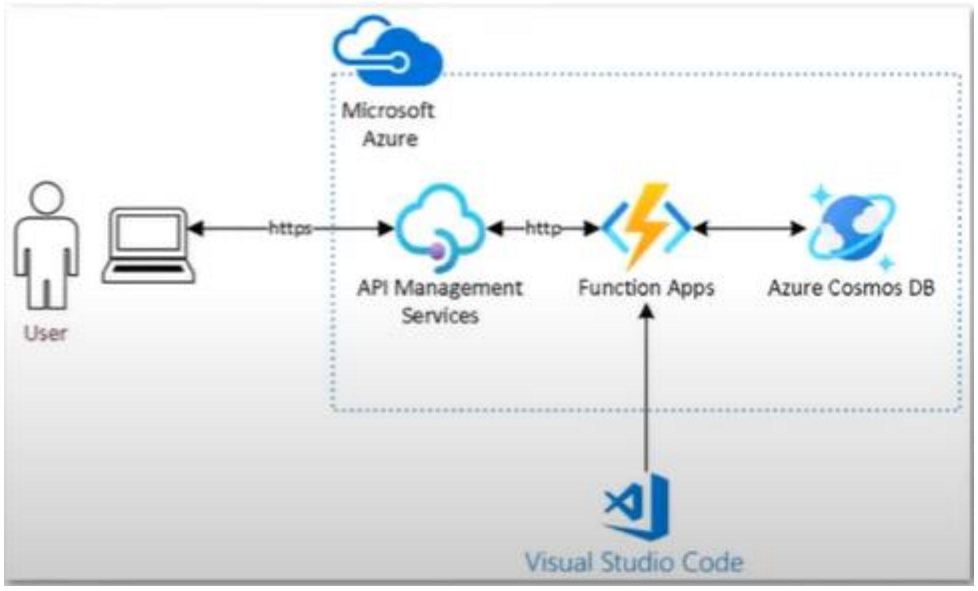
Azure APIM

Azure API Management (APIM) integrates existing back-end services into modern **API gateways.**



It follows the API-first approach, **decoupling front-end** and **back-end** teams with the help of API mocking.

Azure API Management handles the full management of APIs.

It **centralizes** the security, versioning, documentation, and compliance of your **back-end - services** in a **single** point.

KEY CONCEPTS:

- API represents a set of **operations**. API Operation connects an API endpoint to its backend.
- Product: A logical grouping of APIs
- A single or a group of APIs make up a product, which is how APIs are presented to developers. It can be either public or private.
- Backend represents back-end services in API.
- Group, used to manage the visibility of the products to developers:
  - **Administrators** have full access to the API management.
  - **Developers**, users with access to the developer's portal with permissions to build applications.
  - **Guests,** users without access to the developer's portal but with reading permissions in some services.
- **Developer** belongs to one or more Product groups, and each developer has a primary and secondary key to call the product's APIs.

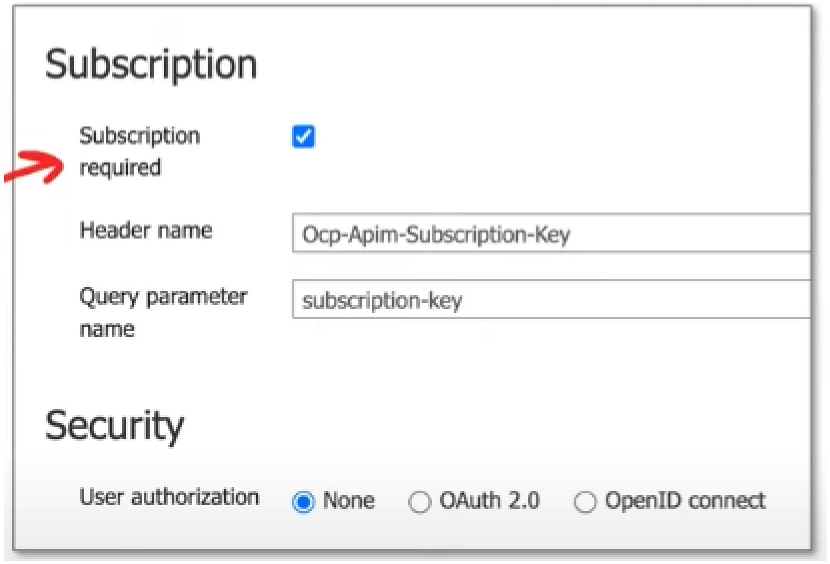
- **Policies:** configurations and validations that are applied in progress to incoming requests and the outcome responses.
- **Named Values:** key-value pairs used with policies. Valued can be a result of an expression.
- **Gateway:** where API calls are received, and policies are applied to incoming requests.
- **Developer Portal:** where developers can access all APIs and products listed by APIM alongside its API's operation and documentation. Developers can also request access to APIs from the developer's portal.

Echo API service

When the APIM gateway is created by default Echo API is created which is a non-production Azure service that is used to test Azure API Management.

API Authentication

In order to authenticate with our APIs, we configure those settings under the subscription section.



Subscription

Subscription required: ✓

Header name: Ocp-Apim-Subscription-Key

Query parameter name: subscription-key

Security

User authorization: [✓ None] OAuth 2.0 OpenID connect

If the subscription is required, only developers with a valid access key can use it.

If it is not checked, **anonymous** requests are allowed.

Here, we can configure where the API will receive the access keys, which can be sent as a header or query string.

Groups

Groups are used to manage the visibility of products to developers.

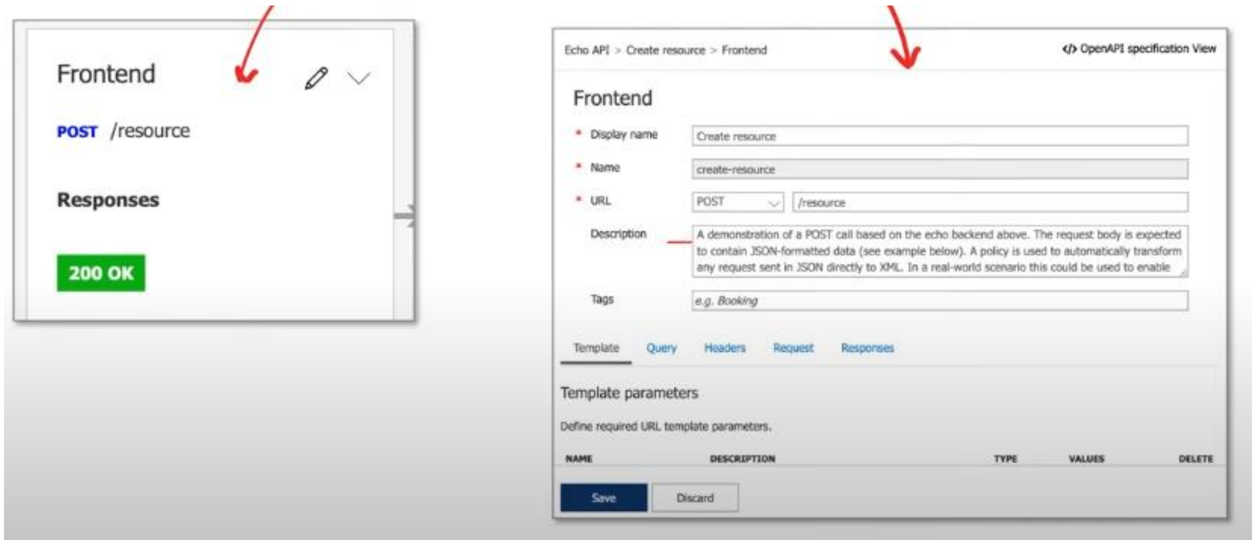
1. Administrators: Manage API management service instances and create the APIs, operations, and products that are used by developers.
2. Developers: Authenticated developer portal, users that build applications using APIs. Developers are granted access to the developer portal and build applications that call the operations of an API.
3. Guests: Unauthenticated developer portal users, such as prospective customers visiting the developer portal. They can be granted certain read-only access, such as the ability to view APIs but not to call them.

Administrators can also create custom groups or use external groups in an associated Azure Active Directory tenant to give developers visibility and access to API products.

A user can belong to more than one group.

Frontends

Frontends define the **route/endpoint** and the **documentation and configuration** around that endpoint.


