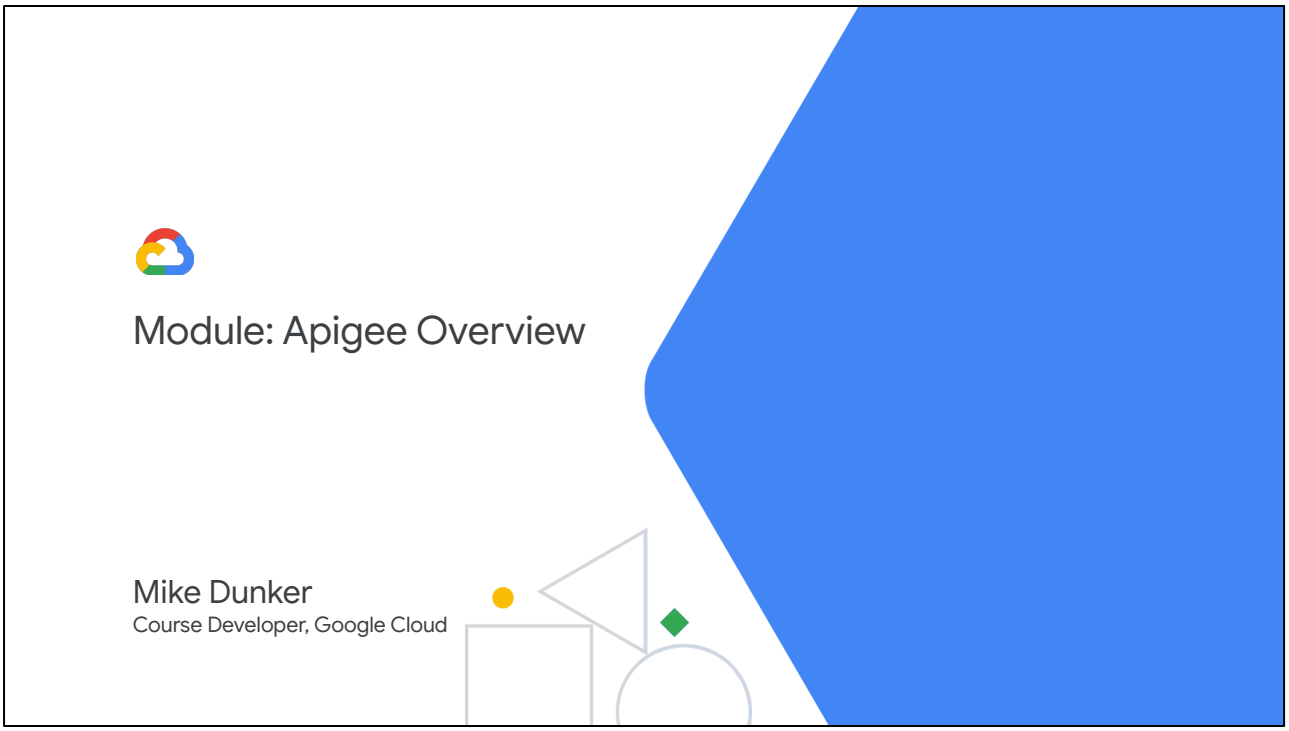
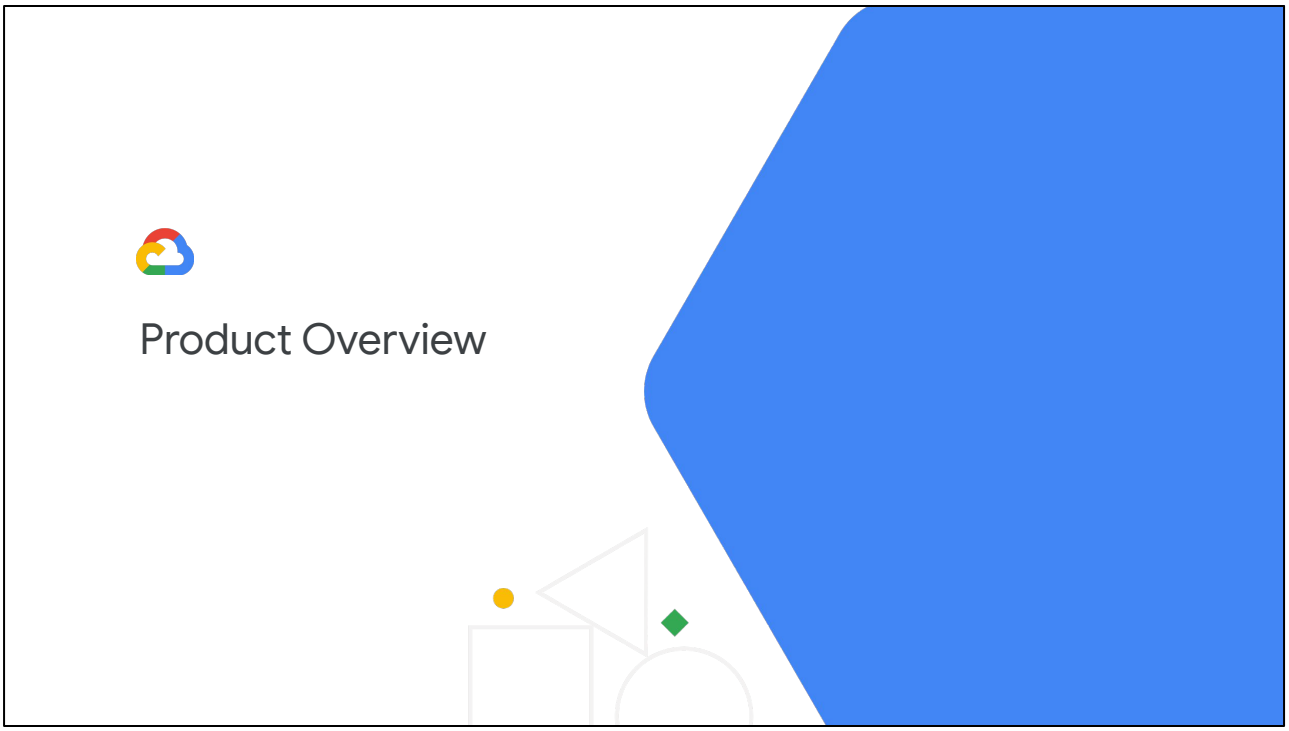


In this module, we'll introduce you to Apigee, Google Cloud's API Management platform.

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



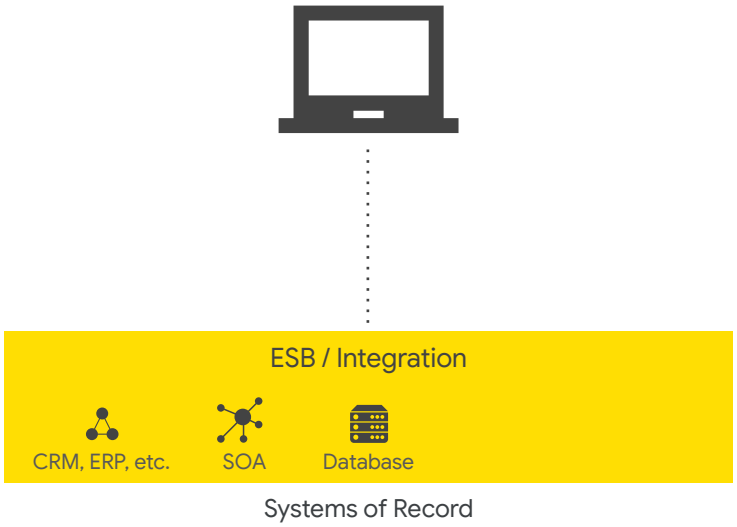
In this lecture you will learn about Apigee, Google Cloud's API Management Platform.

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.

Success, back then



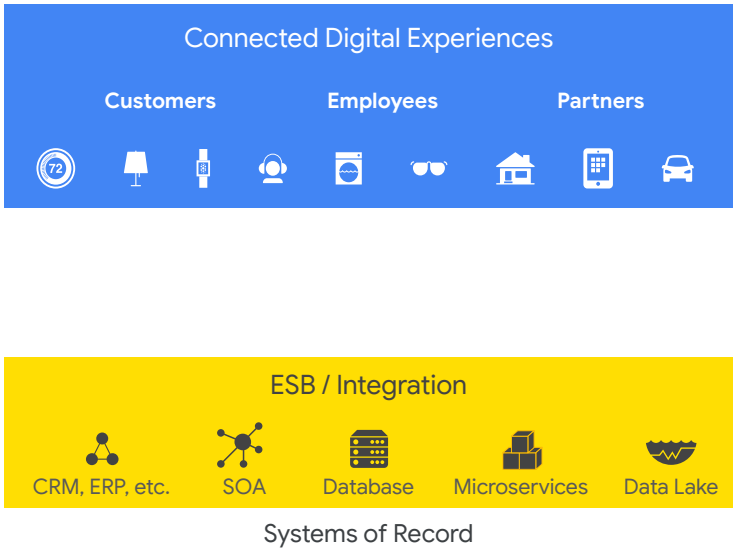
Not too long ago, having a digital presence just meant you had a website.

