
# Bringing Integration Anywhere

## with Azure Logic Apps v2

# Thank you!

VISUG

codit

cegeka
IN CLOSE COOPERATION

Capgemini

team4talent

involved

ORDINA

Xpirit

AXXES

Microsoft

Massimo Crippa

# Hi, nice to meet you!

# IoT solutions

## Building intelligent vessels

# Cloud platforms

Identity at your fingertips


# Advanced Analytics

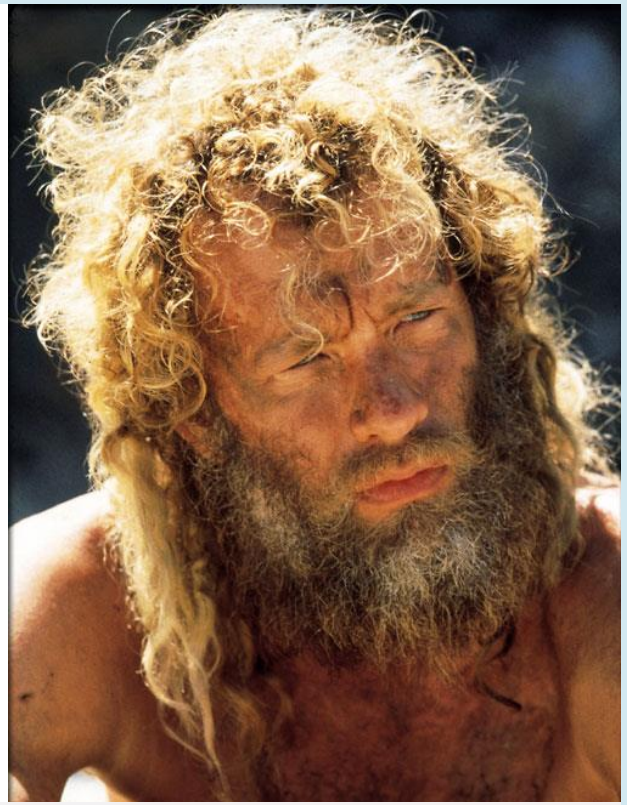
## Securing European waters

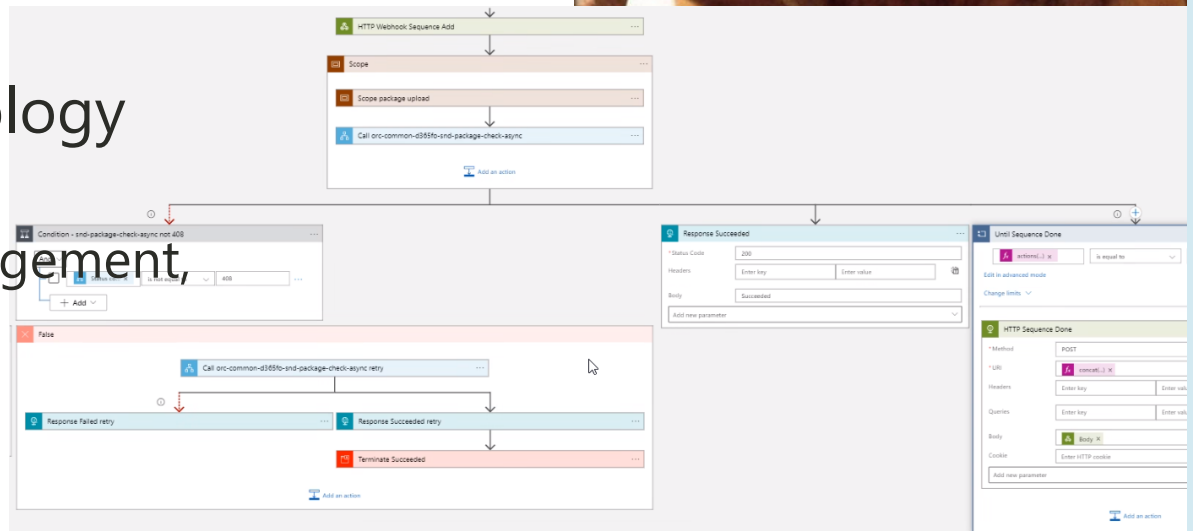
# Enterprise Integration

Connecting systems and people

# Enterprise Integration

- Applications rarely live in isolation
  - Procurement needs to connect with suppliers
  - Shipment needs to sync with the Logistic partner
  - …
- It's all about using messaging to connect applications
  - Message construction, transformation, routing, channel, …
- Azure Integration Services is our technology stack
  - LogicApps, Service Bus, Functions, API Management, Event Grid, Data Factory





# Agenda

- Overview + Terminology
- Logic Apps Standard (Runtime)
  - General Overview
    - Runtime
    - Scaling & Pricing
    - …
  - Connectors: Built-in vs. Cloud
  - Stateful vs. Stateless
  - Local Development & Debugging
  - Networking
  - Monitoring
- Conclusion

# Overview + Terminology

# Logic Apps Offerings

- Logic Apps Consumption (~multi-tenant)
  - Serverless offer
- Logic Apps ISE
  - Dedicated environment
- Logic Apps Standard (~single-tenant, v2, new runtime)
  - The new offering

# Logic Apps Consumption

- Deployment unit: Single Logic App
- Scaling: Serverless handled by MS
- Cost: Pay per action
- Easiest to get started
- API Connections hosted in Azure to connect to other services.

# Logic Apps ISE

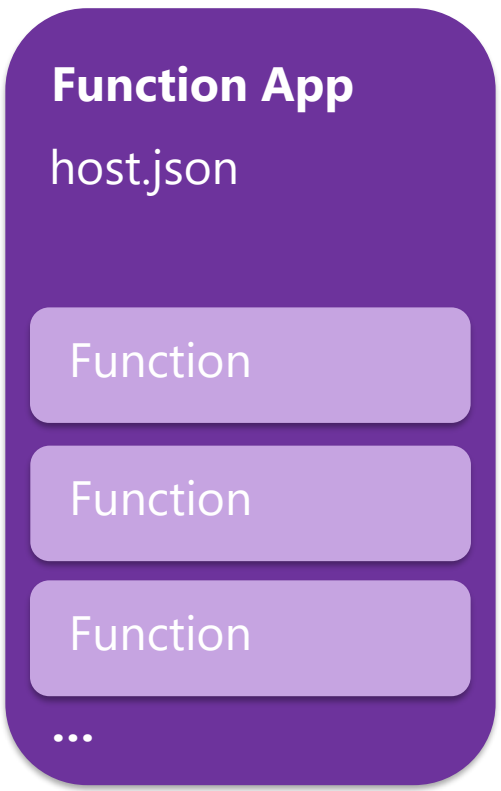
- Deployment unit: Single Logic App
- Deployed inside the (dedicated) ISE environment
- (Optional) Network Integration
- 💰Costly! (Premium starts at ~4750 euros per month)
- ‘Free’ Integration account
- Scale-Out only

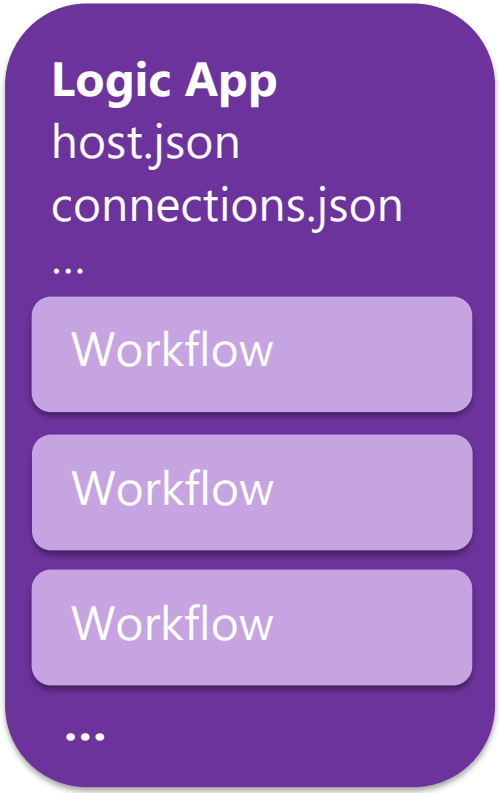
# Logic Apps Standard

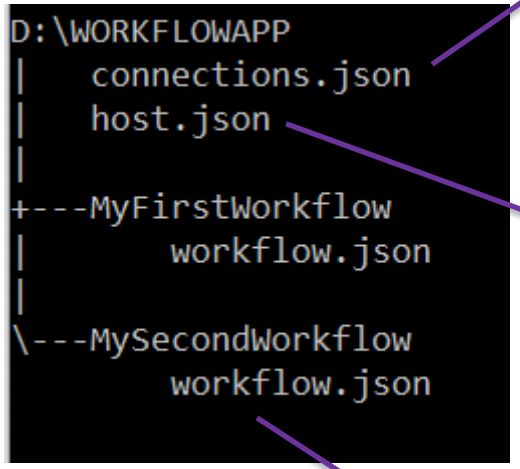
- Deployment unit: Single Logic App with workflow(s)!
- Based on Azure Functions ⚡ runtime
- Run anywhere (Azure, Docker, Kubernetes, … )
- Built-in connectors vs. Cloud connectors

