Decide a resource hierarchy for your Google Cloud landing zone

Release Notes

A resource hierarchy helps to organize your resources in Google Cloud. This document describes how to choose your resource hierarchy as part of your landing zone (/architecture/landing-zones) design. It's intended for cloud system administrators, architects, and technical practitioners who are involved in designing the resource hierarchy. This document is part of a series on landing zones. It includes sample hierarchies that demonstrate common ways that businesses can structure resources in Google Cloud.

Design factors for resource hierarchy

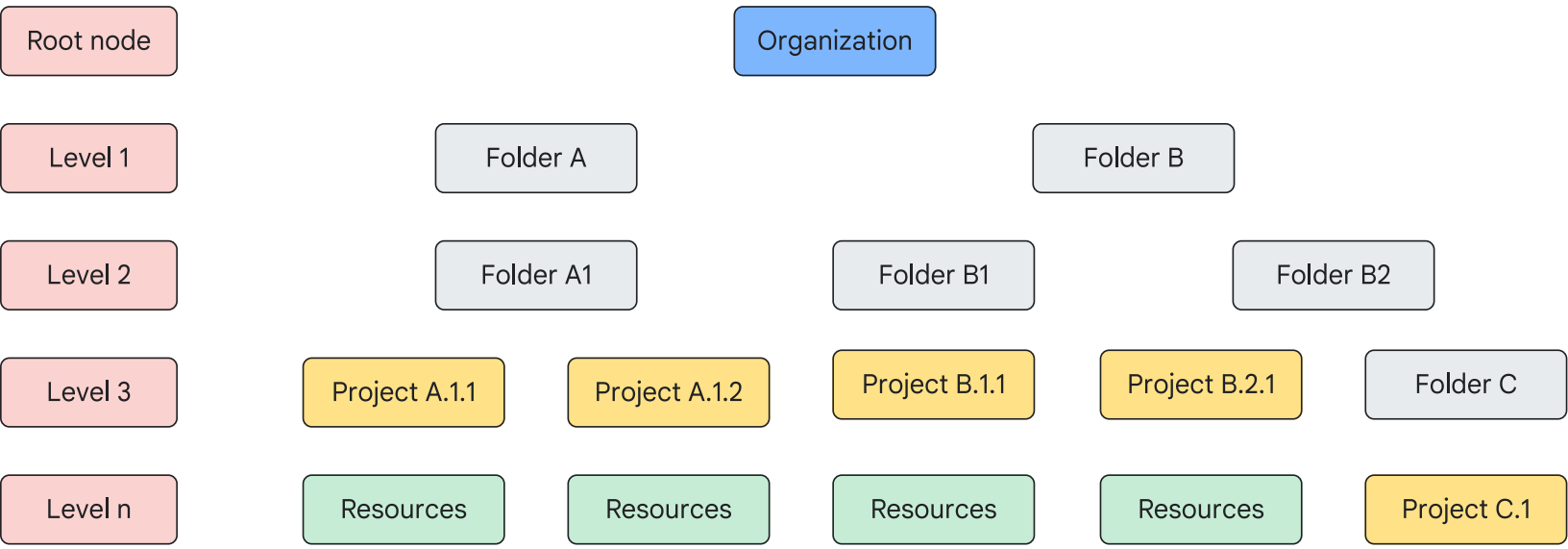
When you define your resource hierarchy (/resource-manager/docs/cloud-platform-resource-hierarchy) in Google Cloud, you must consider how your organization works today and the ideal end state of your cloud transformation. The best way to manage resources is based on your organization's intended way of working in the cloud. As every organization is different, there is no single, best approach for resource hierarchy.

However, we recommend that you avoid mapping your corporate organization structure to the resource hierarchy. Instead, focus on your business needs and operations in Google Cloud.

Google Cloud resource hierarchy

The resource hierarchy (/resource-manager/docs/cloud-platform-resource-hierarchy) in Google Cloud starts at the root node, which is called the *organization*. We recommend that businesses have only one root node, except for in specific situations (#single-org-node). You define lower levels of the hierarchy using *folders* (/resource-manager/docs/cloud-platform-resource-hierarchy#folders) and *projects* (/resource-manager/docs/cloud-platform-resource-hierarchy#projects), and you create folders within folders to build your hierarchy. You can create the projects that host your workloads at any level of the hierarchy.