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



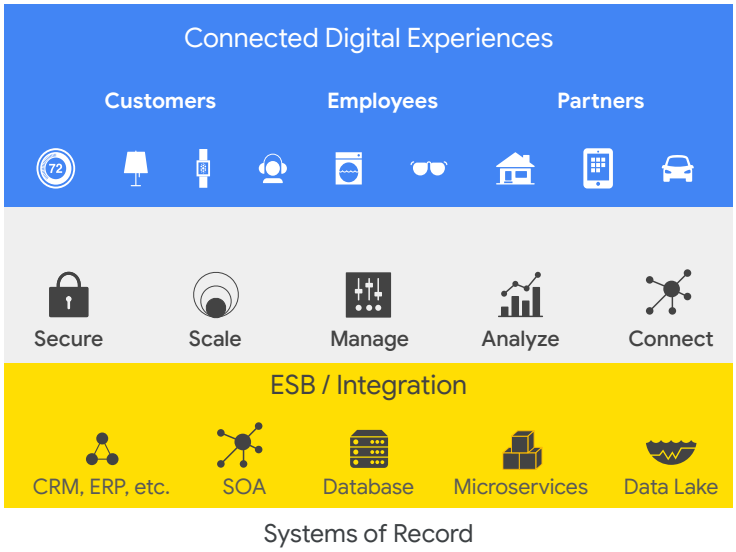
What about today?

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.

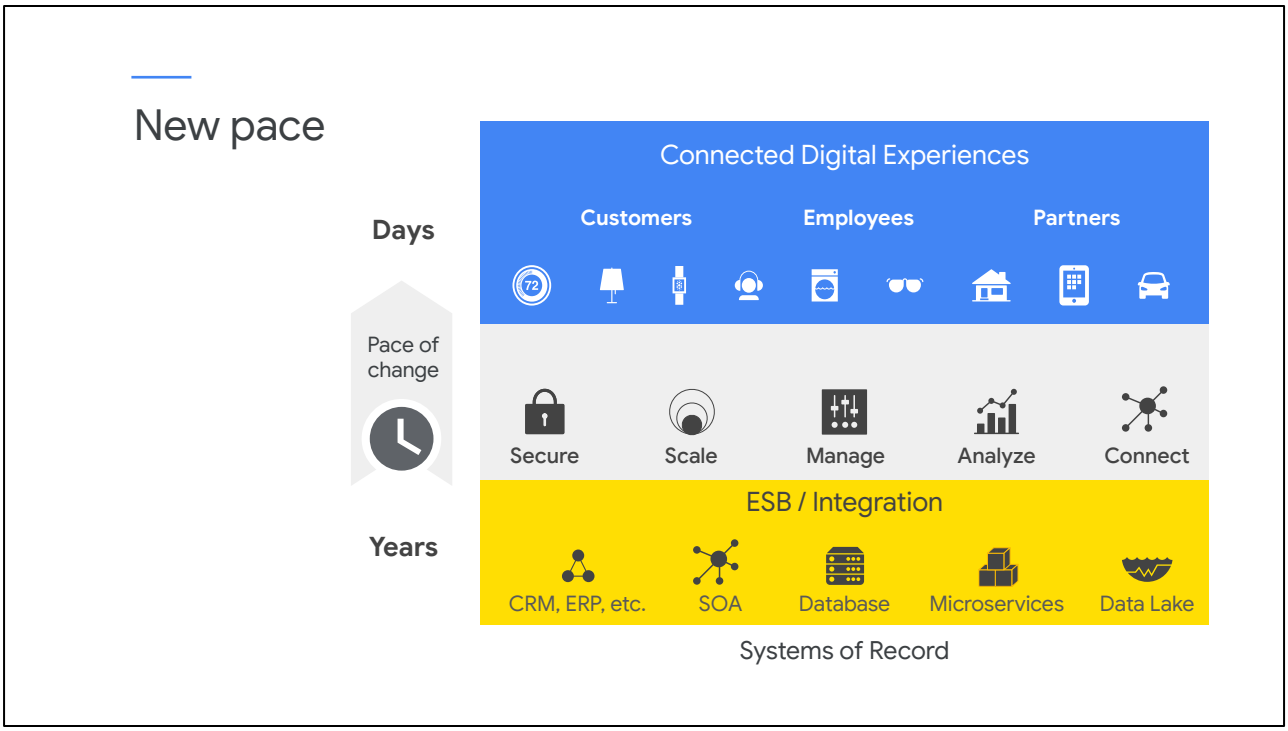
New challenges



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.

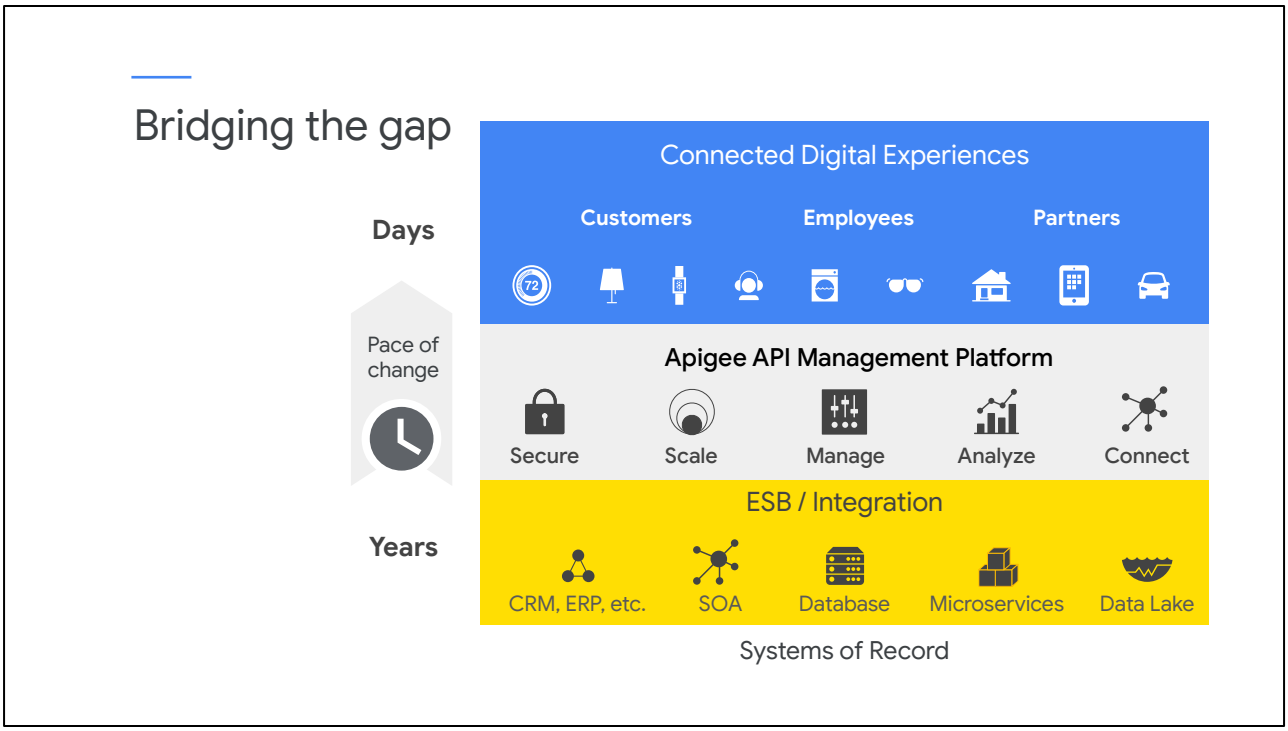
All of these challenges add to the complexity and diversity of requirements that backend systems need to handle, ...



...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.



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.

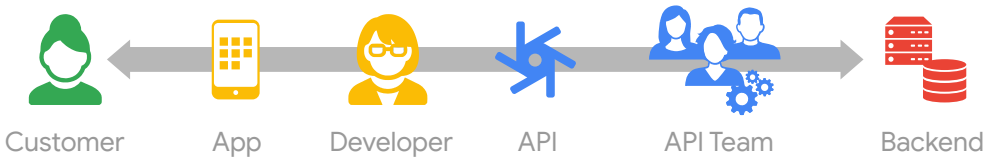
APIs that are implemented on Apigee leverage a rich set of capabilities and features, including security, caching, transformation, and mediation.

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.

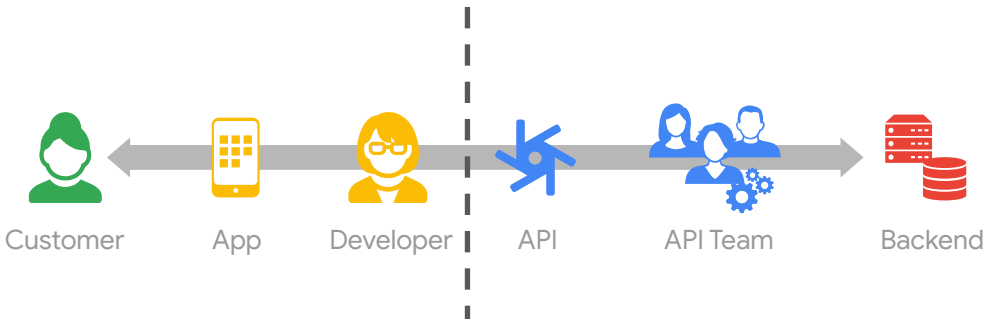
Applications range from web and mobile apps to large enterprise systems and connected devices. Some of these applications are built by developers at your company. Other applications may represent systems used by partners or customer-facing products that they’ve built.

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.

