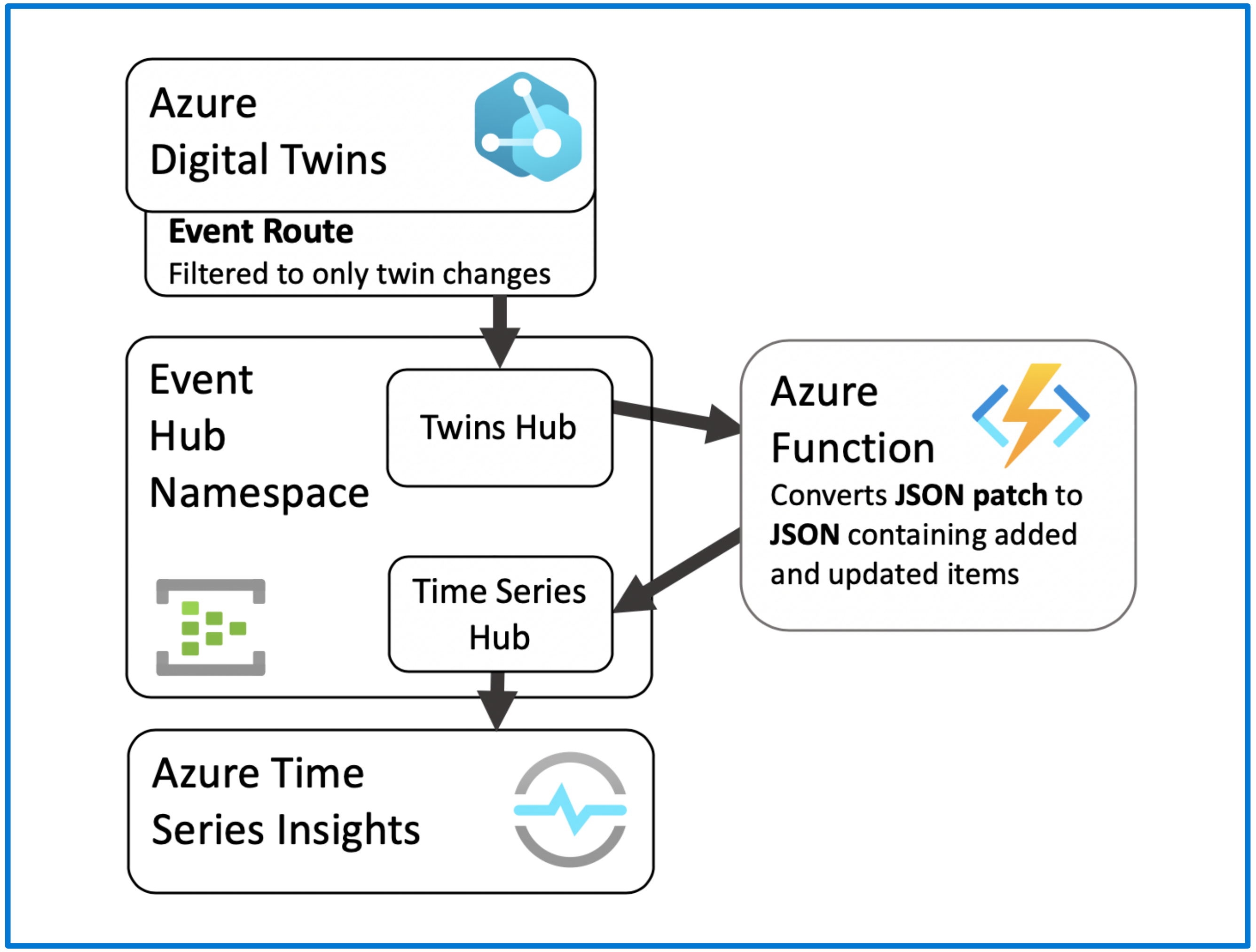
Examine data ingress and egress processes

Data Egress (downstream service support – Time Series Insights)

ADT streams data to downstream services by routing events through an Event Hubs Namespace

The Event Hubs Namespace will include an Event Hub that receives events from ADT and an Event Hub that feeds events to the downstream service. An Azure Function is used to prepare message data and apply event formatting that is appropriate for the downstream service

A downstream service, such as Time Series Insights, consumes the events from the second Event Hub


Azure Digital Twins
Event Route
Filtered to only twin changes
Event Hub Namespace
Twins Hub
Time Series Hub
Azure Function
Converts JSON patch to JSON containing added and updated items
Azure Time Series Insights