Frontend

POST /resource

Responses

200 OK

Echo API > Create resource > Frontend

</> OpenAPI specification View

Frontend

* Display name	Create resource
* Name	create-resource
* URL	POST /resource
Description	A demonstration of a POST call based on the echo backend above. The request body is expected to contain JSON-formatted data (see example below). A policy is used to automatically transform any request sent in JSON directly to XML. In a real-world scenario this could be used to enable
Tags	e.g. Booking

Template parameters

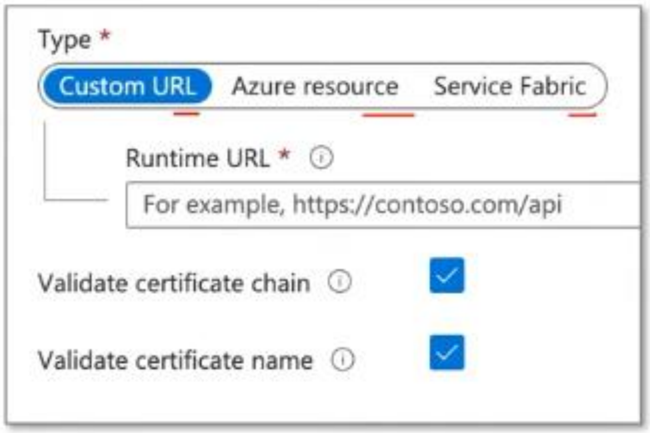
Define required URL template parameters.

NAME	DESCRIPTION	TYPE	VALUES	DELETE

API does not host APIs; it creates facades for APIs.

Backends

For Backends, we have:


Type *

Custom URL | Azure resource | Service Fabric

Runtime URL *

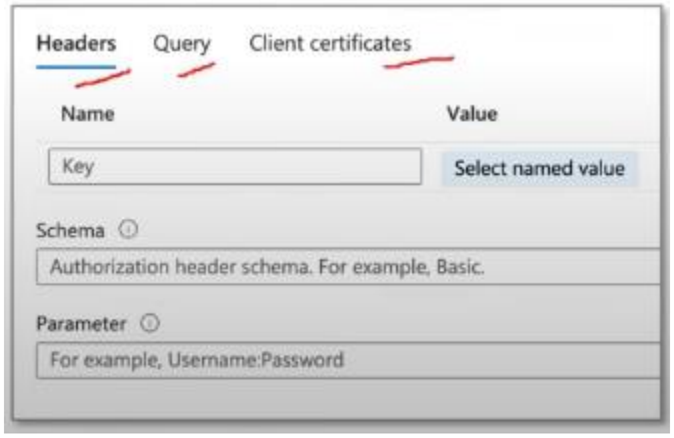
For example, https://contoso.com/api

Validate certificate chain ✓

Validate certificate name ✓

- Custom URL: point to the server where the service is running.
- Azure Resource: Integrate directly to an Azure resource eg.
    - Azure Functions
    - App Service
    - Container App
    - Logic App
- Azure Service Fabric

Authorization credentials present authorization request credentials to the backend service.


Headers | Query | Client certificates

Name	Value
Key	Select named value

Schema

Authorization header schema. For example, Basic.

Parameter

For example, Username:Password

- Headers: HTTP headers
    - Can fetch from Named values.
- Query: query string
    - Can fetch from Named values.
- Client certificates: x.509 certificates
    - Certificates stored in Azure key vault.

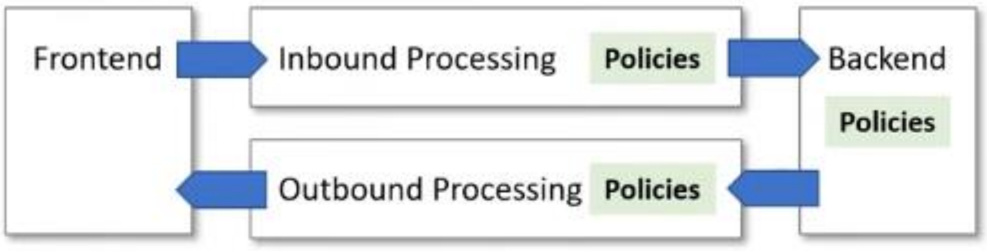
Policies

API Management Policies allow to change the behavior at multiple stages of an endpoint's request lifecycle.

Any part of the request and response messages can be updated eg. headers, body, URLs, etc.

These are four areas where policies can be applied:

- Inbound: for incoming requests.
- Backend: before requests reach to backend.
- Outbound: before sending a response back to the client.
- Error: when a request encounters an error.



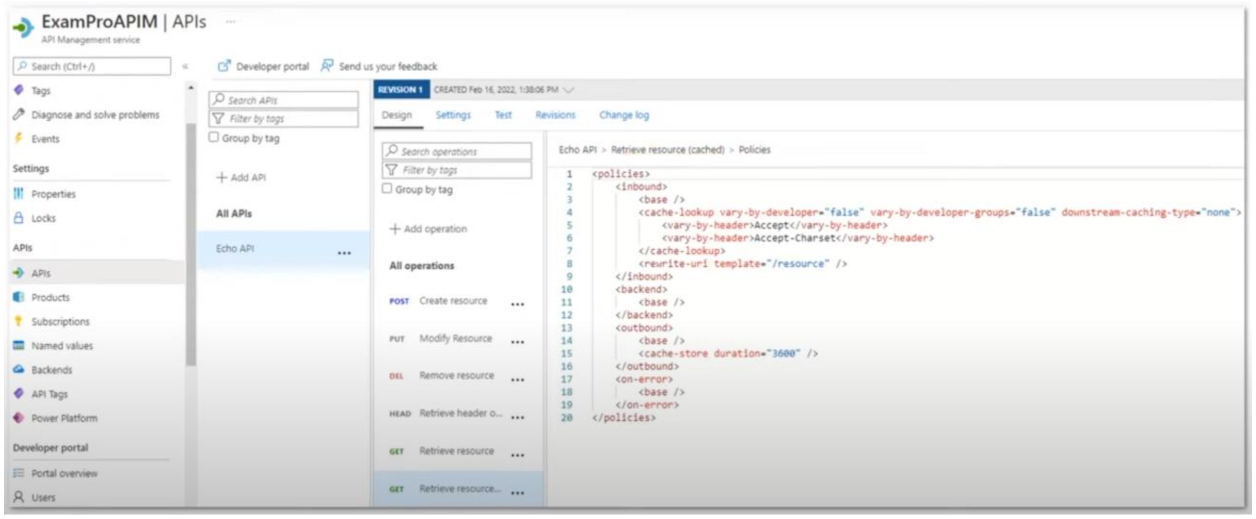
Azure has a collection of policy groups which contain many policies we can apply:

- Access restriction Policies
- Advanced Policies
- Authentication Policies
- Caching Policies
- Cross-Domain Policies
- Transformation Policies
- Dapr Integration Policies
- Validation Policies
- Graph QL Validation Policies

When an error occurs, no other policies are applied except the error policies; however, if other policies were in effect before the error, they will not be removed.

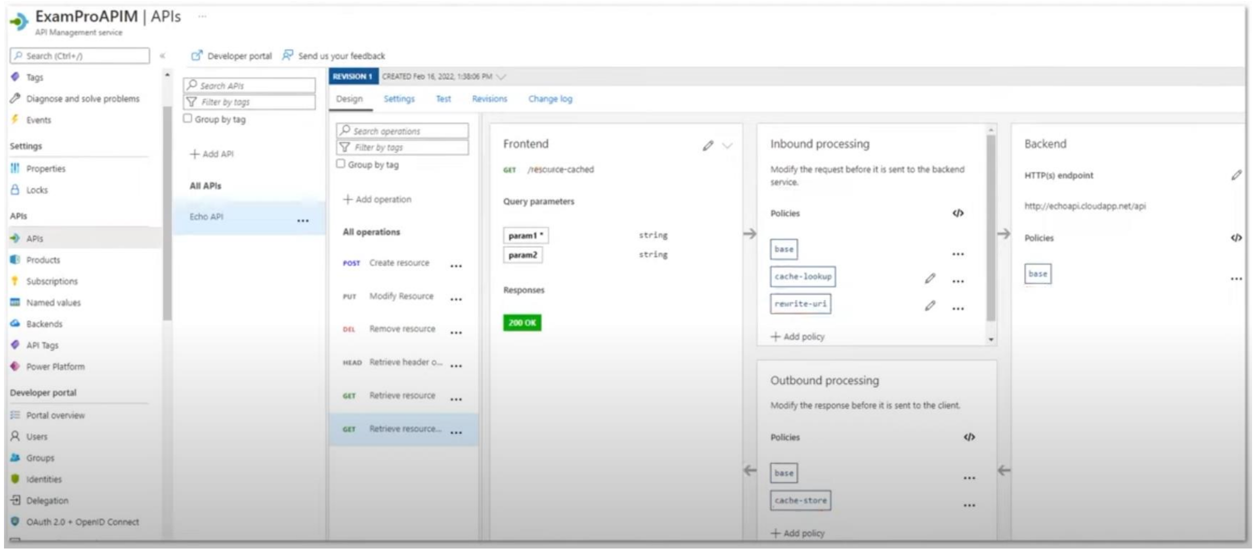
Product-level policies apply to all API operations within a product.

