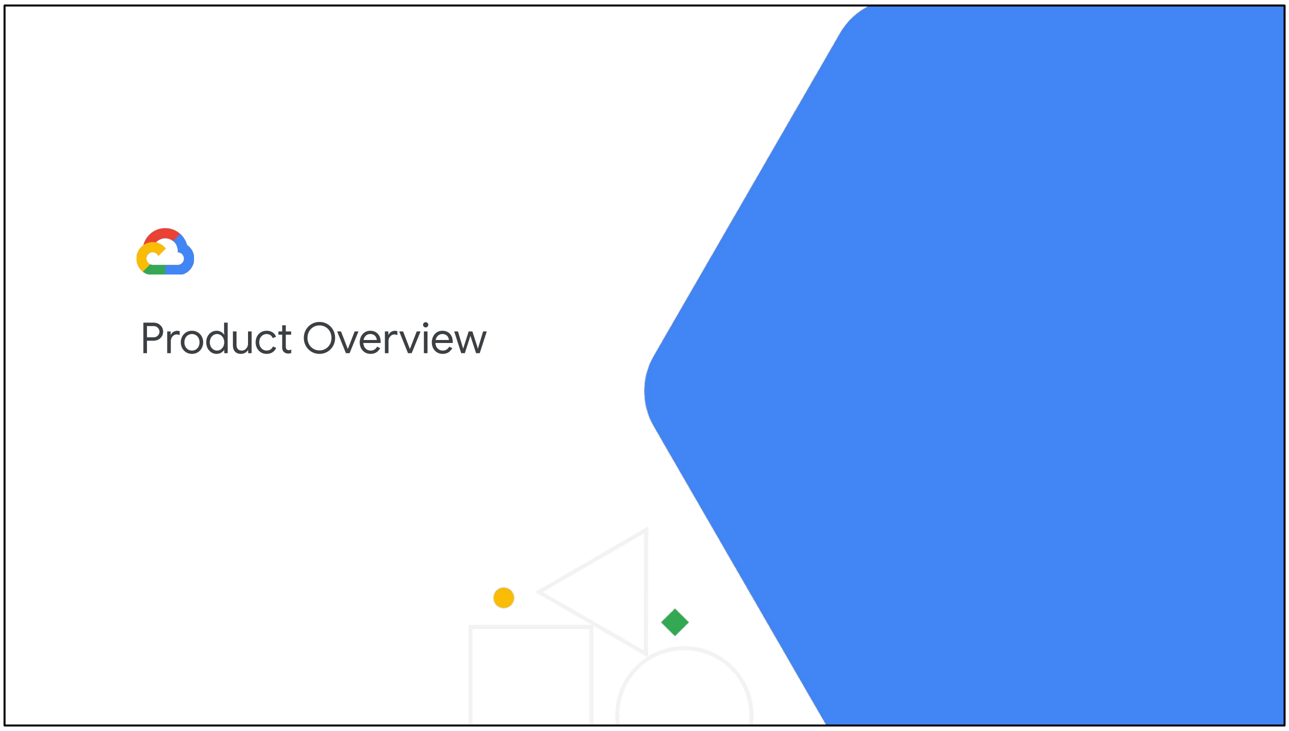
Module: Apigee Overview

Mike Dunker
Course Developer, Google Cloud

In this module, we'll introduce you to Apigee, Google Cloud's API Management platform.

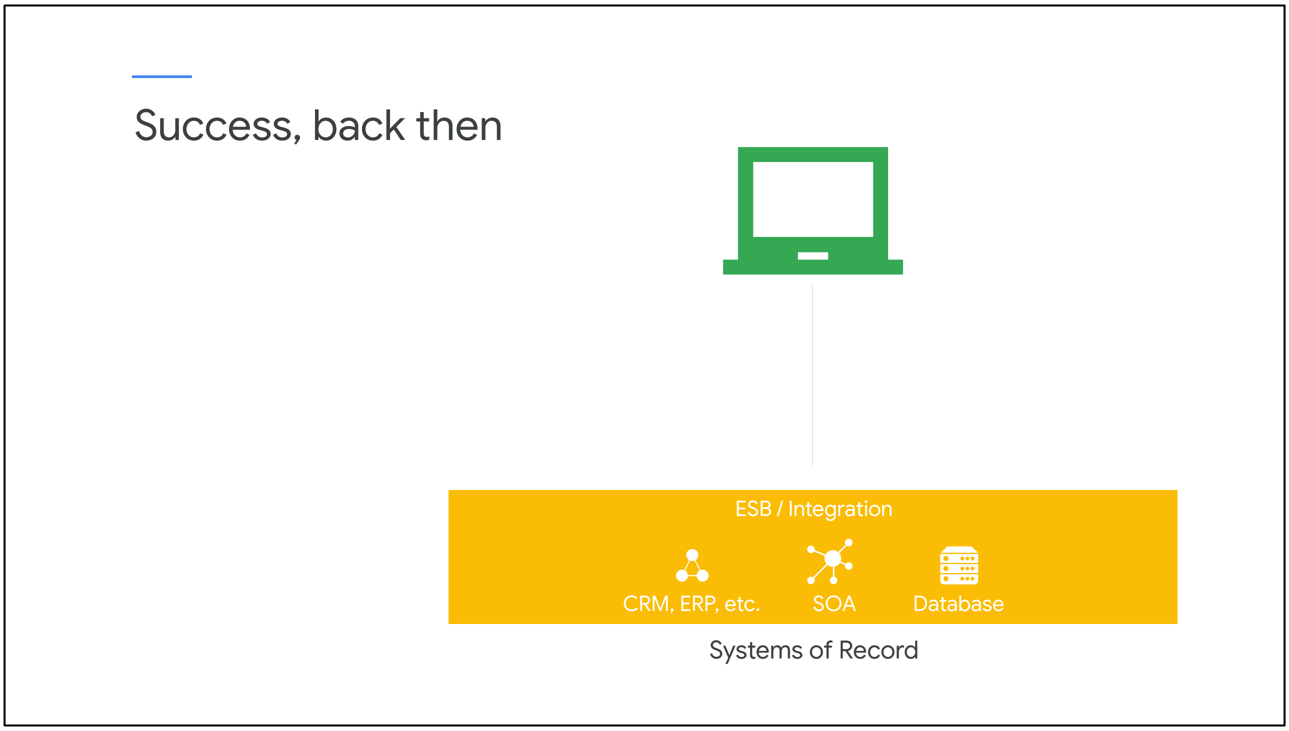
You will learn about the API lifecycle, as well as Apigee organizations and the entities that they contain.



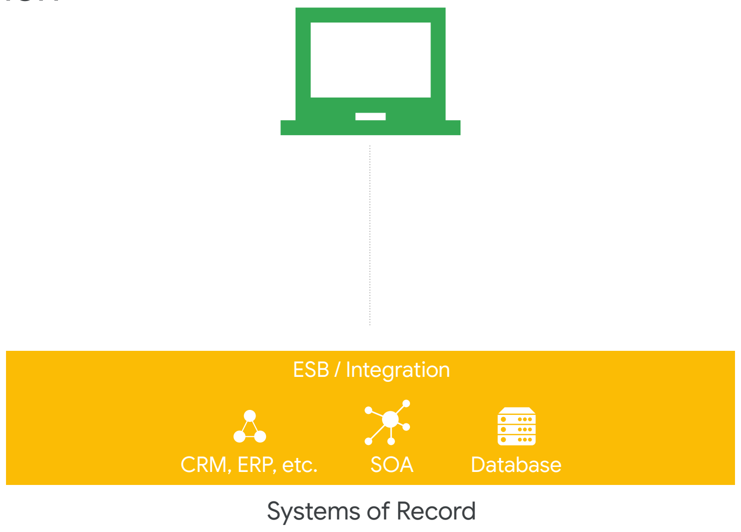
Product Overview

In this lecture you will learn about Apigee, Google Cloud's API Management Platform.

We'll discuss the business problems that can be solved using Apigee, see many of the features that Apigee provides, and introduce the components of Apigee and deployment options for the Apigee platform.



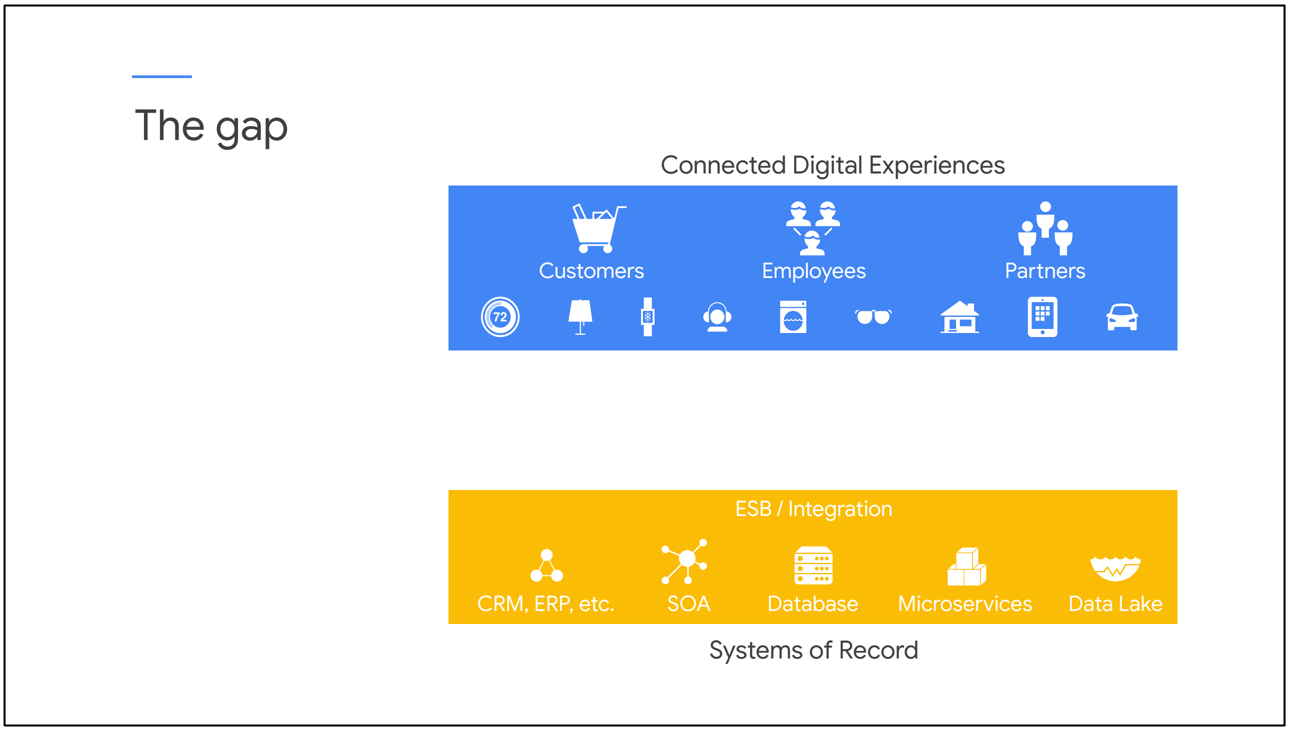
Success, back then



Before we discuss the role of APIs and API Management in today's enterprise landscape, it is important for you to understand where we are, and how we got here.

Not too long ago, having a digital presence just meant you had a website. At the time, success was expressed in simple terms, such as the number of visits to the site or the number of users registered over months or years.

The relatively slow pace of change of the web channel allowed an IT organization to plan and execute changes to backend systems. The pace allowed them to keep up with demand.



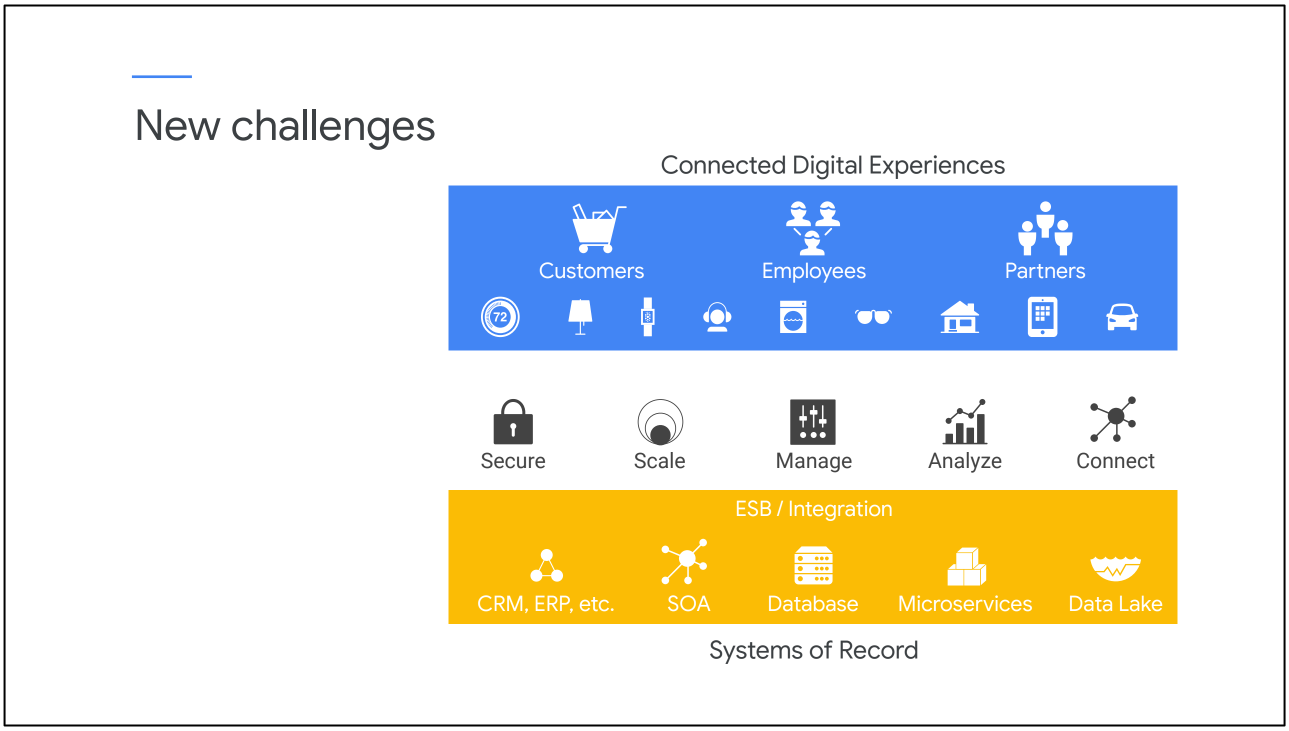
The gap

What about today?

The number of customer-facing applications has dramatically increased and diversified, opening the door to a new era of connected digital experiences.

Today, most companies embrace multiple methods of interaction as part of their digital strategy.

In addition to traditional web and mobile applications, companies are finding new channels for users to interact with data and services, powered by smart connected devices.

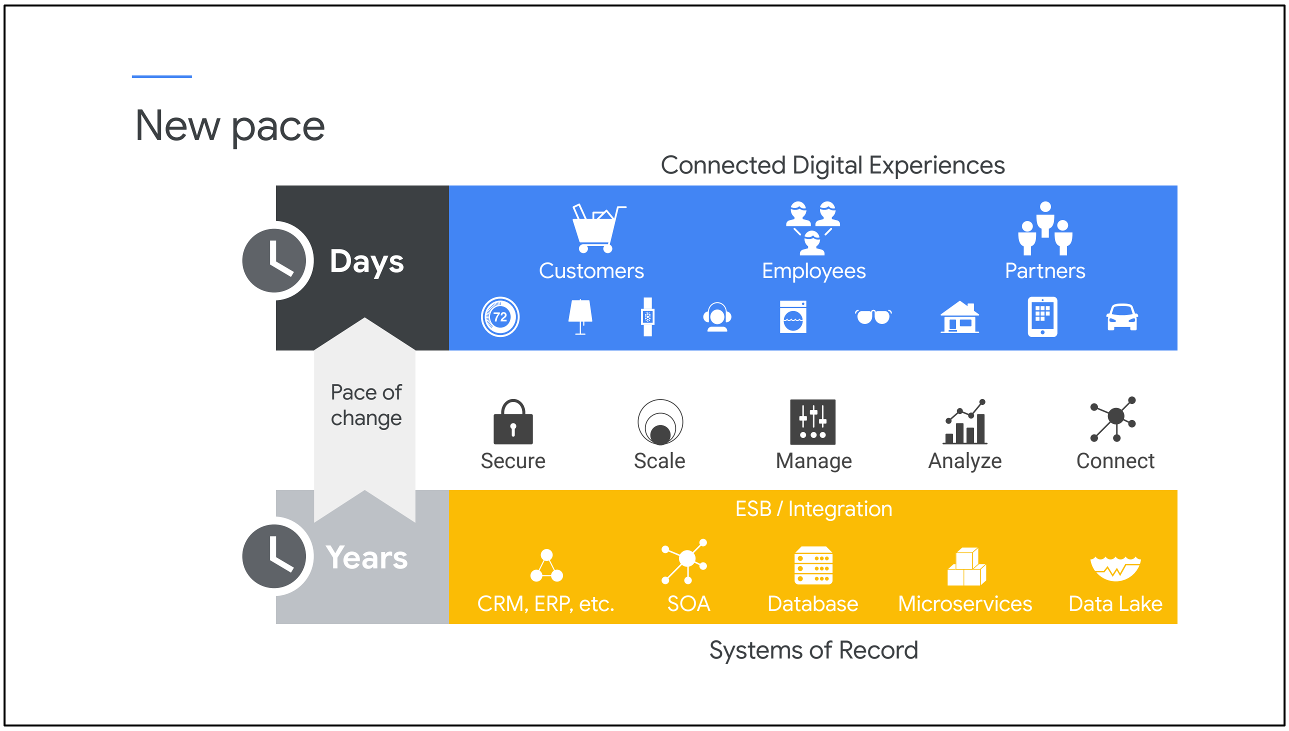


New challenges

With all of these new apps and channels come new challenges, including:

- securing communication and access for new channels and devices;
- increasing scale to handle higher traffic and usage;
- managing new channels, customers, partners, and apps;
- improving visibility of business and technical metrics to allow data-driven business decisions;
- and, leveraging ecosystems and platforms to increase reach.

All of these challenges add to the complexity and diversity of requirements that backend systems need to handle, ...

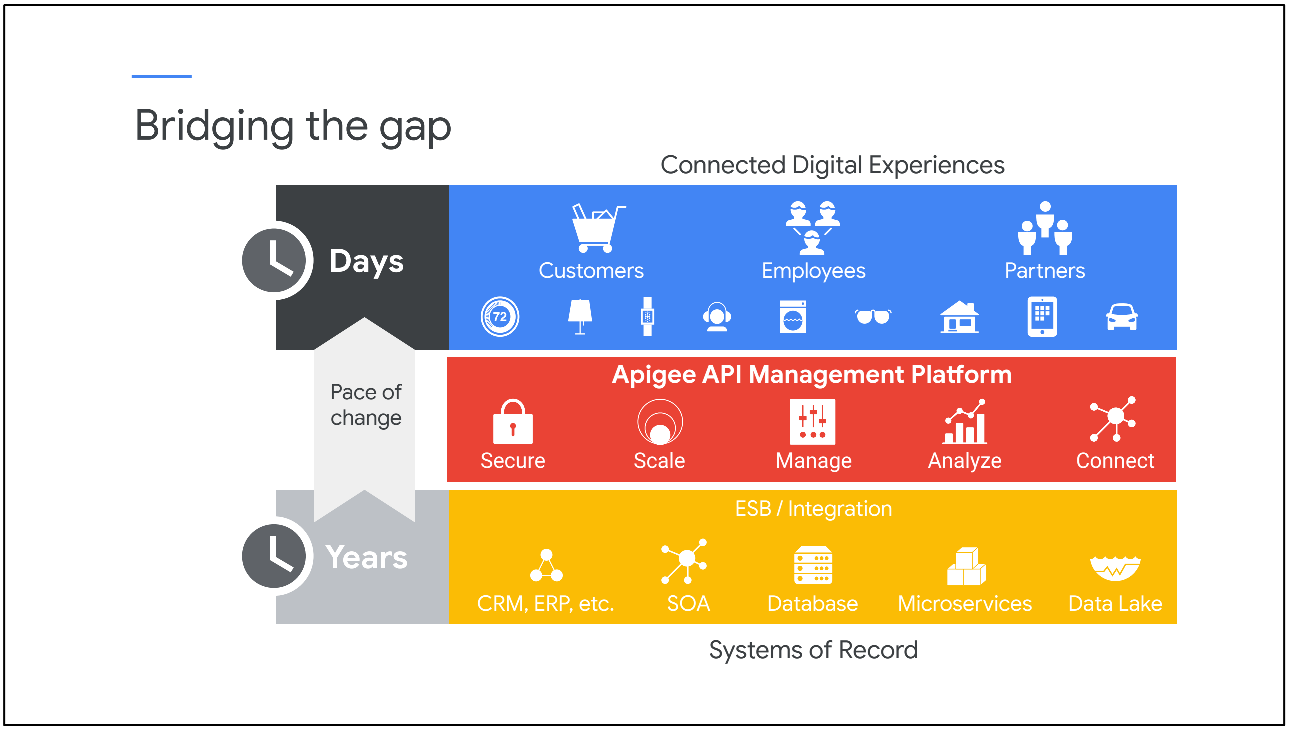


New pace

...while the pace of change gets faster and faster.