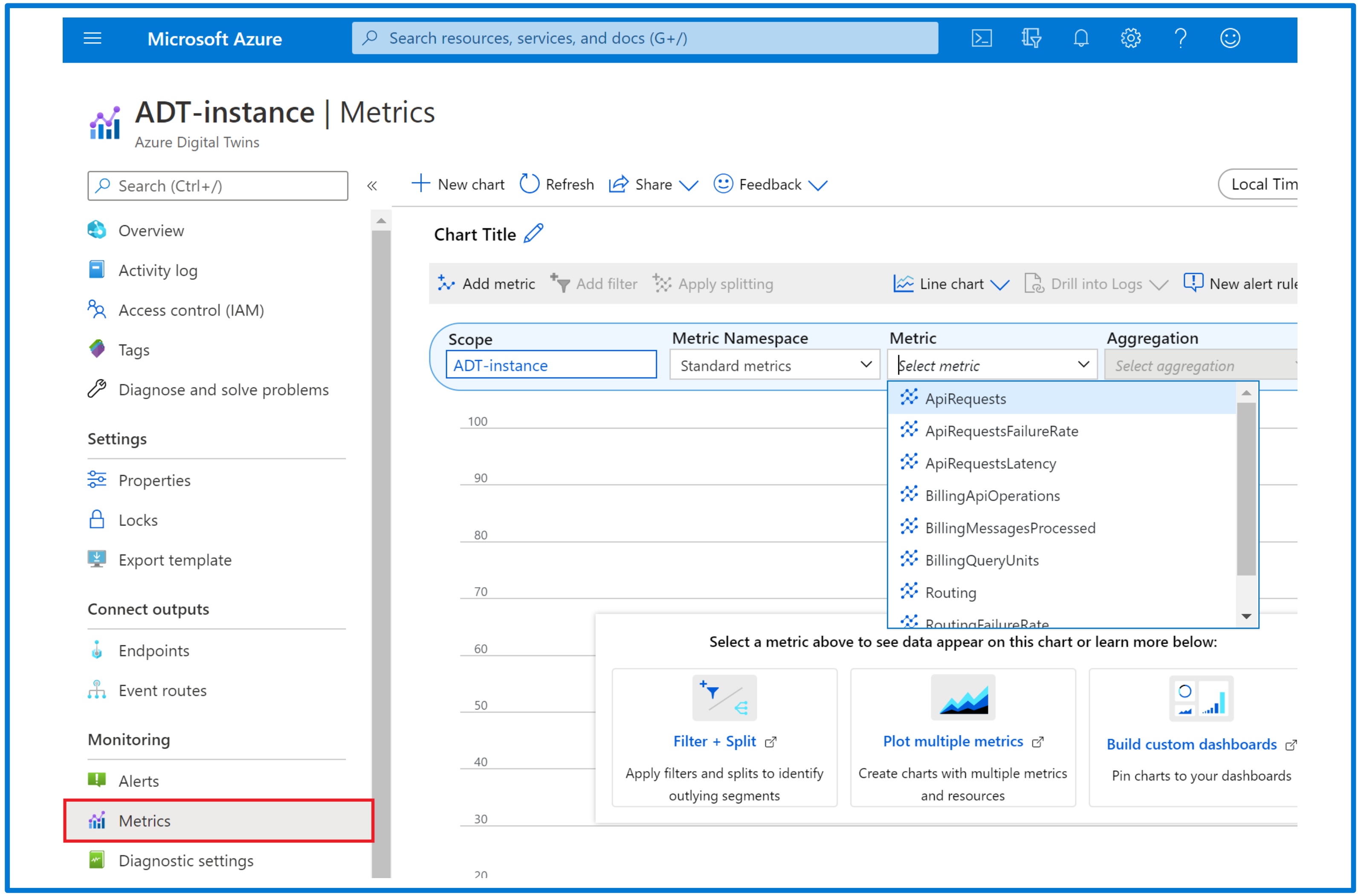
Lesson 4: Monitor and troubleshoot ADT

Examine the Azure Digital Twins metrics

Metrics for tracking service limits can be used, for example, when you're approaching a published service limit for some aspect of your solution

Metrics for tracking data ingress can be used, for example, when you need to monitor the number of incoming telemetry events

Metrics for tracking routing operations can be used, for example, when you need to monitor the number of messages routed to an endpoint