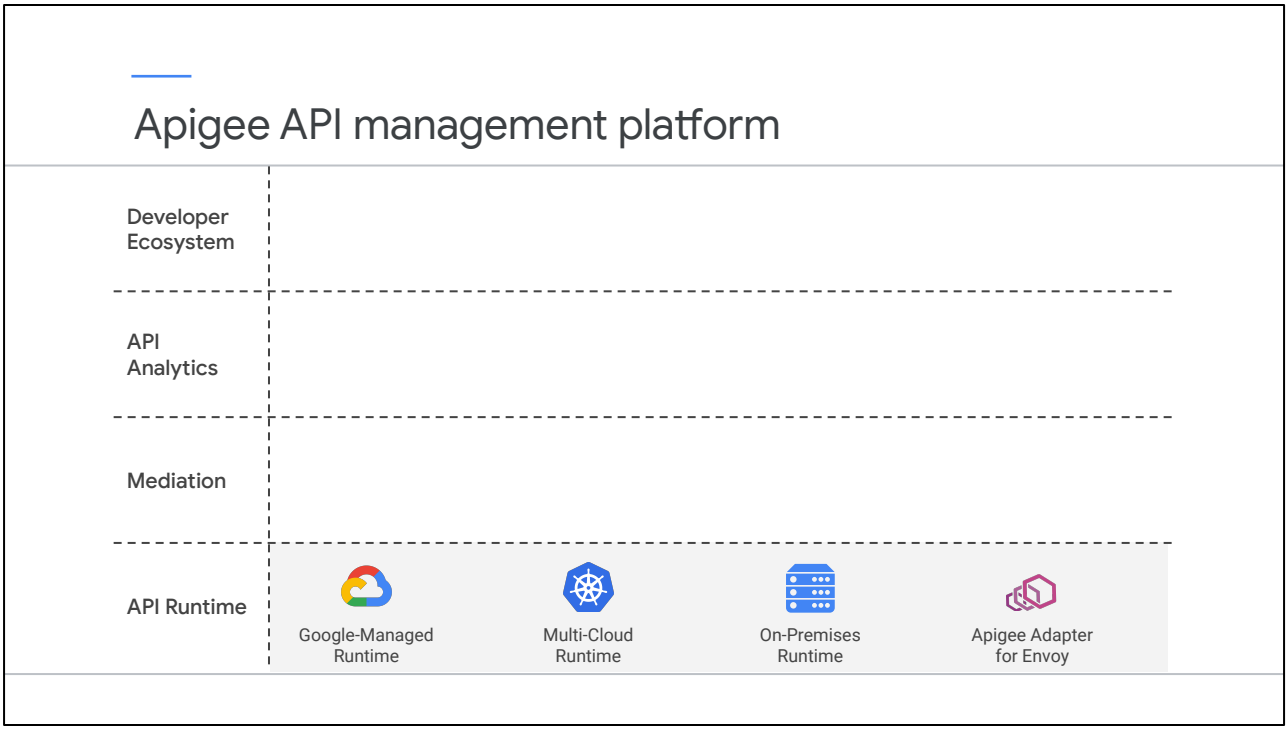
Digital value chain



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.



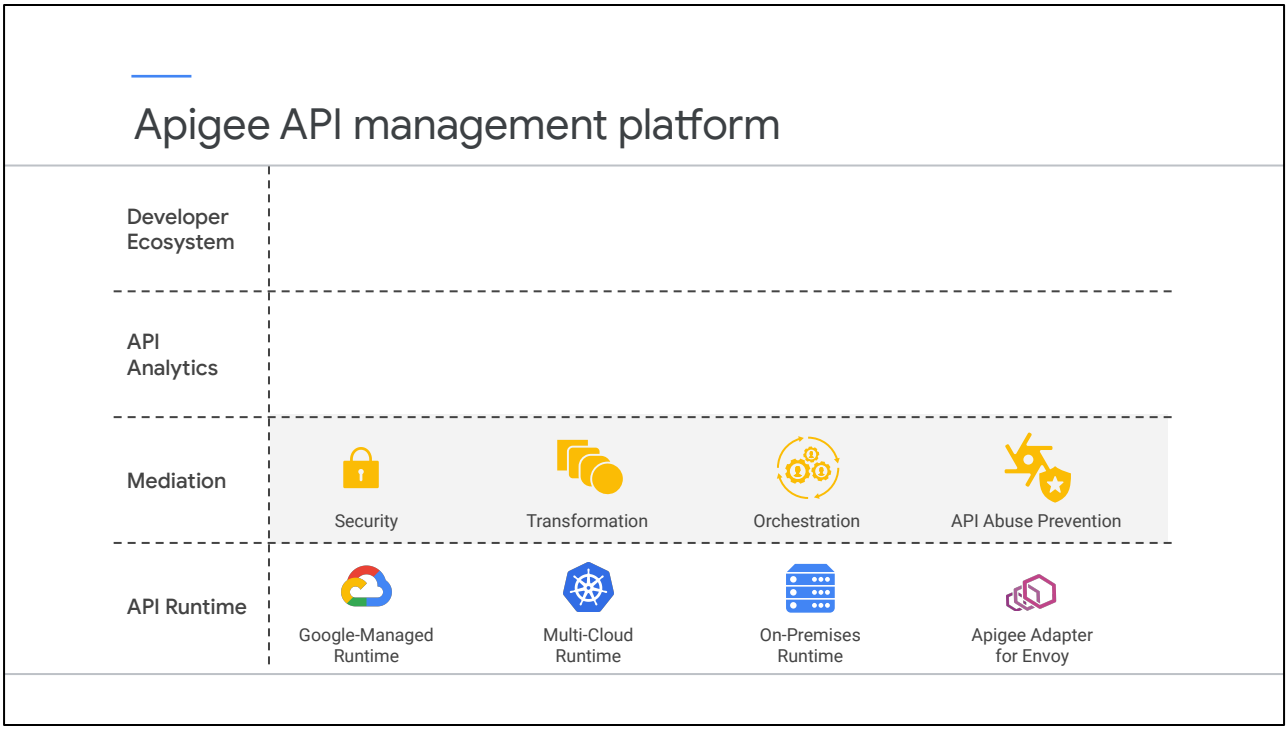
The Apigee platform is composed of four core capabilities.

Starting at the foundation, Apigee offers multiple deployment models for the API runtime. The API runtime is responsible for handling runtime API traffic.

Apigee's enterprise-level platform can be run as a software-as-a-service offering running in Google Cloud and fully managed by Google.

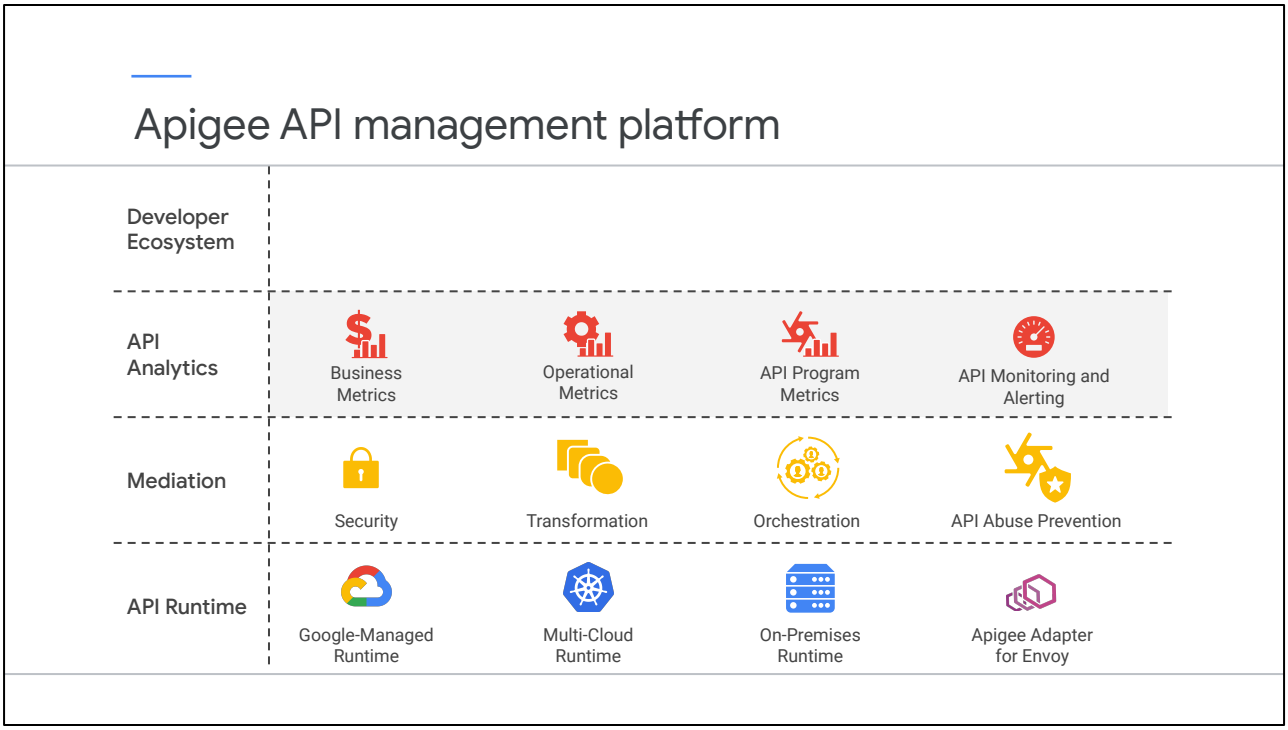
API runtimes can also be deployed in customer private clouds or in an on-premises data center, with the runtime deployment managed by the customer, and the management services managed by Google and running in Google Cloud.