Applications that power connected digital experiences tend to evolve at an accelerating rate.

This need for speed is driven by business opportunities, competition, and evolving customer needs.



Bridging the gap

Apigee's API Management Platform is designed to bridge the gap.

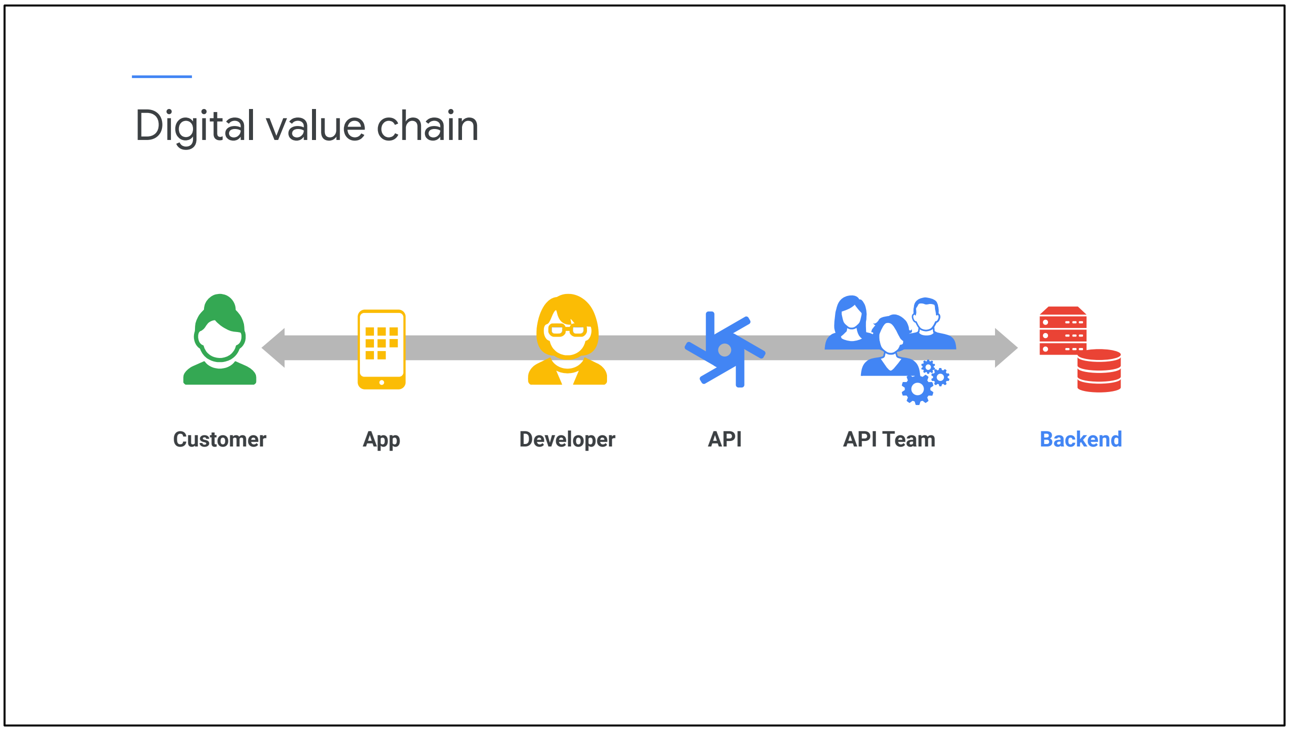
By building APIs for connected experiences, you can create abstraction layers that help reduce the complexity required of backend systems.

APIs that are implemented on Apigee leverage a rich set of capabilities and features, including security, caching, transformation, and mediation.

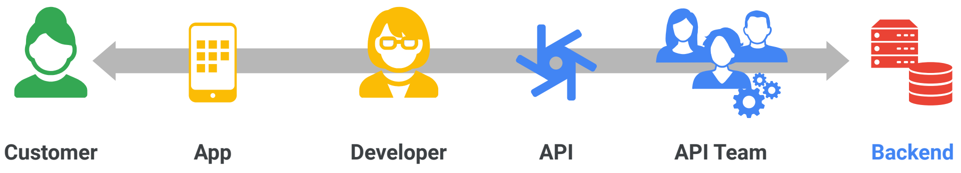
These features allow you to build APIs tailored to the needs of individual applications and react to changing business requirements, while reducing the need for customization and modification of backend services.

With all API calls passing through Apigee, you can gain insights into technical and business challenges.

APIs also improve your ability to participate in or create ecosystems, driving even more business and success.



Digital value chain



The Digital Value Chain allows us to visualize how connected digital experiences are realized.

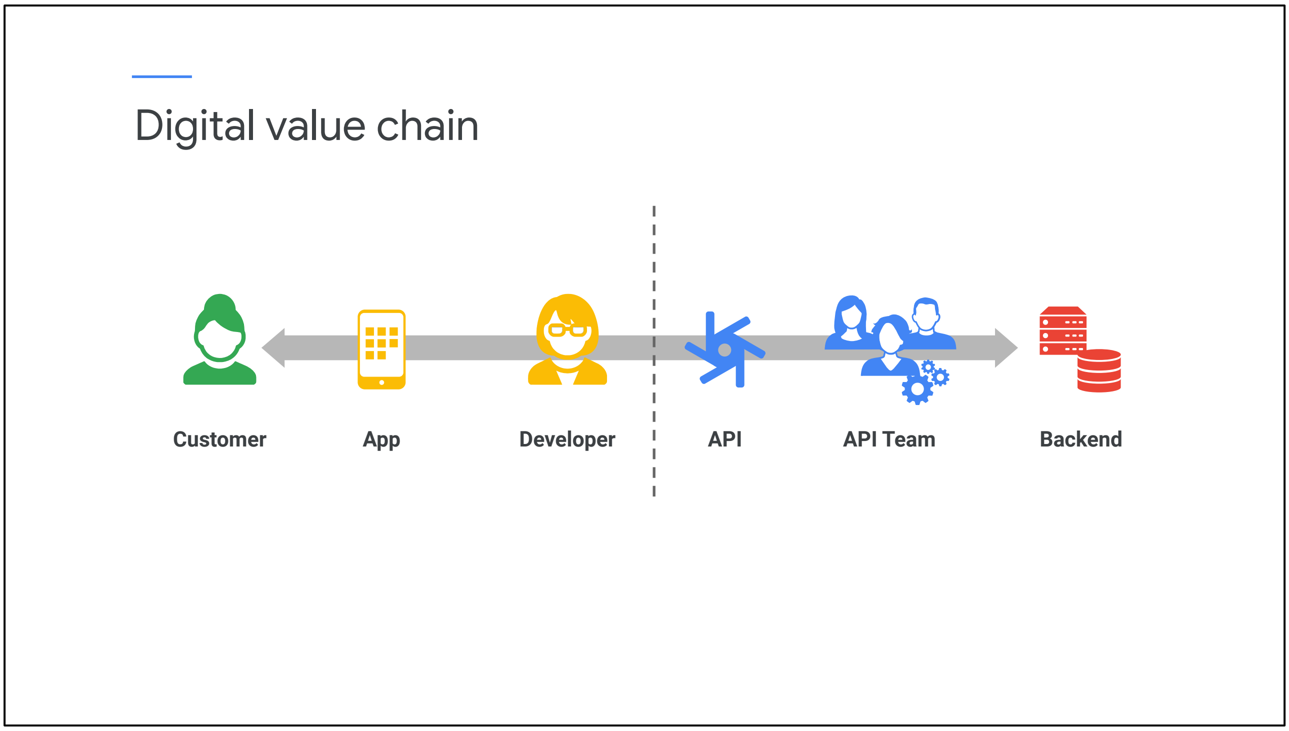
In a digitally connected world, you interact with **customers**, or “end users,” using applications.

Applications range from web and mobile apps to large enterprise systems and connected devices. Some of these applications are built by developers at your company. Other applications may represent systems used by partners or customer-facing products that they’ve built.

Application developers leverage the APIs offered by your company.

These **APIs** are built and managed by a cross-functional team we call the **API Team**. APIs built by the API team make use of **backend** resources, while shielding application developers from unnecessary complexity.

Because some application developers using your APIs are generally external to your company, the apps they create are not actually under your control.

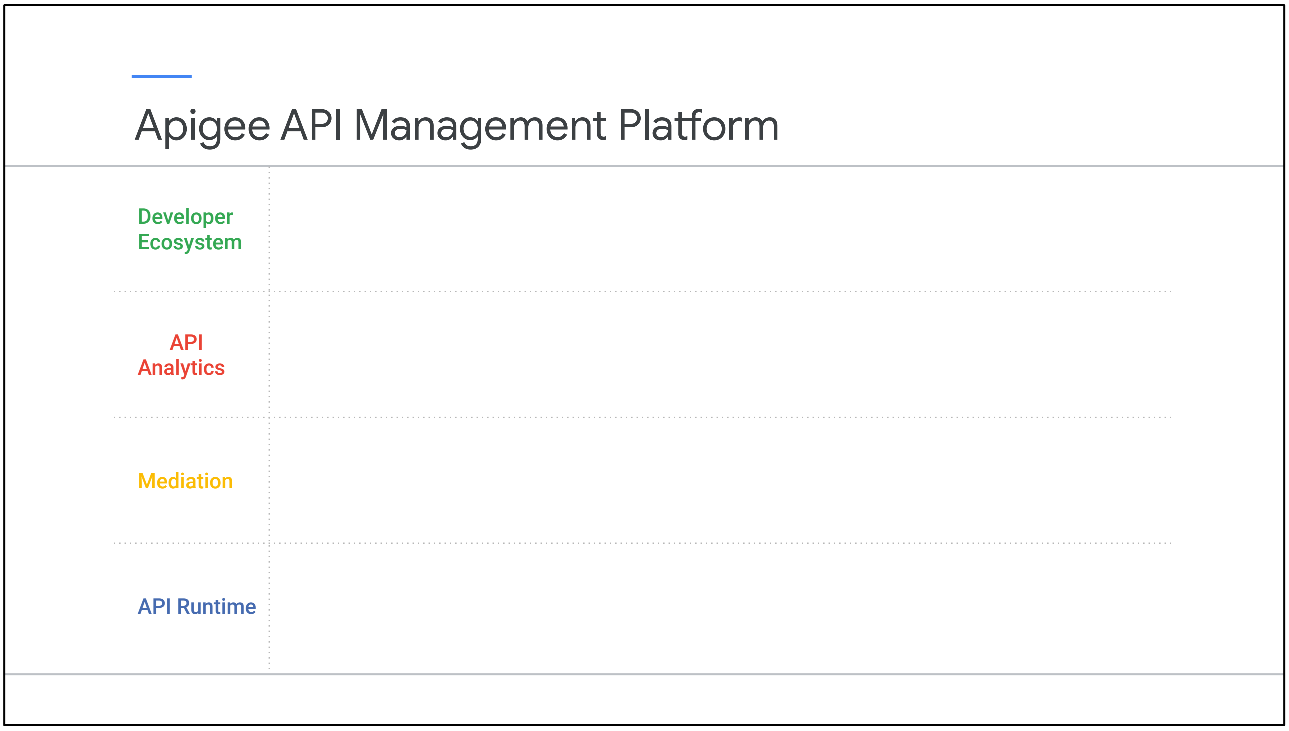


Digital value chain

The real boundary of a company's control over enforcement of policies and management of access to resources ends at the API layer.

APIs are the digital products you present to app developers.

As products, they should follow a life cycle, and you should manage them as you would manage other products produced by the company, marketing them to internal and external audiences.



Apigee API Management Platform

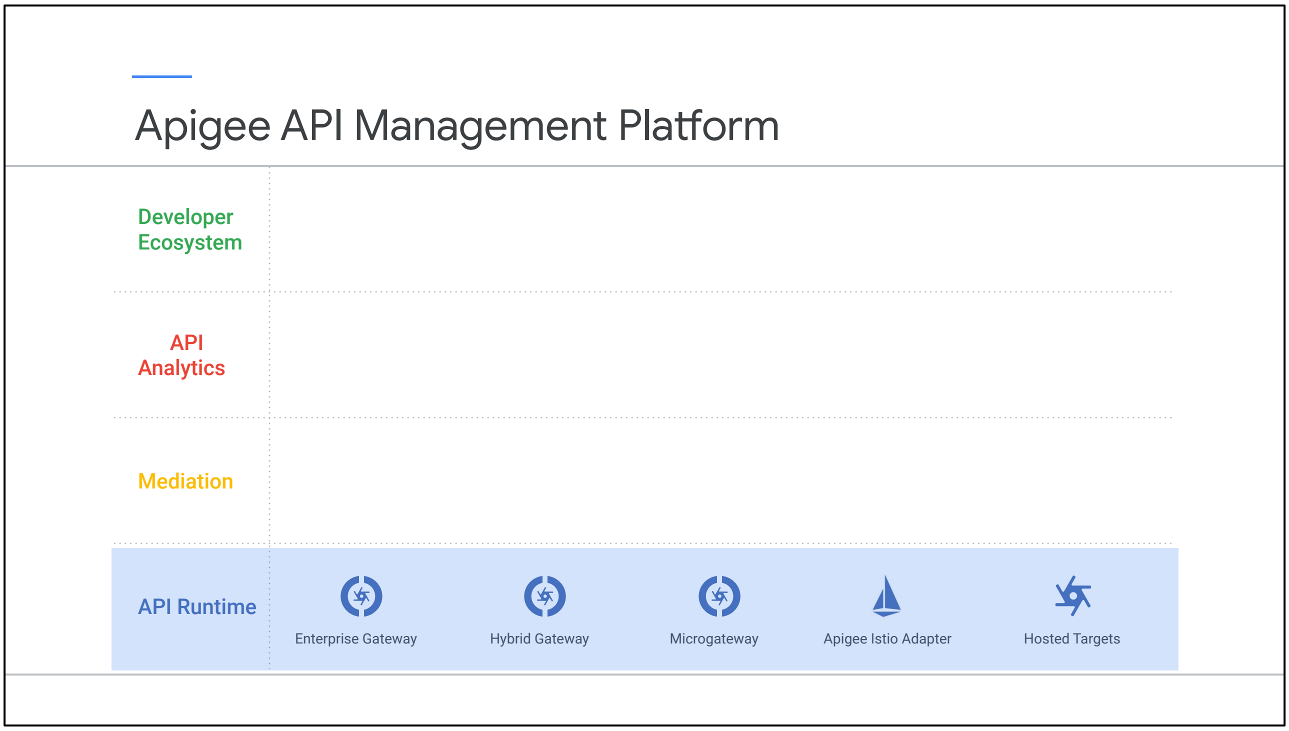
Developer Ecosystem

API Analytics

Mediation

API Runtime

Apigee is composed of four core capabilities, offered with multiple deployment options.



Apigee API Management Platform

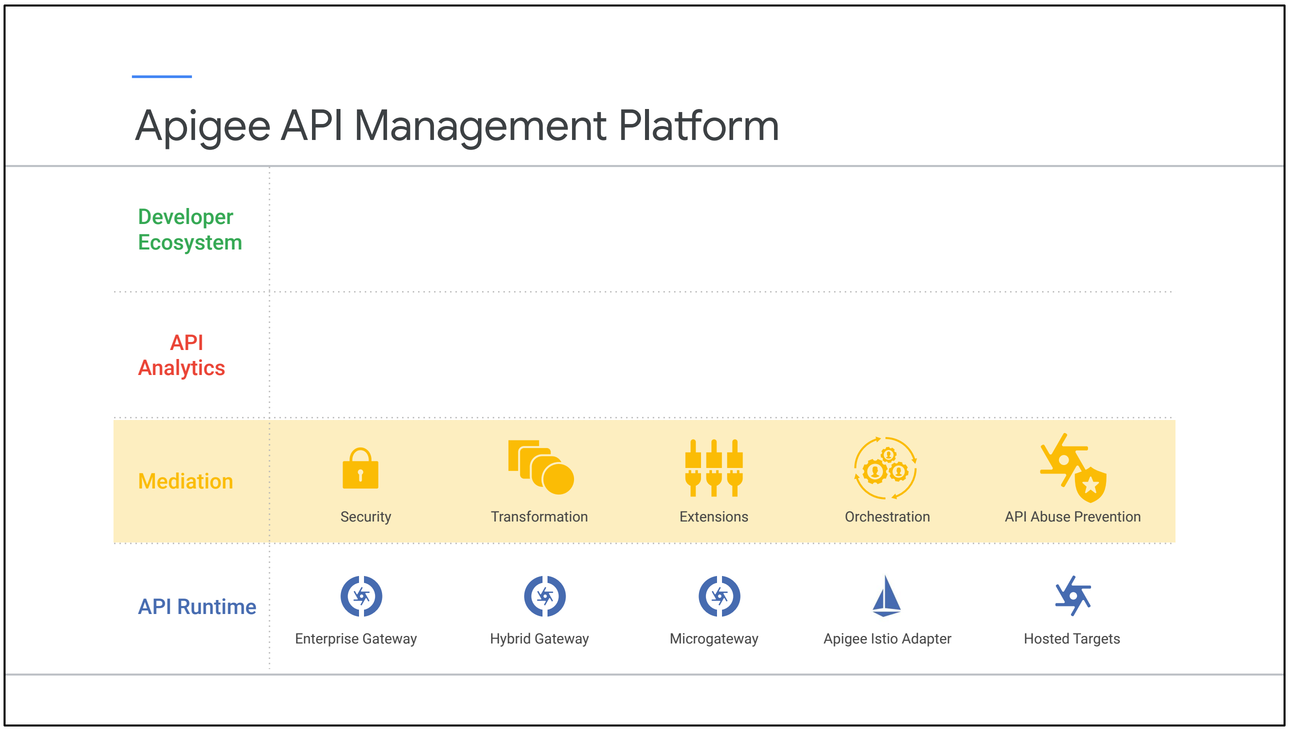
Starting at the bottom, Apigee offers multiple API gateways, responsible for handling runtime API traffic. APIs execute in the gateways.

Apigee's enterprise-level platform can be run as a fully managed, software-as-a-service offering running in the cloud, or as a self-managed on-premises installation in a customer's data center.

It can also be deployed as a hybrid installation, with runtime services managed by the customer and running in customer data centers or private clouds, and management services managed by Google and running in Google Cloud.

Apigee also provides support for microgateways and an Apigee Istio adapter. These lightweight deployment options provide API management functionality that can be deployed close to backend services.

The Hosted Targets feature allows creation of node.js applications and APIs that run as lightweight services inside the cloud enterprise gateway.

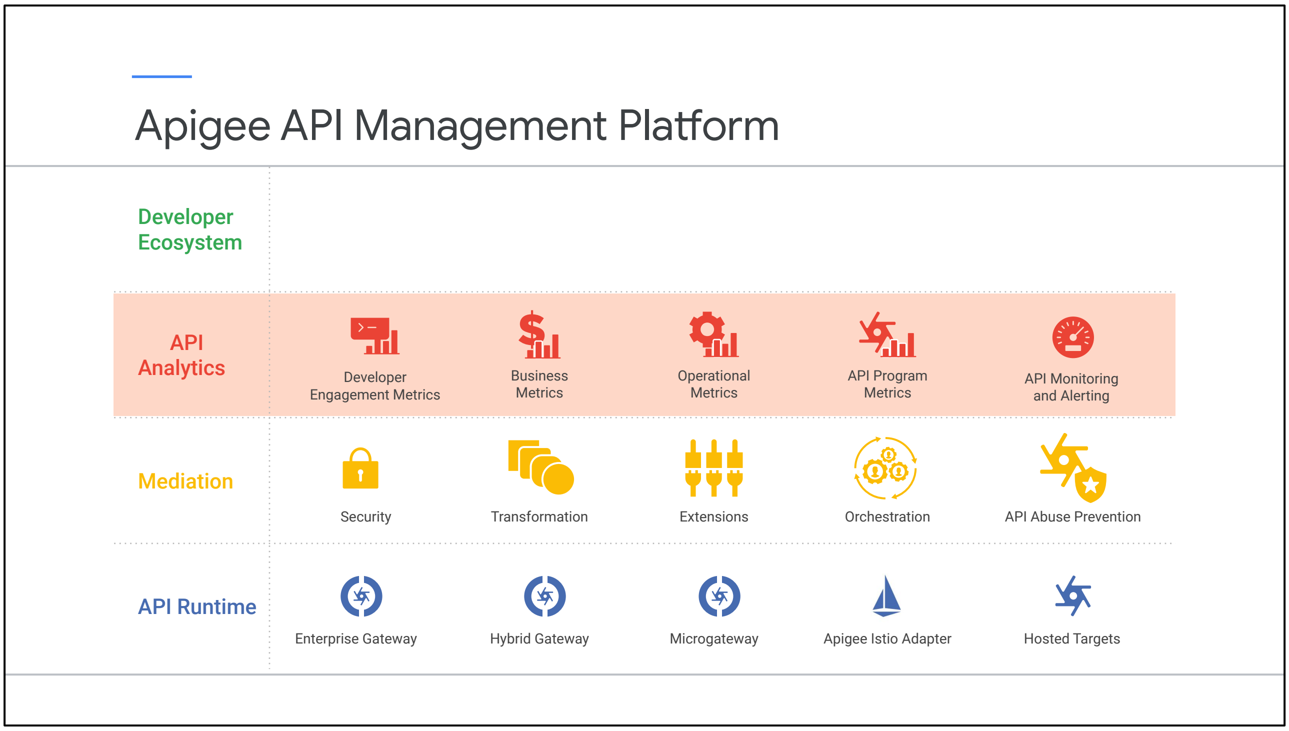


Apigee API Management Platform

Developer Ecosystem					
API Analytics					
Mediation	Security	Transformation	Extensions	Orchestration	API Abuse Prevention
API Runtime	Enterprise Gateway	Hybrid Gateway	Microgateway	Apigee Istio Adapter	Hosted Targets

Mediation represents the ability to parse and manipulate the requests and responses of API calls passing through Apigee.