Scope: ADT-instance
Metric Namespace: Standard metrics
Metric: Select metric
Aggregation: Select aggregation

- ApiRequests
- ApiRequestsFailureRate
- ApiRequestsLatency
- BillingApiOperations
- BillingMessagesProcessed
- BillingQueryUnits
- Routing
- RoutingFailureRate

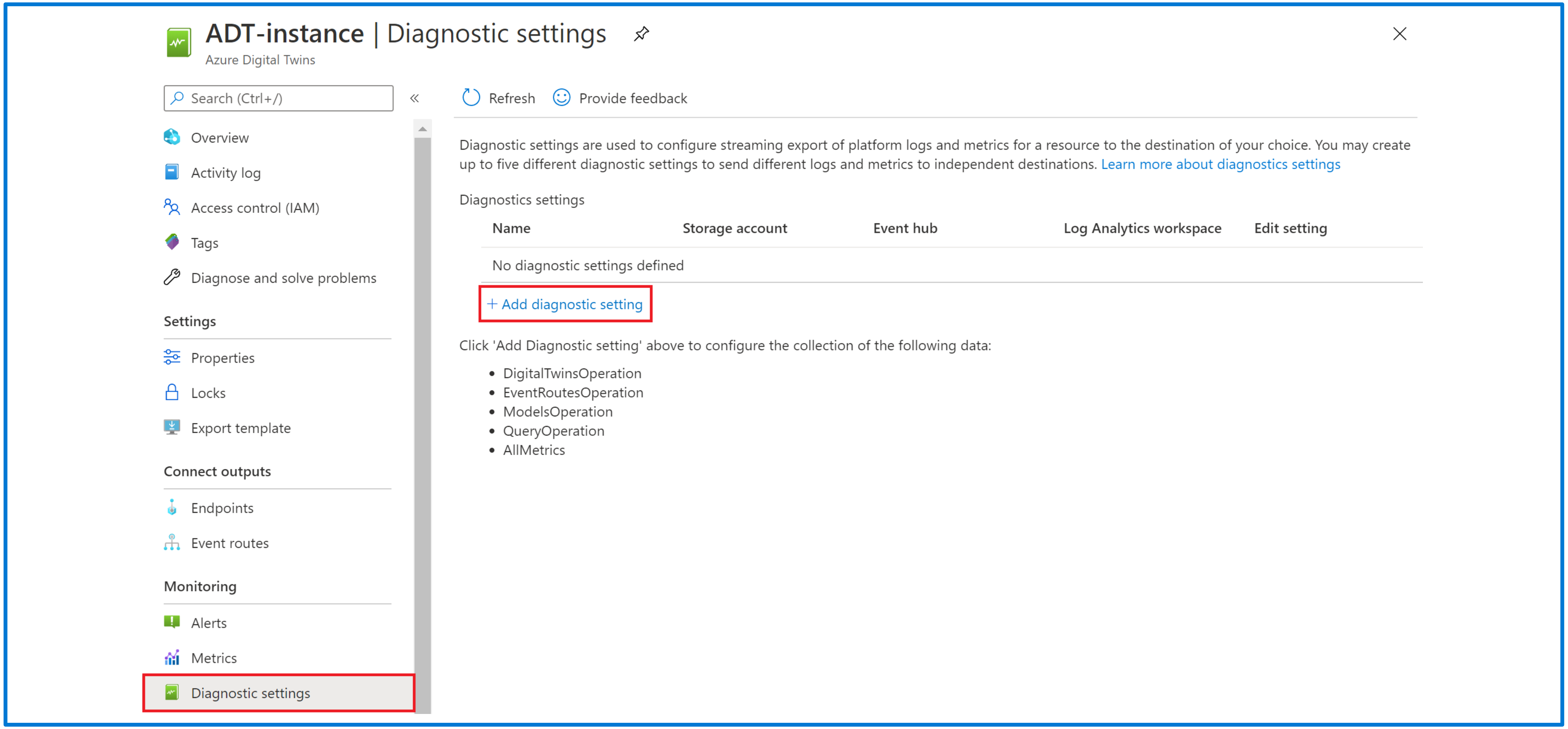
Select a metric above to see data appear on this chart or learn more below:

Filter + Split — Apply filters and splits to identify outlying segments

Plot multiple metrics — Create charts with multiple metrics and resources

Build custom dashboards — Pin charts to your dashboards

Examine the Azure Digital Twins diagnostic settings



ADT-instance | Diagnostic settings

Azure Digital Twins

Diagnostic settings are used to configure streaming export of platform logs and metrics for a resource to the destination of your choice. You may create up to five different diagnostic settings to send different logs and metrics to independent destinations. Learn more about diagnostics settings

Diagnostics settings

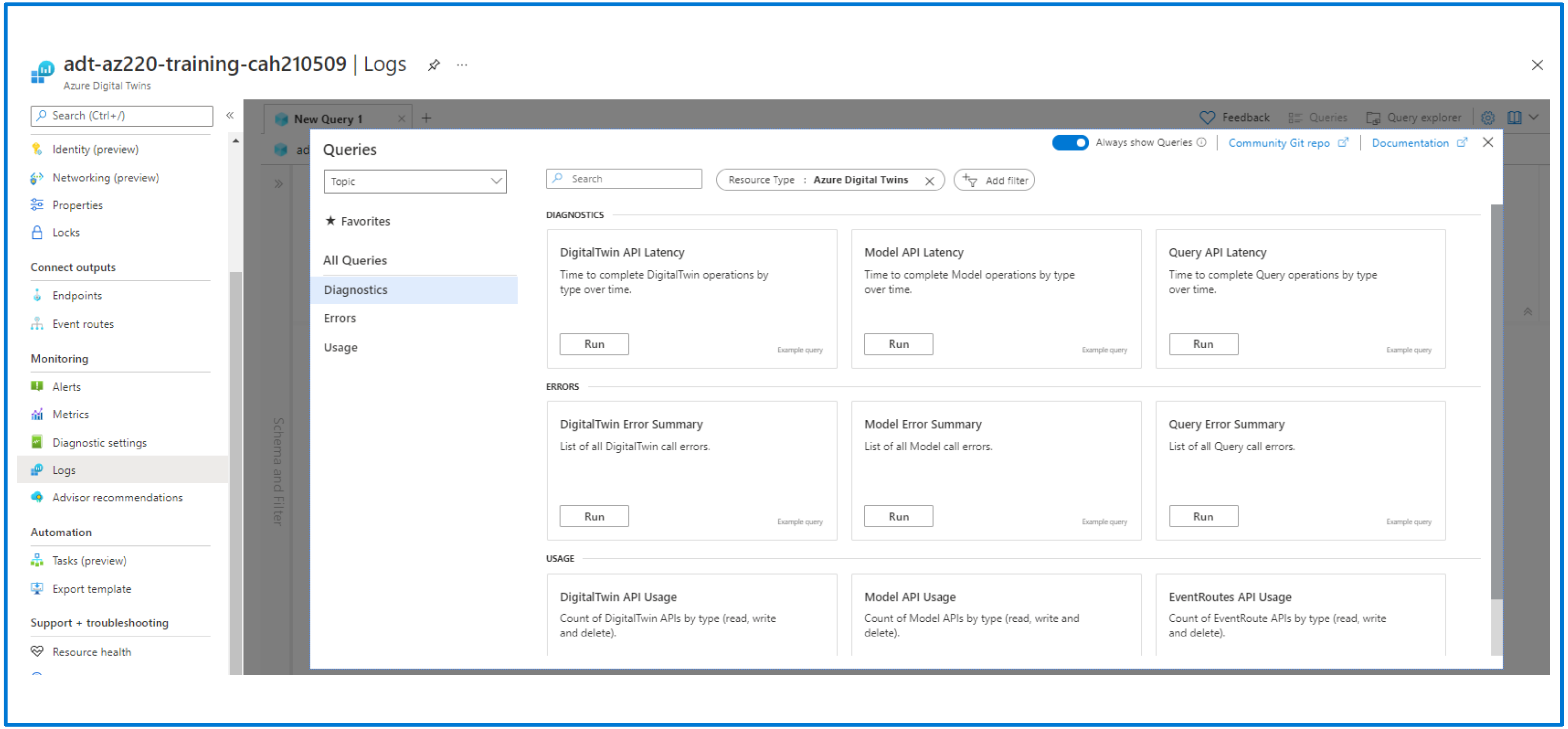
Name	Storage account	Event hub	Log Analytics workspace	Edit setting
No diagnostic settings defined				

+ Add diagnostic setting

Click 'Add Diagnostic setting' above to configure the collection of the following data:

- DigitalTwinsOperation
- EventRoutesOperation
- ModelsOperation
- QueryOperation
- AllMetrics