# Logic Apps Standard: deployment unit

- In LA Standard: Logic App (~infra) can have multiple workflows (~logic)
  - Similar to Function App having multiple Functions



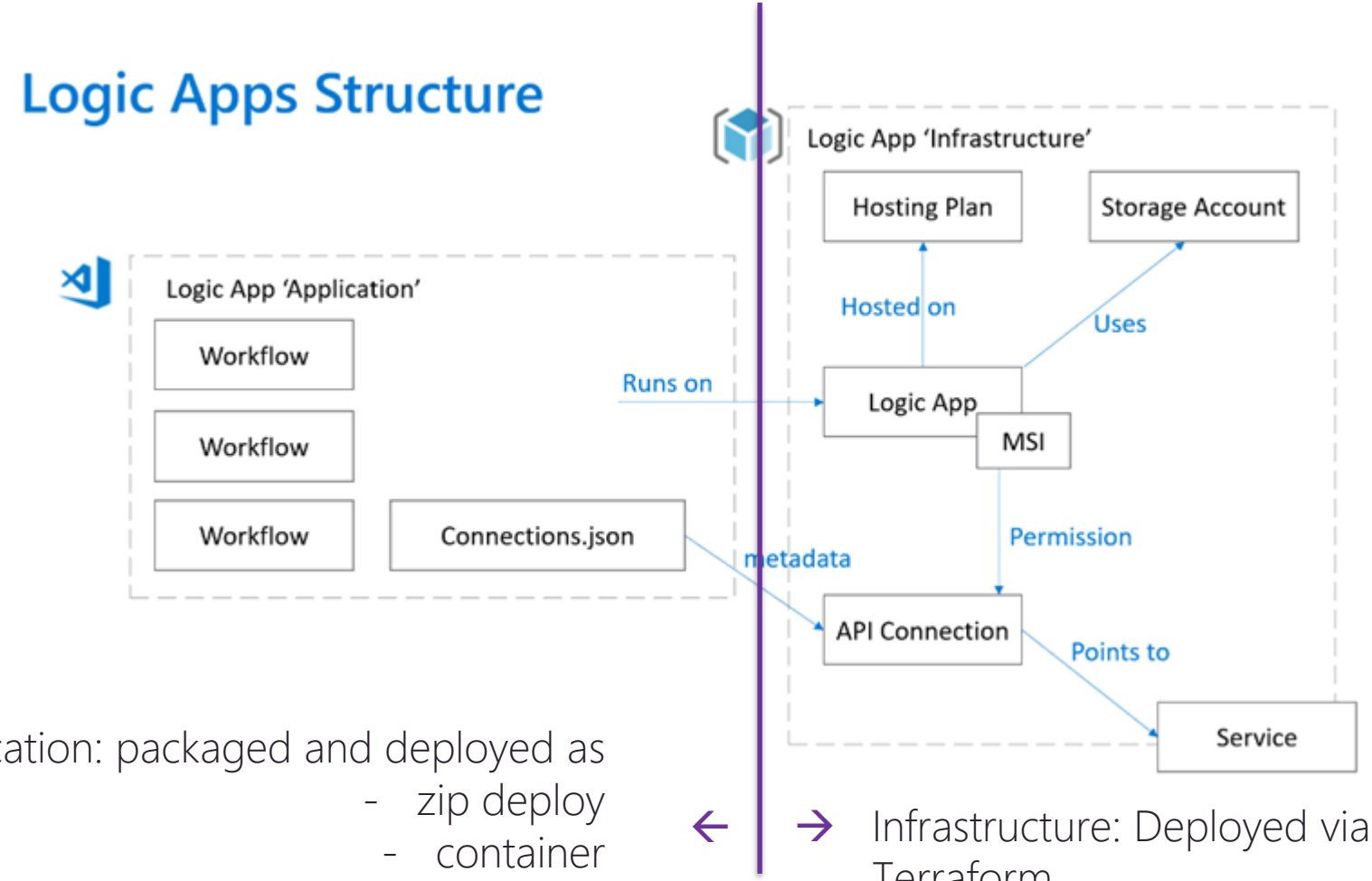




Configure Connectors

Runtime-specific configuration (Logging, Retries, ...)