APIM Authentication Policies Example:



Echo API > Retrieve resource (cached) > Policies

```
<policies>
    <inbound>
        <base />
        <cache-lookup vary-by-developer="false" vary-by-developer-groups="false" downstream-caching-type="none">
            <vary-by-header>Accept</vary-by-header>
            <vary-by-header>Accept-Charset</vary-by-header>
        </cache-lookup>
        <rewrite-uri template="/resource" />
    </inbound>
    <backend>
        <base />
    </backend>
    <outbound>
        <base />
        <cache-store duration="3600" />
    </outbound>
    <on-error>
        <base />
    </on-error>
</policies>
```



Frontend: GET /resource-cached

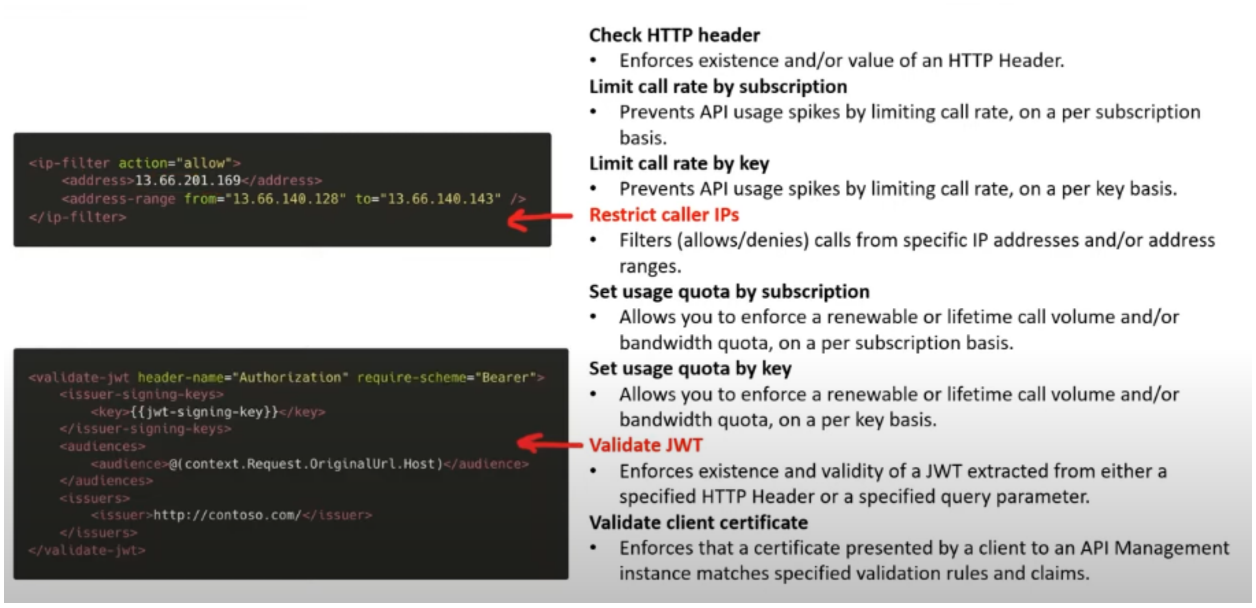
Query parameters: param1 * string; param2 string

Inbound processing: Modify the request before it is sent to the backend service. Policies: base, cache-lookup, rewrite-uri

Backend: HTTP(s) endpoint http://echoapi.cloudapp.net/api Policies: base

Outbound processing: Modify the response before it is sent to the client. Policies: base, cache-store

Access Restriction Policies



```
<ip-filter action="allow">
    <address>13.66.201.169</address>
    <address-range from="13.66.140.128" to="13.66.140.143" />
</ip-filter>
```

```
<validate-jwt header-name="Authorization" require-scheme="Bearer">
    <issuer-signing-keys>
        <key>{{jwt-signing-key}}</key>
    </issuer-signing-keys>
    <audiences>
        <audience>@(context.Request.OriginalUrl.Host)</audience>
    </audiences>
    <issuers>
        <issuer>http://contoso.com/</issuer>
    </issuers>
</validate-jwt>
```

Check HTTP header
- Enforces existence and/or value of an HTTP Header.

Limit call rate by subscription
- Prevents API usage spikes by limiting call rate, on a per subscription basis.

Limit call rate by key
- Prevents API usage spikes by limiting call rate, on a per key basis.

Restrict caller IPs
- Filters (allows/denies) calls from specific IP addresses and/or address ranges.

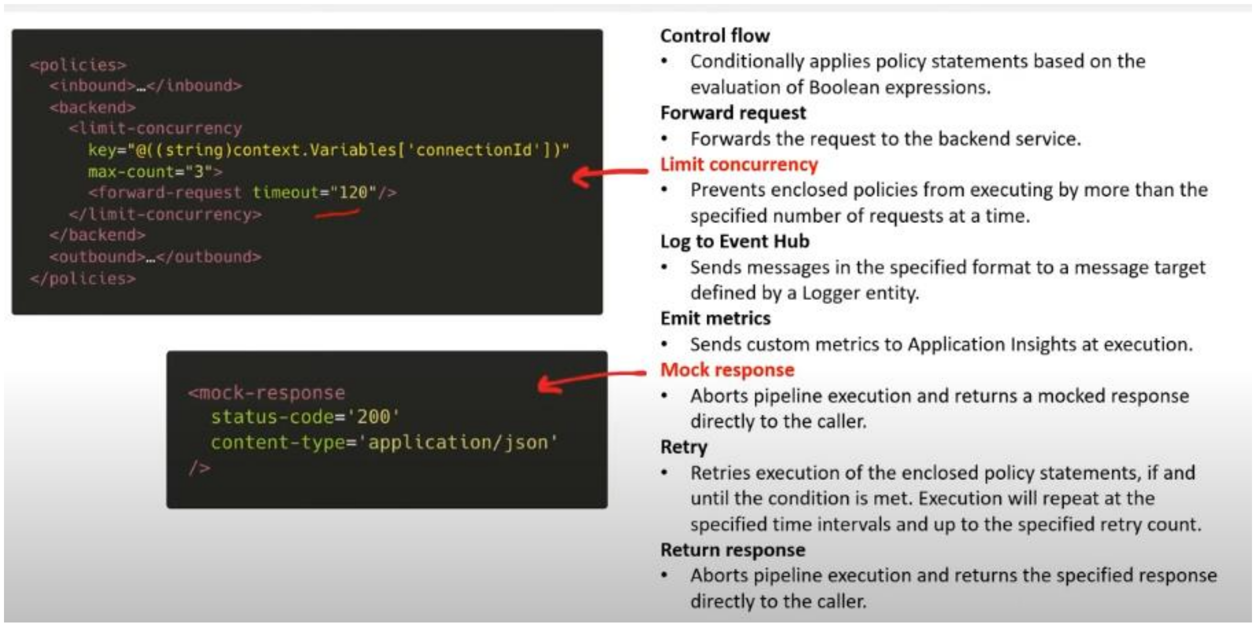
Set usage quota by subscription
- Allows you to enforce a renewable or lifetime call volume and/or bandwidth quota, on a per subscription basis.

Set usage quota by key
- Allows you to enforce a renewable or lifetime call volume and/or bandwidth quota, on a per key basis.

Validate JWT
- Enforces existence and validity of a JWT extracted from either a specified HTTP Header or a specified query parameter.

Validate client certificate
- Enforces that a certificate presented by a client to an API Management instance matches specified validation rules and claims.

Advanced Policies



```
<policies>
  <inbound>…</inbound>
  <backend>
    <limit-concurrency
      key="@((string)context.Variables['connectionId'])"
      max-count="3">
      <forward-request timeout="120"/>
    </limit-concurrency>
  </backend>
  <outbound>…</outbound>
</policies>
```

```
<mock-response
  status-code='200'
  content-type='application/json'
/>
```

Control flow
- Conditionally applies policy statements based on the evaluation of Boolean expressions.