The following diagram shows a root node called *Organization*, and folders at levels one, two, and three. Projects and subfolders are created under the folders in level two.



Resource hierarchy factors

When you decide your resource hierarchy, consider the following factors:

- Who is responsible for cloud resources? Is it your departments, subsidiaries, technical teams, or legal entities?
- What are your compliance needs?
- Do you have upcoming business events, such as mergers, acquisitions, and spin-offs?

Understand resource interactions throughout the hierarchy

Organization policies (/resource-manager/docs/organization-policy/overview) are inherited by descendants in the resource hierarchy, but can be superseded by policies defined at a lower level. For more information, see understanding hierarchy evaluation (/resource-manager/docs/organization-policy/understanding-hierarchy). You use organization policy constraints to set guidelines around the whole organization or significant parts of it and still allow for exceptions.

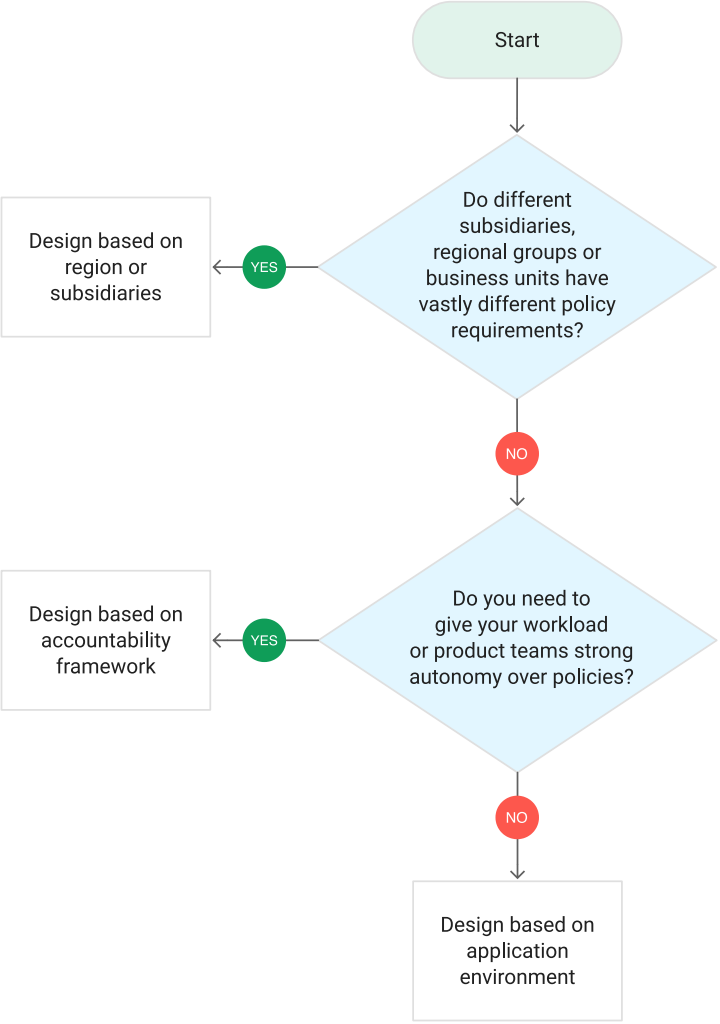
Allow policies, formerly known as IAM policies (/iam/docs/overview) are inherited by descendants, and allow policies at lower levels are additive. However, allow policies can be superseded by deny policies (/iam/docs/deny-overview), which let you restrict permissions at the project, folder, and organization level. Deny policies are applied before allow policies.

You also need to consider the following:

- Cloud Logging (/logging/docs) includes aggregated sinks (/logging/docs/export/aggregated_sinks) that you can use to aggregate logs at the folder or organization level. For more information, see decide the security for your Google Cloud landing zone (/architecture/landing-zones/decide-security#aggregate-logs).
- Billing is not directly linked to the resource hierarchy, but assigned at the project level. However, to get aggregated information at the folder level, you can analyze your costs by project hierarchy (/billing/docs/how-to/reports-project-hierarchy) using billing reports.
- Hierarchical firewall policies (/vpc/docs/firewall-policies) let you implement consistent firewall policies throughout the organization or in specific folders. Inheritance is implicit, which means that you can allow or deny traffic at any level or you can delegate the decision to a lower level.

Decision points for resource hierarchy design

The following flowchart shows the things that you must consider to choose the best resource hierarchy for your organization.