Workflow JSON definition
Not an ARM-template ⚠

# Logic Apps Standard: deployment unit




Application: packaged and deployed as
- zip deploy
- container

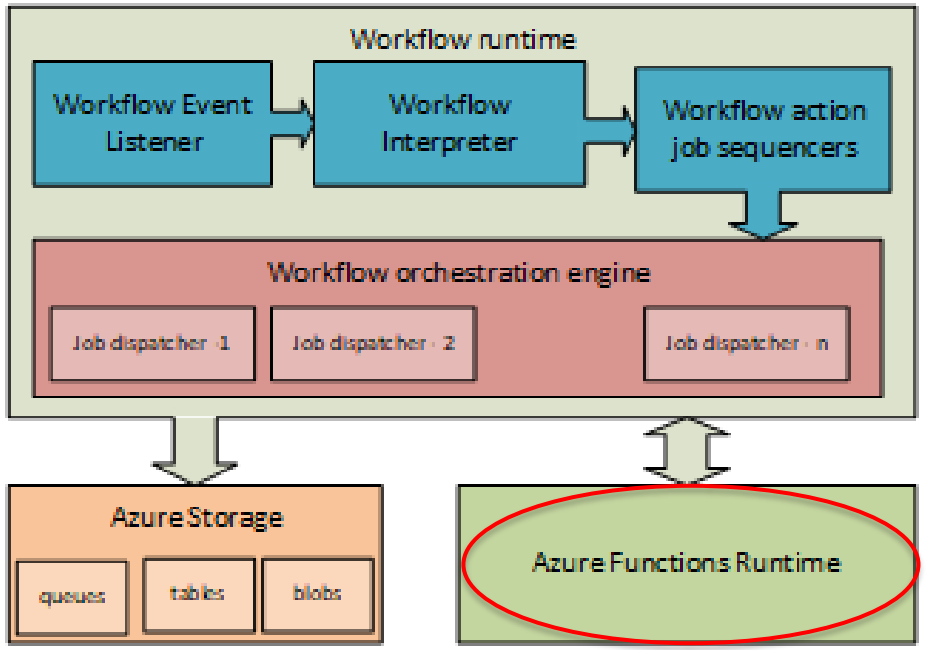
← →

Infrastructure: Deployed via ARM/Bicep, Terraform, …

Separation of concerns

# Logic Apps Standard: workflow runtime

Trigger = Event listener

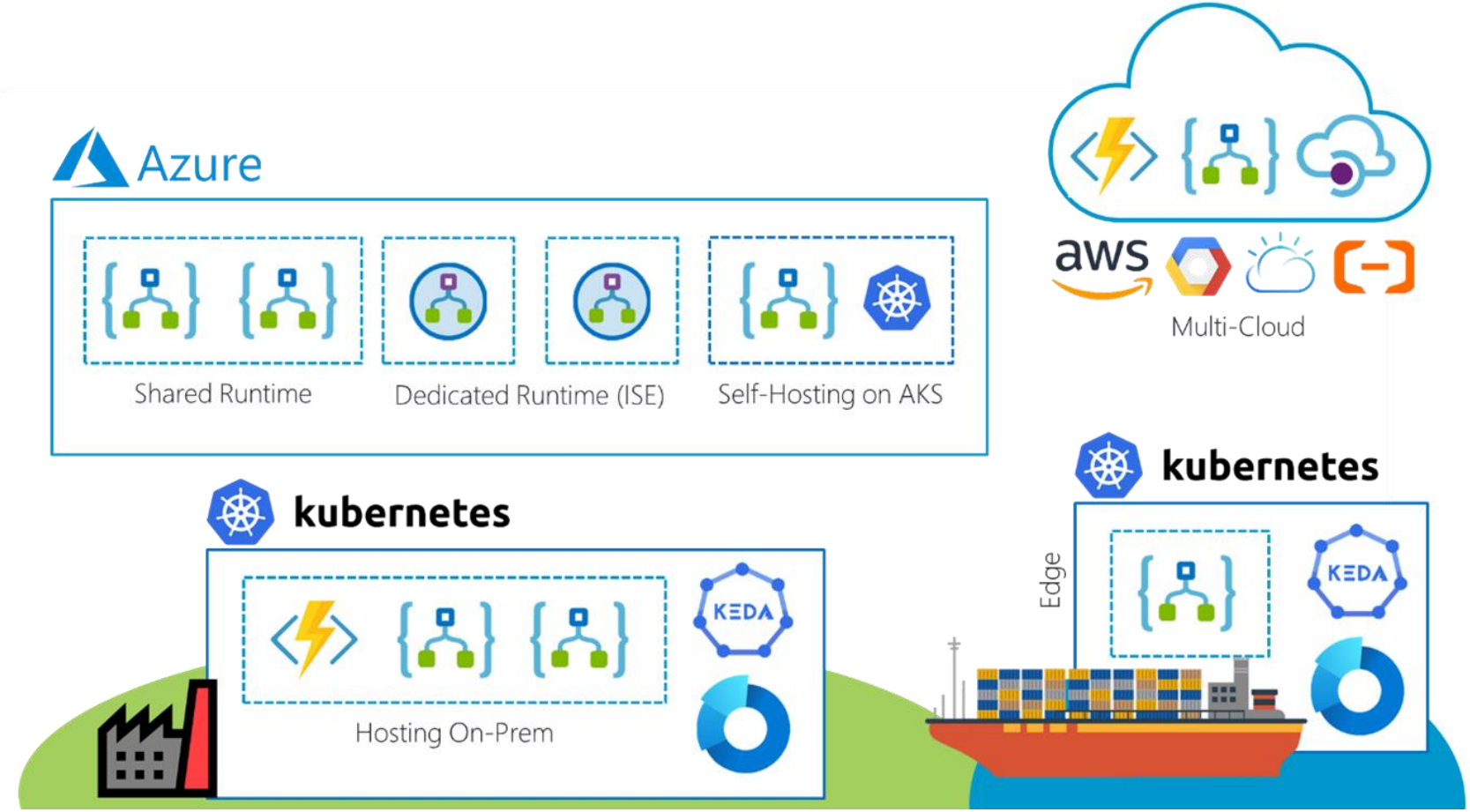



Used for persistence / state

SQL-support in public preview

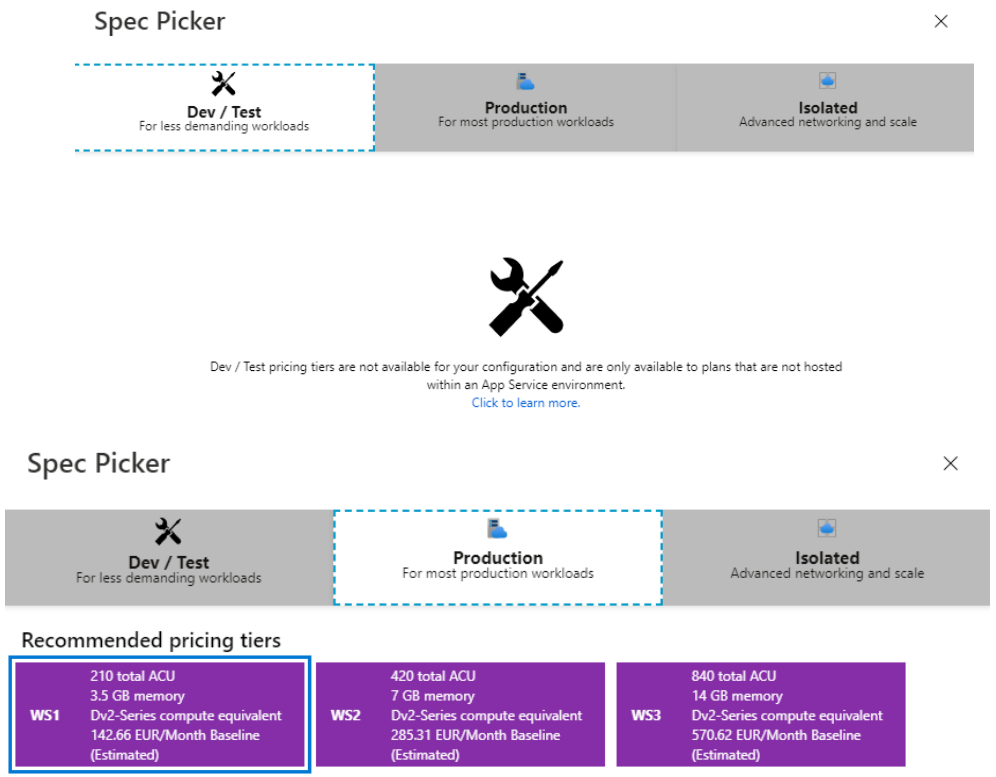
- Each action is a job, job handles input/output/requests/error handling/…
- Job dependencies are mapped via Direct Acyclic Graph.
  Handled by Job sequencer
- Job dispatcher executes a job

# Logic Apps Standard : run anywhere



# Scaling & Pricing

- Microsoft (currently) offers 3 “Workflow Standard” plans
  - Supposedly more predictable (stable) costs
  - Quite expensive ⚠
  - No Dev/Test plans ⚠
- Scale up and Scale out
  - Min and Max instances (plan scale)
  - Min and Max warm instances (app scale)
- Possible to configure other plans via ARM/Bicep.



# Cost Prediction

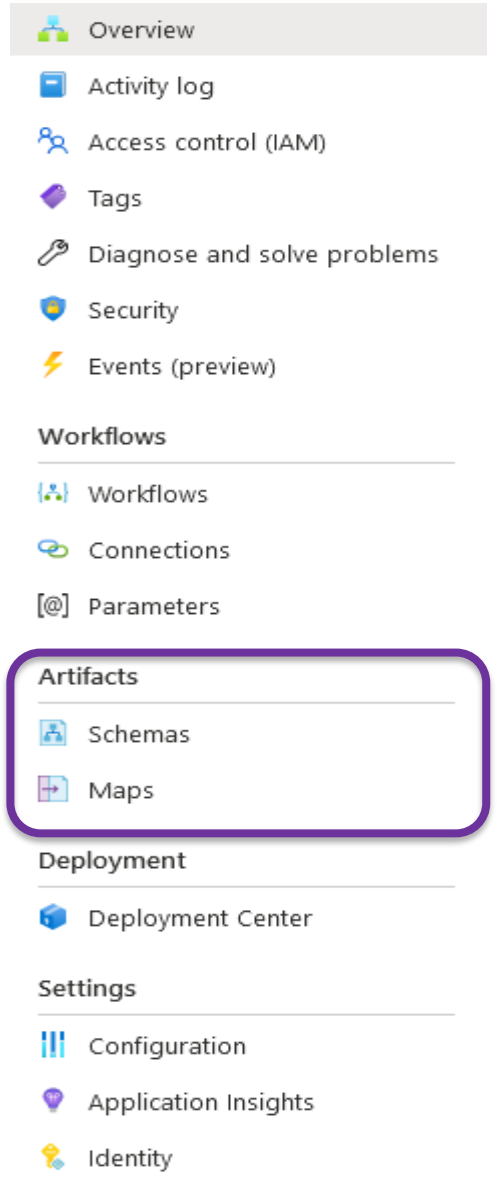
- ~ Fixed costs = easier to predict?
- Typically: workload is known (# runs per day)
  - LA Consumption: # actions known → Predict cost
  - LA Standard: Pricing plan known → Predict cost?
    - What load can app plan run?
    - What is the optimal app plan?
    - Autoscaling: when triggered? For how long?
    - Extra hidden costs (Storage account transactions, API connections)?

⚠ Costs are harder to predict on LA standard

⚠ Dev/Test environments (less activity) will probably end up costing more.