The Apigee adapter for Envoy is a lightweight API gateway that provides limited API management functionality that can be deployed close to backend services.



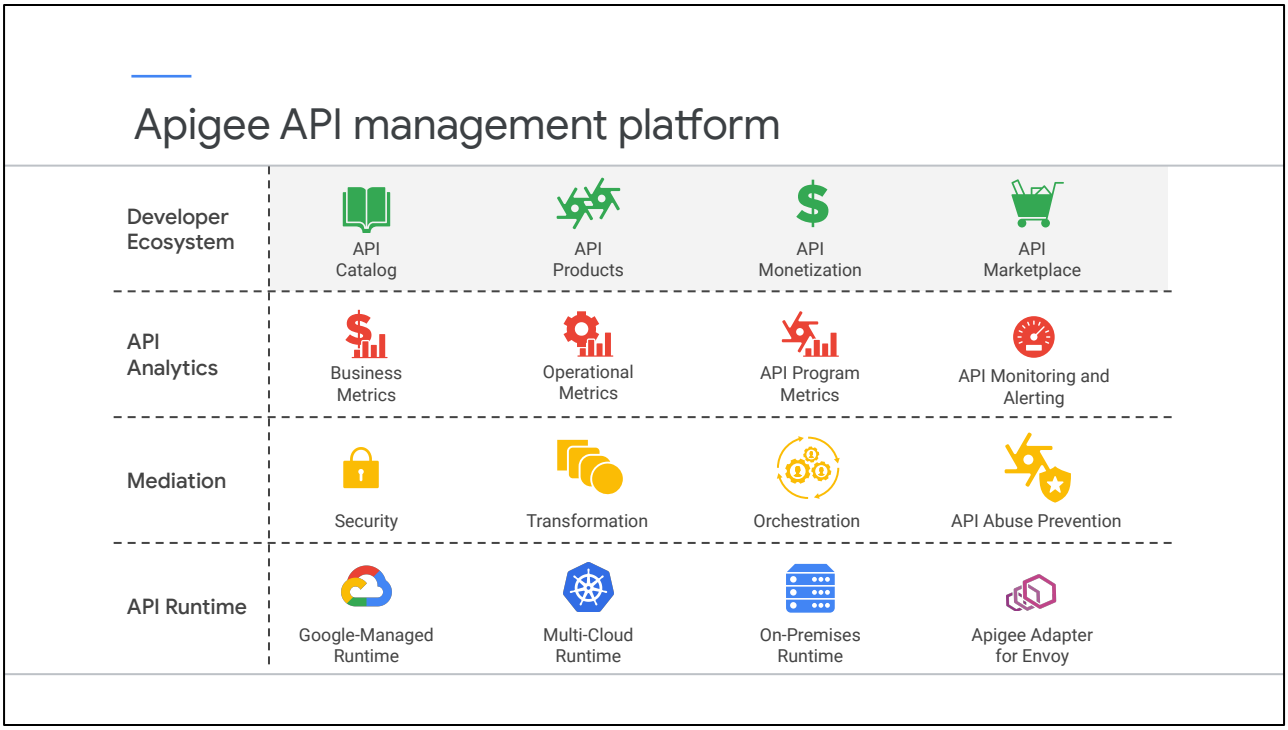
Mediation provides the ability to parse and manipulate the requests and responses of API calls passing through Apigee.

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



Every API call that passes through Apigee generates analytics data.

Analytics data generated by the system can be used by operations teams and business users to make data-driven decisions about APIs and their API program.

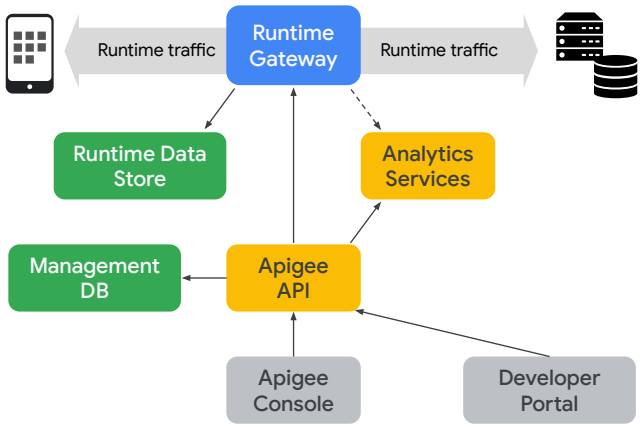


The developer ecosystem is an important factor in the success of your APIs.

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 logical components by looking at their capabilities and relationships.

The **Runtime 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 reveal a wealth of information about APIs, apps, and backend system calls, and are used for analytics reports and visualization.

The **Apigee API** is used to manage the API platform. The API is used to deploy and undeploy API proxy revisions, monitor APIs, configure environments, manage users, and more.

The Apigee Console, the developer portal, and other management processes use the Apigee API.

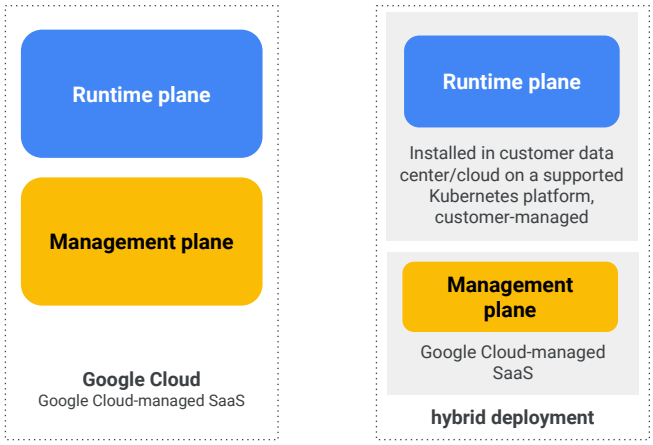
The Apigee API is also fully documented and available to customers. Developers and operations teams make use of this API for automation, such as continuous integration/continuous deployment, or CI/CD.

The **Management database** stores configuration changes. The runtime will poll for changes and update itself when changes are detected.

The **Apigee Console** 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 Apigee Console. The console can be used to view and control all aspects of your APIs, including controlling the API lifecycle and building and viewing analytics reports.

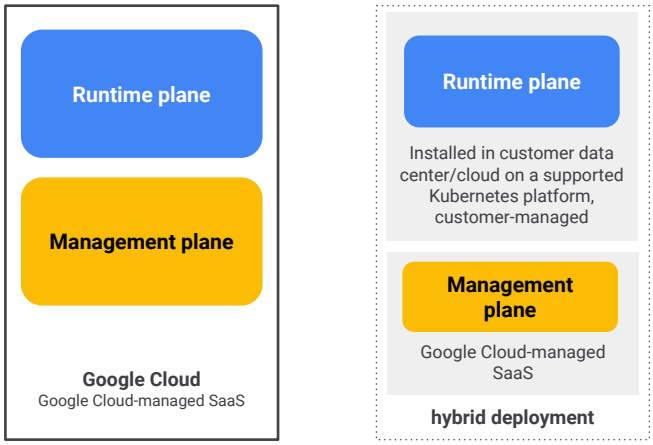
The **developer portal** is a web interface dedicated to addressing the needs of application developers. The API team publishes documentation about your organization's APIs to the developer portal, where application developers can register their applications and sign up to use your API products.

Flexible deployment



Apigee offers flexible platform deployment options.

Flexible deployment



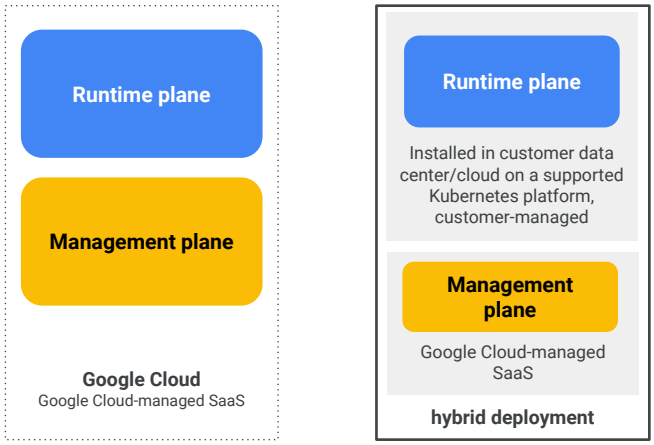
Apigee's Google-managed software-as-a-service deployment simplifies customer adoption and dramatically accelerates time to market for new APIs.

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.

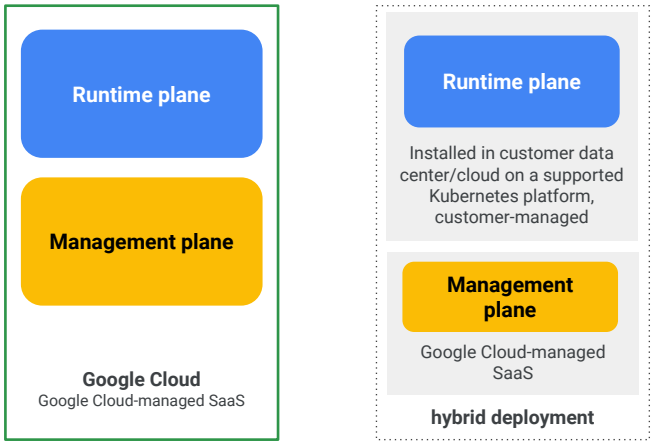
Flexible deployment



Customers who want or need to provide access to their APIs in multiple clouds or on-premises can choose the hybrid deployment model.