Forward request
- Forwards the request to the backend service.

Limit concurrency
- Prevents enclosed policies from executing by more than the specified number of requests at a time.

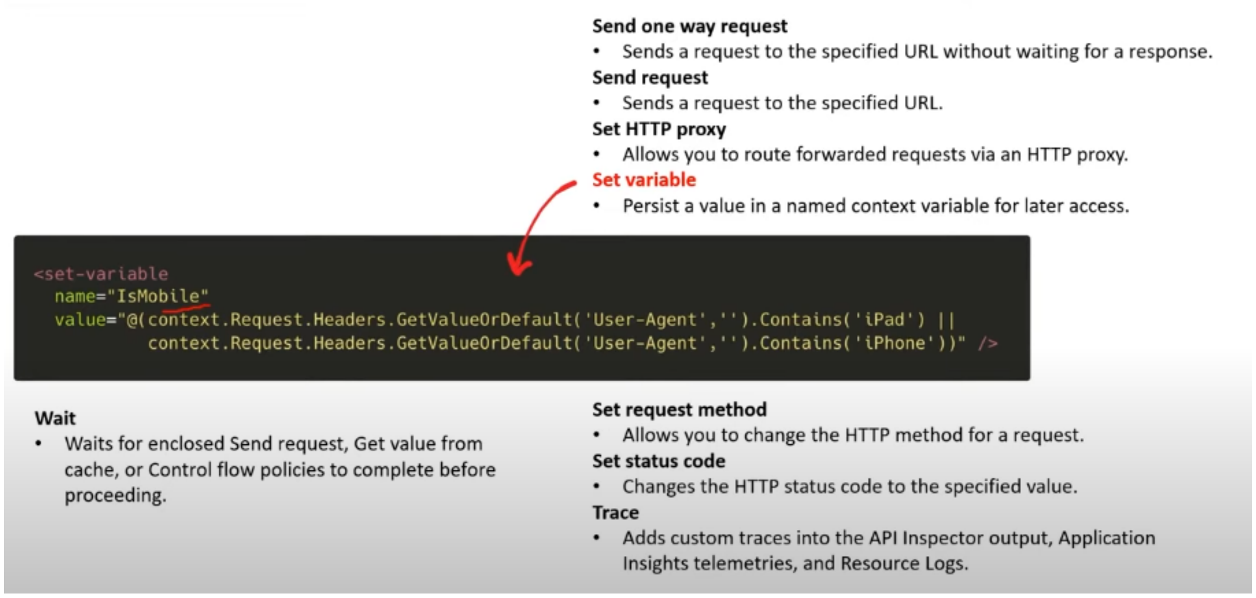
Log to Event Hub
- Sends messages in the specified format to a message target defined by a Logger entity.

Emit metrics
- Sends custom metrics to Application Insights at execution.

Mock response
- Aborts pipeline execution and returns a mocked response directly to the caller.

Retry
- Retries execution of the enclosed policy statements, if and until the condition is met. Execution will repeat at the specified time intervals and up to the specified retry count.

Return response
- Aborts pipeline execution and returns the specified response directly to the caller.



Send one way request
- Sends a request to the specified URL without waiting for a response.

Send request
- Sends a request to the specified URL.

Set HTTP proxy
- Allows you to route forwarded requests via an HTTP proxy.

Set variable
- Persist a value in a named context variable for later access.

```
<set-variable
  name="IsMobile"
  value="@(context.Request.Headers.GetValueOrDefault('User-Agent','').Contains('iPad') ||
          context.Request.Headers.GetValueOrDefault('User-Agent','').Contains('iPhone'))" />
```

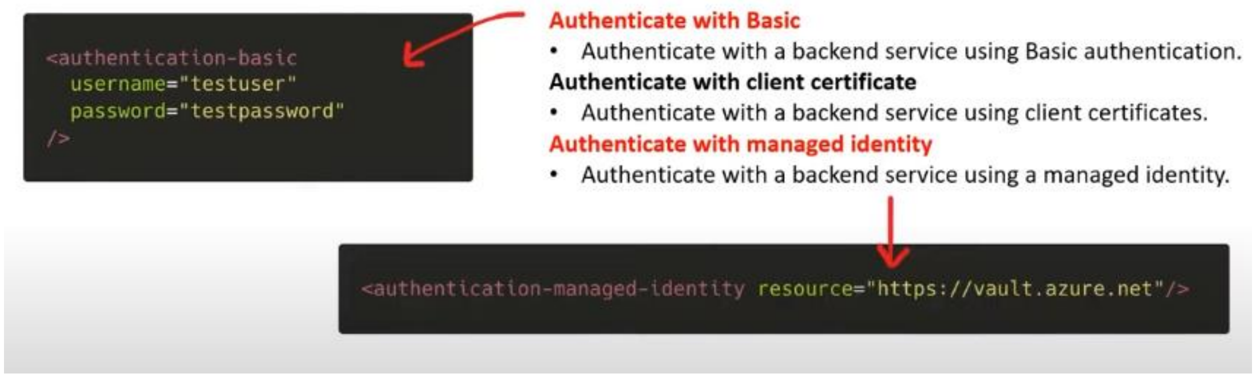
Wait
- Waits for enclosed Send request, Get value from cache, or Control flow policies to complete before proceeding.

Set request method
- Allows you to change the HTTP method for a request.

Set status code
- Changes the HTTP status code to the specified value.

Trace
- Adds custom traces into the API Inspector output, Application Insights telemetries, and Resource Logs.

Authentication Policies



```
<authentication-basic
  username="testuser"
  password="testpassword"
/>
```

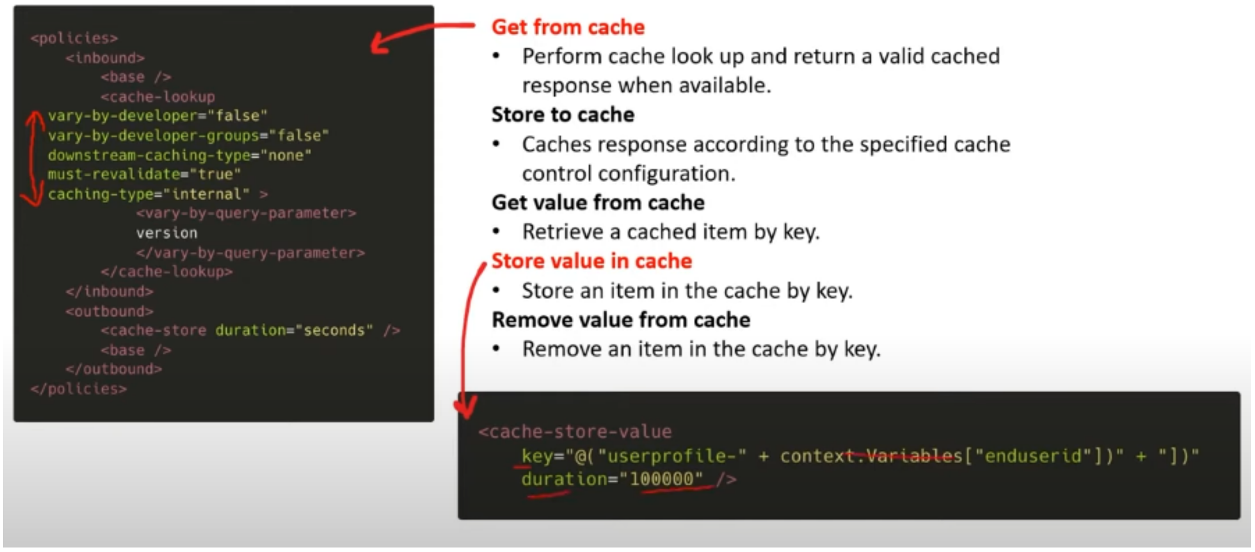
Authenticate with Basic
- Authenticate with a backend service using Basic authentication.

Authenticate with client certificate
- Authenticate with a backend service using client certificates.

Authenticate with managed identity
- Authenticate with a backend service using a managed identity.

```
<authentication-managed-identity resource="https://vault.azure.net"/>
```

Caching Policies



```
<policies>
    <inbound>
        <base />
        <cache-lookup
vary-by-developer="false"
vary-by-developer-groups="false"
downstream-caching-type="none"
must-revalidate="true"
caching-type="internal" >
            <vary-by-query-parameter>
            version
            </vary-by-query-parameter>
        </cache-lookup>
    </inbound>
    <outbound>
        <cache-store duration="seconds" />
        <base />
    </outbound>
</policies>
```

Get from cache

- Perform cache look up and return a valid cached response when available.