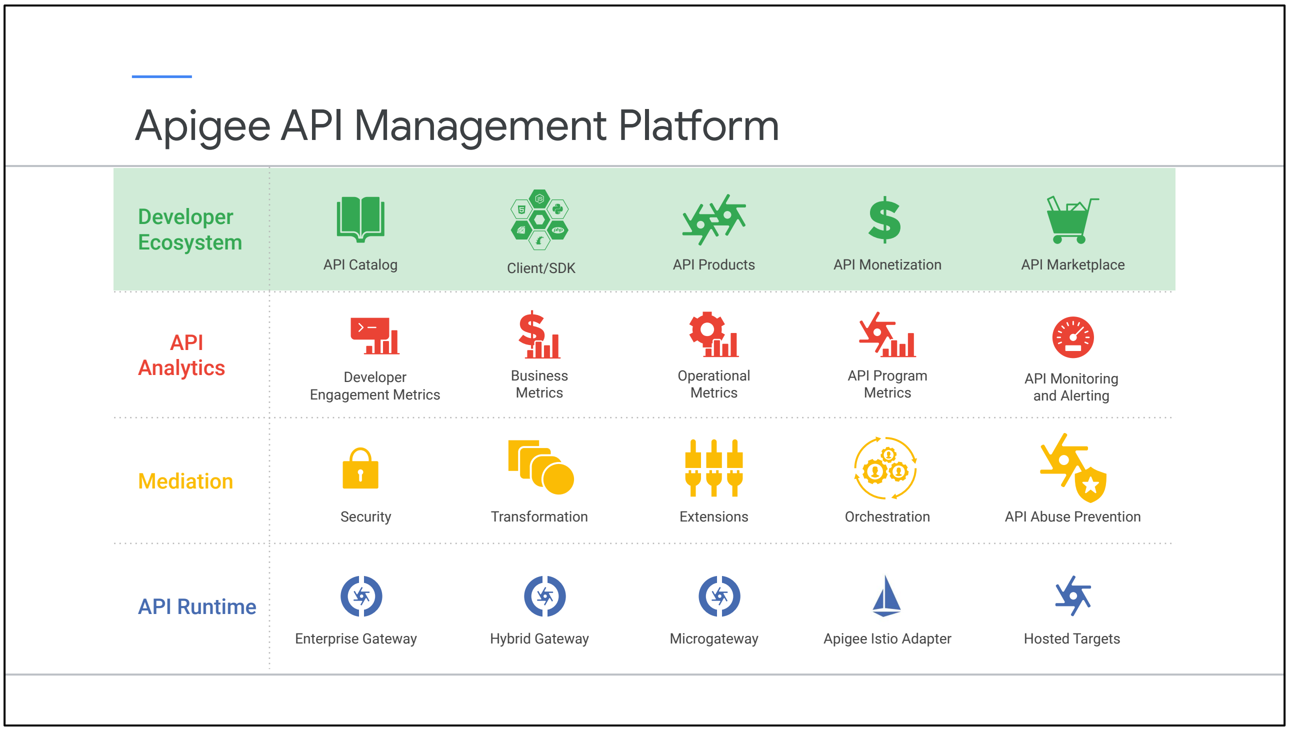
Mediation allows API teams to perform enrichment, transformation, orchestration, enforcement of security, caching, handling of faults, and more.



Apigee API Management Platform

Every API call that passes through Apigee generates analytics data.

Analytics data generated by the system can be consumed by developers, operations teams, and business users to make data-driven decisions about APIs.



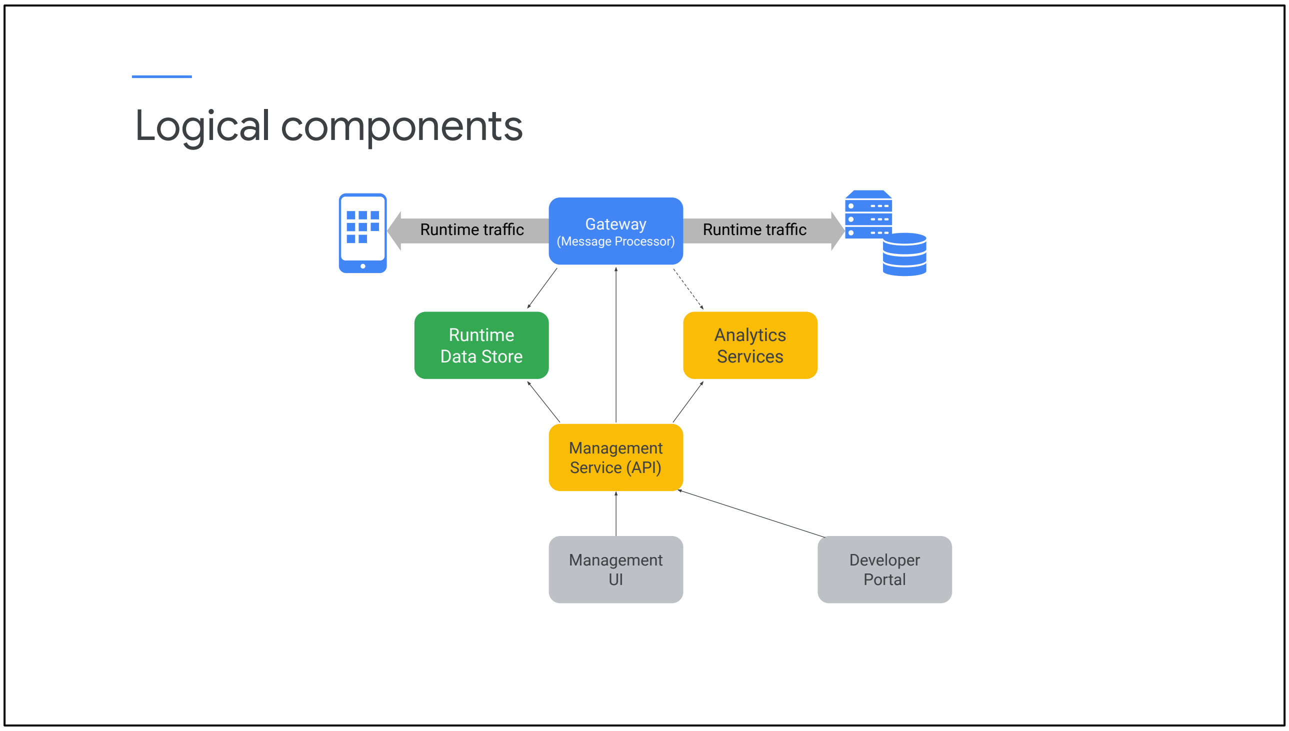
Apigee API Management Platform

Developer Ecosystem	API Catalog	Client/SDK	API Products	API Monetization	API Marketplace
API Analytics	Developer Engagement Metrics	Business Metrics	Operational Metrics	API Program Metrics	API Monitoring and Alerting
Mediation	Security	Transformation	Extensions	Orchestration	API Abuse Prevention
API Runtime	Enterprise Gateway	Hybrid Gateway	Microgateway	Apigee Istio Adapter	Hosted Targets

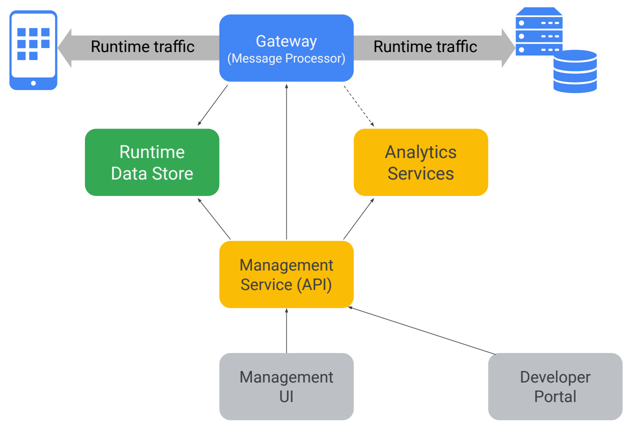
The developer ecosystem is an important factor in the success of your APIs.

APIs built and deployed on Apigee are bundled into API products, which can be deployed to a developer portal.

The developer portal facilitates the discovery and consumption of APIs and offers developers access to API documentation.



Logical components



Let's introduce Apigee's components by looking at their capabilities and relationships.

The **gateway** sits in the critical path of runtime traffic. The gateway's main component is the **Message Processor**, which is responsible for executing APIs in response to API requests.

Data used by APIs during runtime is stored in the **runtime data store**. This includes API keys, OAuth tokens, cache, and configuration.

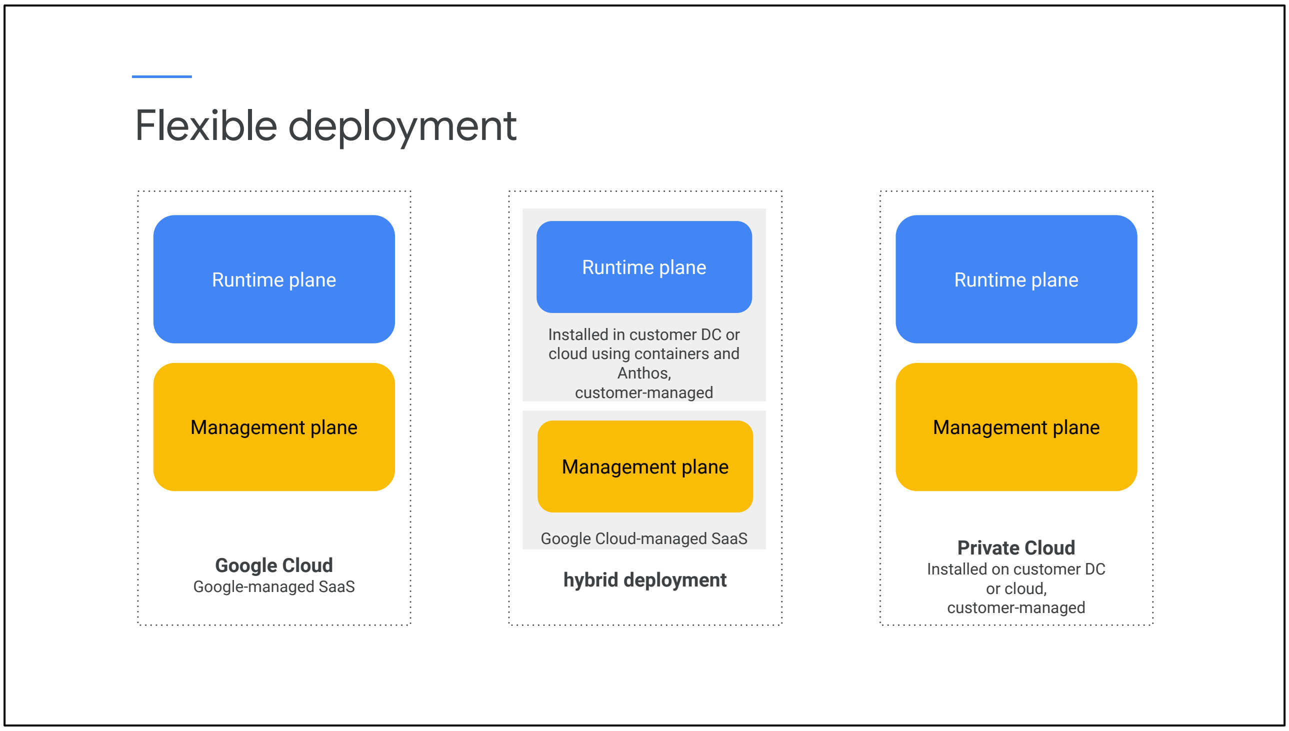
As APIs are executed by a Message Processor, **analytics** events are generated and processed asynchronously. These events contain a wealth of information about APIs, apps, and backend system calls, and are used for analytics reports and visualization.

The **Management Service** facilitates management of this distributed infrastructure. The Management Service interacts with the rest of the components via APIs and exposes a **management API** for use by the management UI, the developer portal, and other management processes. The management API is fully documented and available to customers. Developers and operations teams make use of this API for automation, such as continuous integration/continuous deployment, or CI/CD.

The **Management UI** is the main web interface for administration and development. Developers can use it to create, develop, and manage APIs. Operations, security, and business users also access the management UI. The management UI uses the management API and leverages its role-based access control to allow the appropriate level of access to its users. The management UI can be used to view and control all aspects of your APIs, including controlling the API lifecycle, building and viewing analytics reports, and managing roles and users.

The **developer portal** is a web interface dedicated to addressing the needs of application developers. The API team publishes documentation about your company's APIs to the developer portal, where application developers can register their applications and sign up to use your API products.

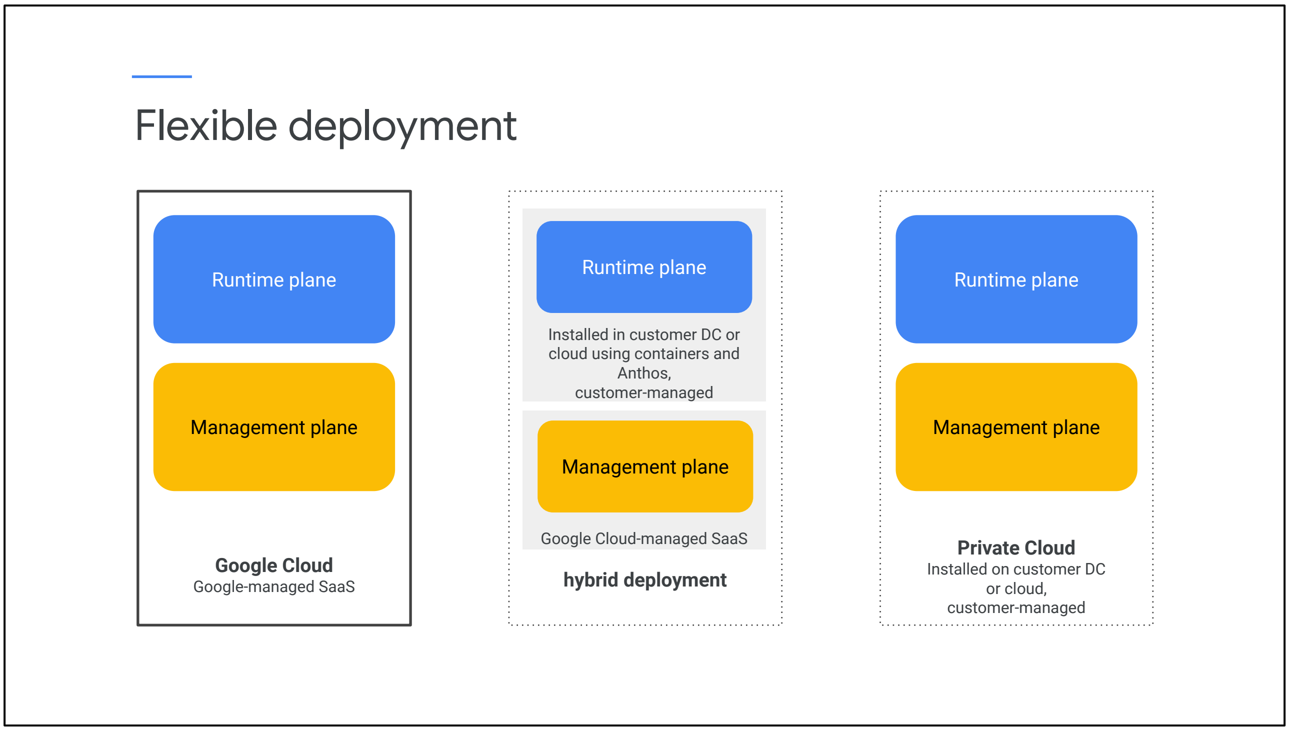
Each of these components can be horizontally scaled to meet high availability and resiliency requirements.



Flexible deployment

Runtime plane
Management plane
Google Cloud
Google-managed SaaS
Runtime plane
Installed in customer DC or cloud using containers and Anthos, customer-managed
Management plane
Google Cloud-managed SaaS
hybrid deployment
Runtime plane
Management plane
Private Cloud
Installed on customer DC or cloud, customer-managed

Apigee offers flexible platform deployment options.



Flexible deployment

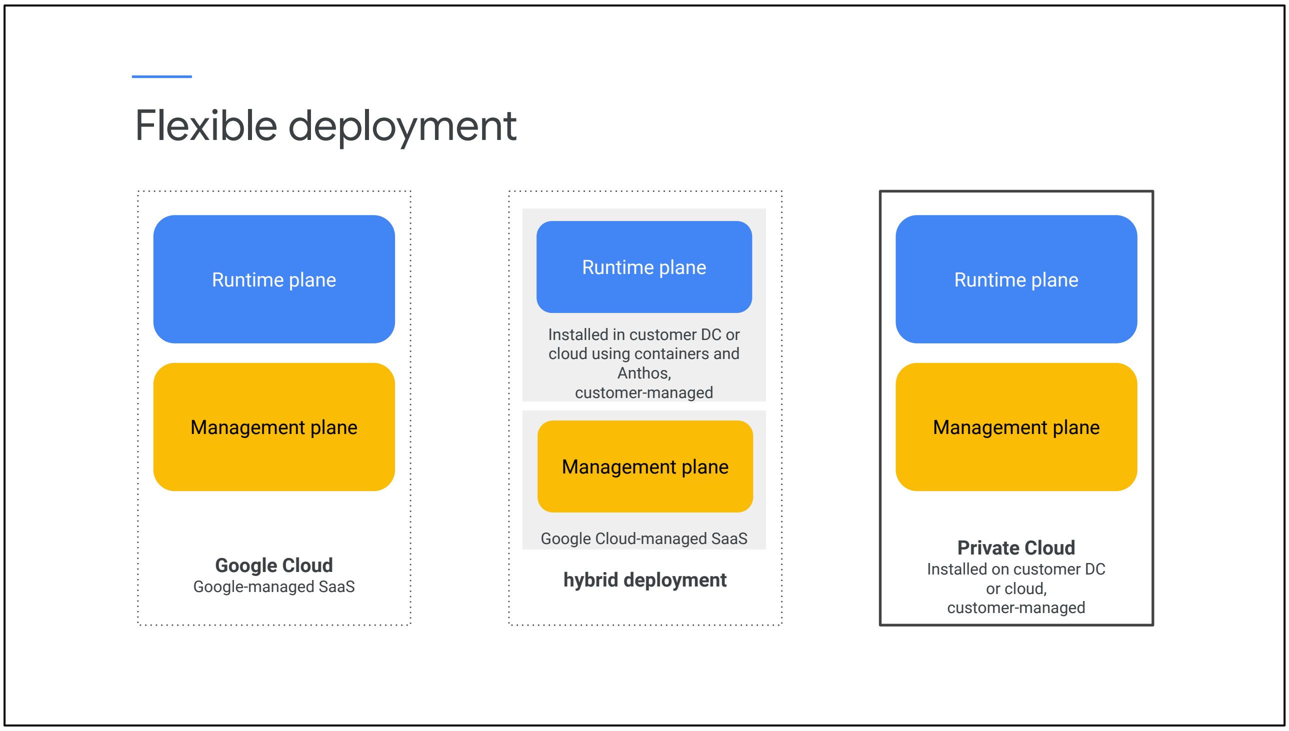
Runtime plane
Management plane
Google Cloud
Google-managed SaaS
Runtime plane
Installed in customer DC or cloud using containers and Anthos, customer-managed
Management plane
Google Cloud-managed SaaS
hybrid deployment
Runtime plane
Management plane
Private Cloud
Installed on customer DC or cloud, customer-managed

Apigee's Google-managed software-as-a-service deployment simplifies customer adoption and dramatically accelerates time to market for new APIs.