The preceding diagram outlines the following decision points:

1. Do different subsidiaries, regional groups, or business units have very different policy requirements?
    a. If yes, follow the design based on region or subsidiaries (#option2).
    b. If no, go to the next decision point.
2. Do your workload or product teams require strong autonomy over their policies? For example, you don't have a central security team that determines policies for all workload or product teams.
    a. If yes, see the design based on accountability framework (#option3).
    b. If no, see the design based on application environment (#option1).

Your specific use case might lead you to another resource hierarchy design than what the decision chart suggests. Most organizations choose a hybrid approach and select different designs at different levels of the resource hierarchy, starting with the design that most affects policies and access.

Option 1: Hierarchy based on application environments

In many organizations, you define different policies and access controls for different application environments, such as development, production, and testing. Having separate policies that are standardized across each environment eases management and configuration. For example, you might have security policies that are more stringent in production environments than in testing environments.

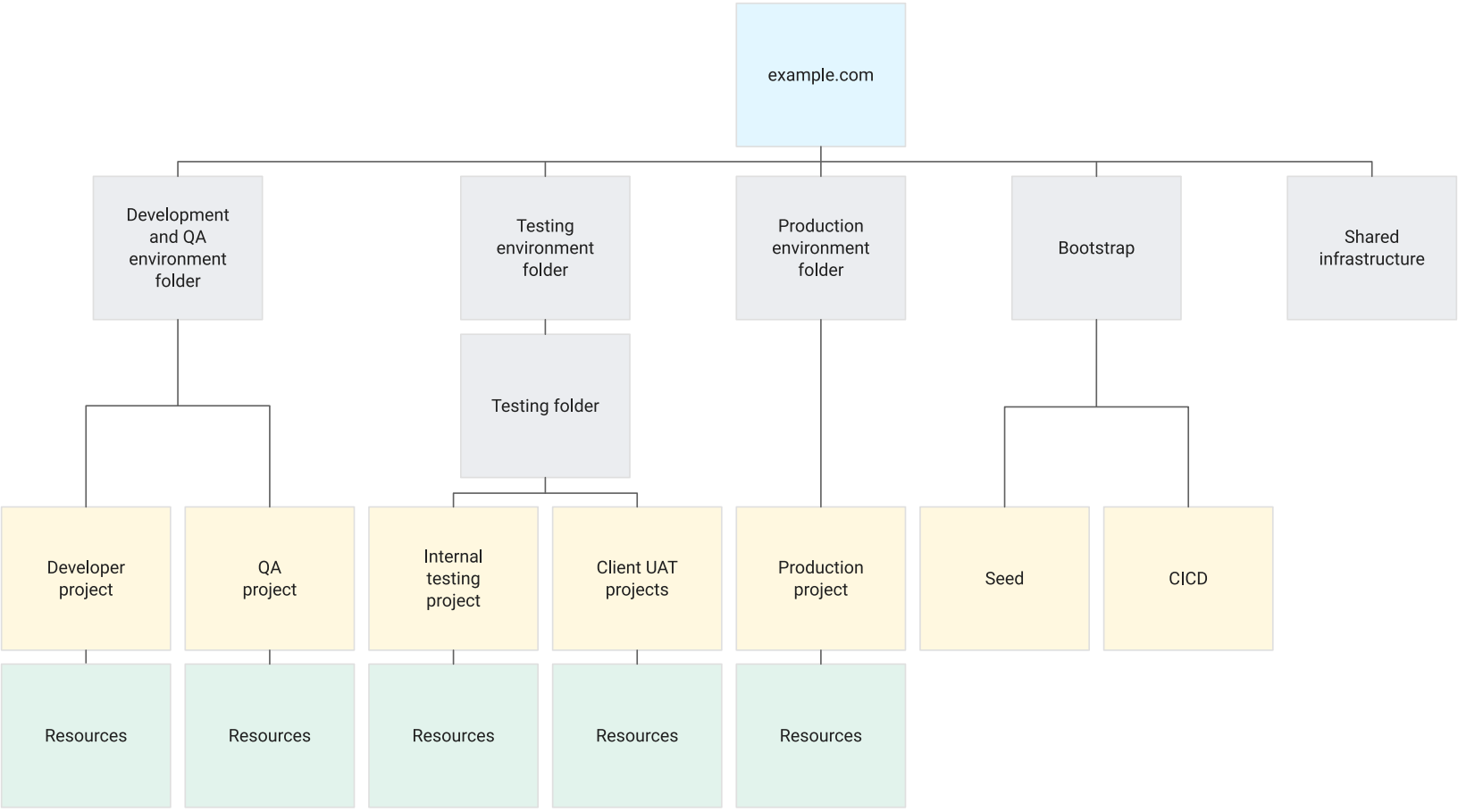
Use a hierarchy based on application environments if the following is true:

- You have separate application environments that have different policy and administration requirements.
- You have centralized security or audit requirements that a central security team must be able to enforce consistently on all production workloads and data.
- You require different Identity and Access Management (IAM) roles to access your Google Cloud resources in different environments.

Avoid this hierarchy if the following is true:

- You don't run multiple application environments.
- You don't have varying application dependencies and business processes across environments.
- You have strong policy differences for different regions, teams, or applications.

The following diagram shows a hierarchy for example.com, which is a fictitious financial technology company.



As shown in the preceding diagram, example.com has three application environments that have different policies, access controls, regulatory requirements, and processes. The environments are as follows:

- **Development and QA environment:** This environment is managed by developers who are both internal employees and consultants. They continuously push code and are responsible for quality assurance. This environment is never available to your business' consumers. The environment has less strict integration and authentication requirements than the production environment, and developers are assigned approved roles with suitable permissions. The Development and QA environment is designed only for standard application offerings from example.com.
- **Testing environment:** This environment is used for regression and application testing, and supports the business-to-business (B2B) offerings of example.com clients who use example.com APIs. For this purpose, example.com creates two project types:
  - Internal testing projects to complete internal regression, performance, and configuration for B2B offerings.

- Client UAT projects with multi-tenant support so that B2B clients can validate their configurations and align with example.com requirements for user experience, branding, workflows, reports, and so on.

- **Production environment:** This environment hosts all product offerings that are validated, accepted, and launched. This environment is subject to Payment Card Industry Data Security Standard (PCI DSS) regulations, uses hardware security modules (HSMs), and integrates with third-party processors for items such as authentication and payment settlements. The audit and compliance teams are critical stakeholders of this environment. Access to this environment is tightly controlled and limited mostly to automated deployment processes.

For more information about designing a hierarchy that is based on application environments, see Organization structure (/architecture/security-foundations/organization-structure) in the enterprise foundations blueprint.

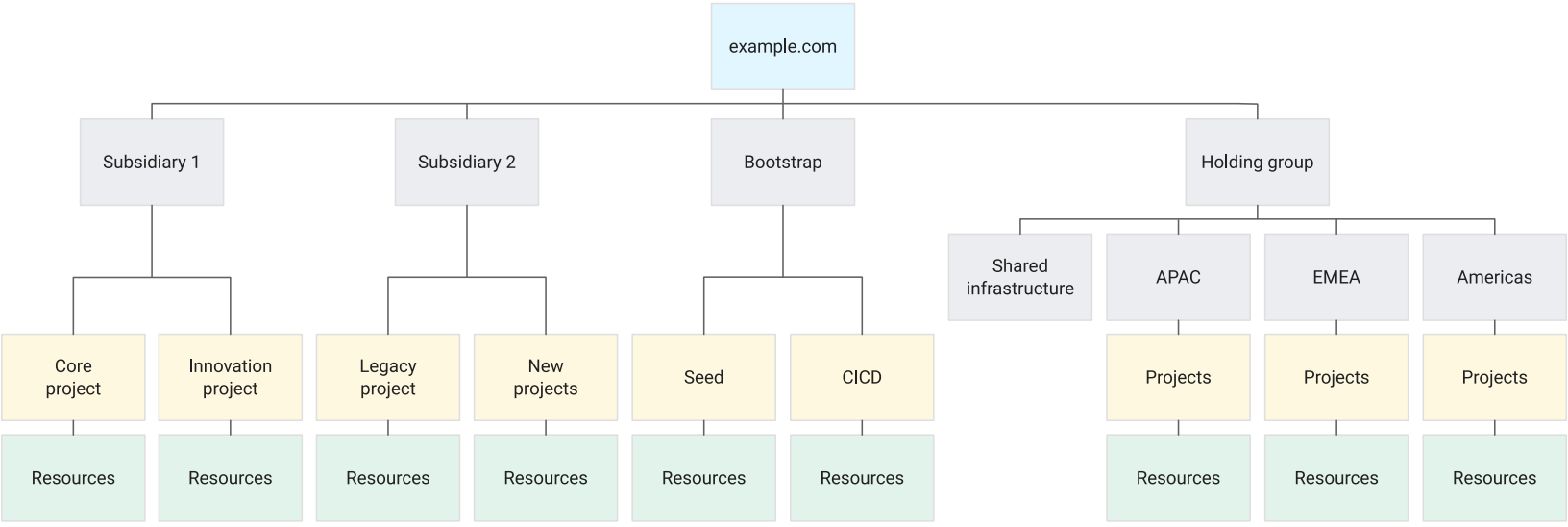
Option 2: Hierarchy based on regions or subsidiaries