# Artifacts

- Schemas & mappings used by workflows. No support for:
  - FF schema/mapping
  - EDI
- Mappings / Schema validations
  - executed locally, no need for Integration Account
- Integration Account
  - ⚠ No possibility to associate Integration Account With Logic App Standard!
  - ⚠ No actions for Integration Account available!
- EDI, FF en-/decoding
  - Call Logic App Consumption to handle this task 👎



# Parameters vs. App settings

**Parameters:**

- Key-value pairs
- Value can be: string, int, float, bool, object, array
- ⚠ No support for “SecureString” and “SecureObject”!
- Accessible via all workflows

```
"inputs": {
    "method": "GET",
    "uri": "@{concat(parameters('hostName'), '/path' )}"
},
```

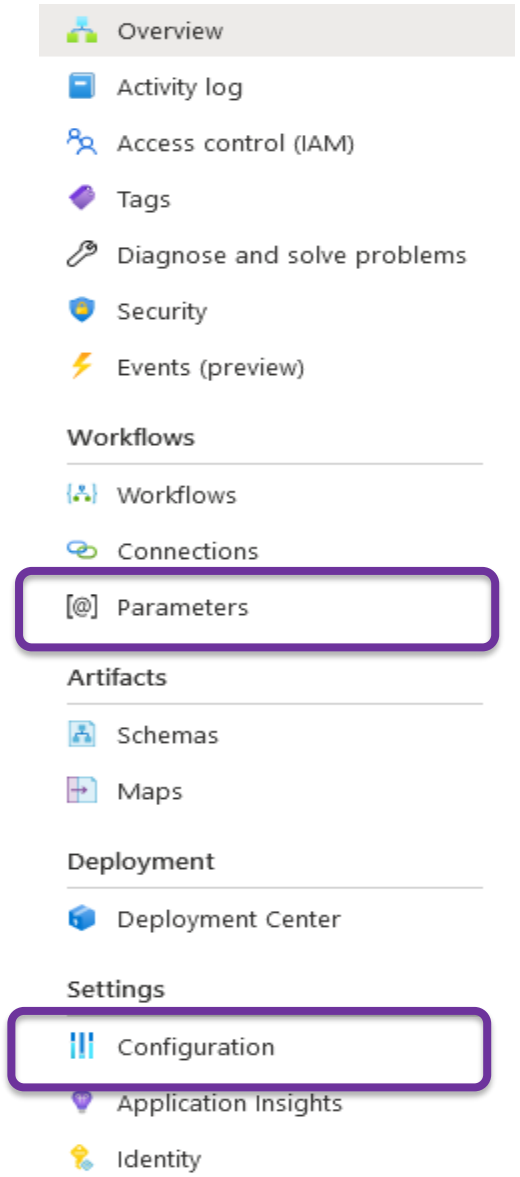
- No KeyVault integration
- Defined in ‘parameters.json’ file
- Deployed via zip deploy (part of application)

**App Settings:**

- Key-value pairs
- Value can only be string
- Accessible via all workflows

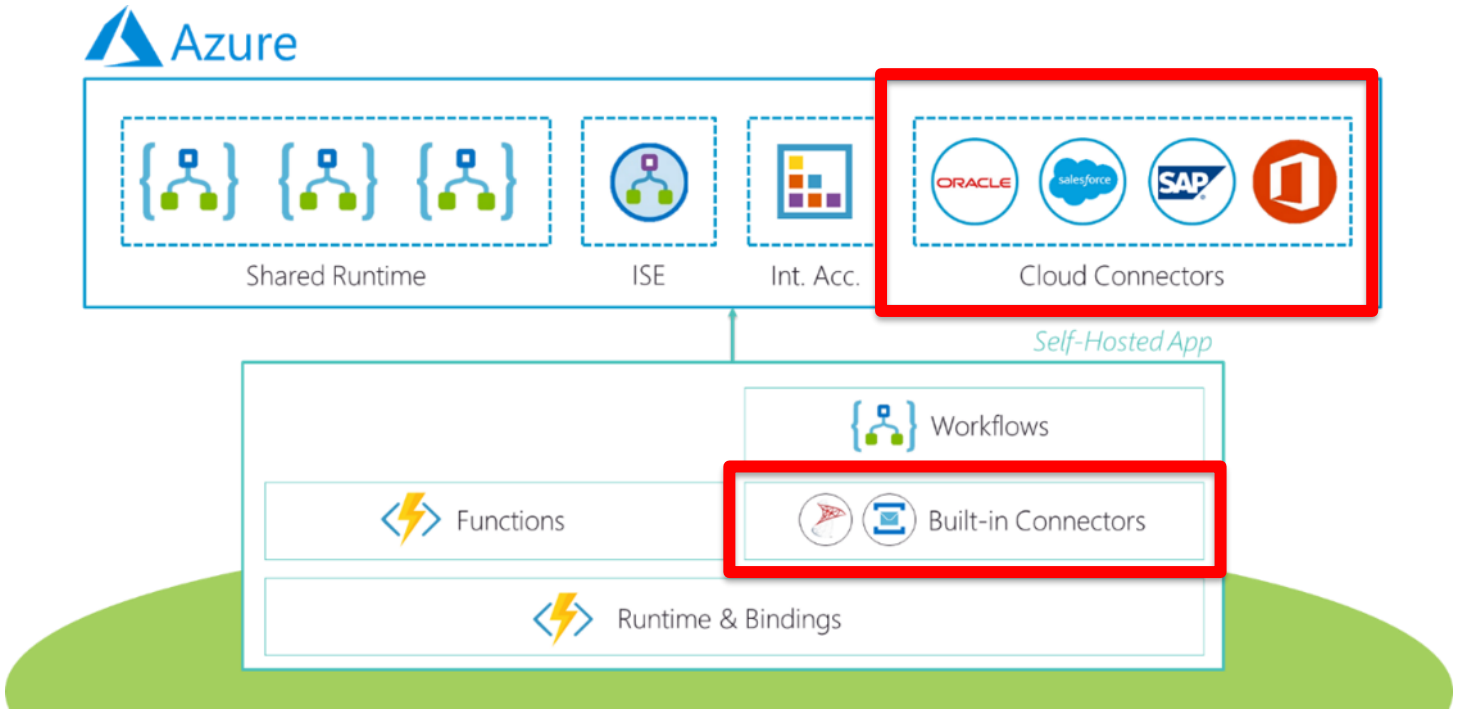
```
"inputs": {
    "method": "GET",
    "uri": "@{concat(appsetting('hostName'), '/path' )}"
},
```

- KeyVault integration 🔑
- Defined in ARM/Bicep
- Deployed via ARM/Bicep (part of infra)