Store to cache

- Caches response according to the specified cache control configuration.

Get value from cache

- Retrieve a cached item by key.

Store value in cache

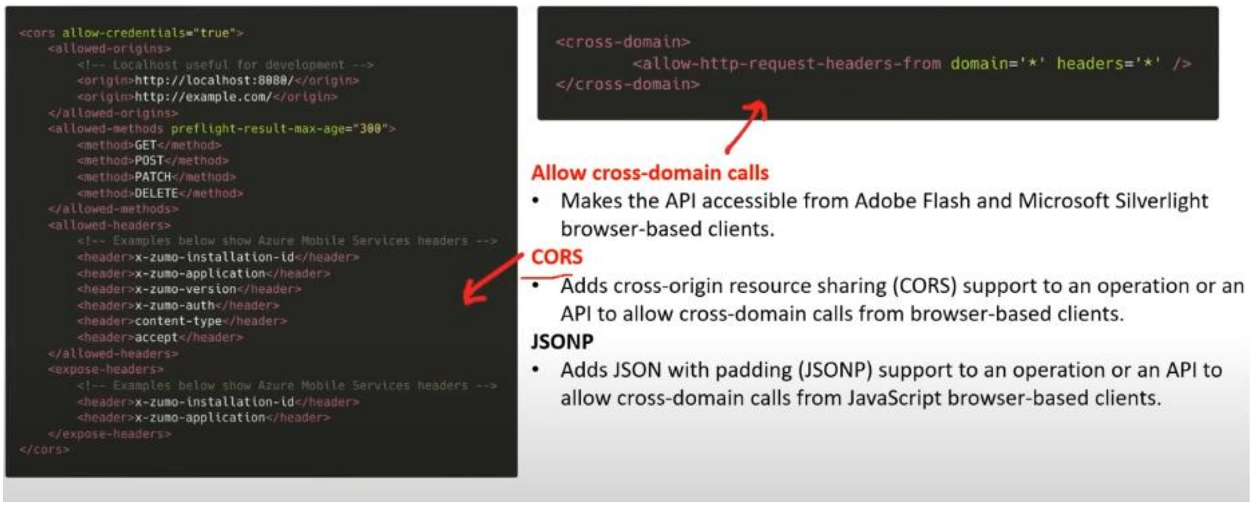
- Store an item in the cache by key.

Remove value from cache

- Remove an item in the cache by key.

```
<cache-store-value
    key="@("userprofile-" + context.Variables["enduserid"])" + "])"
    duration="100000" />
```

Cross Domain Policies



```
<cors allow-credentials="true">
    <allowed-origins>
        <!-- Localhost useful for development -->
        <origin>http://localhost:8080/</origin>
        <origin>http://example.com/</origin>
    </allowed-origins>
    <allowed-methods preflight-result-max-age="300">
        <method>GET</method>
        <method>POST</method>
        <method>PATCH</method>
        <method>DELETE</method>
    </allowed-methods>
    <allowed-headers>
        <!-- Examples below show Azure Mobile Services headers -->
        <header>x-zumo-installation-id</header>
        <header>x-zumo-application</header>
        <header>x-zumo-version</header>
        <header>x-zumo-auth</header>
        <header>content-type</header>
        <header>accept</header>
    </allowed-headers>
    <expose-headers>
        <!-- Examples below show Azure Mobile Services headers -->
        <header>x-zumo-installation-id</header>
        <header>x-zumo-application</header>
    </expose-headers>
</cors>
```

```
<cross-domain>
        <allow-http-request-headers-from domain='*' headers='*' />
</cross-domain>
```

Allow cross-domain calls

- Makes the API accessible from Adobe Flash and Microsoft Silverlight browser-based clients.

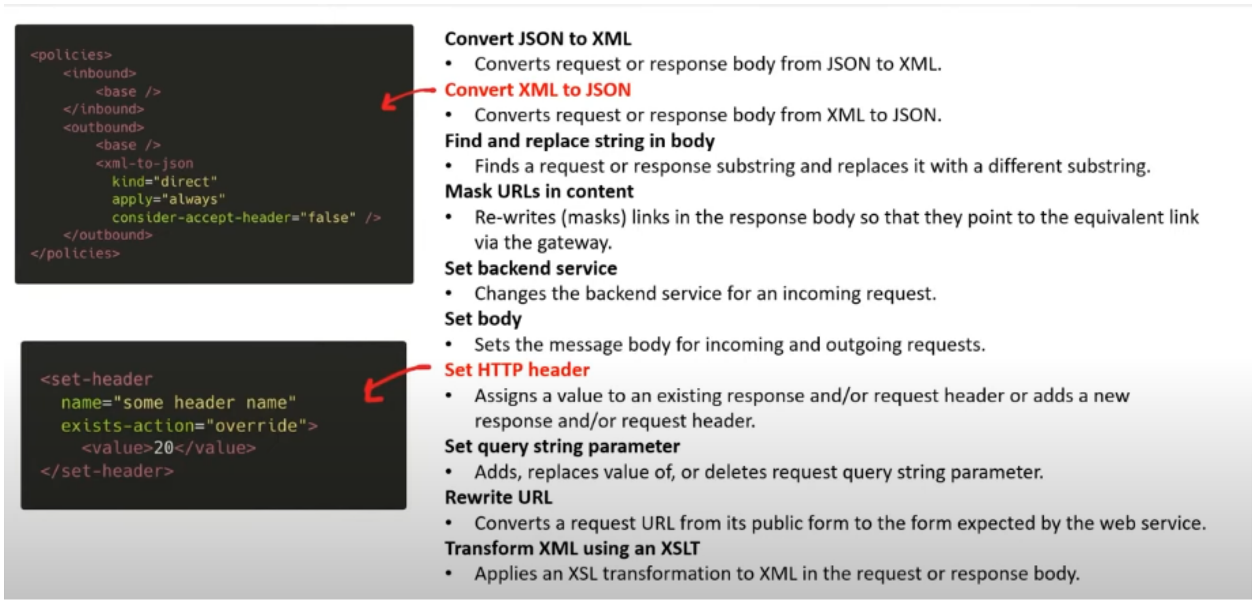
CORS

- Adds cross-origin resource sharing (CORS) support to an operation or an API to allow cross-domain calls from browser-based clients.

JSONP

- Adds JSON with padding (JSONP) support to an operation or an API to allow cross-domain calls from JavaScript browser-based clients.

Transformation Policies



```
<policies>
    <inbound>
        <base />
    </inbound>
    <outbound>
        <base />
        <xml-to-json
          kind="direct"
          apply="always"
          consider-accept-header="false" />
    </outbound>
</policies>
```

```
<set-header
  name="some header name"
  exists-action="override">
    <value>20</value>
</set-header>
```

Convert JSON to XML
- Converts request or response body from JSON to XML.

Convert XML to JSON
- Converts request or response body from XML to JSON.

Find and replace string in body
- Finds a request or response substring and replaces it with a different substring.

Mask URLs in content
- Re-writes (masks) links in the response body so that they point to the equivalent link via the gateway.

Set backend service
- Changes the backend service for an incoming request.

Set body
- Sets the message body for incoming and outgoing requests.