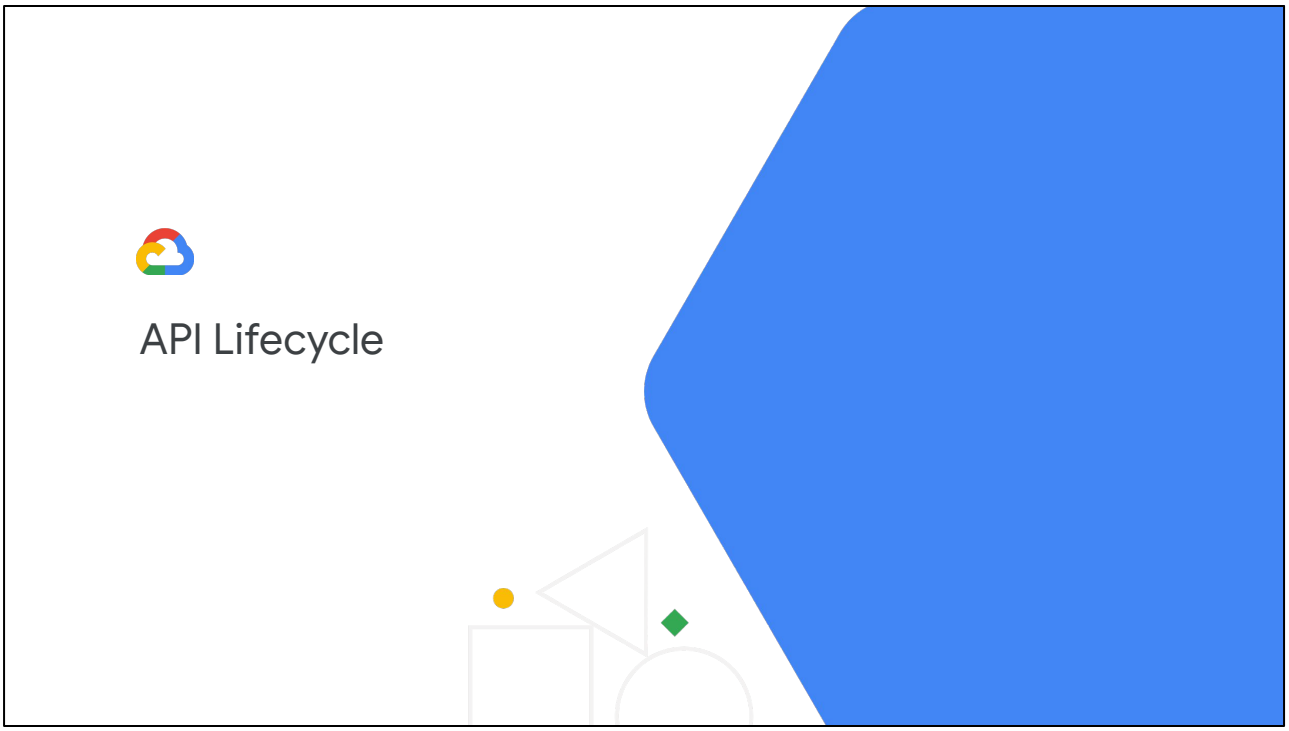
This model allows the customer to manage and deploy containerized versions of the API runtime on Kubernetes, while delegating the management plane operations to Google.

Flexible deployment



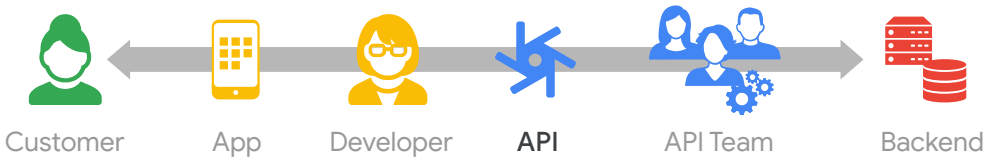
During this course, you will use the Google Cloud–hosted deployment for your labs.

Operational management differs between the Google Cloud–managed and the hybrid deployments, but the experience you will have as an API developer is virtually identical.



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

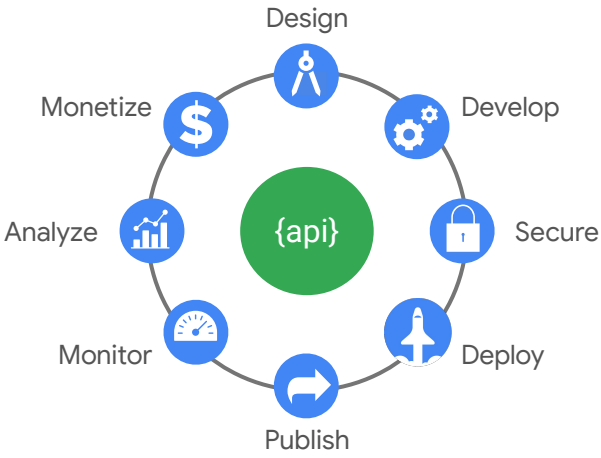
Digital value chain



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

OPEN API INITIATIVE

- /categories GET
- /categories/{categoryId} GET
- /orders POST
- /orders/...
  - {orderId} GET
  - {orderId} DELETE
- /products GET
- /products/...
  - {productId} GET
  - {productId} PATCH
- /stores GET
- /stores/{storeId} GET

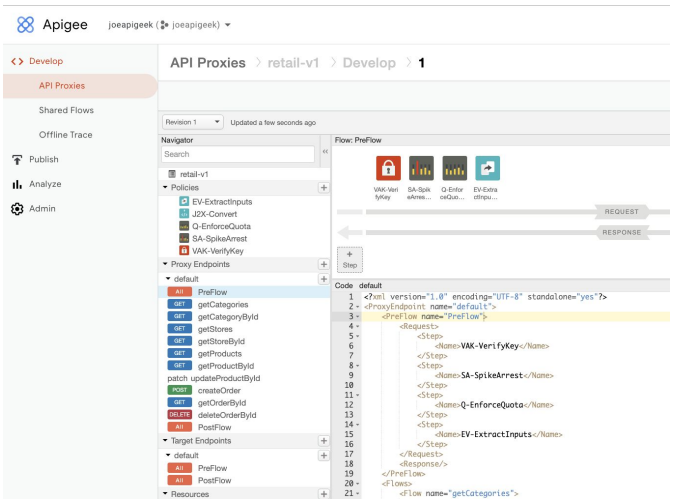
One tool often used when designing a REST API is an OpenAPI specification.

An OpenAPI spec can be used to define the interface and capabilities of your REST APIs, without focusing on the implementation.

The spec may then be used in a developer portal to allow 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



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

Apigee

Cloud Armor WAF

Identity Platform

Apigee and Google provide multi-layer security for your APIs.

Apigee proxies can utilize many built-in policies and features which allow you to create secure APIs, even if your backend APIs are not fully secured.

Policies are available to add OAuth, SAML, JSON Web Token, and HMAC authentication and authorization to your APIs.

Other policies provide threat protection against content-based attacks, detecting malicious request payloads and rejecting the requests before they are sent to your backend services.

Apigee allows sensitive data to be masked, so that operations teams do not see user data or passwords when tracing live API traffic.

Because Apigee is hosted in Google Cloud, Google Cloud's security features can be leveraged to further protect your APIs.

For example, Cloud Armor is a Google-grade web application firewall that protects web and API traffic against distributed denial-of-service attacks, allows rejection of traffic based on geographic origin or IP address, and provides firewall rules to protect against many common types of attacks.

Cloud Armor benefits from Google's extensive experience protecting key internet

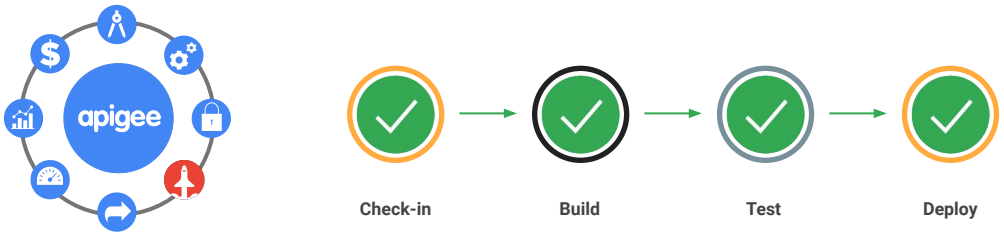
services like Google Search, GMail, and YouTube.

Identity Platform is a customer identity and access management platform that helps organizations easily add identity and access management functionality to their APIs and applications.

It protects users by supporting multi-factor authentication, and provides support for many authentication methods, including SAML, OpenID Connect, email and password, or custom implementations.

These are just a couple of examples of the Google Cloud security features you can leverage for your APIs.