Some organizations operate across many regions and have subsidiaries doing business in different geographies or have been a result of mergers and acquisitions. These organizations require a resource hierarchy that uses the scalability and management options in Google Cloud, and maintains the independence for different processes and policies that exist between the regions or subsidiaries. This hierarchy uses subsidiaries or regions as the highest folder level in the resource hierarchy. Deployment procedures are typically focused around the regions.

Use this hierarchy if the following is true:

- Different regions or subsidiaries operate independently.
- Regions or subsidiaries have different operational backbones, digital platform offerings, and processes.
- Your business has different regulatory and compliance standards for regions or subsidiaries.

The following diagram shows an example hierarchy of a global organization with two subsidiaries and a holding group with a regionalized structure.



The preceding diagram has the following hierarchy structure:

- The following folders are on the first level:
  - The `Subsidiary 1` and `Subsidiary 2` folders represent the two subsidiaries of the company that have different access permissions and policies from the rest of the organization. Each subsidiary uses IAM (/iam/docs/resource-hierarchy-access-control) to restrict access to their projects and Google Cloud resources.
  - The `Holding group` folder represents the groups that have high-level common policies across regions.

    - The `Bootstrap` folder represents the common resources that are required to deploy your Google Cloud infrastructure, as described in the enterprise foundations blueprint (/architecture/security-foundations/organization-structure#folders).
- On the second level, within the *Group Holding* folder, there are the following folders:
    - The `APAC`, `EMEA`, and `Americas` folders represent the various regions within the group that have different governance, access permissions, and policies.
    - The `Shared infrastructure` folder represents resources that are used globally across all regions.
- Within each folder are various projects that contain the resources that these subsidiaries or regions are responsible for.

You can add more folders to separate different legal entities, departments, and teams within your company.

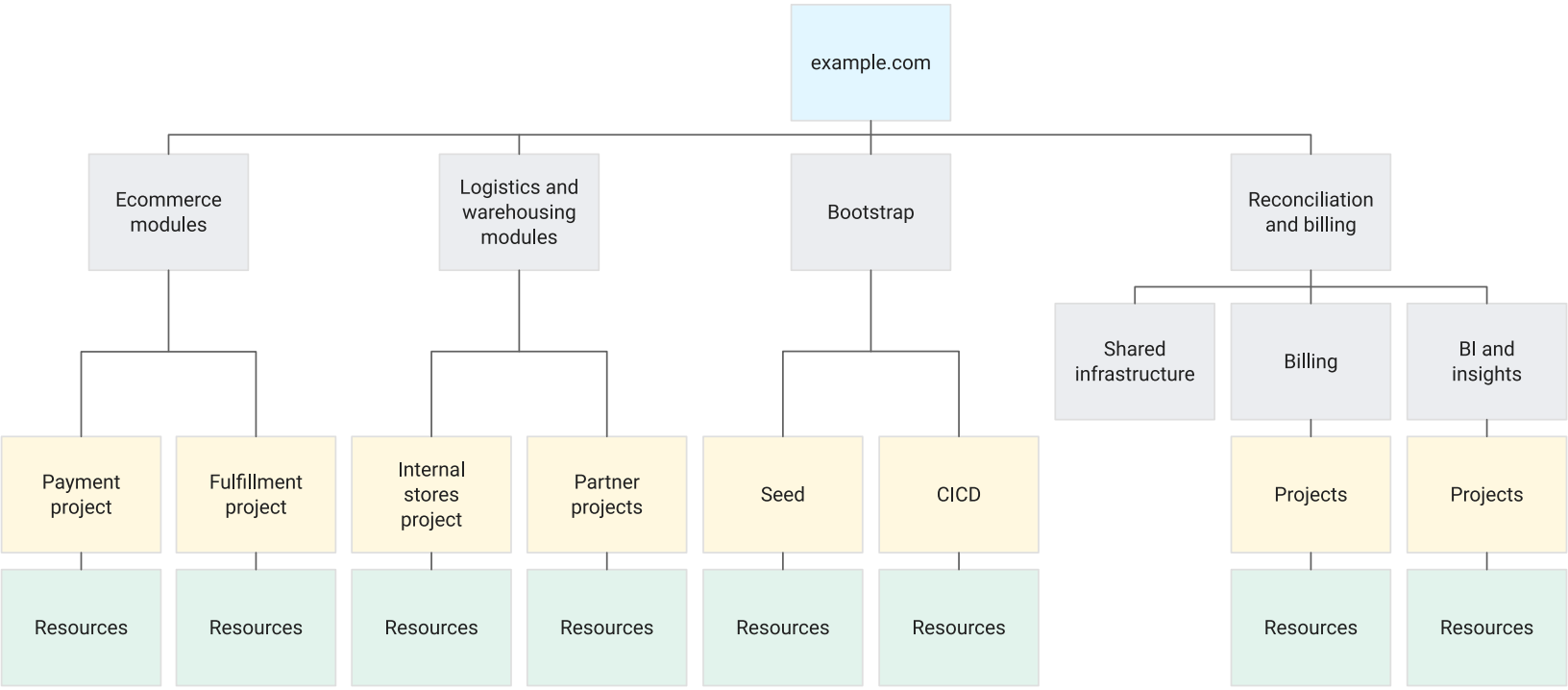
Option 3: Hierarchy based on an accountability framework