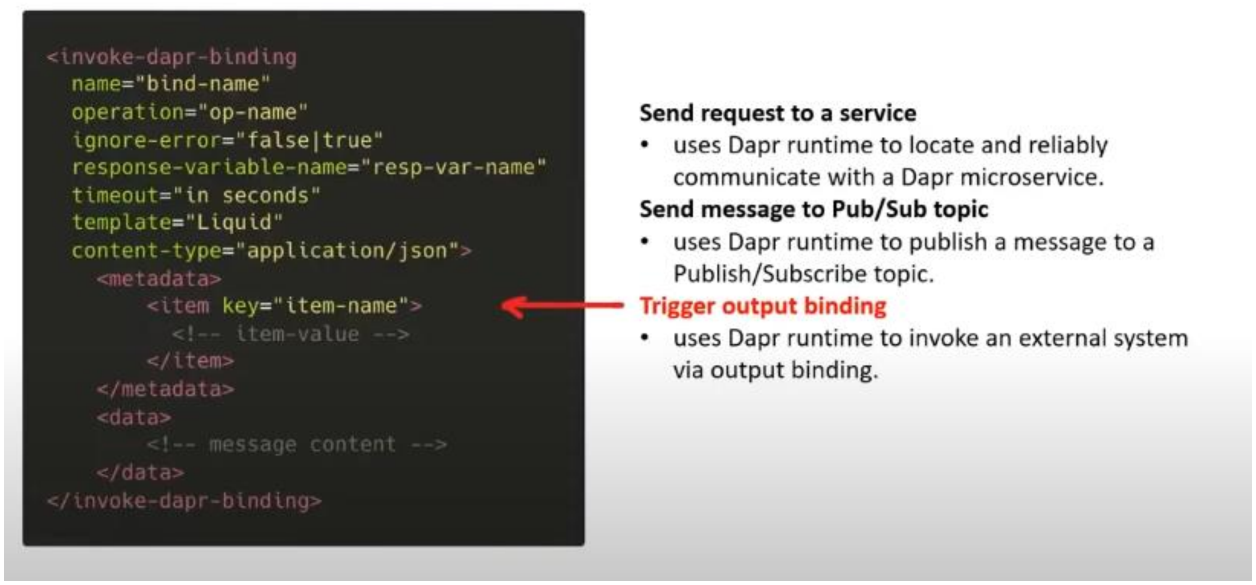
Set HTTP header
- Assigns a value to an existing response and/or request header or adds a new response and/or request header.

Set query string parameter
- Adds, replaces value of, or deletes request query string parameter.

Rewrite URL
- Converts a request URL from its public form to the form expected by the web service.

Transform XML using an XSLT
- Applies an XSL transformation to XML in the request or response body.

Dapr Integration Policies



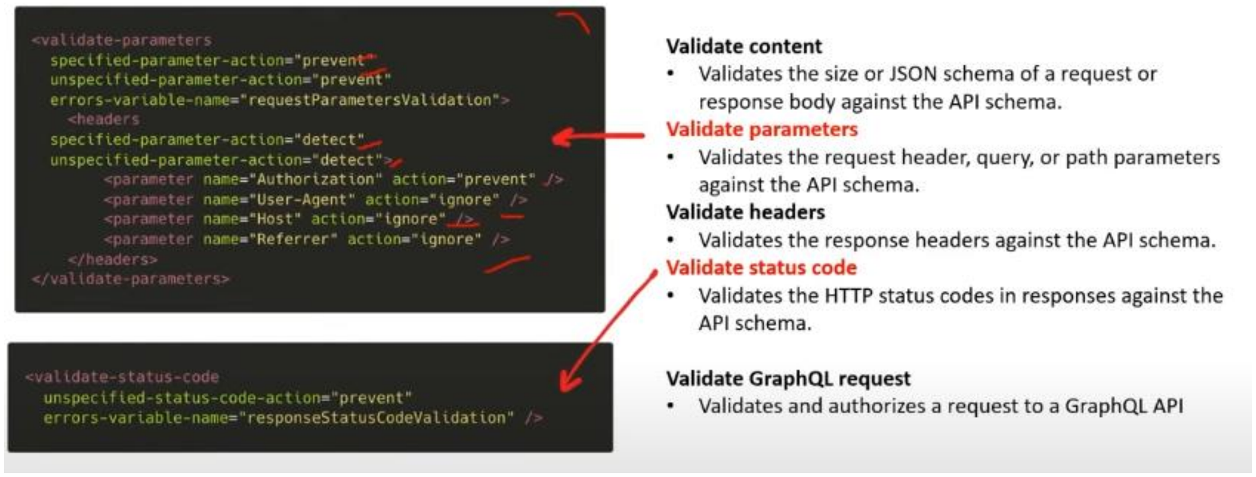
```
<invoke-dapr-binding
  name="bind-name"
  operation="op-name"
  ignore-error="false|true"
  response-variable-name="resp-var-name"
  timeout="in seconds"
  template="Liquid"
  content-type="application/json">
    <metadata>
        <item key="item-name">
          <!-- item-value -->
        </item>
    </metadata>
    <data>
        <!-- message content -->
    </data>
</invoke-dapr-binding>
```

Send request to a service
- uses Dapr runtime to locate and reliably communicate with a Dapr microservice.

Send message to Pub/Sub topic
- uses Dapr runtime to publish a message to a Publish/Subscribe topic.

Trigger output binding
- uses Dapr runtime to invoke an external system via output binding.

Validation Policies



```
<validate-parameters
  specified-parameter-action="prevent"
  unspecified-parameter-action="prevent"
  errors-variable-name="requestParametersValidation">
    <headers
  specified-parameter-action="detect"
  unspecified-parameter-action="detect">
        <parameter name="Authorization" action="prevent" />
        <parameter name="User-Agent" action="ignore" />
        <parameter name="Host" action="ignore" />
        <parameter name="Referrer" action="ignore" />
    </headers>
</validate-parameters>
```

```
<validate-status-code
  unspecified-status-code-action="prevent"
  errors-variable-name="responseStatusCodeValidation" />
```

Validate content
- Validates the size or JSON schema of a request or response body against the API schema.

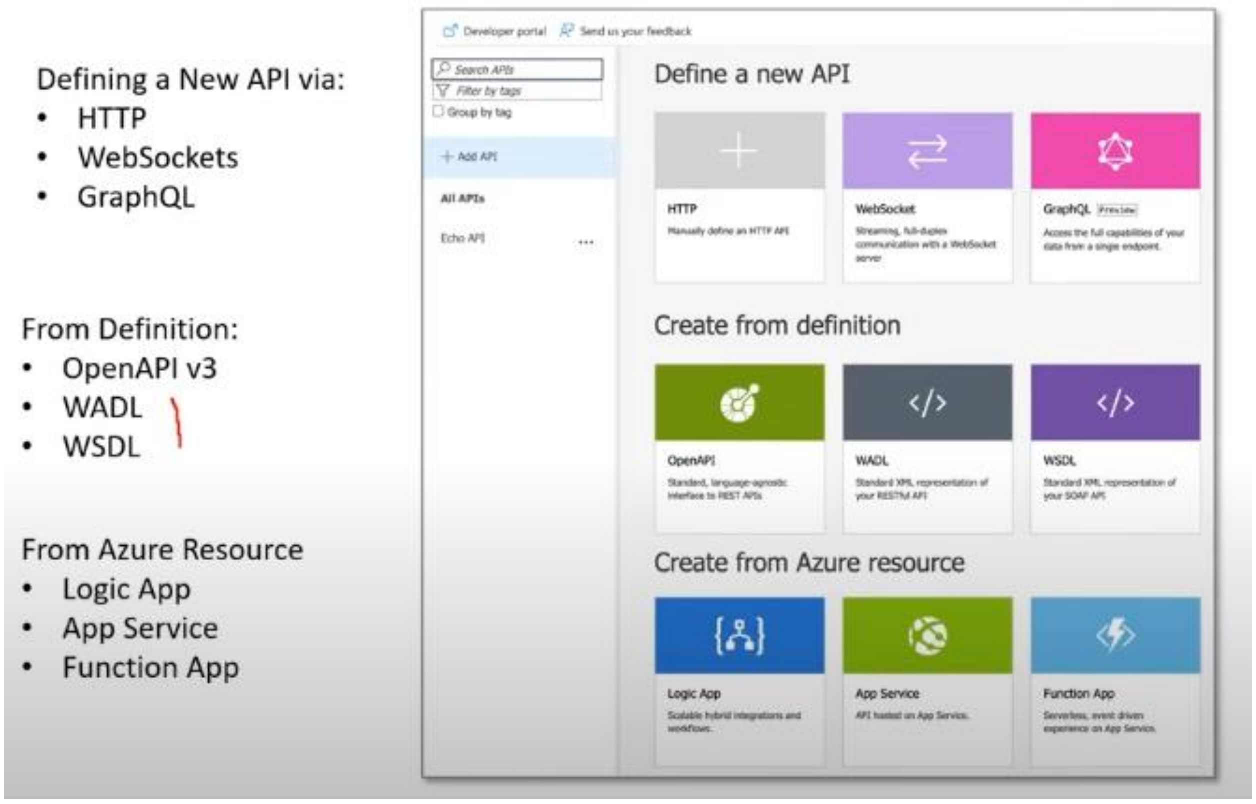
Validate parameters
- Validates the request header, query, or path parameters against the API schema.

Validate headers
- Validates the response headers against the API schema.

Validate status code
- Validates the HTTP status codes in responses against the API schema.