Deploy

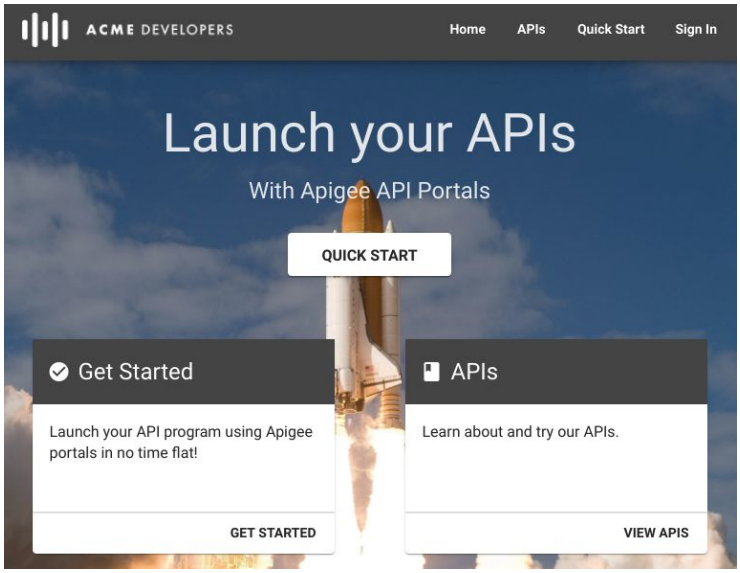


When your API has been built and secured, you will need to deploy your API proxy into production.

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 management APIs 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

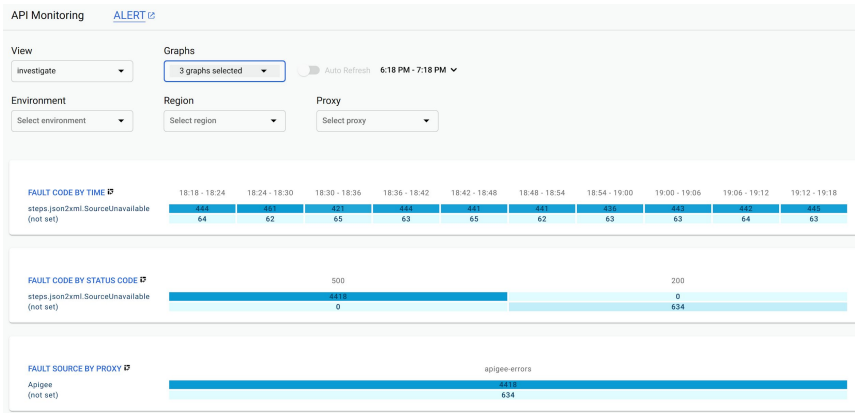


Apigee's developer portal helps your app developers discover your APIs and register apps to use them.

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



After APIs are built and launched on Apigee, they need to be monitored to ensure that they are available and performing as expected.

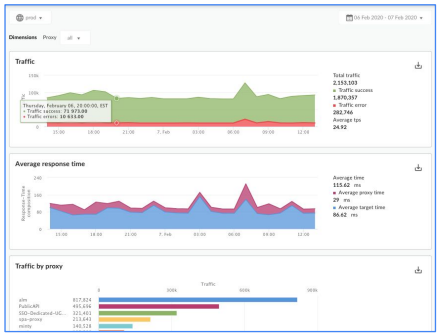
Apigee's API Monitoring provides near real-time insights into API traffic and performance by monitoring API performance and usage, automatically capturing API and backend latencies, error rates, and call volume, among other types of operational metrics.

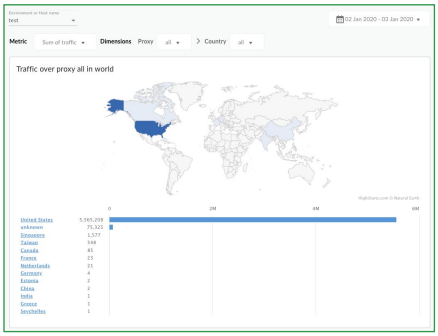
Alerts can be used to keep you informed of unusual events or patterns, such as spikes in traffic or latencies.

Alerted events can be analyzed in the console, and you can use notification channels in Google Cloud's Operations Suite to make sure the right people are notified quickly.

API monitoring helps you diagnose issues before your app developers and users of their apps notice them.

Analyze



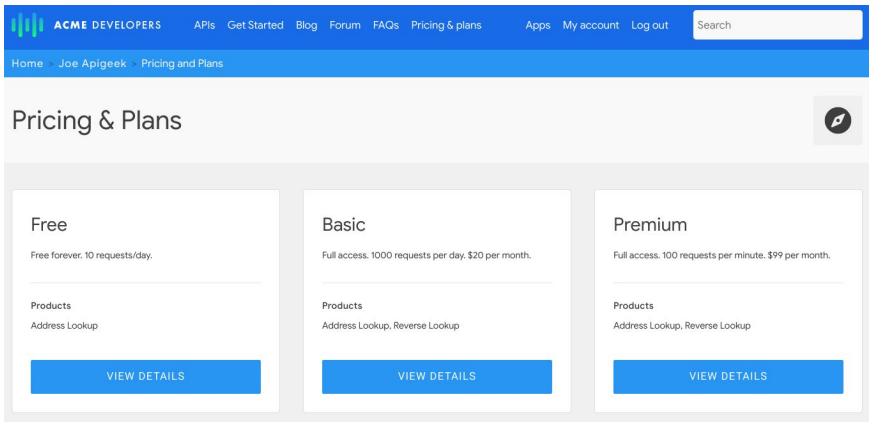


In addition to API and performance metrics, Apigee captures business metrics, tracking the apps and app developers using your APIs and the device types and geolocation of the users of those apps. Other metrics specific to your business can be captured by collecting custom data within your API proxies.

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



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.

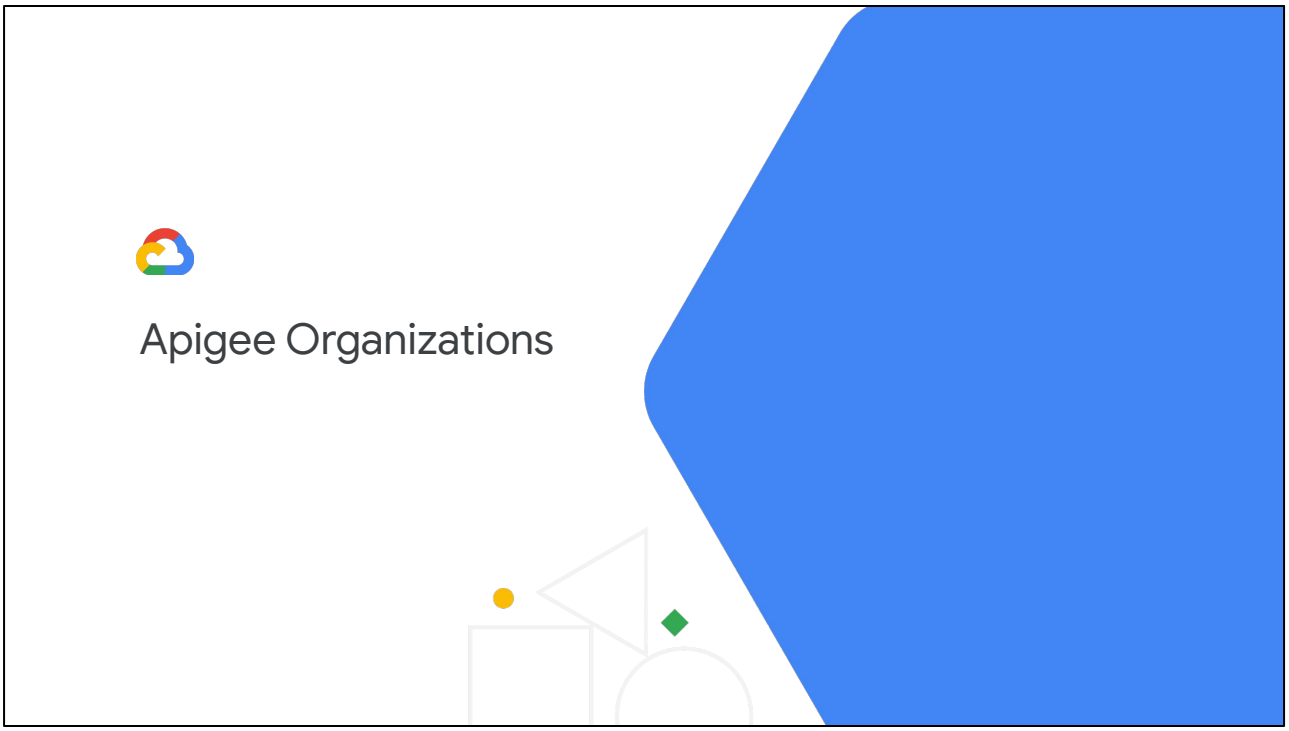
Apigee monetization allows you to charge for API usage or share revenue with app developers that drive your business.

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.



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.

An Apigee **organization** is the top-level entity for Apigee. When you use the Apigee Console, you are working within the context of an organization.

This Apigee organization is not the same as the Google Cloud organization.