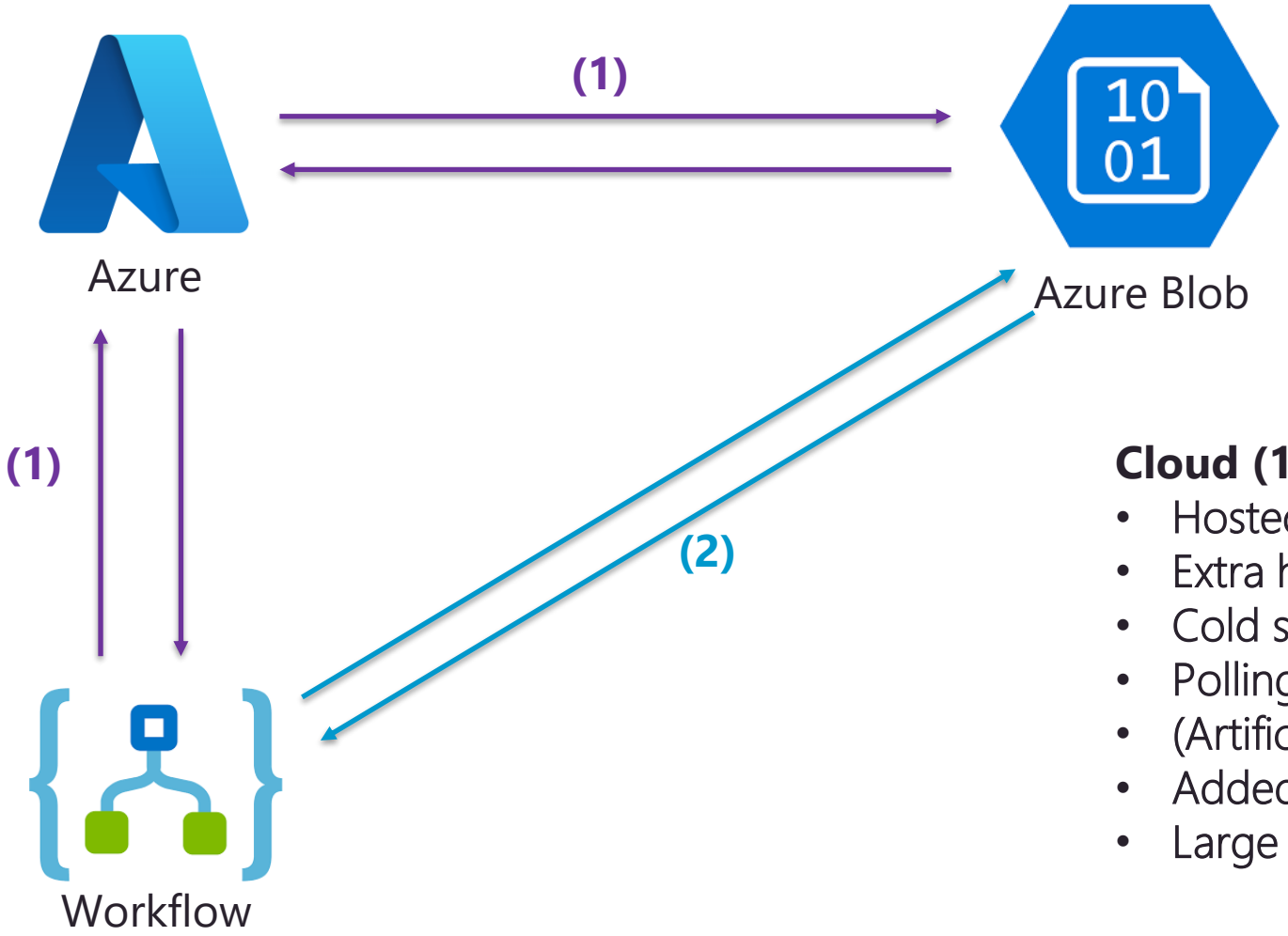
# Logic Apps Standard Connectors: Built-in vs. Cloud

# Built-in vs. Cloud connectors




- Built-in connectors “execute locally”
- Cloud connectors require extra hop to the Cloud (performance!)

# Built-in vs. Cloud connectors




**Cloud (1):**

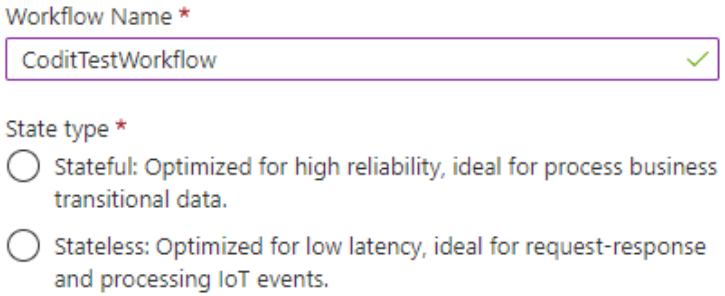
- Hosted in Azure
- Extra hop
- Cold start
- Polling
- (Artificial) Throttling
- Added latency
- Large selection

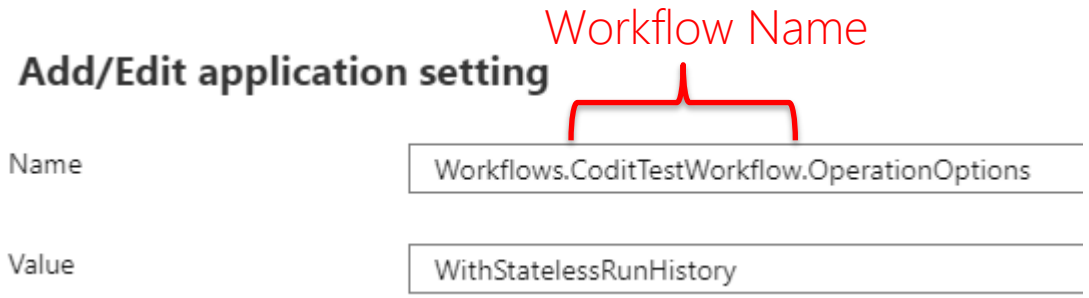
**Built-in (2):**

- Runs locally
- Single hop
- Always on
- Polling + Event listeners
- No (artificial) throttling
- Better performance
- Small selection

# Logic Apps Standard
# Stateful vs. Stateless

# Stateful vs. Stateless workflows




Debug purposes only
(not stable)

Stateful:

- Persistance (StorAcc)
- Resubmits
- Resilience
- Run History
- All triggers
- Lower throughput
- Higher latency

→ Use in most scenarios.

Stateless:

- In-memory
- No Resubmits
- Volatile
- Run History off by default
- Only built-in triggers
- Higher throughput
- Lower Latency

→ Use in specific (high-throughput) scenarios.
(e.g. high volumes of non-critical IoT data)

# Stateless workflows

- Only use **stateless** when:
  - Small messages (<64kb)
  - Non-critical data
  - Need for high performance/throughput
  - Trigger is built-in connector (not cloud)

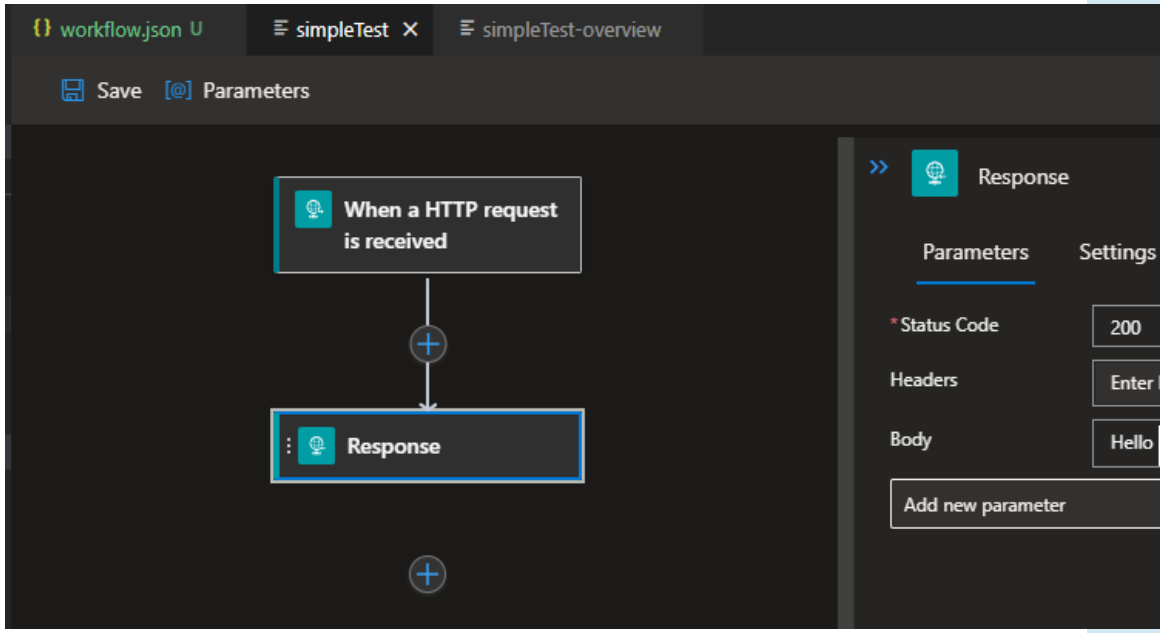
> ⓘ **Important**
>
> A stateless workflow provides the best performance when handling data or content, such as a file, that doesn't exceed 64 KB in *total* size. Larger content sizes, such as multiple large attachments, might significantly slow your workflow's performance or even cause your workflow to crash due to out-of-memory exceptions. If your workflow might have to handle larger content sizes, use a stateful workflow instead.