Developers can get started immediately building and running APIs at scale.

This is the most popular deployment option.

It allows customers to focus on addressing business needs, while letting Google manage the operational overhead of running the software at scale in a secure and reliable way.

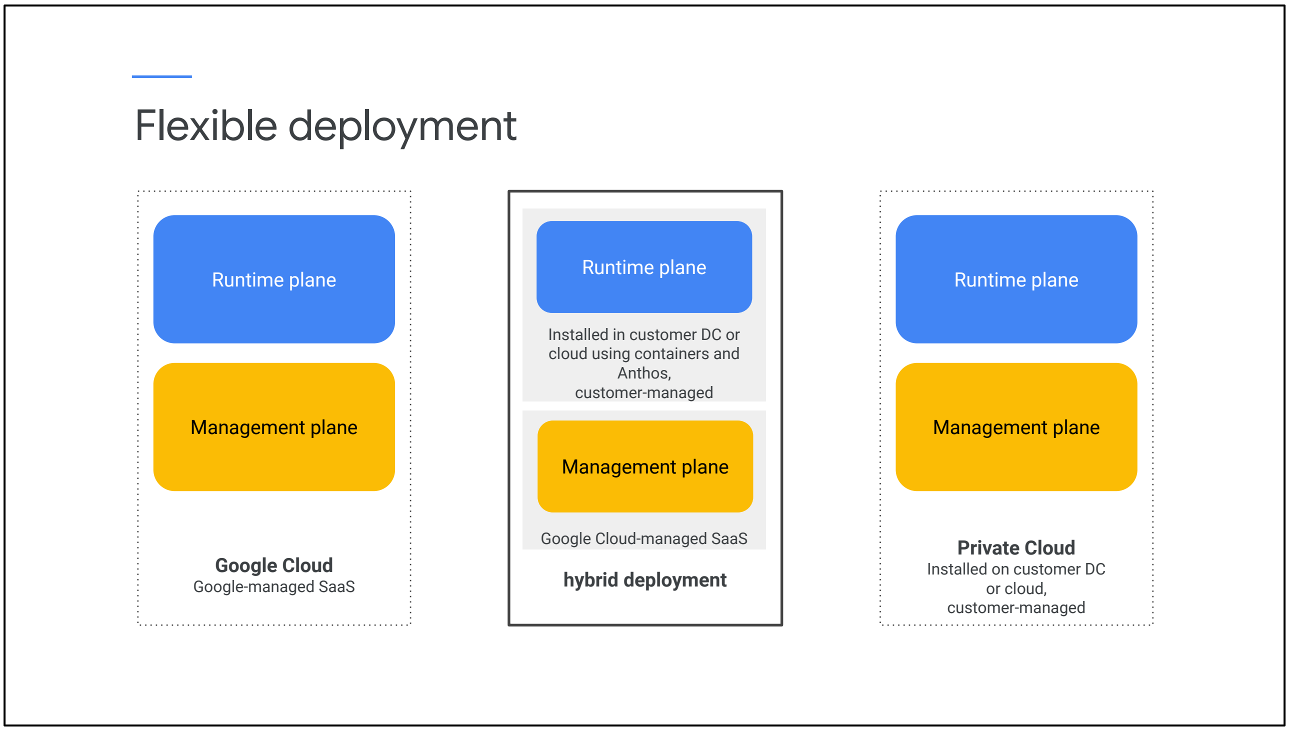


Flexible deployment

Runtime plane
Management plane
Google Cloud
Google-managed SaaS
Runtime plane
Installed in customer DC or cloud using containers and Anthos, customer-managed
Management plane
Google Cloud-managed SaaS
hybrid deployment
Runtime plane
Management plane
Private Cloud
Installed on customer DC or cloud, customer-managed

For customers who are willing to manage the Apigee infrastructure, Apigee can be deployed and managed in a private cloud.

Apigee can be installed in a customer data center or in an infrastructure-as-a-service cloud.



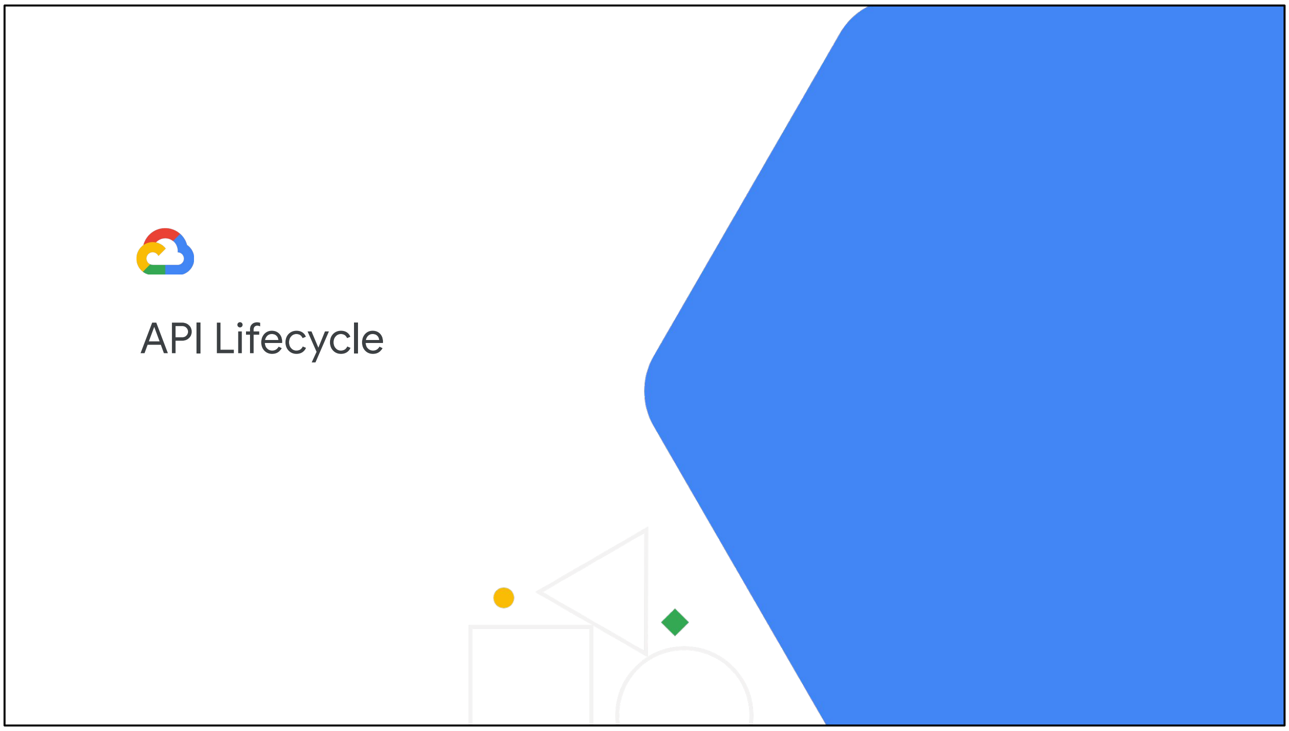
Flexible deployment

Runtime plane
Management plane
Google Cloud
Google-managed SaaS
Runtime plane
Installed in customer DC or cloud using containers and Anthos, customer-managed
Management plane
Google Cloud-managed SaaS
hybrid deployment
Runtime plane
Management plane
Private Cloud
Installed on customer DC or cloud, customer-managed

Customers who want to reduce infrastructure management costs but retain the flexibility of running APIs in multiple clouds or on-premises can choose the hybrid deployment model.

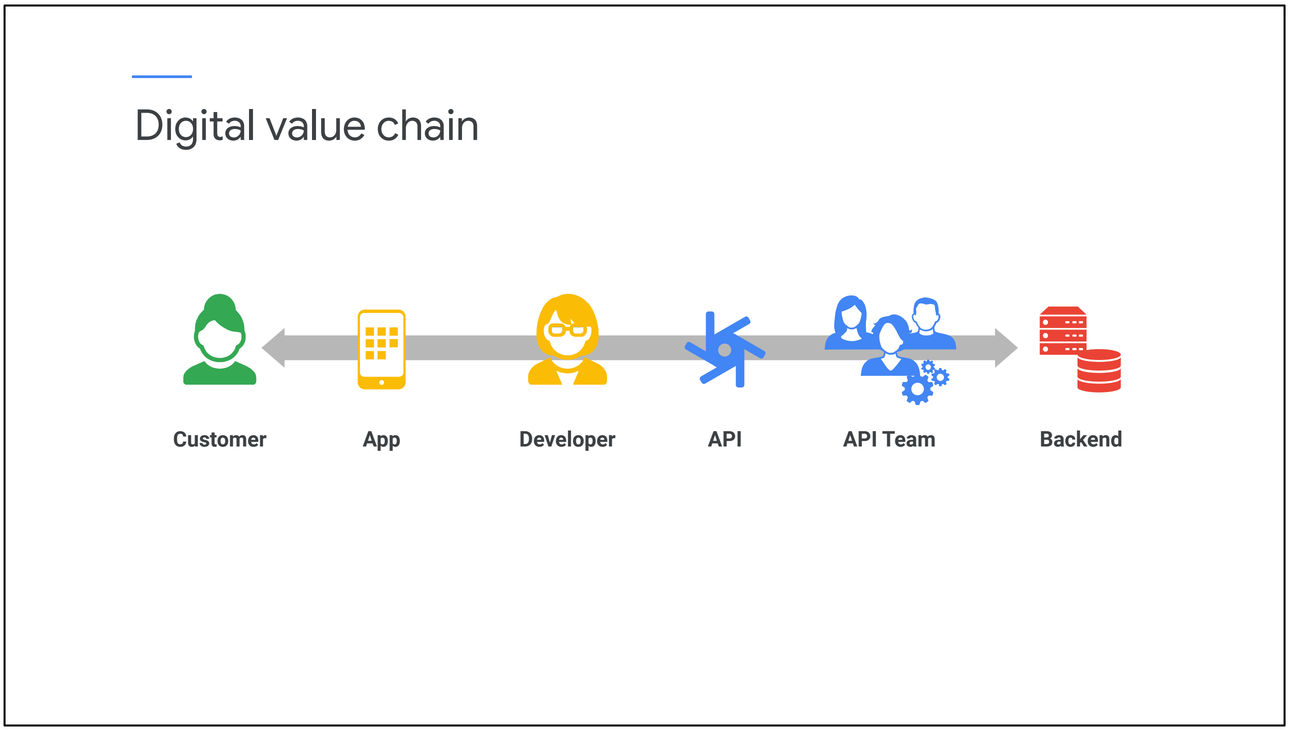
This model allows the customer to manage and deploy containerized versions of the API runtime wherever necessary, while delegating the management plane operations to Google.

Apigee's hybrid deployment model takes advantage of Anthos, Google Cloud's hybrid and multi-cloud application platform, which simplifies the management of runtime components across multiple clouds and data centers.

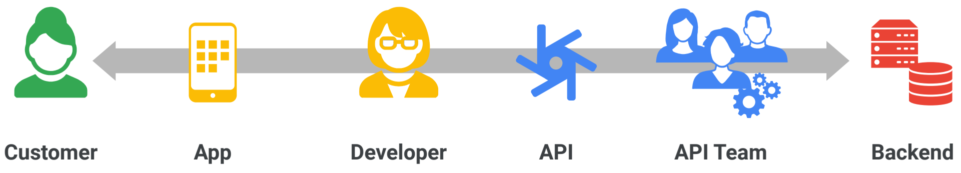


API Lifecycle

During this lecture we will discuss the API lifecycle and see how Apigee can help with development of your APIs and API programs.

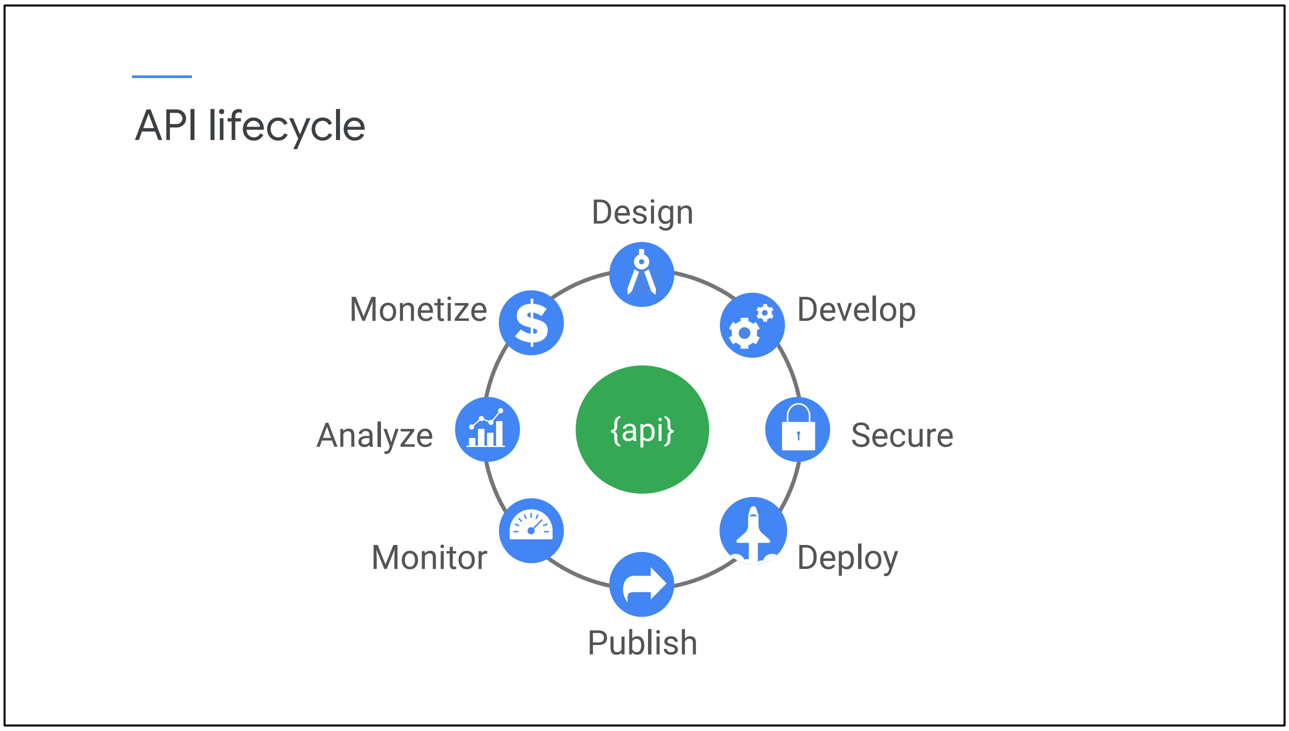


Digital value chain

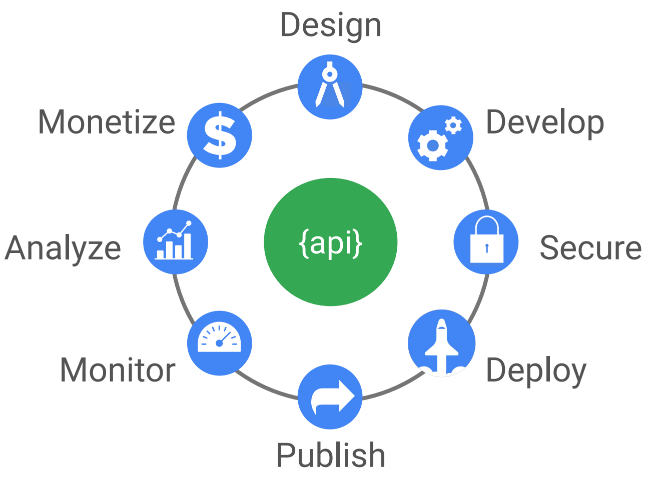


APIs can play a key role in your business and your ability to drive connected digital experiences.

It is important to continually improve your APIs to adjust to changing customer and business needs.



API lifecycle



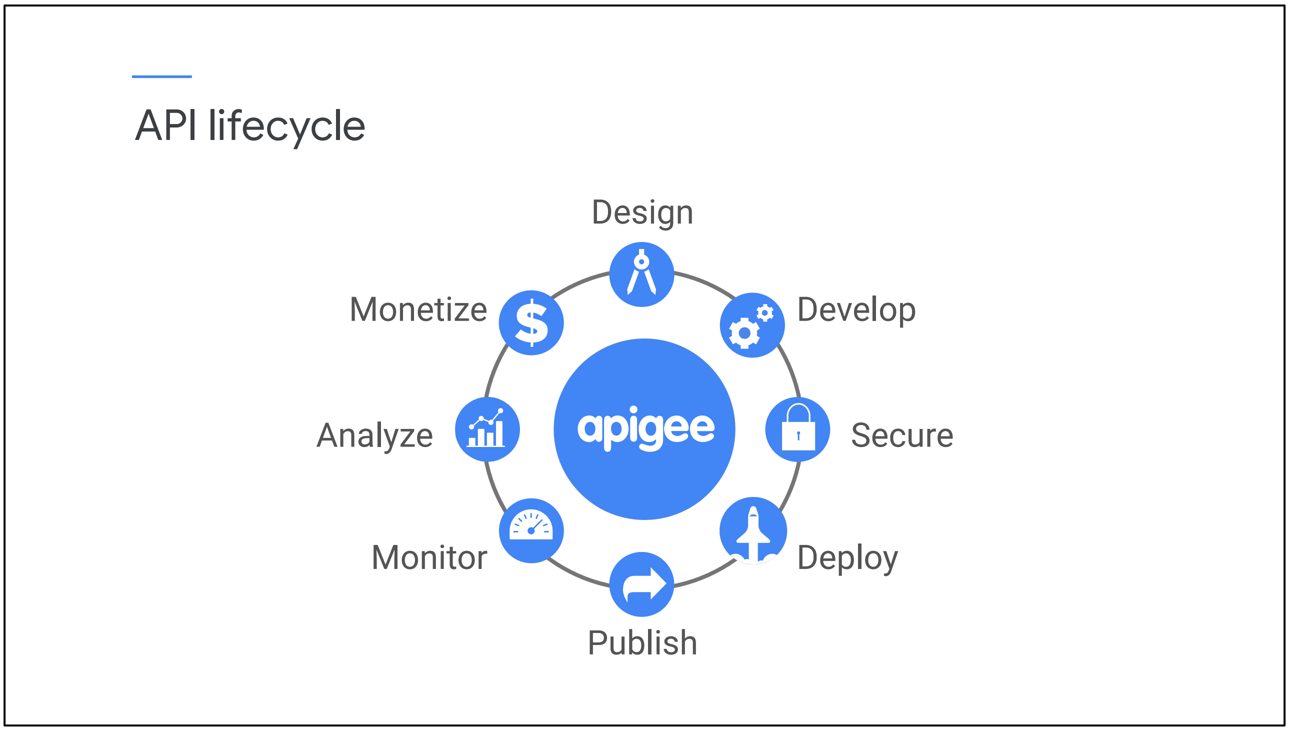
You may find it beneficial to think of your API development in terms of a life cycle.

Start at the top with the design of the API, and move clockwise. After the design has been reviewed and approved by stakeholders, you can develop your APIs and build security into them.

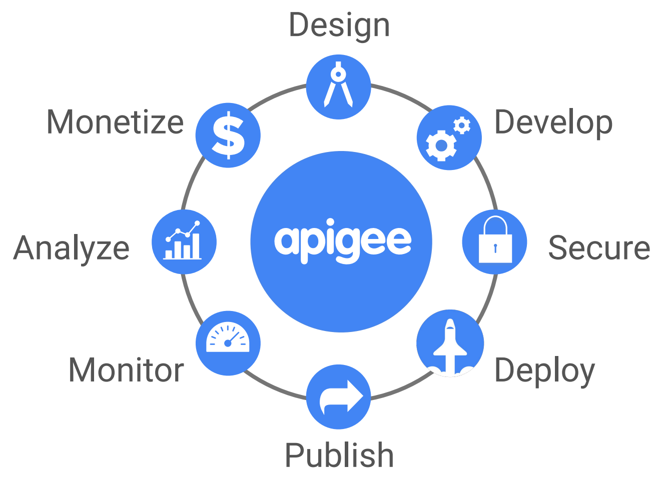
Your API is launched by deploying it into production and publishing it to app developers. When your API is in production, you must make sure to monitor the health and usage of your API.

Analytics can be used to determine your API's level of adoption and how it can be improved. Depending on your business model, it may make sense to monetize your API—charging for its use or sharing revenue with app developers who are driving new business.

With the feedback you receive from your app developers and the insights you gain from monitoring and analyzing your API program, you will have an understanding of necessary and desired changes. You can design new features for your API, beginning the cycle again.