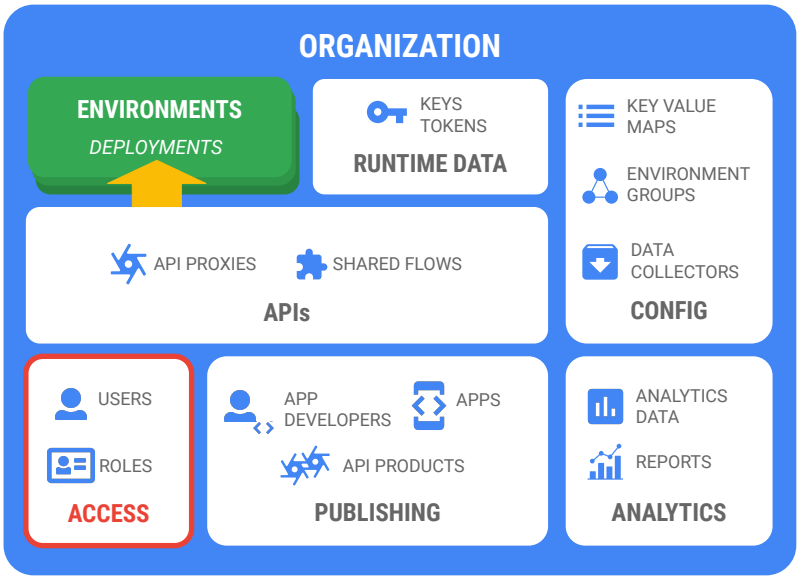
An Apigee organization is associated with a single Google Cloud project.

An organization contains many types of entities. Some entities live inside environments, which are used as 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.
- An Apigee Organization Admin has superuser access within the organization.
- Other built-in roles are designed for operations, business, or API development team members.



Users can be granted access to one or more organizations.

Users are associated with one or more **roles** within an organization.

A role specifies a set of permissions that is granted to a user.

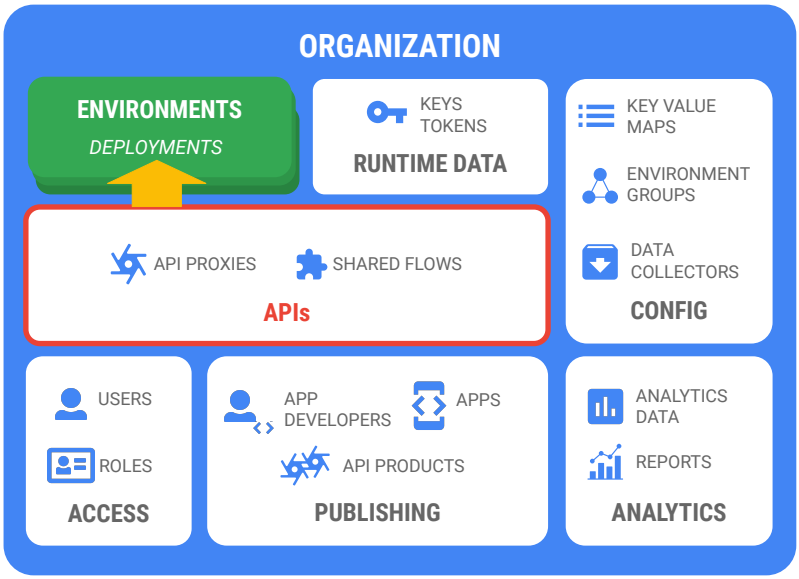
Users and roles are managed using Google Cloud's **Identity and Access Management**, or IAM.

An **Apigee Organization Admin** has superuser access within the organization.

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

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.



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.

Environments

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

An **environment** is a runtime execution context for your APIs.

API proxies and shared flows 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

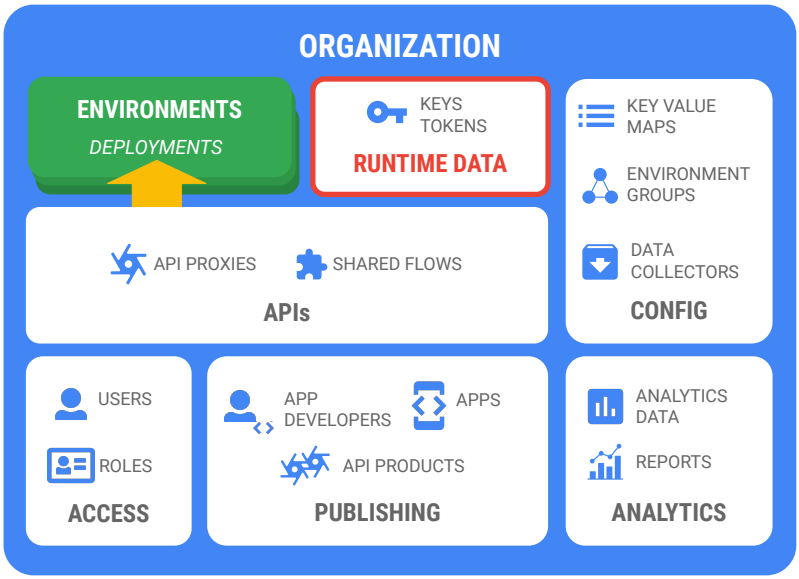
- APIs are productized by exposing them in your developer portal as API products.
- App developers register apps to use one or more API products.

Before publishing your APIs to the developer portal, you group and productize your APIs by creating **API products**. 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.



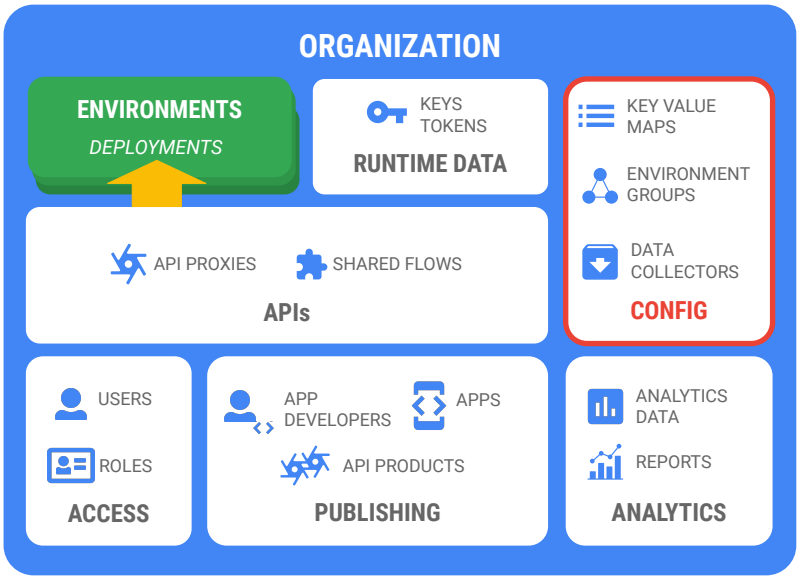
Apps present **API keys** and **OAuth tokens** to access APIs. When an API key or OAuth token is verified in an API proxy, the app making the request is identified, as is the associated API product. This allows proxies to control functionality based on API product or app.

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.
- Environment groups map hostnames to Apigee environments.
- Data collectors are used when collecting custom data for use in reports.



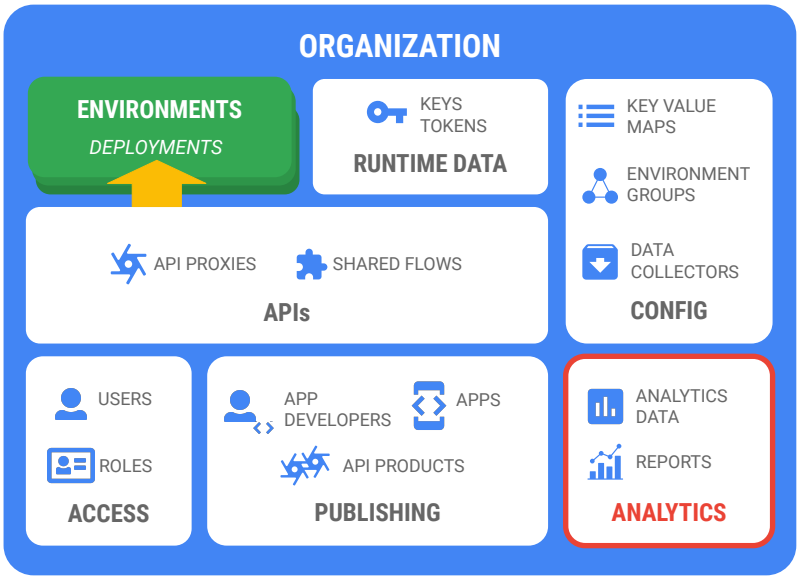
Organization-scoped **key value maps**, or KVMs, can store organization-wide configuration. KVMs are encrypted, so they are appropriate for storing passwords or other sensitive information.

Environment groups are used to create a mapping from hostnames to one or more Apigee environments. For example, the hostname api.example.com could be mapped to the production environment, and test.example.com could be mapped to the test environment. When an API request is received with the hostname api.example.com, the request would be routed to a proxy in the production environment.

Data collectors are defined locations used to store data collected during the processing of an API call. A DataCapture policy may be used to store a value in a data collector. The data collected may be used in custom reports.

Analytics

- Analytics data is captured to provide visibility for all of your API traffic.
- Custom reports can be created for your organization.