**⚠Usage of Cloud connectors ➔ Latency & Throttling!**

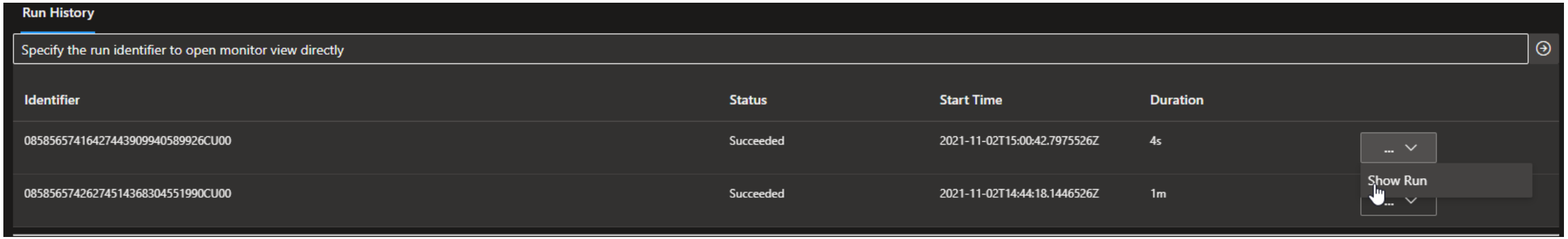
# Logic Apps Standard
# Local Development & Debugging

# Local Development

- VSCode
  - Logic Apps (Standard) extension
- Ability to run locally using Azure Functions Core Tools (F5)
- However, URL doesn't include any sig, api-version, …
  - Can be found in 'overview'
  - Overview – Run History
- Inspect local runs in a similar way to the portal
- Ability to set break points
  - Only set breakpoints before and after an action (not in)
  - Complex structure, not readable, …

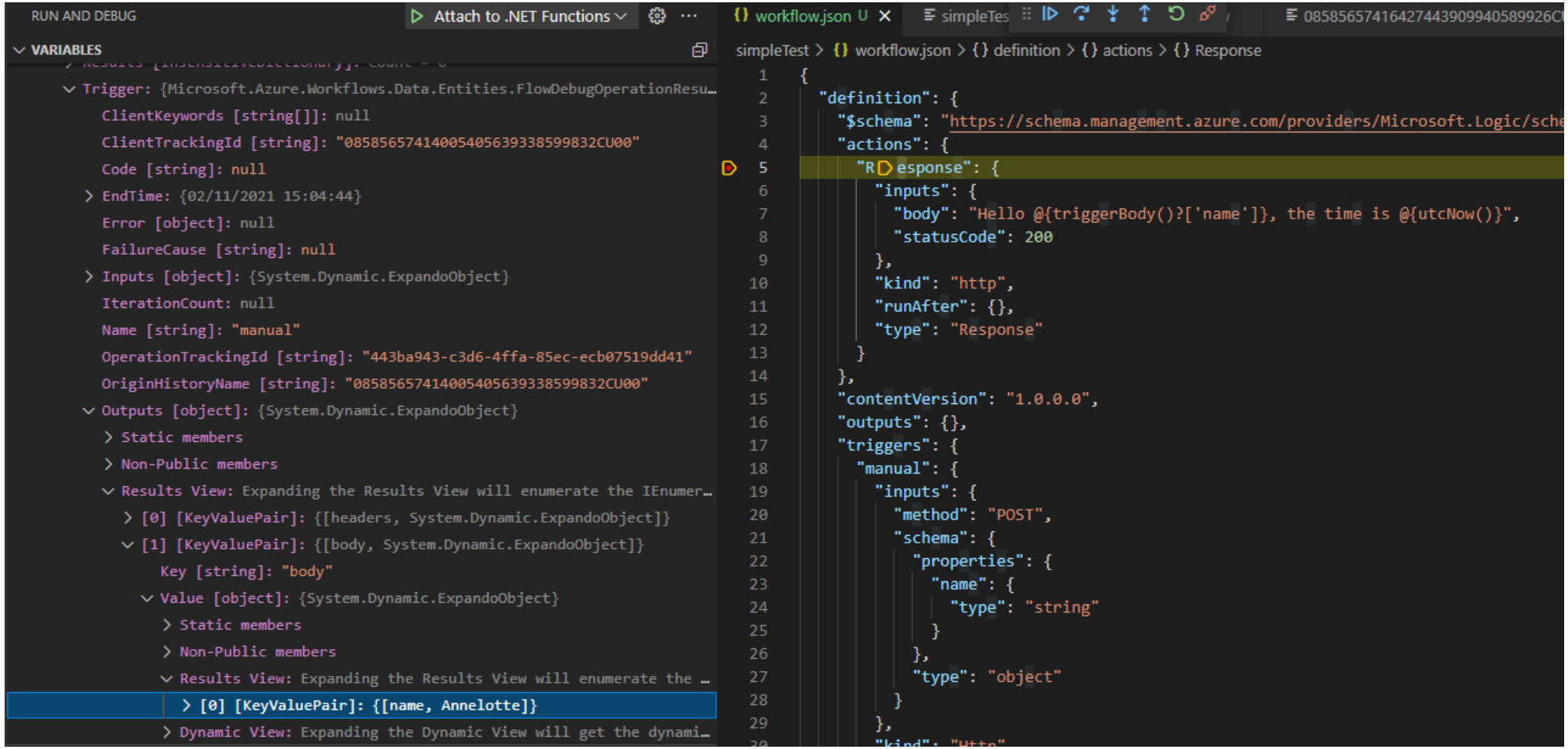



```
Functions:

        simpleTest: [HEAD,POST] http://localhost:7071/api/simpleTest/triggers/manual/invoke

        WorkflowDispatcher: edgeWorkflowRuntimeTrigger

For detailed output, run func with --verbose flag.
[2021-11-02T14:57:29.481Z] Host lock lease acquired by instance ID '000000000000000000000000004369392F'.
```



# Local Debugging

- Ability to set break points



# Demo

Local development (vscode and docker)

# Logic Apps Standard Network integration

# Network integration

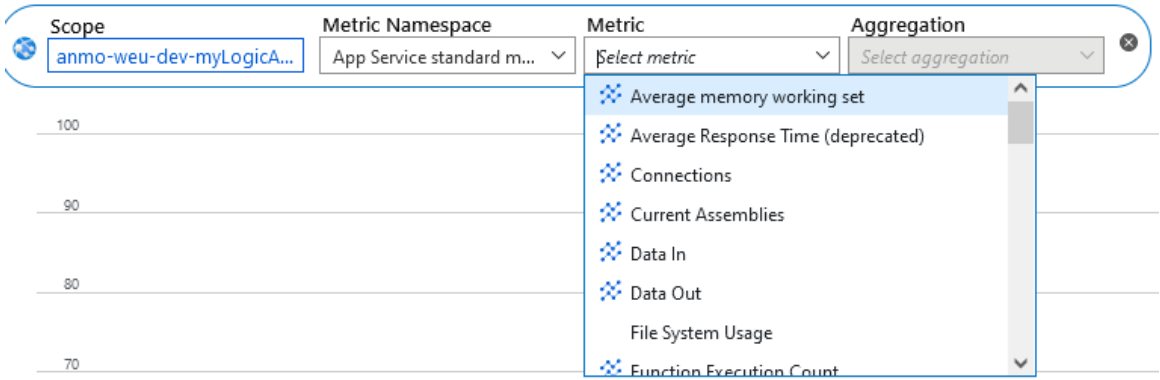
- Inbound
  - IP restriction (IPs, Service Tags, VNET / Service Endpoint)
  - Private endpoint / Private Link
  - Triggered by VNET resources
- Outbound
  - VNET integration (Regional, Gateway)
  - Hybrid connections (Relay)
  - IP restrictions (force all the traffic to be sent into your virtual network)

| Feature | Workflow plan | ASE | Kubernetes |
|---|---|---|---|
| Inbound IP restrictions and private site access | ✅ Yes | ✅ Yes | ✅ Yes |
| Virtual network integration | ✅ Yes (Regional) | ✅ Yes | ✅ Yes |
| Virtual network triggers (non-HTTP) | ✅ Yes | ✅ Yes | ✅ Yes |
| Hybrid connections (Windows only) | ✅ Yes | ✅ Yes | ✅ Yes |
| Outbound IP restrictions | ✅ Yes | ✅ Yes | ✅ Yes |