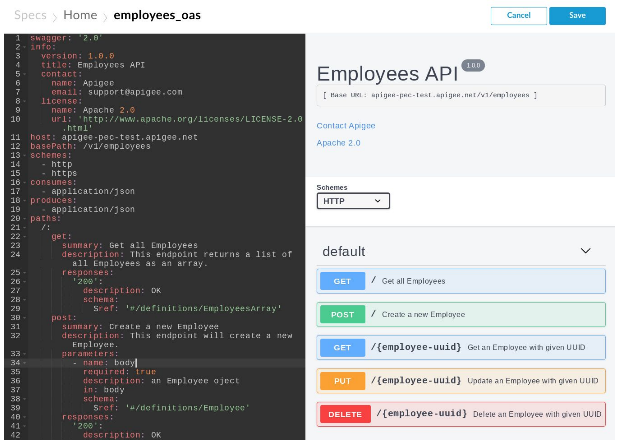
API lifecycle



Apigee has been designed to provide all of these capabilities, so you can manage the API lifecycle for your APIs.

Let's take a look at each stage of the life cycle and see how Apigee is used.

Design



```
swagger: '2.0'
info:
  version: 1.0.0
  title: Employees API
  contact:
    name: Apigee
    email: support@apigee.com
  license:
    name: Apache 2.0
    url: 'http://www.apache.org/licenses/LICENSE-2.0
      .html'
host: apigee-pec-test.apigee.net
basePath: /v1/employees
schemes:
  - http
  - https
consumes:
  - application/json
produces:
  - application/json
paths:
  /:
    get:
      summary: Get all Employees
      description: This endpoint returns a list of
        all Employees as an array.
      responses:
        '200':
          description: OK
          schema:
            $ref: '#/definitions/EmployeesArray'
    post:
      summary: Create a new Employee
      description: This endpoint will create a new
        Employee.
      parameters:
        - name: body
          required: true
          description: an Employee oject
          in: body
          schema:
            $ref: '#/definitions/Employee'
      responses:
        '200':
          description: OK
```

Employees API 1.0.0

[Base URL: apigee-pec-test.apigee.net/v1/employees]

Contact Apigee

Apache 2.0

Schemes: HTTP

default

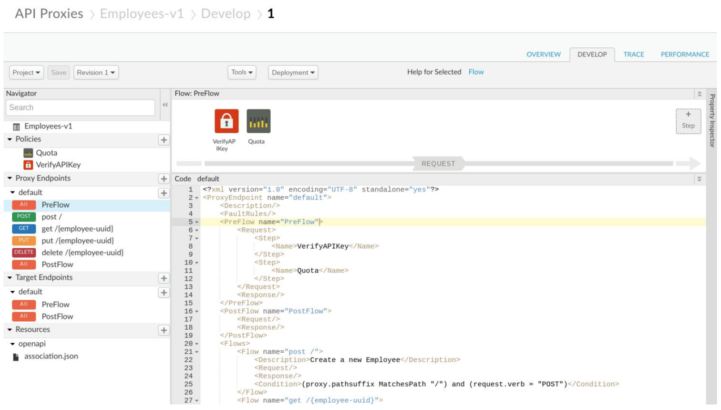
- GET / Get all Employees
- POST / Create a new Employee
- GET /{employee-uuid} Get an Employee with given UUID
- PUT /{employee-uuid} Update an Employee with given UUID
- DELETE /{employee-uuid} Delete an Employee with given UUID

An OpenAPI specification is used to define the interface and capabilities of your APIs, without focusing on the implementation.

OpenAPI specifications can be designed using Apigee's OpenAPI editor, which automatically generates live documentation from the specification. This documentation allows app developers to explore and try out your APIs.

An OpenAPI specification can also be used to generate an API proxy stub. The API proxy stub provides a template for building an API that adheres to the defined specification.