A hierarchy based on an accountability framework works best when your products are run independently or organizational units have clearly defined teams who own the lifecycle of the products. In these organizations, the product owners are responsible for the entire product lifecycle, including its processes, support, policies, security, and access rights. Your products are independent from each other, and organization-wide guidelines and controls are uncommon.

Use this hierarchy when the following is true:

- You run an organization that has clear ownership and accountability for each product.
- Your workloads are independent and don't share many common policies.
- Your processes and external developer platforms are offered as service or product offerings.

The following diagram shows an example hierarchy for an ecommerce platform provider.



The preceding diagram has the following hierarchy structure:

- The following folders are on the first level:

    - The folders that are named `Ecommerce Modules` and `Logistics and Warehousing Modules` represent the modules within the platform offering that require the same access permissions and policies during the product lifecycle.
    - The `Reconciliation and Billing` folder represents the product teams who are responsible for the end-to-end modules for specific business components within the platform offering.
    - The `Bootstrap` folder represents the common resources that are required to deploy your Google Cloud infrastructure, as described in the enterprise foundations blueprint (/architecture/security-foundations/organization-structure#folders).
- Within each folder are various projects that contain the independent modules that different product teams are responsible for.

For more information, see Fabric FAST Terraform framework resource hierarchy (https://github.com/GoogleCloudPlatform/cloud-foundation-fabric/tree/master/fast/stages/1-resman).

Best practices for resource hierarchy

The following sections describe the best practices for designing resource hierarchy that we recommend, regardless of the resource hierarchy that you choose.

For more best practices on how to configure your Cloud Identity and Google Workspace accounts and organizations, see Best practices for planning accounts and organizations (/architecture/identity/best-practices-for-planning).

Use a single organization node

To avoid management overhead, use a single organization node whenever possible. However, consider using multiple organization nodes to address the following use cases:

- You want to test major changes to your IAM levels or resource hierarchy.
- You want to experiment in a sandbox environment that doesn't have the same organization policies.
- Your organization includes sub-companies that are likely to be sold off or run as completely separate entities in the future.

Use standardized naming conventions

Use a standardized naming convention throughout your organization. The security foundations blueprint has a sample naming convention (/architecture/security-foundations/summary#naming-conventions) that you can adapt to your requirements.

Keep bootstrapping resources and common services separate

Keep separate folders for bootstrapping the Google Cloud environment using infrastructure-as-code (IaC) and for common services that are shared between environments or applications. Place the bootstrap folder right below the organization node in the resource hierarchy.

Place the folders for common services at different levels of the hierarchy, depending on the structure that you choose.

