Software used

Software deployed to Azure

FaçadeApi	App Service (Web API)
EventBroker	Function App
ProcConfigBuilder	App Service (Web API)
ProcessingExec	App Service (Web API)
ClientNotifier	App Service (Web API)
*ReqEventViewer	App Service (Web API)
*RespEventViewer	App Service (Web API)
Repository	App Service (Web API)

*The ReqEventViewer and RespEventViewer are clones of the Azure Event Grid Viewer (https://github.com/Azure-Samples/azure-event-grid-viewer), with a modification to the index.cshtml file the word “Requests” or “Responses” appears after “Azure Event Grid Viewer” on the displayed page.

Software executed on local machine

PostMan	https://www.postman.com/
AzureRelayListener	.NET Console app, based on https://learn.microsoft.com/en-us/azure/azure-relay/relay-hybrid-connections-dotnet-get-started
TestClientRepo	.NET Web API to inspect the Repository contents

Supporting software, built into deployed software

CommonModels	Class library of models used across solutions
EventGridPublishClient	Class library for a wrapper class around Microsoft's EventGridPublisherClient
EventHubPublisherClient	Class library for a wrapper class around Microsoft's EventHubProducerClient
RelayPublishClient	Class library for a wrapper class around Microsoft's HybridConnectionClient
RepoClient	Class library for remote access to the Repository
WebHookAbstraction	Abstract class for implementation by APIs implementing the Web Hook interface
FakeDatabase	Simple, hard-coded implementation of a fake database

Construction

- Deploy the ReqEventViewer, RespEventViewer and Repository to Azure. Stop them.
- Deploy An Azure Relay.
    - Create a Hybrid Connection with “Requires Client Authorization” selected
    - For the connection, in Share access policies, create two policies:
        - sendpolicy, with Send Claims
        - managepolicy, with Manage, Send, Listen claims
- Update the Relay settings in appsettings.json for AzureRelayListener.

```
{
  "RelaySettings": {
    "RelayNamespace": "<your relay namespace>.servicebus.windows.net",
    "ConnectionName": "<your hybrid connection name>",
    "KeyName": "managepolicy",
    "Key": "your key value"
  }
}
```

Where:

- Your-namespace is the Azure Relay instance
- Your-connection is the hybrid connection name
- KeyName is the connection’s Shared Access Policy to use (managepolicy)
- Key is the Primary Key for the KeyName

- Deploy the 4 APIs: Façade, ProcConfigBuilder, ProcessingExec, ClientNotifier. Stop them.
- Configure the Repository settings for Façade, ProcConfigBuilder, ProcessingExec (set “repo_url” to the “Default domain” of the Repository, prefixed with https://)
- Instantiate the three Event Grid Topics: UserRequestTopic, ProcConfigCreatedTopic, ProcessingCompleteTopic (enter the Resource group, Name and Region, but leave all other settings unchanged).
- ***(Note that subscriptions can’t be created until the subscribing services are configured and running)***
- ***For each subscription, select a Web Hook type and specify https:// + the “Default domain” from the subscribing API’s Overview page + /api/ + the***

controller name (without including the word "controller"). For the RequestEventGridViewer and ResponseEventGridViewer the controller is "UpdatesController".

- Configure the Relay settings for ClientNotifier
    - connectionName is the hybrid connection name
    - keyName is the connection's Shared Access Policy to use (sendPolicy)
    - key is the Primary Key for the keyName
    - relayNamespace is the name of the relay followed by ".servicebus.windows.net"
- For ProcConfigBuilder, set "EventGridTopicEndpoint" as the Topic Endpoint for the ProcConfigCreatedTopic, and "EventGridTopicKey" as the value of one of the Keys for the Topic.
- Start ProcConfigBuilder. For UserRequestTopic, create 2 Subscriptions: the first for ProcConfigBuilder, the second for RequestEventGridViewer.
- Start ClientNotifier. For ProcessingCompleteTopic, create 2 Subscriptions: the first for ClientNotifier, the second for ResponseEventGridViewer.
- Stop ProcConfigBuilder, ClientNotifier.
- Create the Event Hub Namespace: Basic tier, 1 throughput unit.
- Create a new Shared access policies for the Event Hub Namespace:
    - "listener", with Listen Claims
- Create an Event Hub with 2 partitions. Use only the provided Consumer Group ($Default). Disable the Hub.
- Create a new Shared access policies for the Event Hub:
    - "sender", with Send Claims
- Create the Event Broker Function. Stop it. Configure the TopicEndpoint (from the Topic Overview page) and TopicKey (from the Access keys page) settings for the two Event Grid Topics – UserRequestTopic, ProcessingCompleteTopic. Name the settings as:
    - RequestEventGridTopicEndpoint
    - RequestEventGridTopicKey
    - ResponseEventGridTopicEndpoint

  - ResponseEventGridTopicKey

- For the Event Broker Function, configure “EventHubConnection” setting to match the Event Hub **Namespace** connection string for the “listener” policy, with the following appended:

  ;EntityPath=<your Event Hub name>

- Configure the Event Hub settings for the Façade API, Processing Exec:

  - eh_name – the name of the Event Hub instance

  - eh_partition_id – the default partition to use (“0” for Façade, “1” for ProcessingExec)

  - ehns_connstring – the Event Hub connection string for the “sender” policy

- Start ProcessingExec. For ProcConfigCreatedTopic, create a Subscription for ProcessingExec.