# Logic Apps Standard Monitoring

# Logic App standard: Metrics

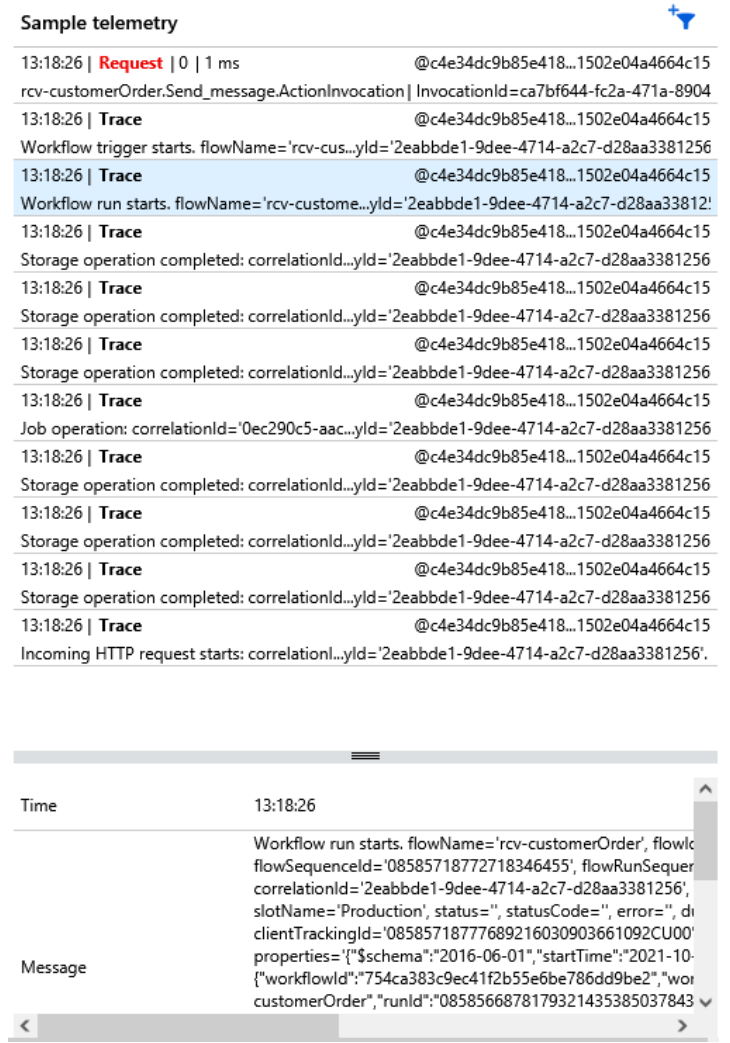
- Only “App service like” metrics
  - CPU/Memory
  - HTTP xxx
  - Response times
  - …




⚠ No specific metrics (yet) for runs/actions/triggers (started/completed/failed), …

➔ Major issue for Production workloads! (Metrics, Alerts, …)

# Application insights – live metrics

# Application insights – Performance / Failures

## Split per action (job)

| OPERATION NAME | DURATION (AVG) | COUNT | PIN |
|---|---|---|---|
| Overall | 280 ms | 6 | |
| rcv-customerOrder.manual.TriggerInvocation | 379 ms | 2 | |
| rcv-customerOrder.Transform_JSON_To_JSON.ActionInvocation | 317 ms | 2 | |
| rcv-customerOrder.Send_message.ActionInvocation | 143 ms | 2 | |

| OPERATION NAME | COUNT (FAILED) | COUNT | PIN |
|---|---|---|---|
| Overall | 2 | 6 | |
| rcv-customerOrder.Send_message.ActionInvocation | 2 | 2 | |

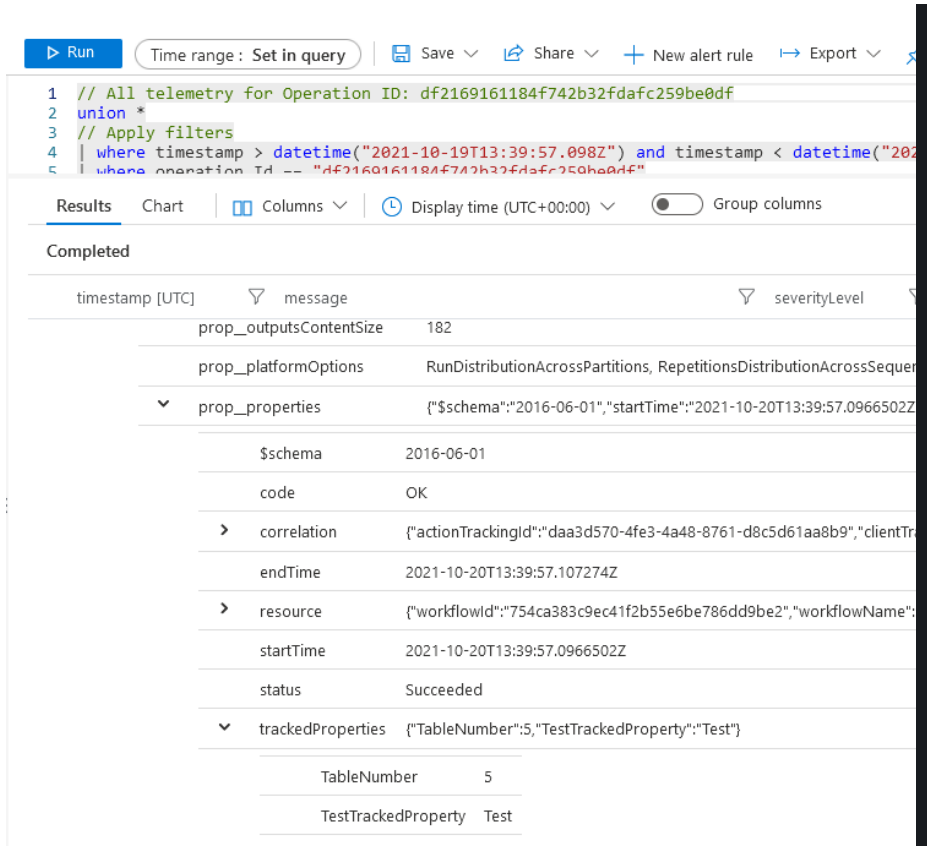
Search results | Learn more | Copy link | Feedback | Leave preview

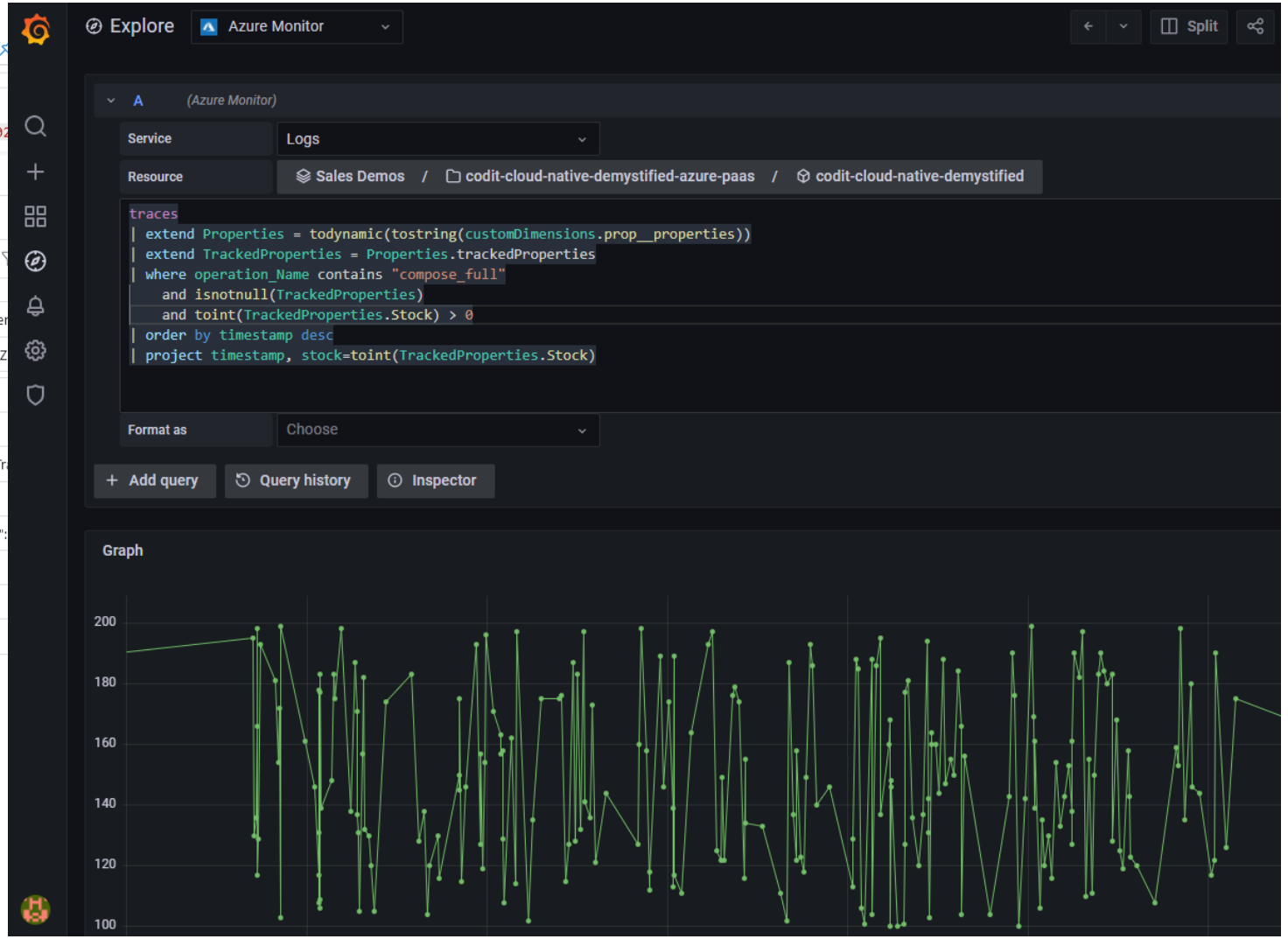
**End-to-end transaction**
Operation ID: 732c9c541009004b8f8202b425d6d9b5