Develop

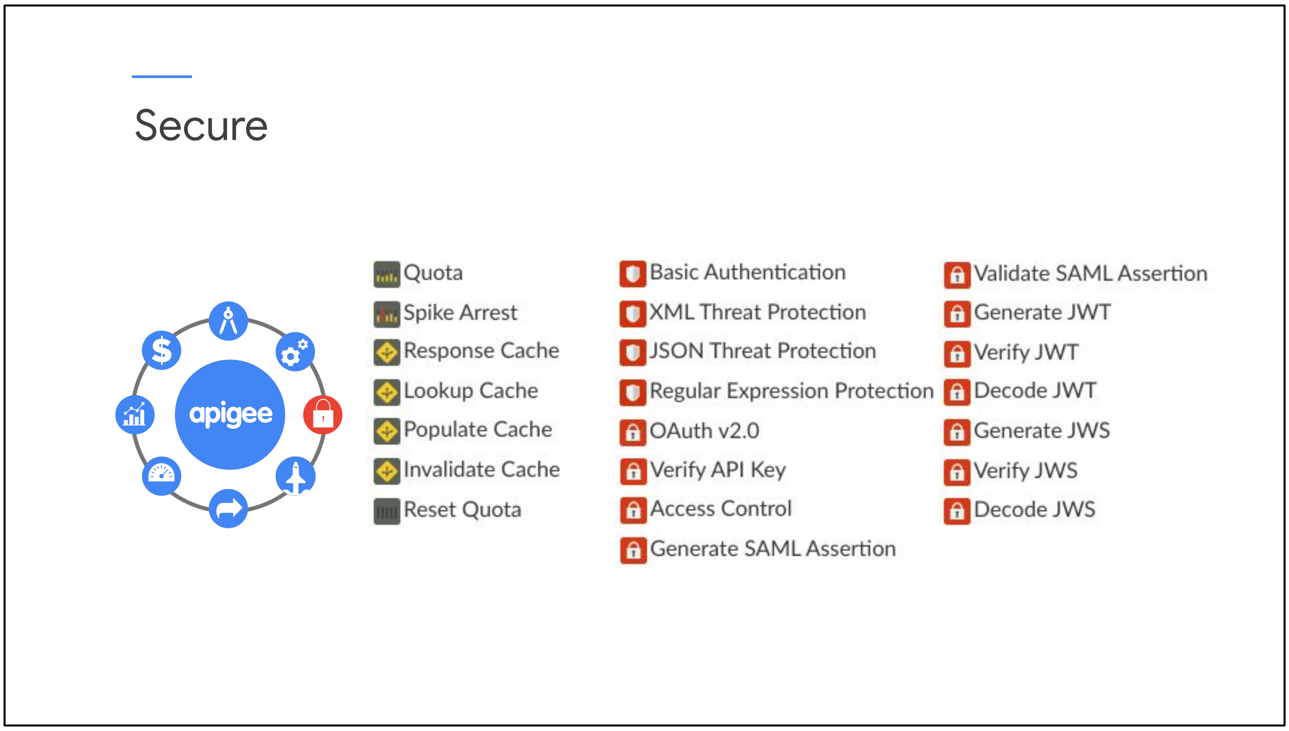


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ProxyEndpoint name="default">
    <Description/>
    <FaultRules/>
    <PreFlow name="PreFlow">
        <Request>
            <Step>
                <Name>VerifyAPIKey</Name>
            </Step>
            <Step>
                <Name>Quota</Name>
            </Step>
        </Request>
        <Response/>
    </PreFlow>
    <PostFlow name="PostFlow">
        <Request/>
        <Response/>
    </PostFlow>
    <Flows>
        <Flow name="post /">
            <Description>Create a new Employee</Description>
            <Request/>
            <Response/>
            <Condition>(proxy.pathsuffix MatchesPath "/") and (request.verb = "POST")</Condition>
        </Flow>
        <Flow name="get /{employee-uuid}">
```

Apigee allows you to build your API proxies using policies, which are pre-built functions that can be configured without code.

Apigee also has built-in support for JavaScript or Java policies, which allow you to write custom code when needed for more complex use cases.

Your proxies can be debugged using Apigee's trace tool, so you can troubleshoot issues during development or in production.



Secure

- Quota
- Spike Arrest
- Response Cache
- Lookup Cache
- Populate Cache
- Invalidate Cache
- Reset Quota

- Basic Authentication
- XML Threat Protection
- JSON Threat Protection
- Regular Expression Protection
- OAuth v2.0
- Verify API Key
- Access Control
- Generate SAML Assertion

- Validate SAML Assertion
- Generate JWT
- Verify JWT
- Decode JWT
- Generate JWS
- Verify JWS
- Decode JWS

Many of Apigee's policies can be used to secure your APIs.

You can use traffic management to block excessive traffic or enforce traffic quotas on applications.

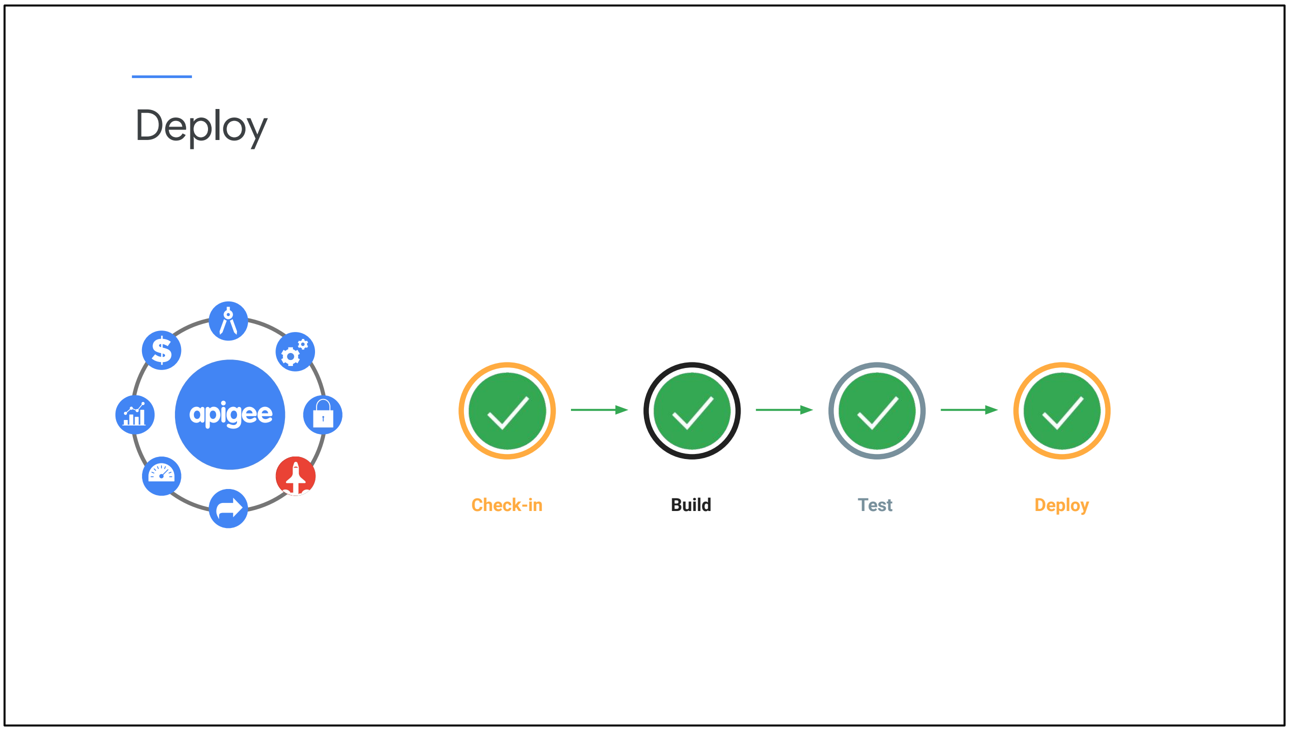
Apigee has threat protection policies that can be used to detect malicious request payloads and reject them before they are sent to your backend services.

Other policies, such as the OAuth policy, allow you to secure your APIs by providing authentication and authorization of applications and their users.

In addition to the security built into your API using policies, Apigee secures your data across the internet using point-to-point encryption.

Apigee provides access control features to help prevent users of your platform from seeing sensitive configuration or user data.

With all of its security features, Apigee can be used to create secure APIs even if your backend services are not fully secured.



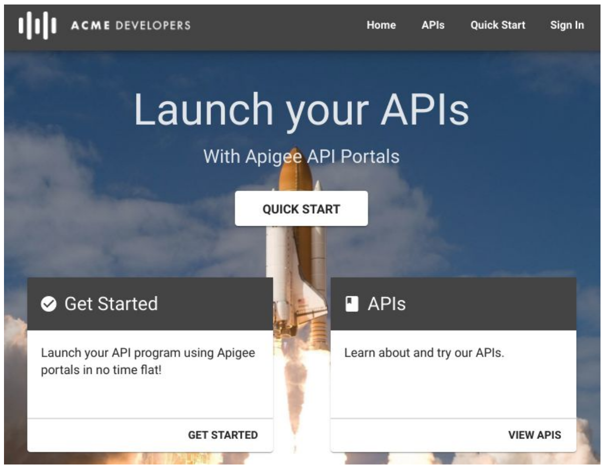
Deploy

When your API has been built, you will need to deploy your API proxy into production.

The deployment process should include testing and should be repeatable. This process can be built into a deployment pipeline, where changes to a proxy are automatically tested before being deployed.

Apigee provides a management API that can be used to create and deploy proxies and configuration artifacts as part of a deployment pipeline, allowing you to build a repeatable process for deploying new or updated APIs.

Publish

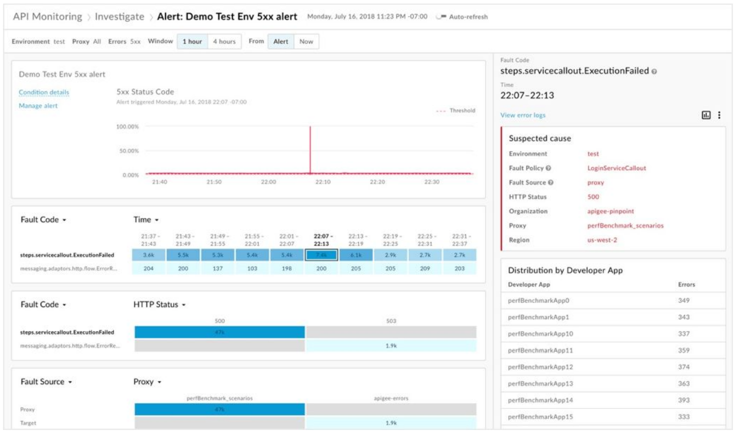


Apigee's developer portal helps your app developers discover your APIs and register apps to use them.

Your OpenAPI specifications can be used to create the live documentation hosted in the developer portal, allowing app developers to try out your APIs.

Apigee provides 2 types of developer portals: a Drupal-based portal that offers a full-featured, customizable content management system; and a hosted, integrated portal, which requires much less effort but lacks some of the features and customization of the Drupal portal.

Monitor


API Monitoring › Investigate › **Alert: Demo Test Env 5xx alert** Monday, July 16, 2018 11:23 PM -07:00 Auto-refresh

Environment test Proxy All Errors 5xx Window 1 hour 4 hours From Alert Now

Demo Test Env 5xx alert

Condition details

Manage alert

5xx Status Code

Alert triggered Monday, Jul 16, 2018 22:07 -07:00

Fault Code	21:37 - 21:43	21:43 - 21:49	21:49 - 21:55	21:55 - 22:01	22:01 - 22:07	**22:07 - 22:13**	22:13 - 22:19	22:19 - 22:25	22:25 - 22:31	22:31 - 22:37
steps.servicecallout.ExecutionFailed	3.6k	5.5k	5.3k	5.4k	5.4k	7.4k	6.1k	2.9k	2.7k	2.7k
messaging.adaptors.http.flow.ErrorR...	204	200	137	103	198	200	205	205	209	203

Fault Code	500	503
steps.servicecallout.ExecutionFailed	47k	
messaging.adaptors.http.flow.ErrorRe...		1.9k

Fault Source	perfBenchmark_scenarios	apigee-errors
Proxy	47k	
Target		1.9k

Fault Code

steps.servicecallout.ExecutionFailed

Time

22:07–22:13

View error logs

Suspected cause

Environment	test
Fault Policy	LoginServiceCallout
Fault Source	proxy
HTTP Status	500
Organization	apigee-pinpoint
Proxy	perfBenchmark_scenarios
Region	us-west-2

Distribution by Developer App

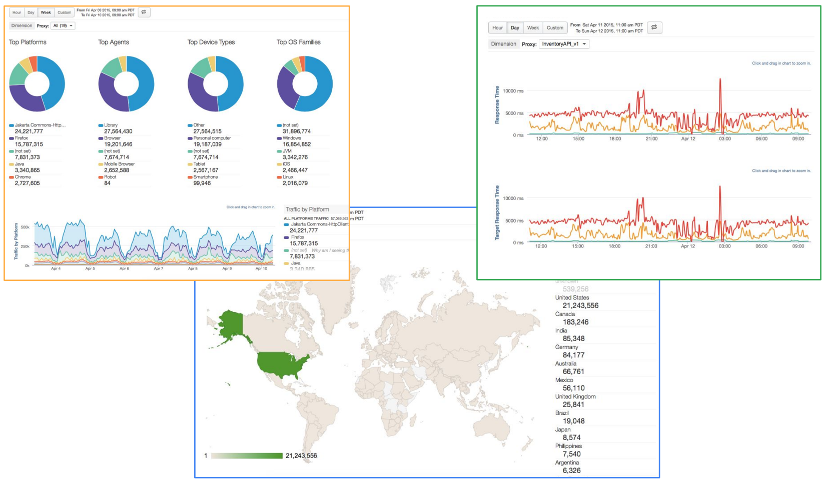
Developer App	Errors
perfBenchmarkApp0	349
perfBenchmarkApp1	343
perfBenchmarkApp10	337
perfBenchmarkApp11	359
perfBenchmarkApp12	374
perfBenchmarkApp13	363
perfBenchmarkApp14	393
perfBenchmarkApp15	333

Apigee monitors API performance and usage, automatically capturing API and backend latencies, error rates, and call volume, among other types of operational metrics.

Apigee also captures business metrics, helping you identify how specific apps and app developers are using your APIs.

Apigee's API monitoring product helps you identify issues before your app developers and users of their apps notice them. It leverages heat maps and alerting to allow you to quickly diagnose problems.

Analyze

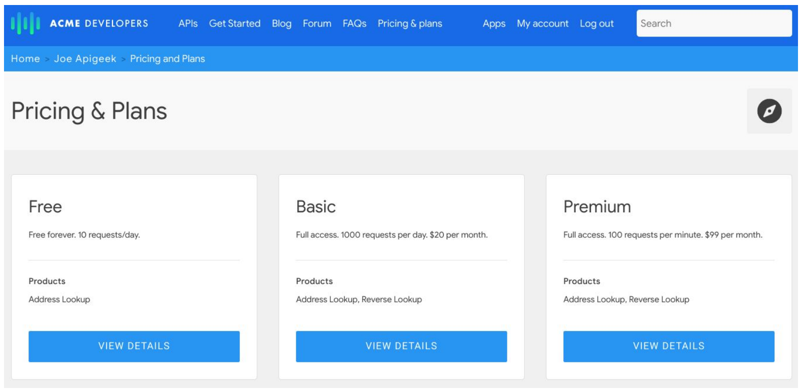


In addition to API and performance metrics, Apigee captures business metrics, tracking the apps and app developers using your APIs and the device types and geolocation of the users of those apps. Other metrics specific to your business can be captured by collecting custom statistics within your API proxies.

Apigee includes a rich set of built-in reports to help gain insights into your APIs and API program. Custom reports can also be created to explore business-specific data.