View and query logs



adt-az220-training-cah210509 | Logs

Azure Digital Twins

Search (Ctrl+/)

- Identity (preview)
- Networking (preview)
- Properties
- Locks

Connect outputs

- Endpoints
- Event routes

Monitoring

- Alerts
- Metrics
- Diagnostic settings
- Logs
- Advisor recommendations

Automation

- Tasks (preview)
- Export template

Support + troubleshooting

- Resource health

New Query 1

Feedback | Queries | Query explorer

Queries

Always show Queries | Community Git repo | Documentation

Topic

Search | Resource Type : Azure Digital Twins | Add filter

★ Favorites

All Queries

- Diagnostics
- Errors
- Usage

DIAGNOSTICS

DigitalTwin API Latency
Time to complete DigitalTwin operations by type over time.
Run — Example query

Model API Latency
Time to complete Model operations by type over time.
Run — Example query

Query API Latency
Time to complete Query operations by type over time.
Run — Example query

ERRORS

DigitalTwin Error Summary
List of all DigitalTwin call errors.
Run — Example query

Model Error Summary
List of all Model call errors.
Run — Example query

Query Error Summary
List of all Query call errors.
Run — Example query

USAGE

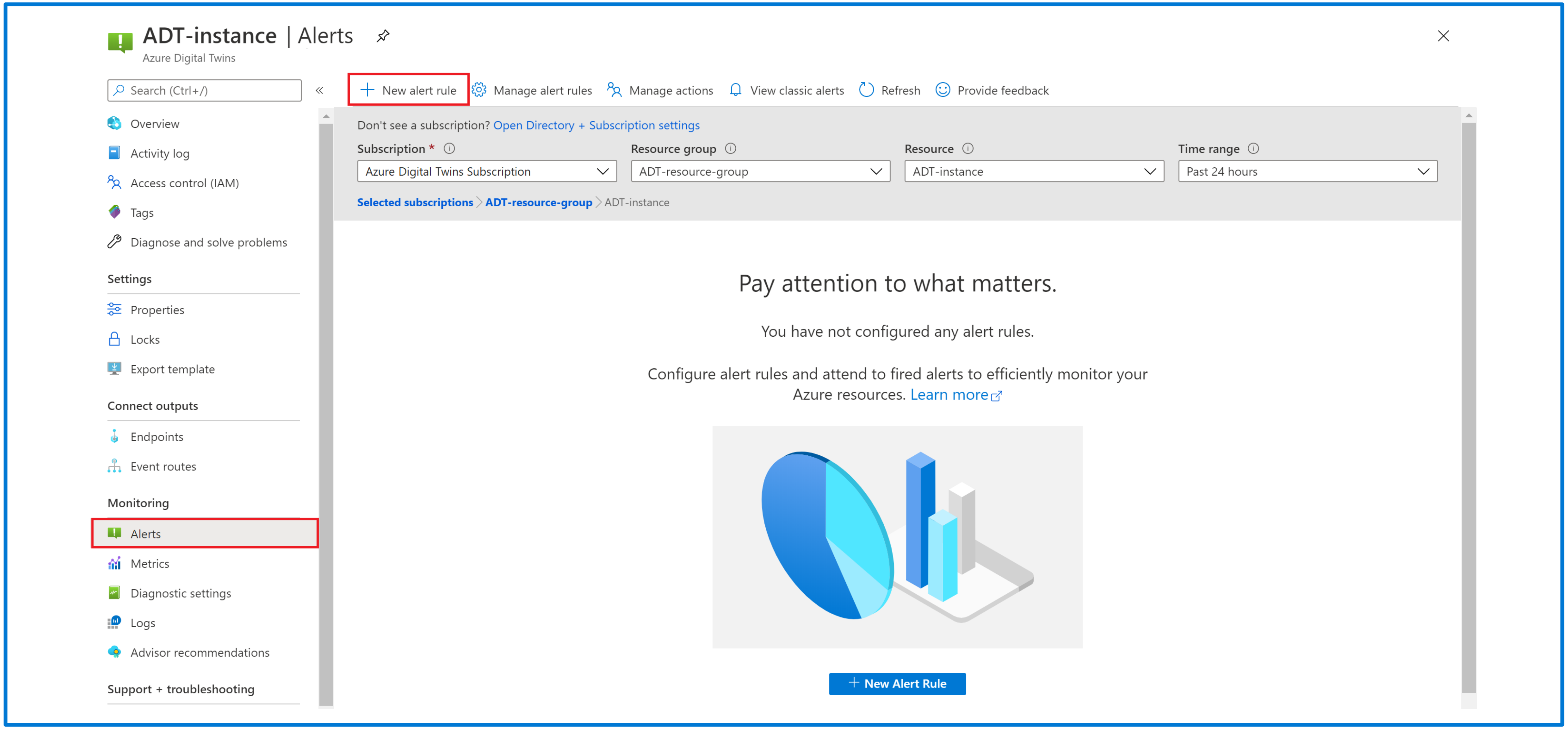
DigitalTwin API Usage
Count of DigitalTwin APIs by type (read, write and delete).