Validate GraphQL request
- Validates and authorizes a request to a GraphQL API

APIs



Defining a New API via:
- HTTP
- WebSockets
- GraphQL

From Definition:
- OpenAPI v3
- WADL
- WSDL

From Azure Resource
- Logic App
- App Service
- Function App

Define a new API

HTTP
Manually define an HTTP API

WebSocket
Streaming, full-duplex communication with a WebSocket server

GraphQL Preview
Access the full capabilities of your data from a single endpoint.

Create from definition

OpenAPI
Standard, language-agnostic interface to REST APIs

WADL
Standard XML representation of your RESTful API

WSDL
Standard XML representation of your SOAP API

Create from Azure resource

Logic App
Scalable hybrid integrations and workflows.

App Service
API hosted on App Service.

Function App
Serverless, event driven experience on App Service.

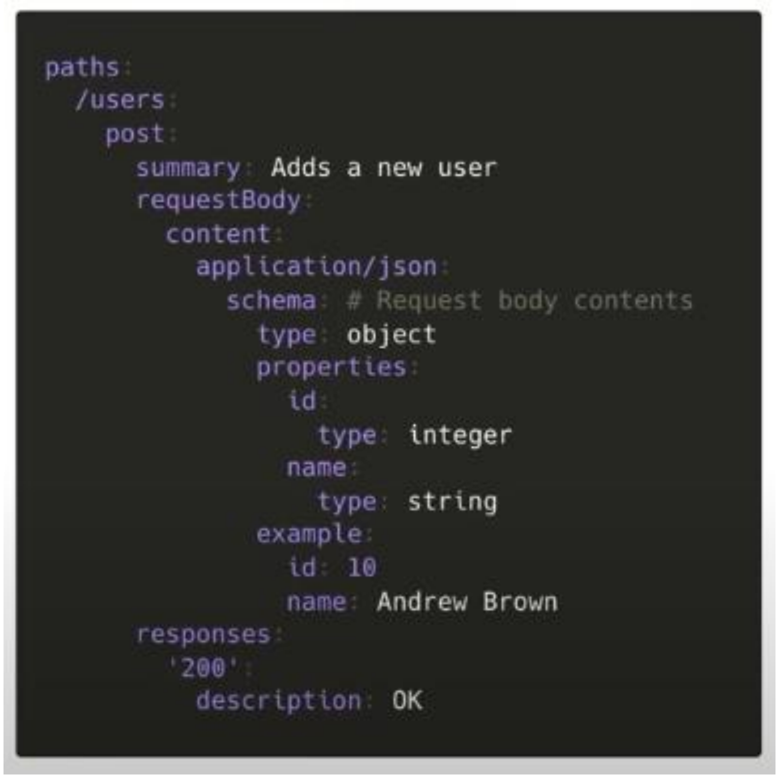
OpenAPI

OpenAPI Specification (OAS) defines a standard, language-agnostic interface to RESTful APIs which allows both humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection.

Swagger and OpenAPI used to be the same thing but as of OpenAPI V3, Swagger and OpenAPI are two different things

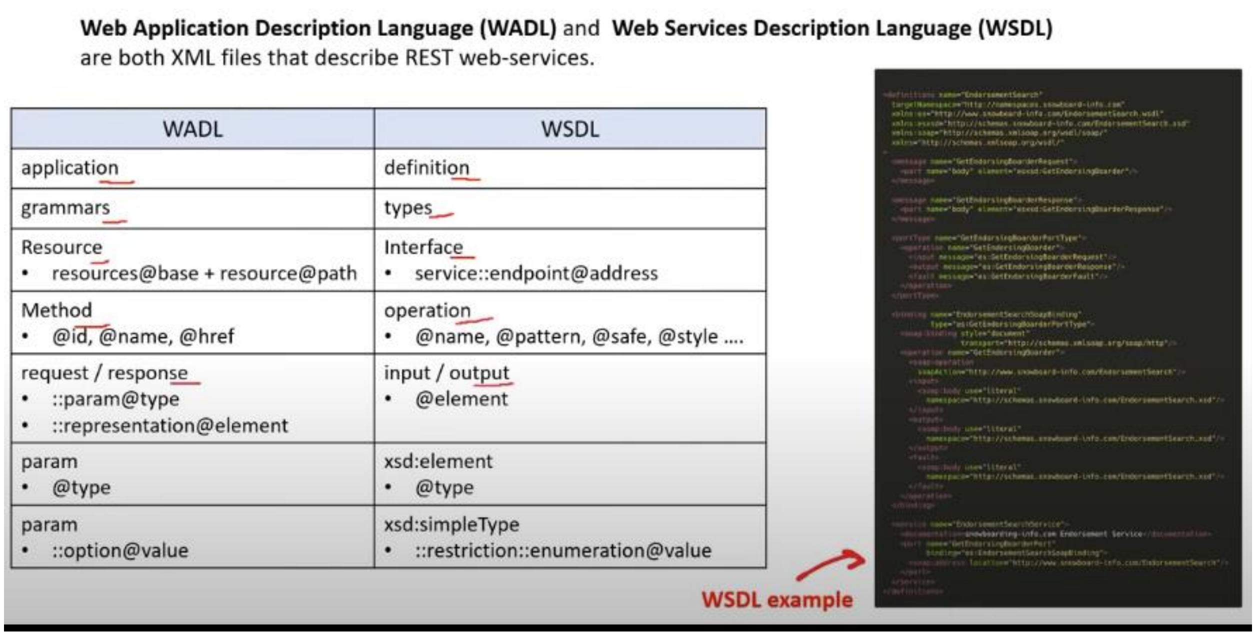
- OpneAPI = Specification
- Swagger = Tools for implementing the specification

OpenAPI can be represented as either **JSON** or **YAML**.



```yaml
paths:
  /users:
    post:
      summary: Adds a new user
      requestBody:
        content:
          application/json:
            schema: # Request body contents
              type: object
              properties:
                id:
                  type: integer
                name:
                  type: string
              example:
                id: 10
                name: Andrew Brown
      responses:
        '200':
          description: OK
```

WADL and WSDL

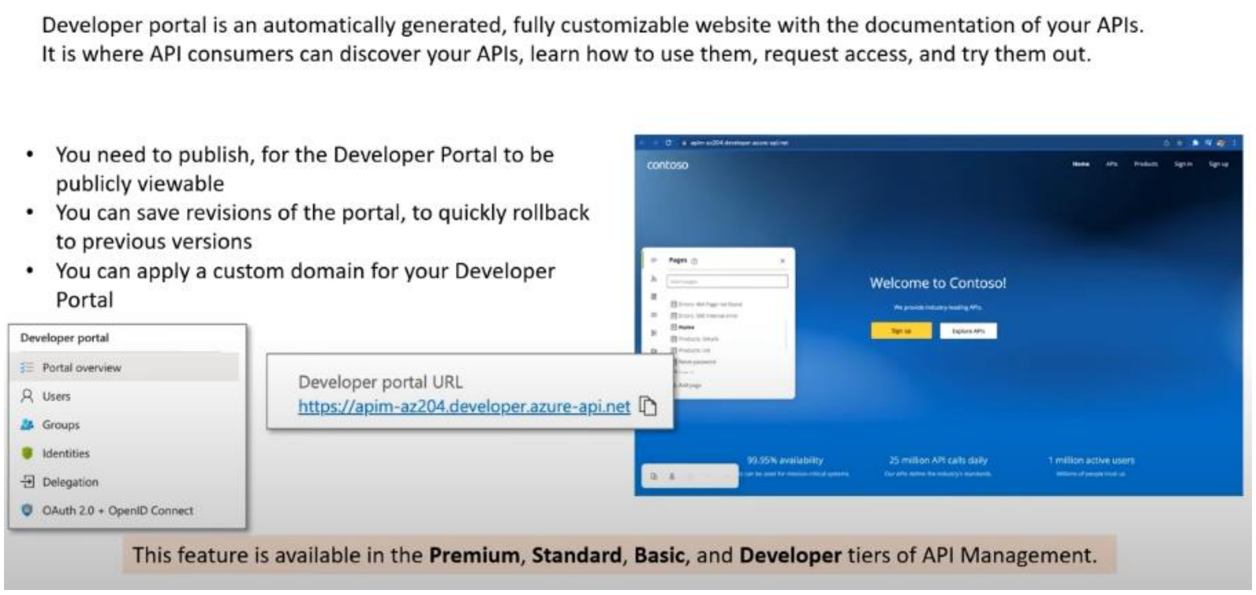


Web Application Description Language (WADL) and **Web Services Description Language (WSDL)** are both XML files that describe REST web-services.