Apigee's Analytics data can be integrated into your own enterprise systems by using the metrics API or by extracting the data into Google’s Cloud Storage or BigQuery.

Monetize



ACME DEVELOPERS APIs Get Started Blog Forum FAQs Pricing & plans Apps My account Log out Search

Home › Joe Apigeek › Pricing and Plans

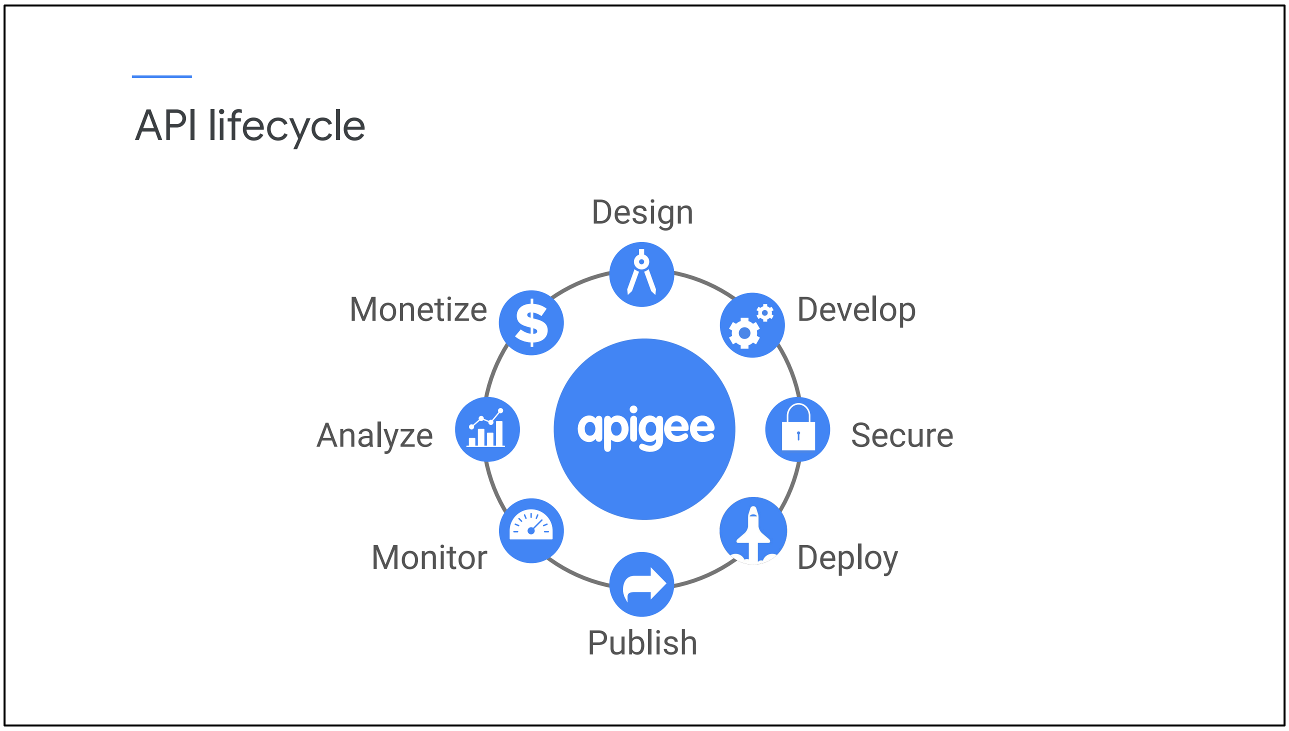
Pricing & Plans

Free	Basic	Premium
Free forever. 10 requests/day.	Full access. 1000 requests per day. $20 per month.	Full access. 100 requests per minute. $99 per month.
Products: Address Lookup	Products: Address Lookup, Reverse Lookup	Products: Address Lookup, Reverse Lookup
VIEW DETAILS	VIEW DETAILS	VIEW DETAILS

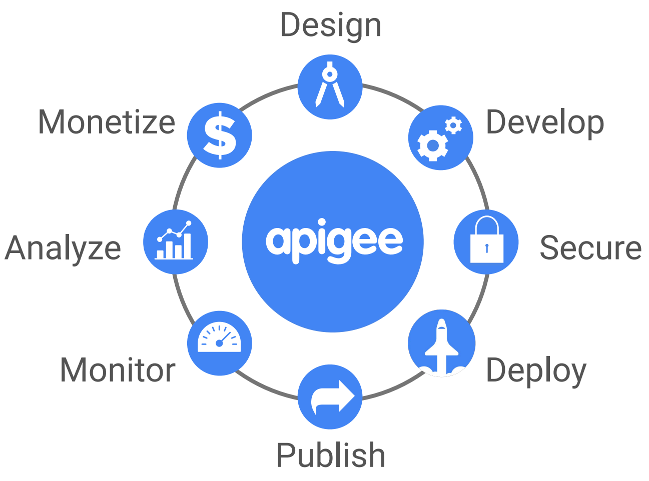
Companies with public API programs, or those offering digital products to partners, can use Apigee's monetization capability to create revenue streams based on API powered digital products.

Apigee monetization allows you to charge for API usage or share revenue with app developers that drive your business.

App developers can easily set up billing, choose rate plans, and process credit card payments from within the developer portal.

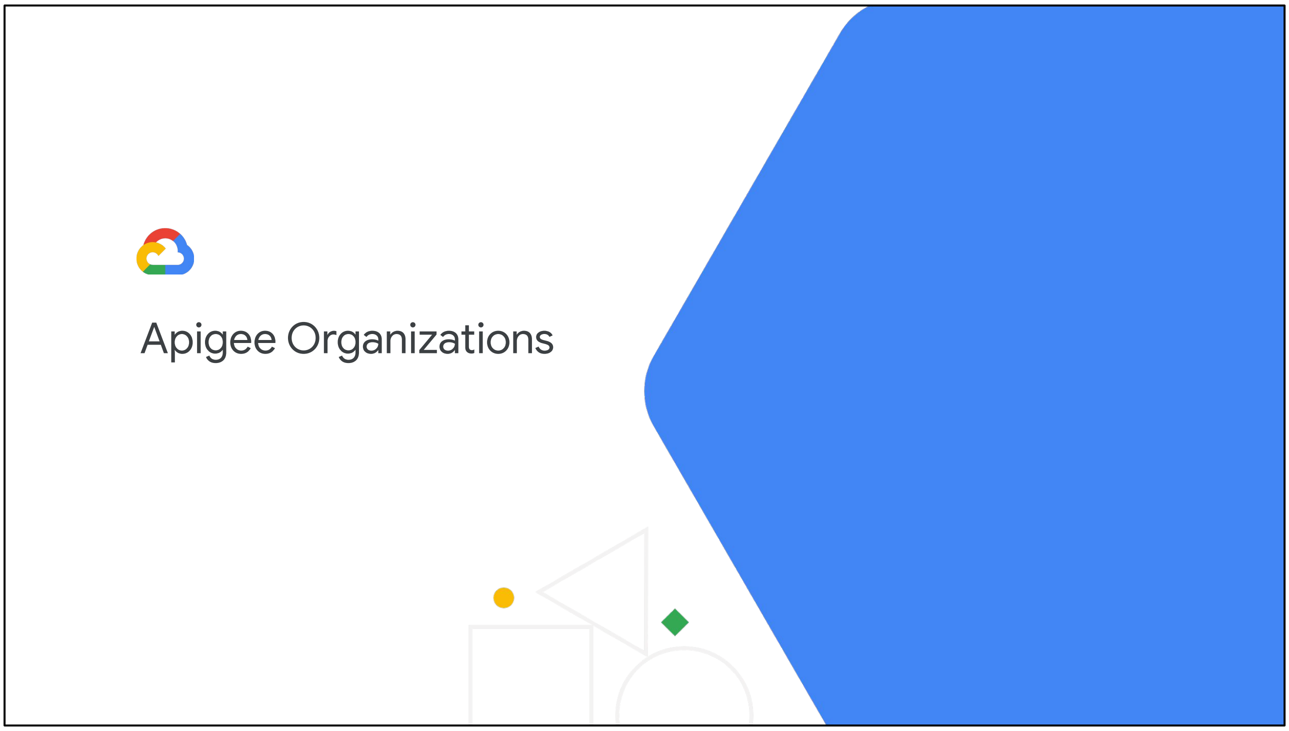


API lifecycle



APIs are key factors in today's digital businesses.

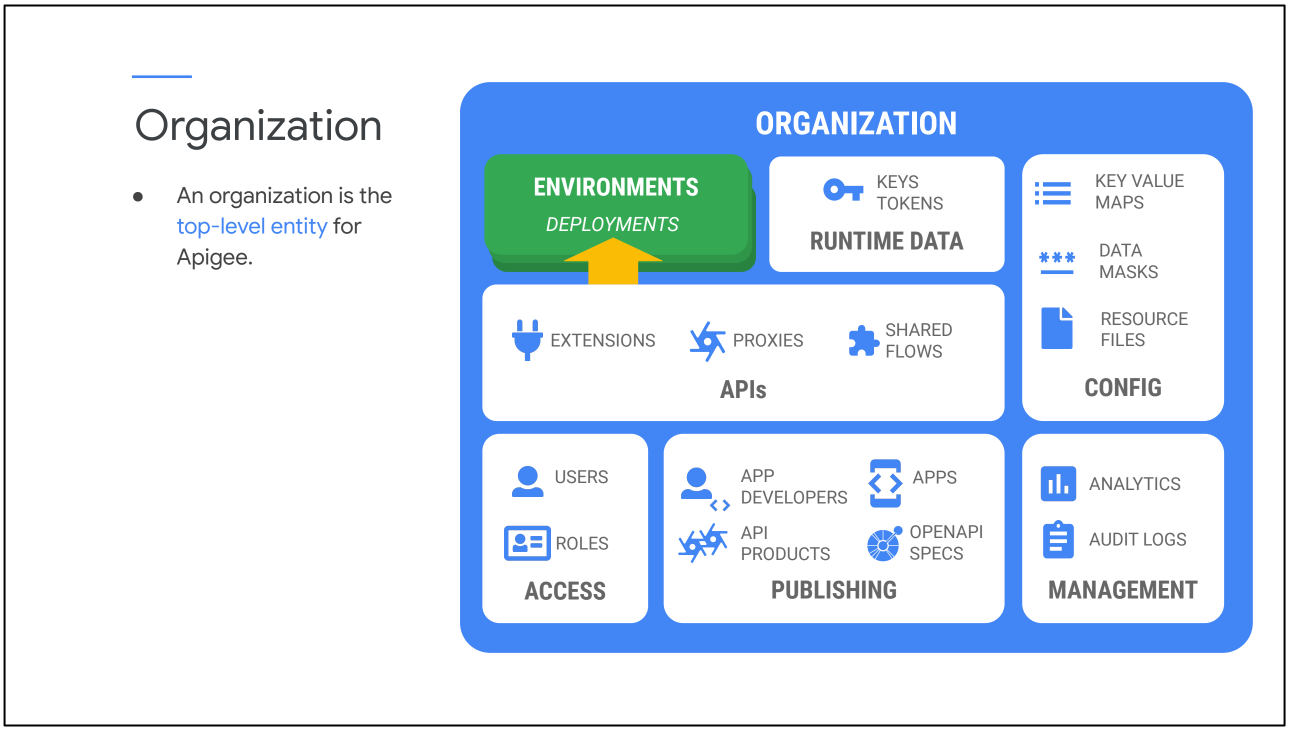
Apigee can help you manage all aspects of the API lifecycle, helping you to improve your APIs, create new APIs to address new opportunities, and grow your API program.



Apigee Organizations

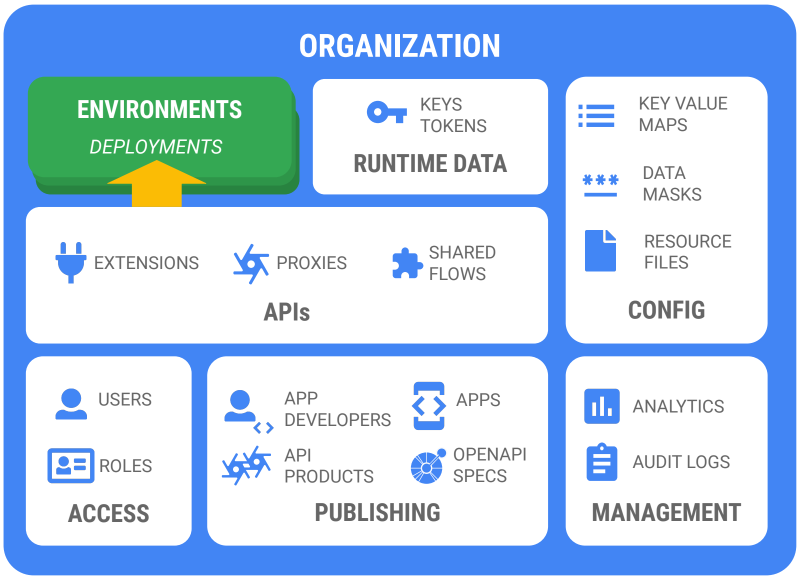
This lecture will be a quick introduction to Apigee organizations and the entities they contain.

You will learn more about all of these entities during this series of courses.



Organization

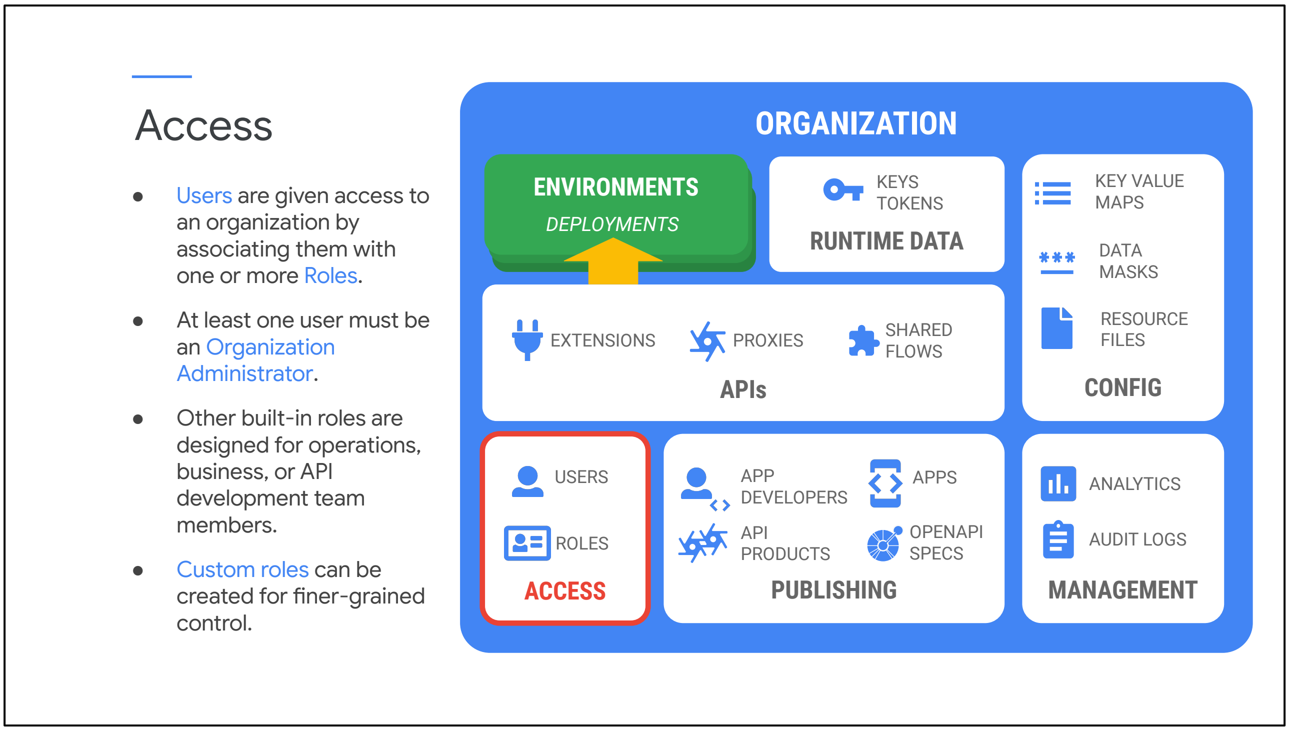
- An organization is the top-level entity for Apigee.



The **organization** is the top-level entity for Apigee. When you use the Apigee management user interface, you are working within the context of an organization.

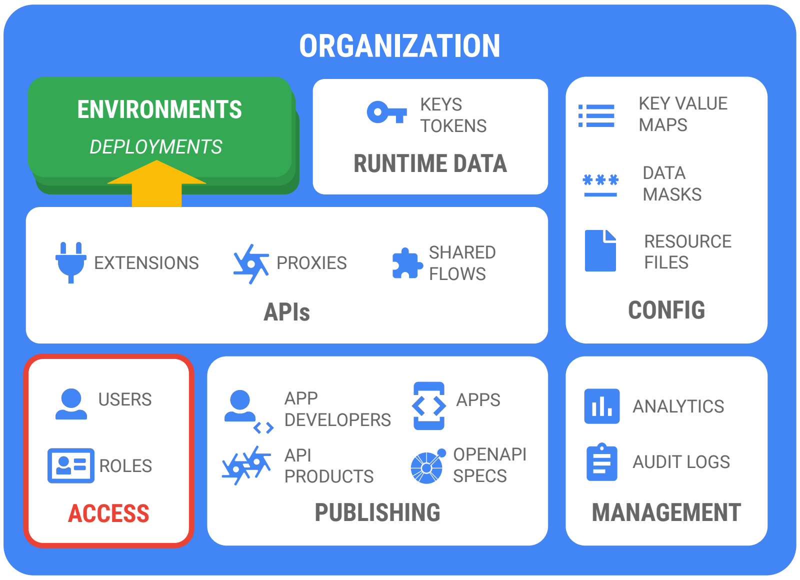
An organization contains many types of entities. Some entities live inside environments, which are runtime execution contexts for your APIs.

Let's quickly review organization and environment entities.



Access

- Users are given access to an organization by associating them with one or more Roles.
- At least one user must be an Organization Administrator.
- Other built-in roles are designed for operations, business, or API development team members.
- Custom roles can be created for finer-grained control.

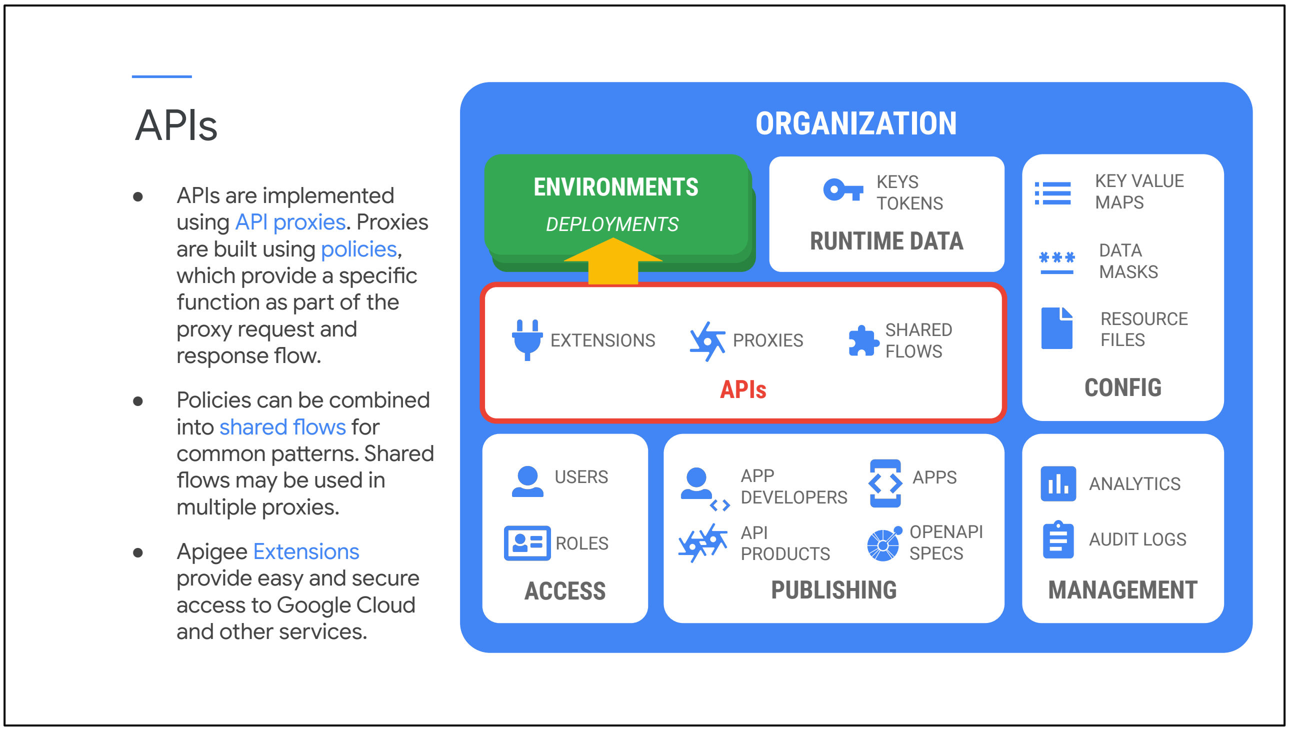


Users can be granted access to one or more organizations. Users are associated with one or more **roles** within an organization. The role specifies the set of permissions that is granted to a user.

At least one user must have the built-in Organization Administrator role, which provides superuser access to the organization.

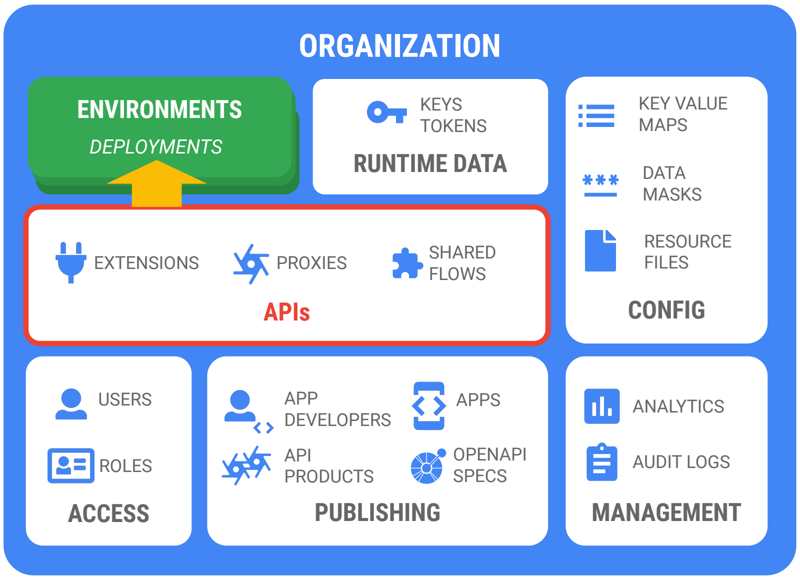
Other built-in roles specify permissions appropriate for other users of Apigee, including operations, business, and API development team members.

Organization administrators can create custom roles, which provide finer-grained control over a user's access within an organization.



APIs

- APIs are implemented using API proxies. Proxies are built using policies, which provide a specific function as part of the proxy request and response flow.
- Policies can be combined into shared flows for common patterns. Shared flows may be used in multiple proxies.
- Apigee Extensions provide easy and secure access to Google Cloud and other services.

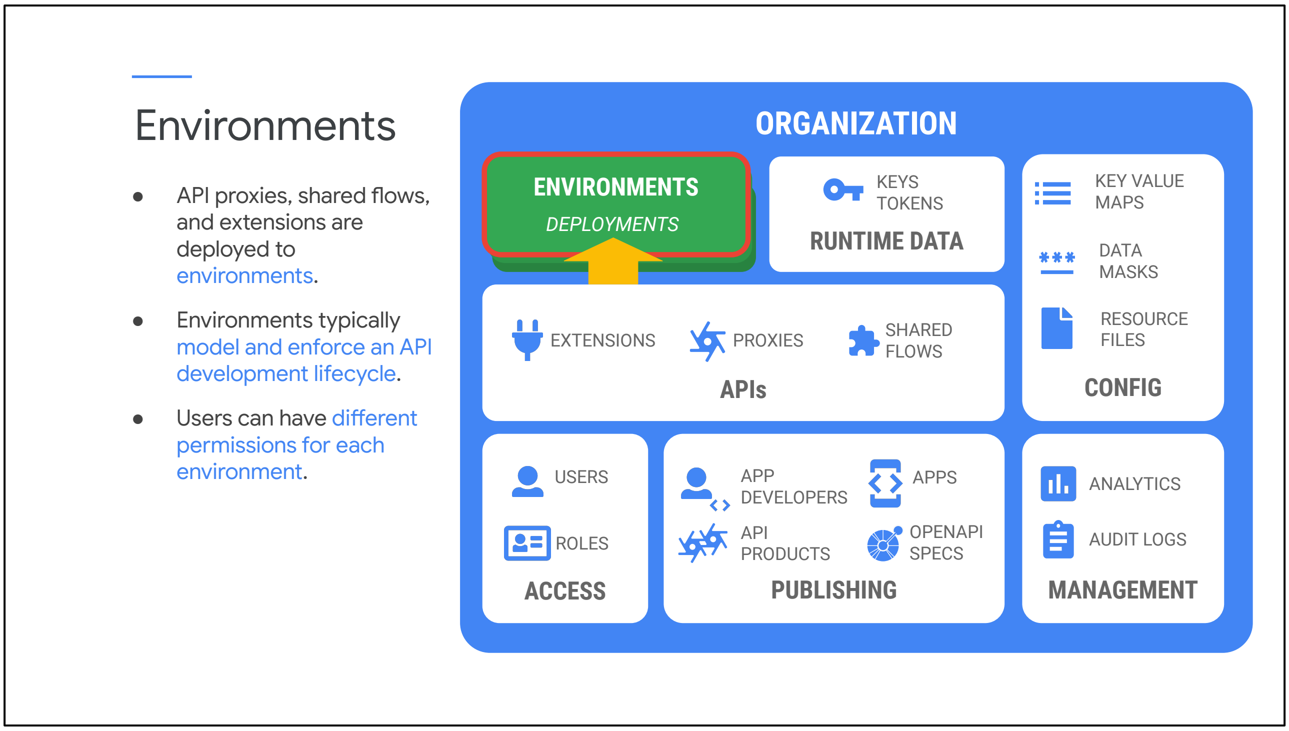


An organization's **API proxies** are scoped at the organization level. APIs are exposed on Apigee by implementing API proxies.

These proxies are built using **policies**, which are pre-built modules that provide features like security, rate-limiting, message transformation, or mediation within the request and response flows of your proxy. Policies allow your APIs to provide rich functionality without your having to write lots of code.

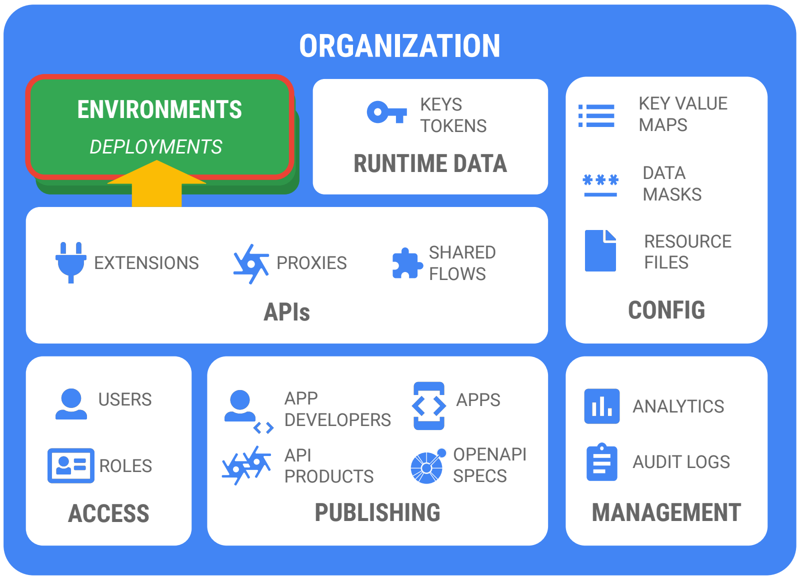
Shared flows can be used to combine a set of policies into a common pattern, allowing reuse of proxy logic in multiple APIs.

Extensions are used within your proxies to simplify access to Google Cloud and other common services.



Environments

- API proxies, shared flows, and extensions are deployed to environments.
- Environments typically model and enforce an API development lifecycle.
- Users can have different permissions for each environment.



API proxies, shared flows, and extensions are deployed to **environments**. API requests are handled by a proxy deployed in a specific environment.

Environments are typically used to model and enforce your API development lifecycle. An organization might have three environments: development, test, and production.

An API developer would work on a new proxy, or changes to an existing proxy, in the development environment. When the API developer is confident that the proxy is working as intended, that revision of the proxy can be deployed to the test environment, where more formal testing could occur. Finally, the tested revision of the proxy can be moved into the production environment.

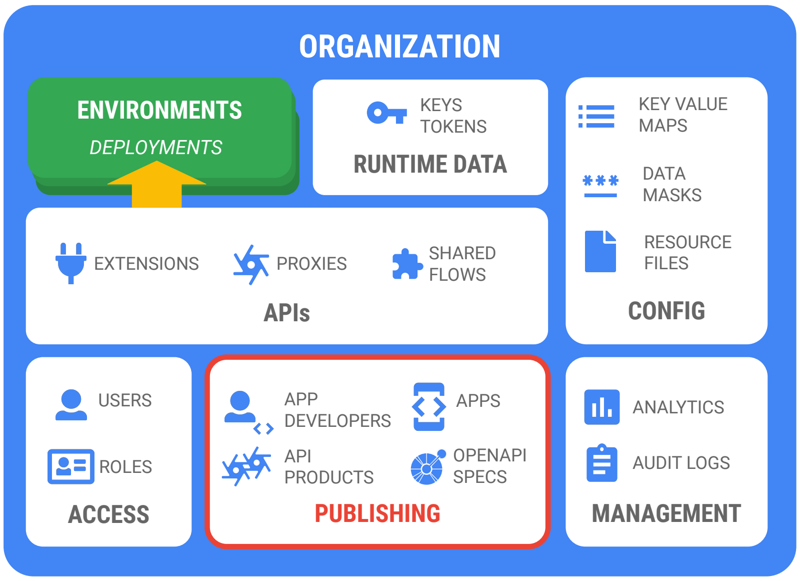
Users can be given different permissions in each environment.

A developer might need full access in the development environment, but should have no write access in production.

The support team might have only read-only access in development, but could trace proxies in production.

Publishing

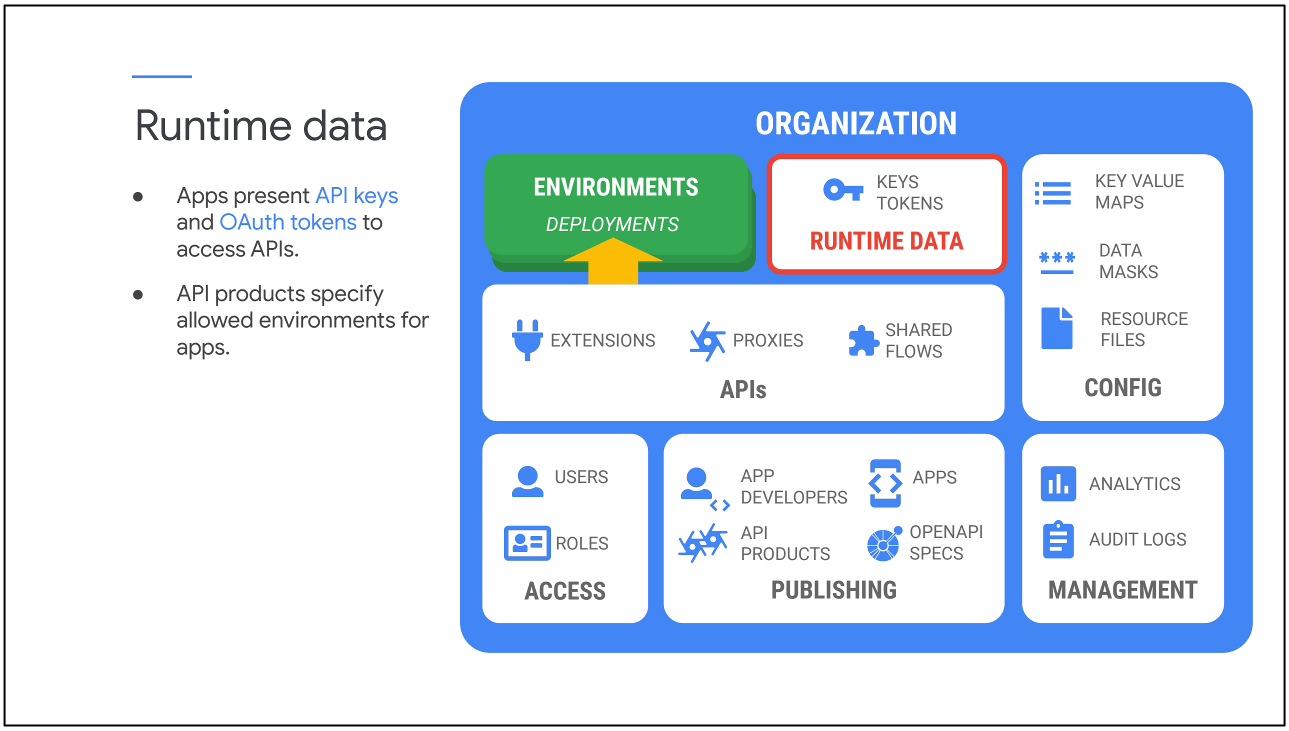
- APIs are documented using OpenAPI specifications.
- APIs are productized by exposing them in your developer portal as API products.
- App developers register apps to use one or more API products.



An **OpenAPI specification** defines an interface for an API. A specification generally details all requests and responses for the API, including for error responses.

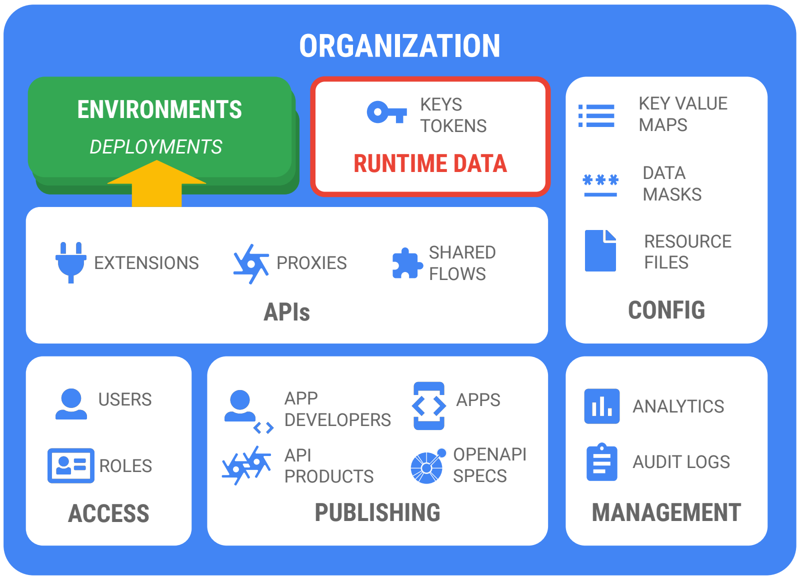