Model API Usage
Count of Model APIs by type (read, write and delete).

EventRoutes API Usage
Count of EventRoute APIs by type (read, write and delete).

Schema and Filter

Enable alerts



Don't see a subscription? Open Directory + Subscription settings

Subscription *: Azure Digital Twins Subscription
Resource group: ADT-resource-group
Resource: ADT-instance
Time range: Past 24 hours

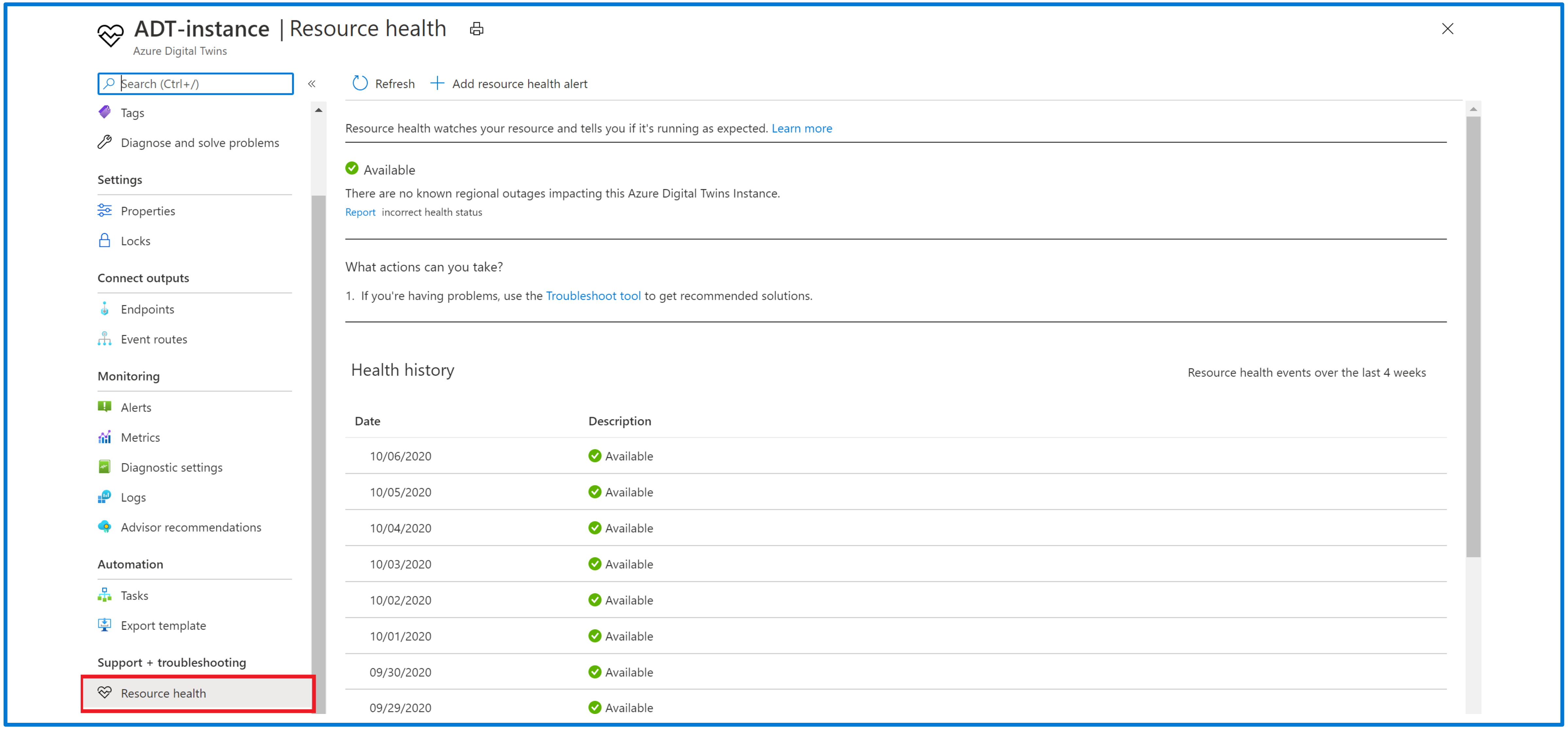
Selected subscriptions > ADT-resource-group > ADT-instance

Pay attention to what matters.