= Request (incoming)

| EVENT | RES. | DURATION | 0 | 1 S | 2 S |
|---|---|---|---|---|---|
| anmo-weu-dev-mylogicapp Microsoft.AspNetCore.Hosting.HttpRequestIn | 0 | 712.3 ms | | | |
| anmo-weu-dev-mylogicapp rcv-customerOrder.Transform_JSON_To_JS | 0 | 626.7 ms | | | |
| anmo-weu-dev-mylogicapp rcv-customerOrder.Send_message.ActionIn | 0 | 285.6 ms | | | |

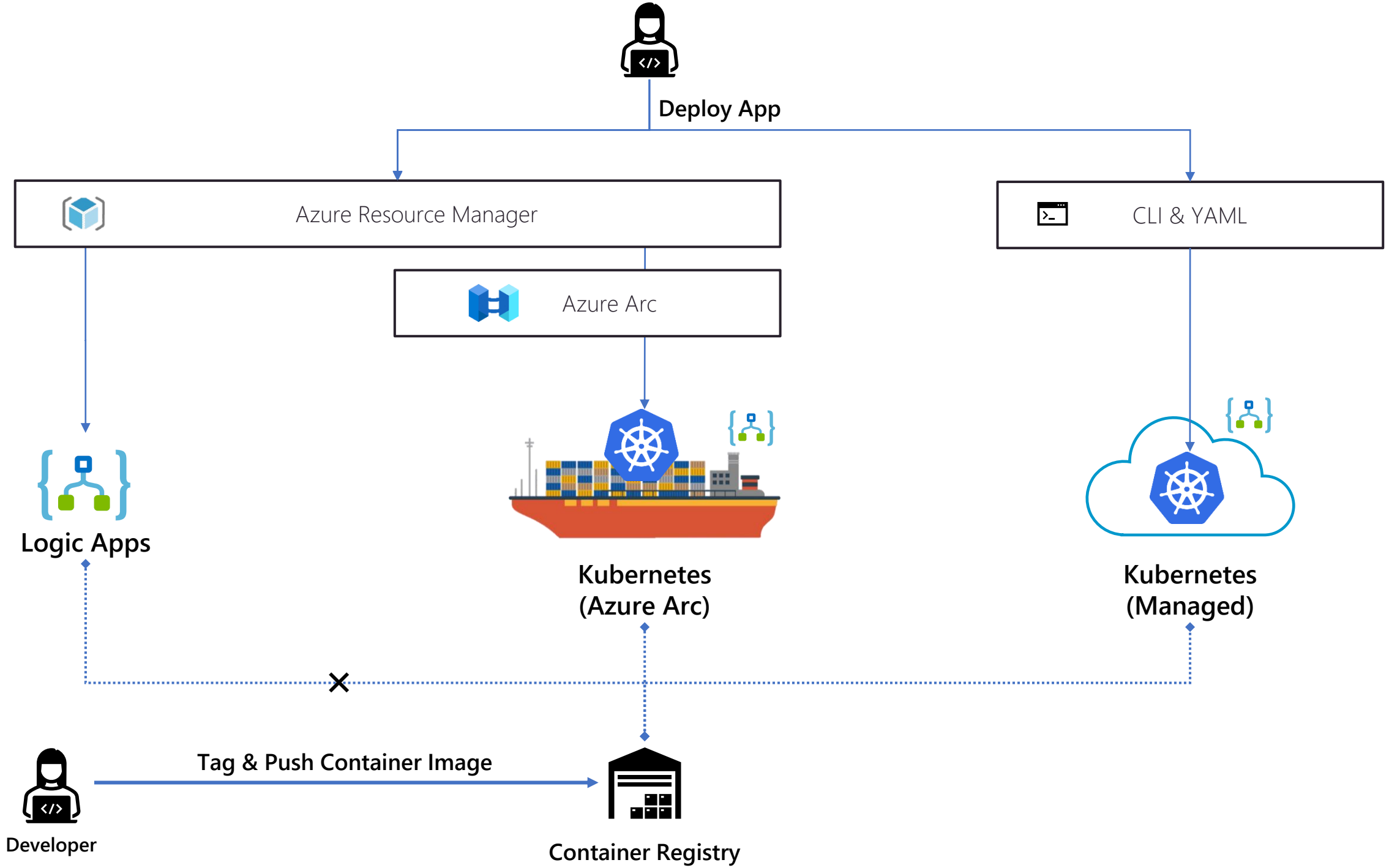
# Application insights – Tracked properties



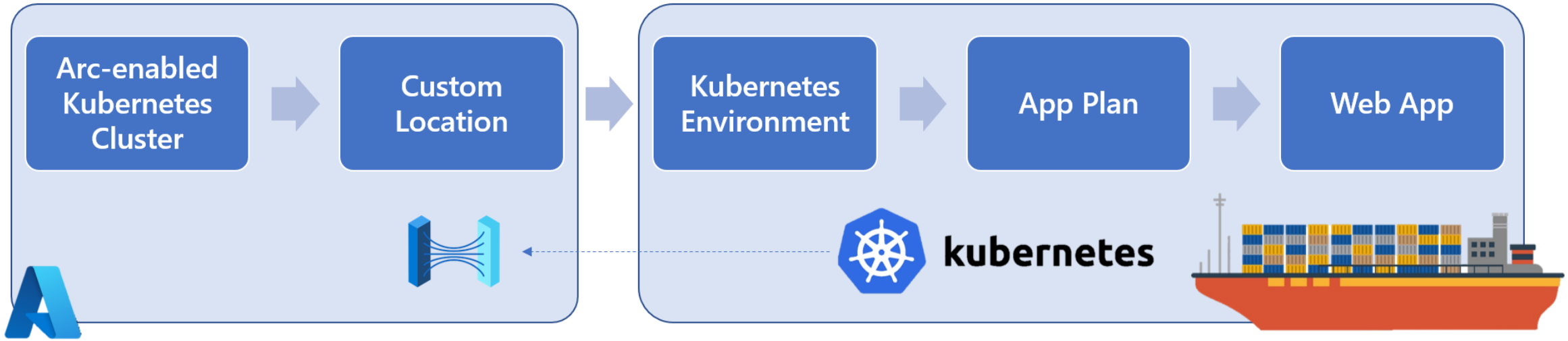
# Demo

Deploy containerized integration on to Azure PaaS, Kubernetes or anywhere


Deploy App
Azure Resource Manager
CLI & YAML
Azure Arc
Logic Apps
Kubernetes
(Azure Arc)
Kubernetes
(Managed)
Tag & Push Container Image
Developer
Container Registry

# Enterprise Integration on Azure ARC

- Same deployment model
- Target a custom location



# Conclusions

# Logic Apps Standard: yay or nay?

- Flexibility and Control
- Run Anywhere
- Network Integration
- Performances
  - built-in connectors
  - stateless
- Local debugging
- KeyVault integration

- Unstable behavior in some cases
- Cost harder to predict
- No offer (yet) for Dev/Test
- Limited options for built-in connectors
- Limited metrics
- Local debugging can be hassle to setup

# Thank you