Before publishing your APIs to the developer portal, you create **API products** for your APIs. API products provide a mechanism for access and authorization for a group of APIs.

App developers access the developer portal to discover your APIs and experiment with them. Within the developer portal, app developers may register **apps** with API products to allow access to your APIs.



Runtime data

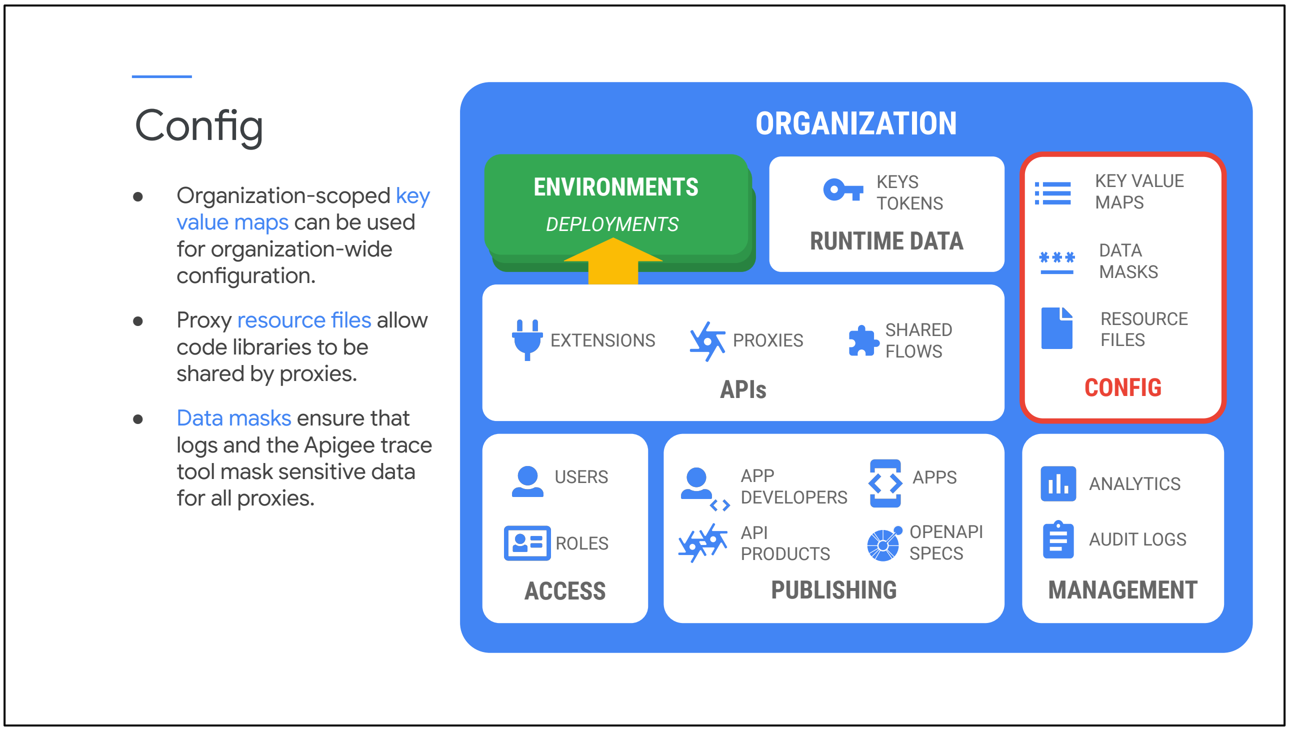
- Apps present API keys and OAuth tokens to access APIs.
- API products specify allowed environments for apps.



Apps present **API keys** and **OAuth tokens** to access APIs. When an API key or OAuth token is verified in an API proxy, the app making the request is identified, as is the associated API product. This allows proxies to control functionality based on API product or app.

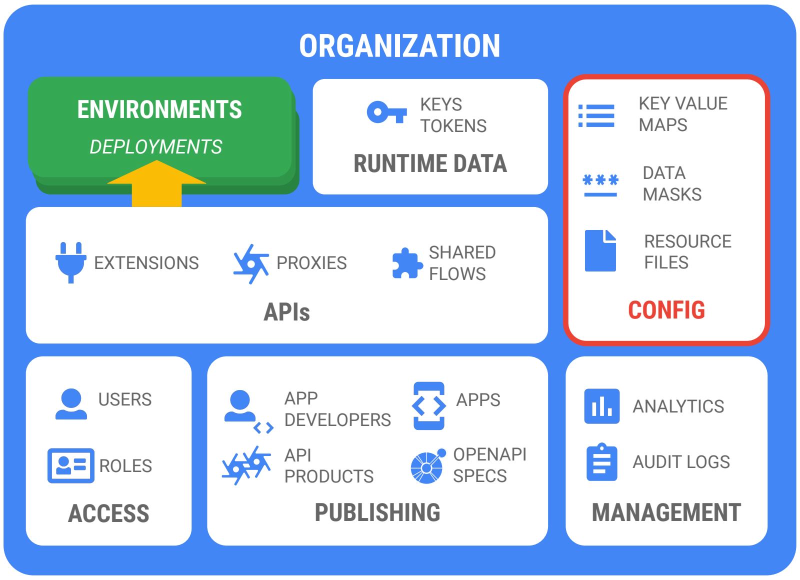
API keys and tokens are stored at the organization level, but are generally associated with a single environment.

The API product associated with the app specifies which environment or environments can be used.



Config

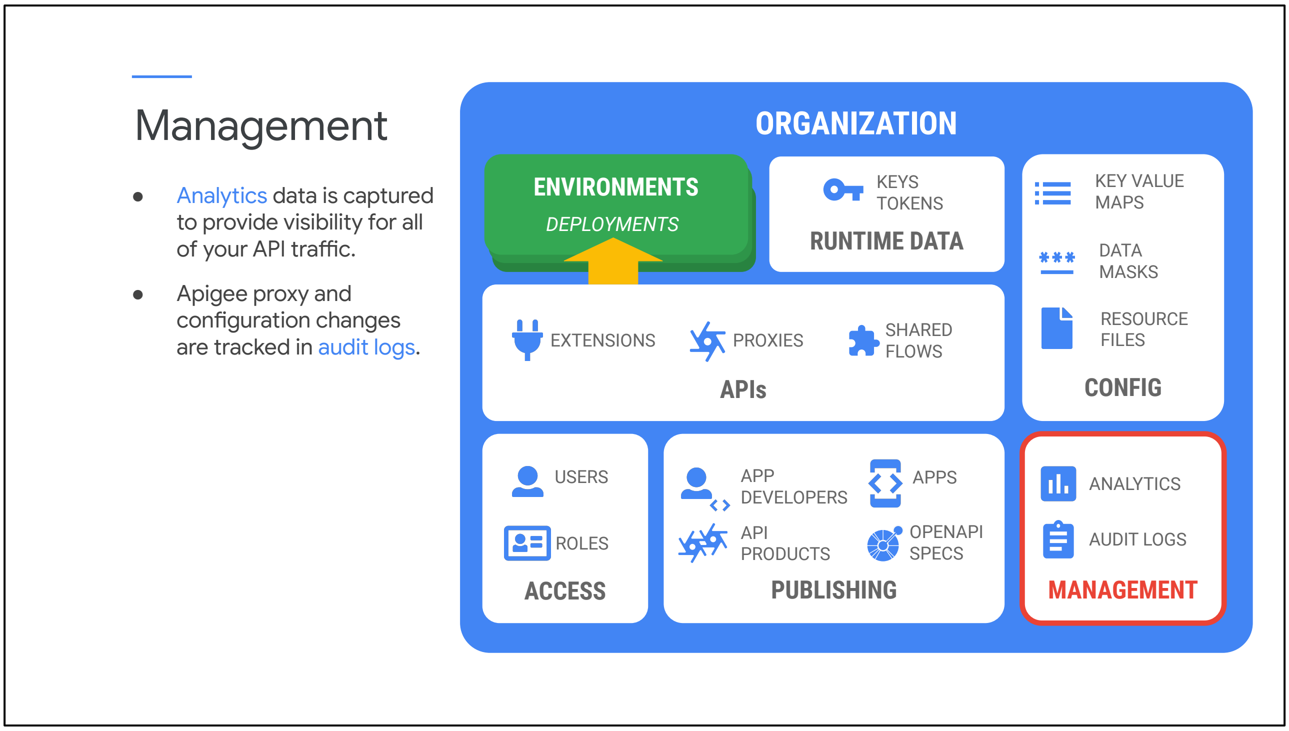
- Organization-scoped key value maps can be used for organization-wide configuration.
- Proxy resource files allow code libraries to be shared by proxies.
- Data masks ensure that logs and the Apigee trace tool mask sensitive data for all proxies.



Organization-scoped **key value maps**, or KVMs, can store organization-wide configuration. KVMs can be encrypted to store passwords or other sensitive information.

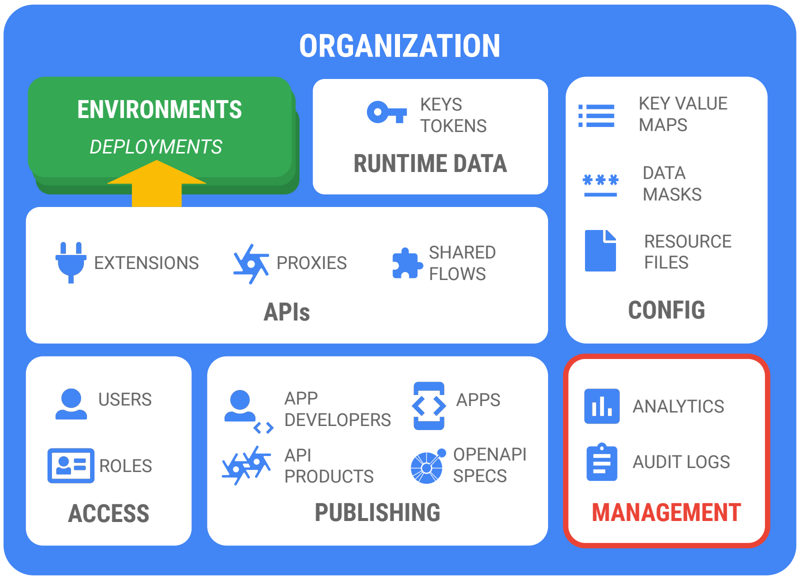
Organization-scoped proxy **resource files** can be used to share code libraries among proxies.

Data masks can be created to mask sensitive data in logs and the trace tool.



Management

- Analytics data is captured to provide visibility for all of your API traffic.
- Apigee proxy and configuration changes are tracked in audit logs.



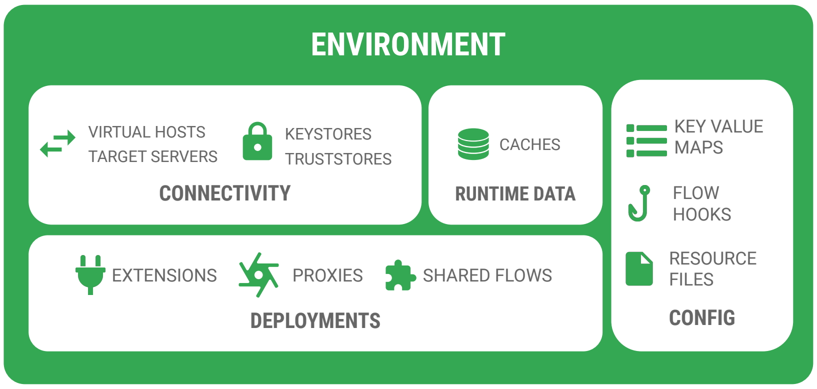
Analytics data provides visibility for all API traffic, from an application through Apigee to your backend services and back.

Operational and business metrics are automatically captured for each API call, and a wide range of provided reports allows you to gain insight into your APIs.

API proxy and configuration changes within an organization are tracked in **audit logs**.

Environment

- An environment is a runtime execution content for API proxies.
- Environments support the API lifecycle.

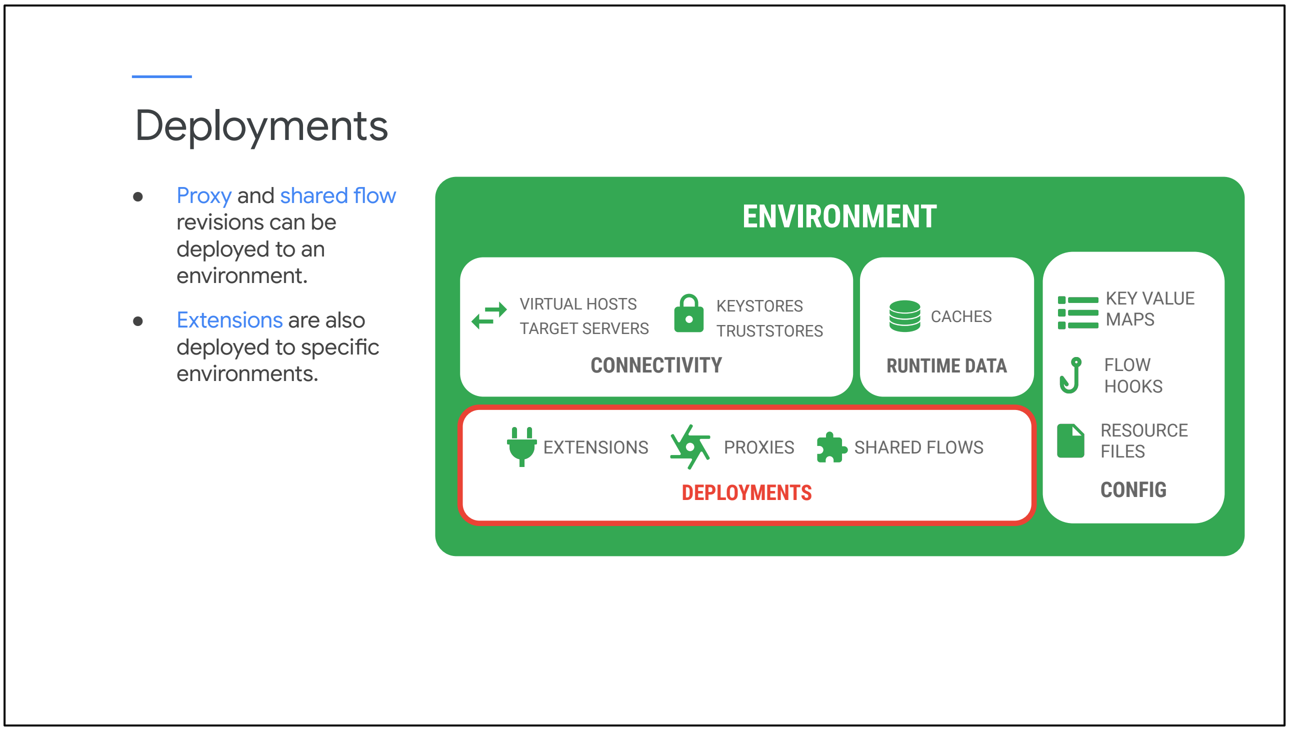


Environments provide a runtime execution context for API proxies.

A proxy only accepts API requests when deployed to an environment.

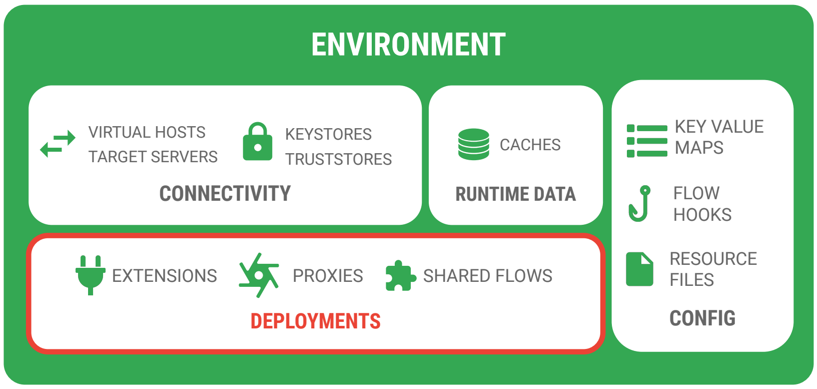
Environments are primarily intended to support the API development lifecycle.

A revision of a proxy can be promoted from a development environment through the testing environment and eventually into production.



Deployments

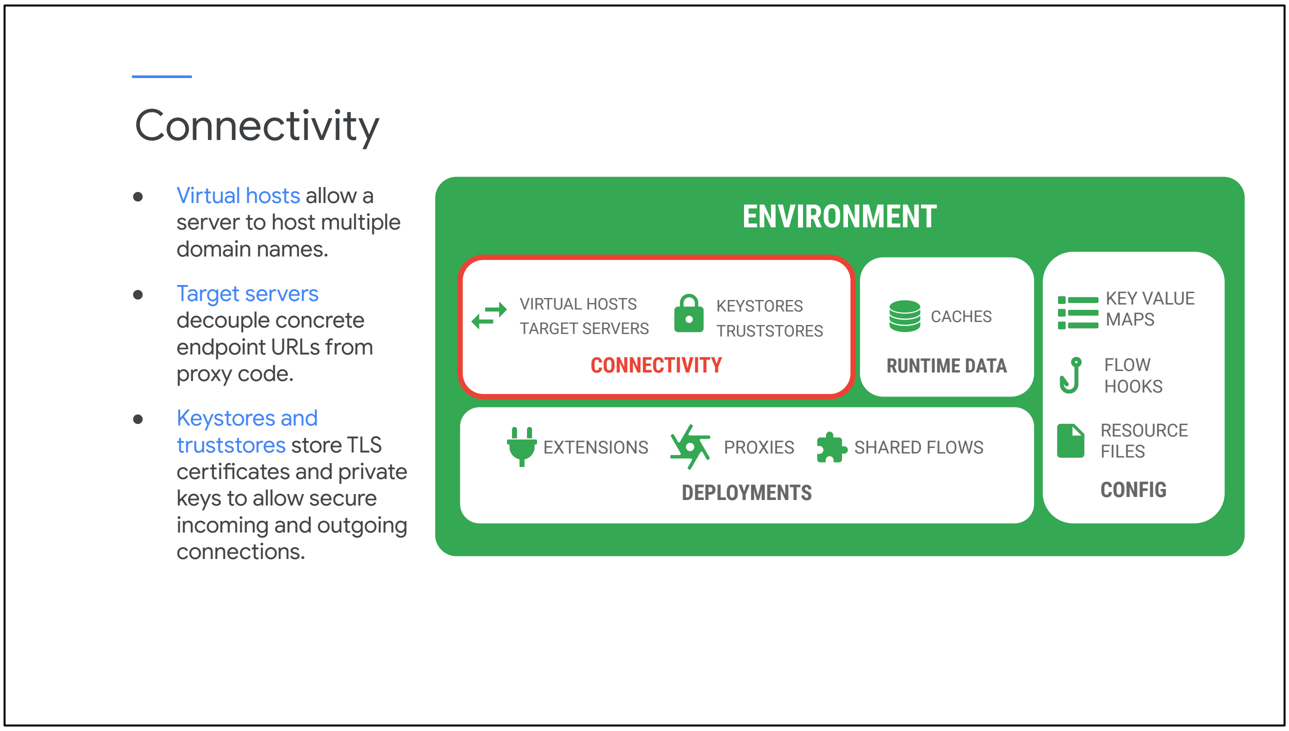
- Proxy and shared flow revisions can be deployed to an environment.
- Extensions are also deployed to specific environments.



A **proxy revision** can be deployed to an environment, where it can start taking traffic.

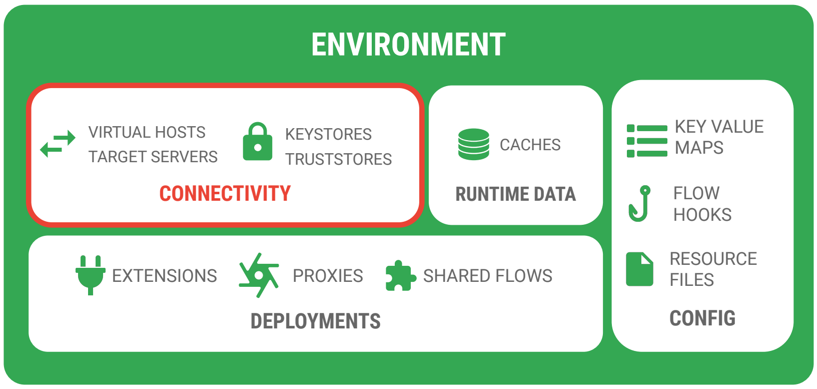
A **shared flow revision** can also be deployed to an environment, making it available for use by proxies in that environment.

Extensions are also deployed to an environment.



Connectivity

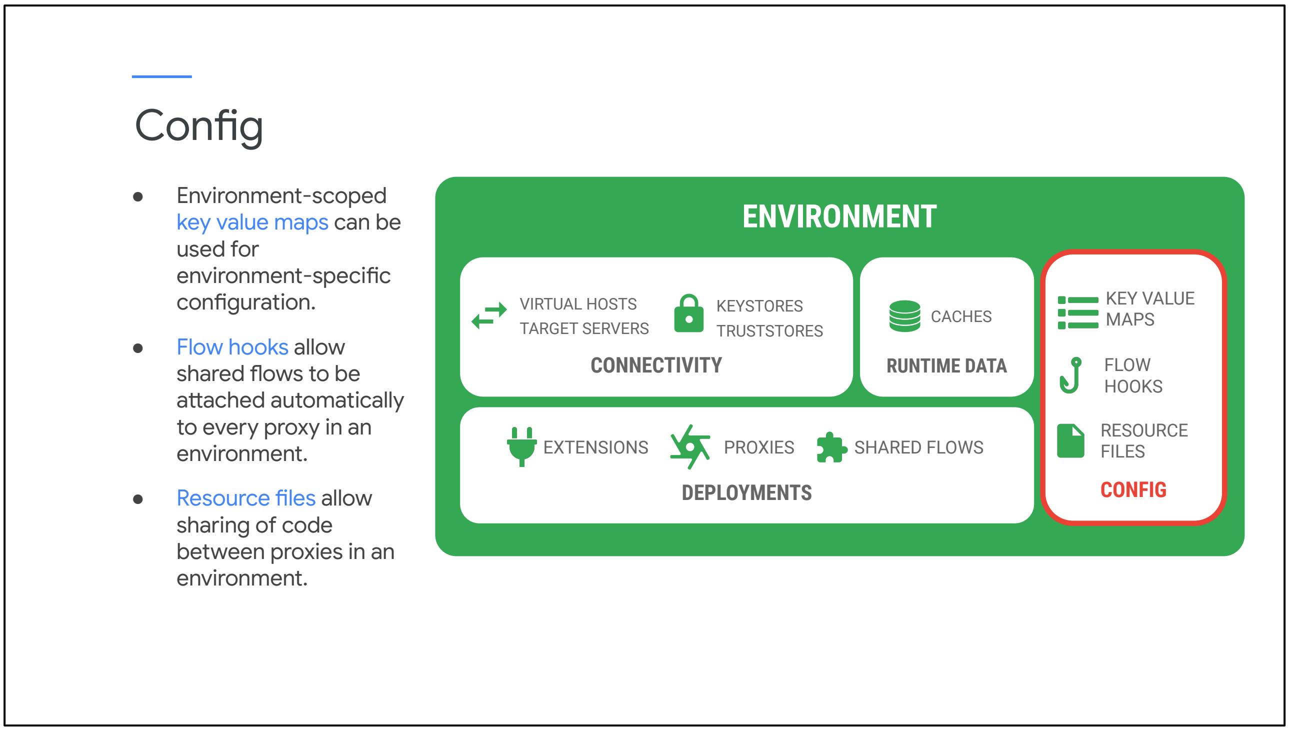
- Virtual hosts allow a server to host multiple domain names.
- Target servers decouple concrete endpoint URLs from proxy code.
- Keystores and truststores store TLS certificates and private keys to allow secure incoming and outgoing connections.



A **virtual host** specifies one or more domain names that should be routed to a specific environment. This allows a server to accept traffic for multiple domain names.

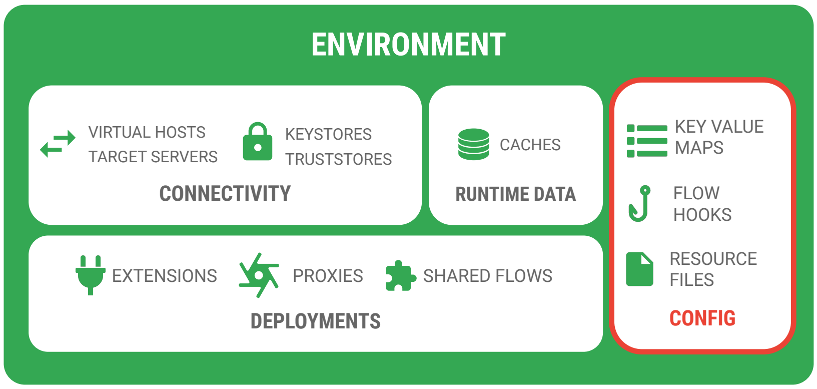
Target servers are used to decouple backend URLs from the API proxy code. This allows the proxy to connect to environment-specific backends without changing proxy code.

Keystores and **truststores** store certificates and private keys to allow point-to-point encryption from apps into Apigee and from Apigee to backend servers.



Config

- Environment-scoped key value maps can be used for environment-specific configuration.
- Flow hooks allow shared flows to be attached automatically to every proxy in an environment.
- Resource files allow sharing of code between proxies in an environment.



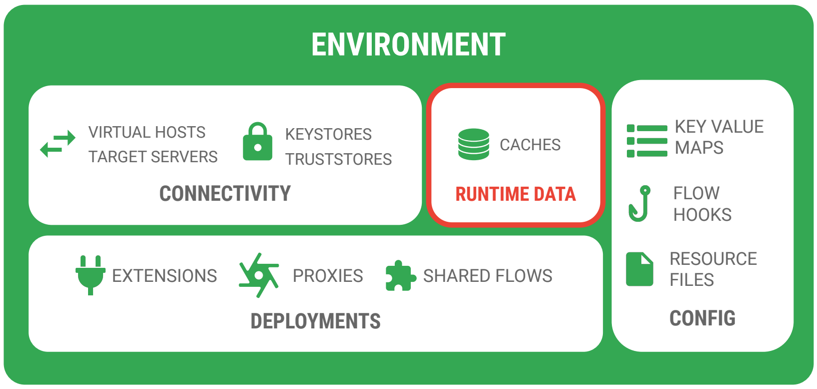
Environment-scoped **key value maps** can be used to store configuration items that change between environments, like backend credentials.

Flow hooks are used to automatically attach shared flows to every proxy in an environment. This allows admins to enforce that security, logging, or other common policies are executed for all proxies.

Resource files allow proxies within an environment to share code libraries.

Runtime Data

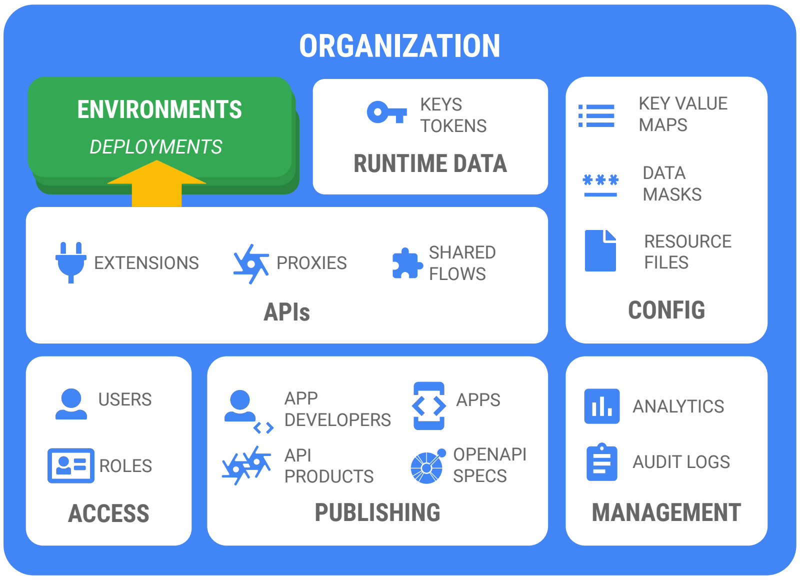
- Use caches to eliminate unnecessary traffic to backends or third-party services.



Caches may be used to eliminate unnecessary traffic to backends or third-party services, resulting in improved latency, reduced costs, and better scalability.

Cached data should not be shared between proxies in separate environments, so caches are only scoped at the environment level.

Organization



I have taken you through a very quick tour of organizations and environments. Don't worry if organizations and environments, as well as all the entities they contain, don't make sense yet. You will learn more about these entities, and how they all fit together, throughout this series of courses.

Review: Apigee Overview

Mike Dunker
Course Developer, Google Cloud

You have learned about Apigee, Google Cloud's API management platform, and the business challenges it can help you overcome.

You learned about the API lifecycle, and how Apigee helps with the development of your APIs.

You also learned about Apigee organizations and environments, and the entities they contain.