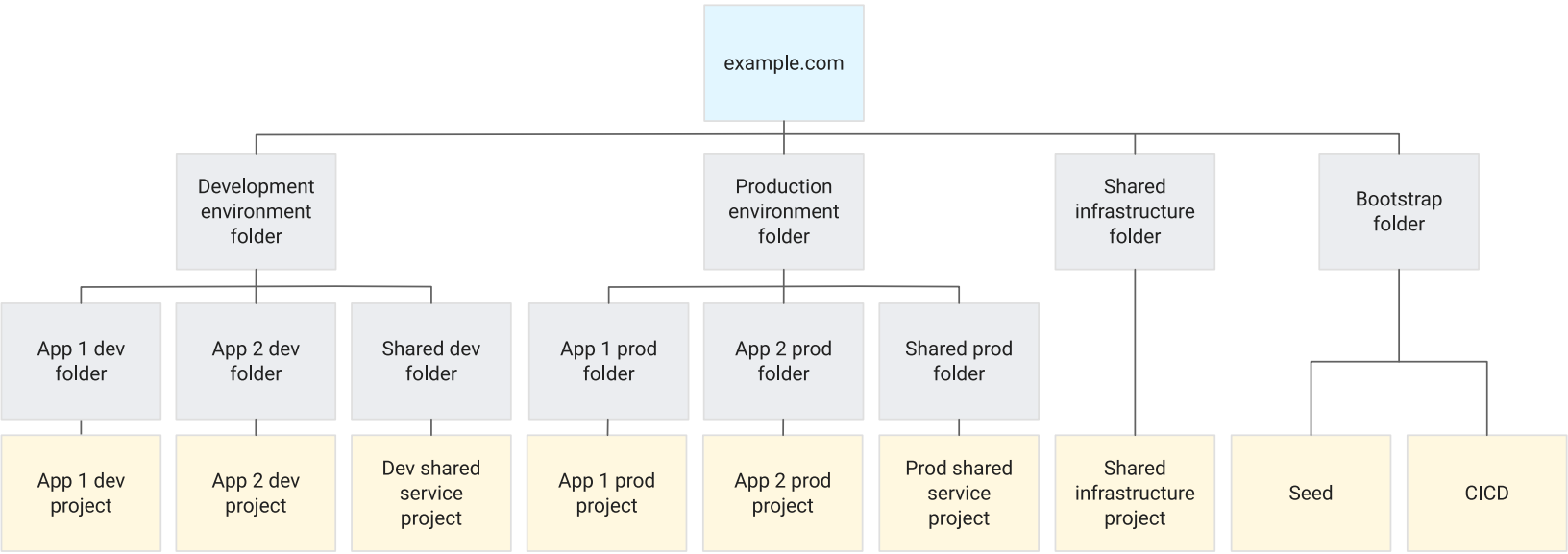
Place the folder for common services right below the organization node when the following is true:

- Your hierarchy uses application environments at the highest level and teams or applications at the second layer.
- You have shared services such as monitoring that are common between environments.

Place the folder for common services at a lower level, below the application folders, when the following is true:

- You have services that are shared between applications and you deploy a separate instance for each deployment environment.
- Applications share microservices that require development and testing for each deployment environment.

The following diagram shows an example hierarchy where there is a shared infrastructure folder that is used by all environments and shared services folders for each application environment at a lower level in the hierarchy:



The preceding diagram has the following hierarchy structure:

- The folders on the first level are as follows:
  - The `Development environment` and `Production environment` folders contain the application environments.
  - The `Shared infrastructure` folder contains common resources that are shared across environments, such as monitoring.
  - The `Bootstrap` folder contains the common resources required to deploy your Google Cloud infrastructure, as described in the enterprise foundations blueprint (/architecture/security-foundations/organization-structure#folders).
- On the second level, there are the following folders:
  - A folder in each environment for each application (`App 1` and `App 2`) which contains the resources for these applications.
  - A `Shared` folder for both application environments that contains services that are shared between the applications but are independent for each environment. For example, you might have a folder-level secrets project (/architecture/security-foundations/organization-structure#projects) so that you can apply different allow policies to your production secrets and non-production secrets.
- Within each application folder are various projects that contain the independent modules that are part of each application.

What's next

- Design the network for your landing zone (/architecture/landing-zones/decide-network-design) (the next document in this series).
- Review the enterprise foundations blueprint (/architecture/security-foundations).
- Read the blueprints and whitepapers that are available in the Google Cloud security best practices center (/security/best-practices).

Last updated 2024-10-31 UTC.