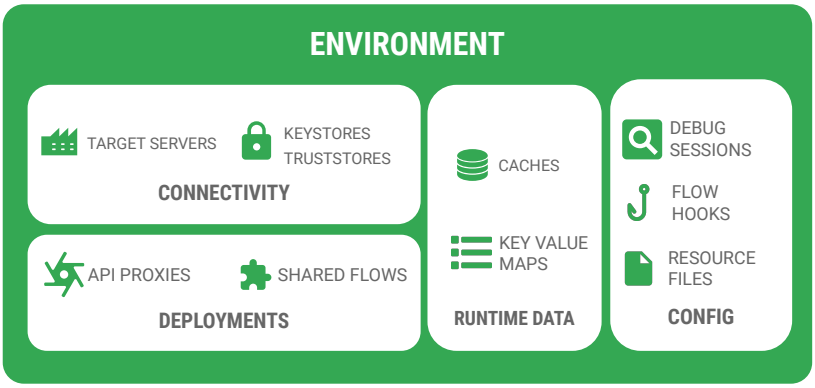
Analytics data provides visibility for all API traffic, from an application through Apigee to your backend services and back.

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.

Custom reports can also be created to allow visualizations of custom data captured in data collectors, or to provide new ways of looking at your data.

Environment

- An environment is a runtime execution content for API proxies.
- Environments may represent different stages of the API lifecycle, like development, testing, and production.



Environments provide a runtime execution context for API proxies.

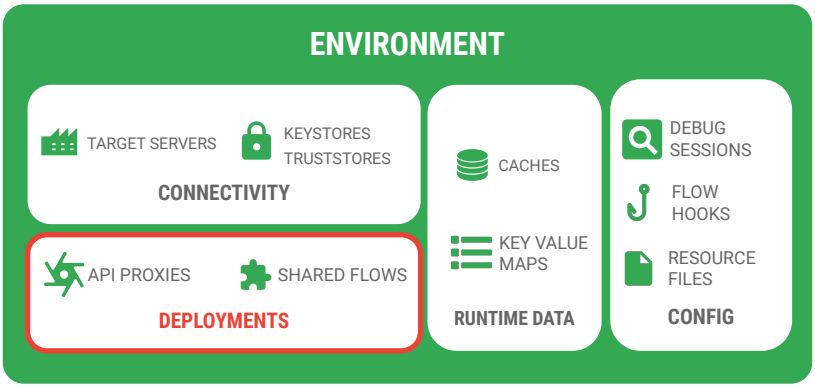
A proxy only accepts API requests when deployed to an environment.

Environments may be used to represent different stages of the API development lifecycle.

For example, a revision of a proxy could be promoted from a development environment to the testing environment and eventually into production.

Deployments

- Proxy and shared flow revisions can be deployed to an environment.
- Deployed revisions are immutable.



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

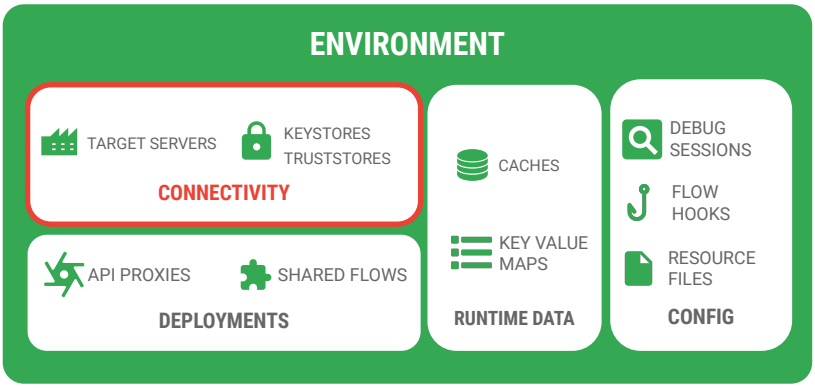
A **shared flow revision** can also be deployed to an environment, making it available for use by proxies in that environment.

A deployed revision of an API proxy or shared flow is immutable.

Further edits to the API proxy or shared flow must be made in a new revision.

Connectivity

- 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.

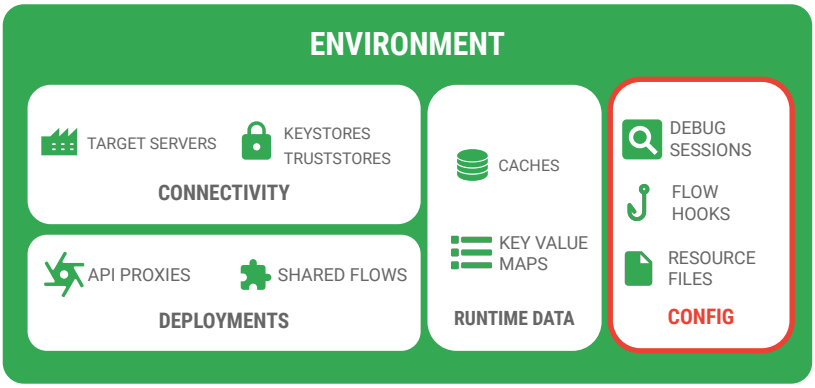


Target servers are used to decouple backend URLs from the API proxy code. This allows the proxy to connect to environment-specific backends without changing proxy code.

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

Config

- Debug sessions capture API traffic received when tracing an API proxy.
- 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.



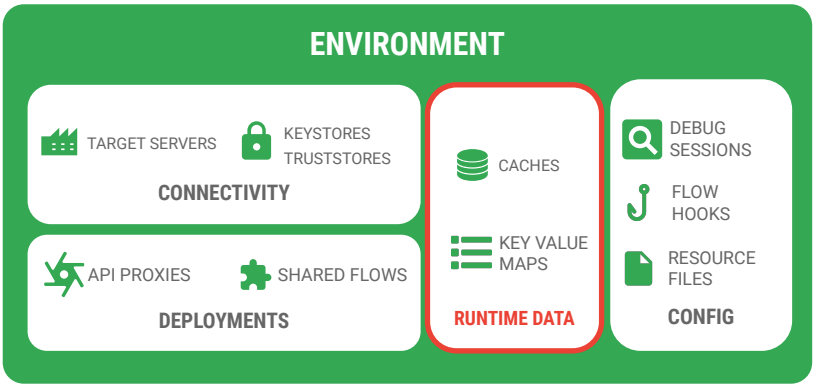
Debug sessions capture requests and responses that are received while tracing an API proxy deployment.

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.
- Environment-scoped key value maps can be used for environment-specific configuration.



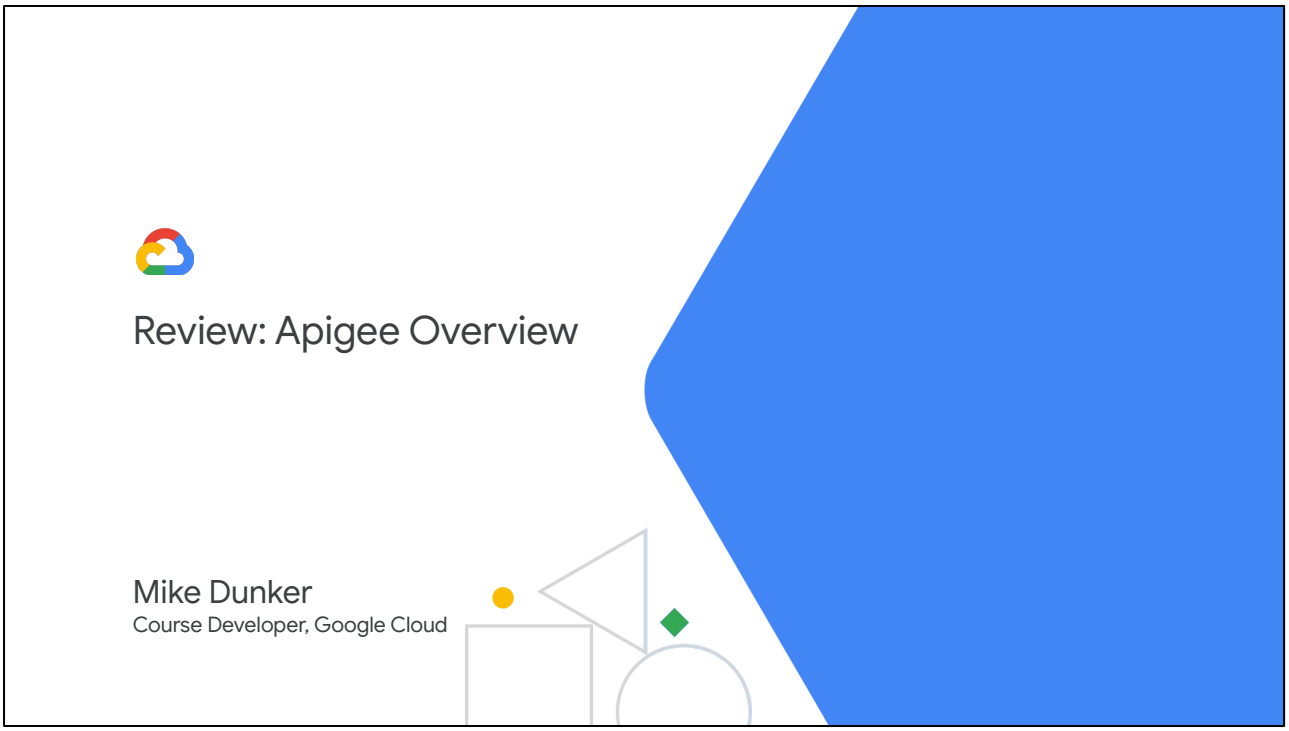
Caches may be used to eliminate unnecessary traffic to backends or third-party services, resulting in improved latency, reduced costs, and better scalability.

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

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

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.



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.