You have not configured any alert rules.

Configure alert rules and attend to fired alerts to efficiently monitor your Azure resources. Learn more

Understand ADT resource health



ADT-instance | Resource health

Azure Digital Twins

Resource health watches your resource and tells you if it's running as expected. Learn more

✓ Available

There are no known regional outages impacting this Azure Digital Twins Instance.

Report incorrect health status

What actions can you take?

1. If you're having problems, use the Troubleshoot tool to get recommended solutions.

Health history

Resource health events over the last 4 weeks

Date	Description
10/06/2020	✓ Available
10/05/2020	✓ Available
10/04/2020	✓ Available
10/03/2020	✓ Available
10/02/2020	✓ Available
10/01/2020	✓ Available
09/30/2020	✓ Available
09/29/2020	✓ Available

Lesson 5: Module Labs

Module 11 Lab

Lab 19: Develop Azure Digital Twins solutions:

- Design and develop digital twin models
- Create and configure digital twins
- Implement ADT graph interactions
- Integrate ADT with upstream and downstream systems

Lesson 6: Module 11 review questions

Module review: Question 11.1

What is the name of the coding format used to define Azure Digital Twins models?

Answer A: Extensible Application Markup Language.	**Answer B:** Azure Digital Twins Modeling Language.	**Answer C:** Digital Twins Definition Language.

Module review: Question 11.2

What is the relationship between an Azure Digital Twins model and a digital twin?

Answer A: An ADT model is an instance of a digital twin.	**Answer B:** A digital twin is an instance of an ADT model.	**Answer C:** An ADT model contains the digital twins for your ADT environment.

Module review: Question 11.3

What happens to the associated digital twins when a developer deletes a model?

Answer A:
The digital twins are automatically removed from the environment.

Answer B:
The properties of the digital twins can no longer be queried.

Answer C:
The properties of the digital twins can no longer be updated.

Module review: Question 11.4

A developer has created an Azure Digital Twins instance in the Azure portal.

In order to manage the Azure Digital Twins service and its data, what role assignment must be configured?

Answer A:
Azure Digital Twins Data Reader

Answer B:
Azure Digital Twins Data Owner

Answer C:
Owner

Module review: Question 11.5

Which of the following choices describes why a developer would query the Azure Digital Twins diagnostics logs?

Answer A:

To troubleshoot Azure Digital Twins service issues and generate insights.

Answer B:

To manage the Azure Digital Twins environment, including models and twins.

Answer C:

To troubleshoot upstream and downstream service issues.

Module review: Question 11.6

Which of the following choices describes the purpose of Azure Digital Twins metrics?

Answer A:	Answer B:	Answer C:
They provide access to properties of the Azure Digital Twins services and the properties of connected upstream and downstream resources.	They provide access to properties of the Azure Digital Twins services and the properties of the connected digital twins.	They provide an overview of the health of your service instance.

Module review: Question 11.7

A digital twins environment includes twins that support telemetry coming from IoT hub devices.

Which Azure service should be used to ingest telemetry from the upstream IoT hub?

Answer A: Azure Device Provisioning Service	**Answer B:** Azure Function	**Answer C:** Azure Digital Twins Explorer

Module review: Question 11.8

A developer has built a digital twins (graph) environment and they want to create some sample queries. They run the following query:

SELECT T, CT FROM DIGITALTWINS T JOIN CT RELATED T.rel_has_caves WHERE T.$dtId = 'factory_1'

Assuming that the query runs as expected, what results are returned?

Answer A:
The "cave" digital twins that have a "rel_has_caves" relationship to the "factory" digital twin with an ID of "factory_1".

Answer B:
The "factory_1" digital twin and all digital twins that are related to it.

Answer C:
The "factory_1" digital twin and all of the digital twins that have a "rel_has_caves" relationship to it.