WADL	WSDL
application	definition
grammars	types
Resource • resources@base + resource@path	Interface • service::endpoint@address
Method • @id, @name, @href	operation • @name, @pattern, @safe, @style ....
request / response • ::param@type • ::representation@element	input / output • @element
param • @type	xsd:element • @type
param • ::option@value	xsd:simpleType • ::restriction::enumeration@value

```
<definitions name="EndorsementSearch"
  targetNamespace="http://namespaces.snowboard-info.com"
  xmlns:es="http://www.snowboard-info.com/EndorsementSearch.wsdl"
  xmlns:esxsd="http://schemas.snowboard-info.com/EndorsementSearch.xsd"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
>
  <message name="GetEndorsingBoarderRequest">
    <part name="body" element="esxsd:GetEndorsingBoarder"/>
  </message>

  <message name="GetEndorsingBoarderResponse">
    <part name="body" element="esxsd:GetEndorsingBoarderResponse"/>
  </message>

  <portType name="GetEndorsingBoarderPortType">
    <operation name="GetEndorsingBoarder">
      <input message="es:GetEndorsingBoarderRequest"/>
      <output message="es:GetEndorsingBoarderResponse"/>
      <fault message="es:GetEndorsingBoarderFault"/>
    </operation>
  </portType>

  <binding name="EndorsementSearchSoapBinding"
           type="es:GetEndorsingBoarderPortType">
    <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="GetEndorsingBoarder">
      <soap:operation
        soapAction="http://www.snowboard-info.com/EndorsementSearch"/>
      <input>
        <soap:body use="literal"
          namespace="http://schemas.snowboard-info.com/EndorsementSearch.xsd"/>
      </input>
      <output>
        <soap:body use="literal"
          namespace="http://schemas.snowboard-info.com/EndorsementSearch.xsd"/>
      </output>
      <fault>
        <soap:body use="literal"
          namespace="http://schemas.snowboard-info.com/EndorsementSearch.xsd"/>
      </fault>
    </operation>
  </binding>

  <service name="EndorsementSearchService">
    <documentation>snowboarding-info.com Endorsement Service</documentation>
    <port name="GetEndorsingBoarderPort"
          binding="es:EndorsementSearchSoapBinding">
      <soap:address location="http://www.snowboard-info.com/EndorsementSearch"/>
    </port>
  </service>
</definitions>
```

Developers Portal



Developer portal is an automatically generated, fully customizable website with the documentation of your APIs.
It is where API consumers can discover your APIs, learn how to use them, request access, and try them out.

- You need to publish, for the Developer Portal to be publicly viewable
- You can save revisions of the portal, to quickly rollback to previous versions
- You can apply a custom domain for your Developer Portal

Developer portal URL
https://apim-az204.developer.azure-api.net

This feature is available in the **Premium**, **Standard**, **Basic**, and **Developer** tiers of API Management.

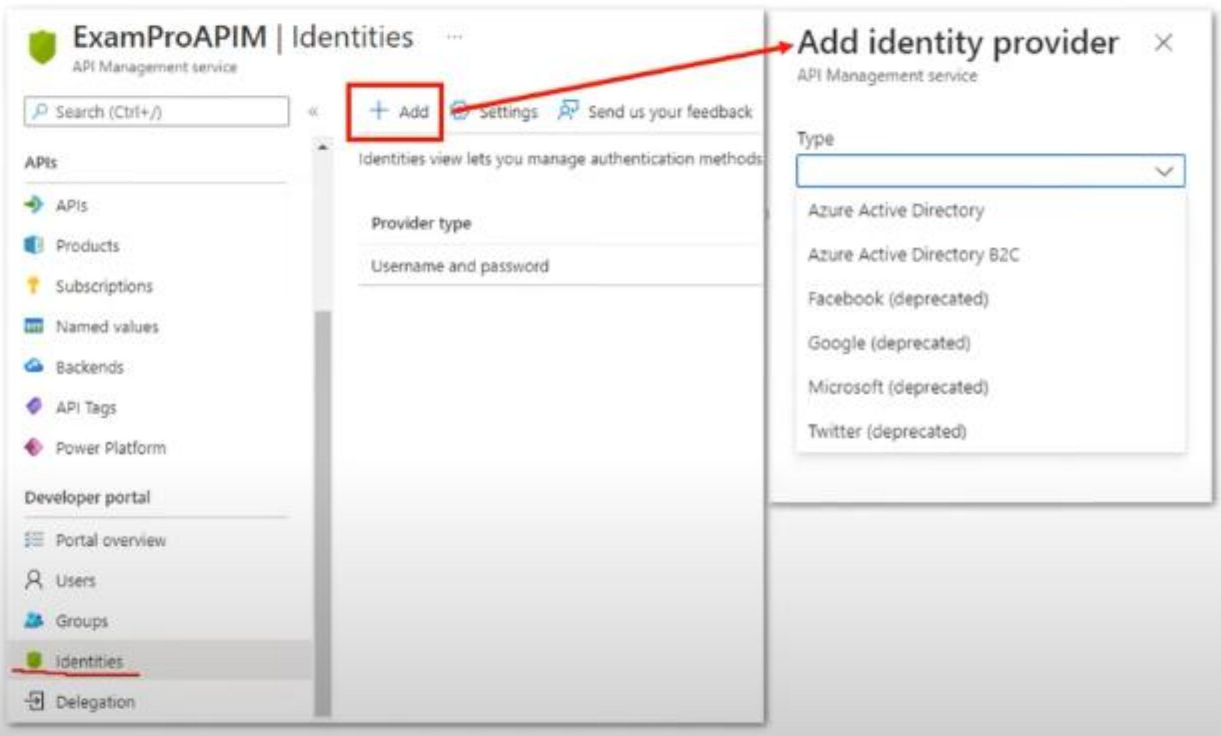
Authentication – Developer Portal

You can configure many **different types** of authentications within APIM:

- Azure Active Directory
- Azure AD B2C
- Identity Providers (IdPs): Google, Microsoft, and Facebook,
- Basic authentication (default)

Basic authentication (Basic Auth) is a built-in authentication method for Azure API Management that requires the developer to register with an email and password in order to obtain an API key, which is then used in requests to authenticate the requestors.

Delegated authentication, allows you to use your own web-app **sign-in sign-up** and **product subscription** instead of the built-in developer portal built-in functionality



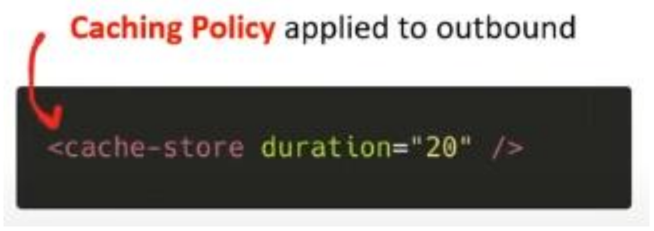
Identities view lets you manage authentication methods

Provider type
Username and password

Builtin Cache

APIs and operations in API Management can be configured with response caching.

Response caching can significantly reduce latency for API callers and backend load for API providers.



```
<cache-store duration="20" />
```

The built-in cache is **volatile** and is **shared by all units in the same region** in the same API Management service.

External Cache

Redis cache can be utilized externally instead of using a built-in cache.

Using an external cache allows to overcome a few limitations of the built-in cache:

- Avoid having cache periodically cleared during API Management updates.
- Have more control over cache configuration.
- Cache more data than your API Management tier allows to
- Use caching with the Consumption tier of API Management
- Enable caching in the API Management self-hosted gateways.

We need to just provide a connection string to Redis Cache.

Practical Implementation

Create APIM service:

1. Create a **Resource group** with a subscription and region.
2. Create a **Log Analytics Workspace** with a subscription, resource group.
    a. Instance details: Name, region.
3. Create an **Application Instance** with a subscription, resource group.
    a. Instance details: Name, Region, Resource mode.
    b. Workspace details: subscription, Log analytics workspace.
4. Create an **APIM service** with a subscription and resource group:
    a. Instance details: region, name, organization name, administrator email.
    b. Pricing Tier: Developer, standard, basic, premium, etc.

It takes 30-40 min. to get ready.

RESOURCES:

1. TLS and SSL: https://www.youtube.com/watch?v=j9QmMEWmcfo
2. Certificates: https://www.youtube.com/watch?v=r1nJT63BFQ0