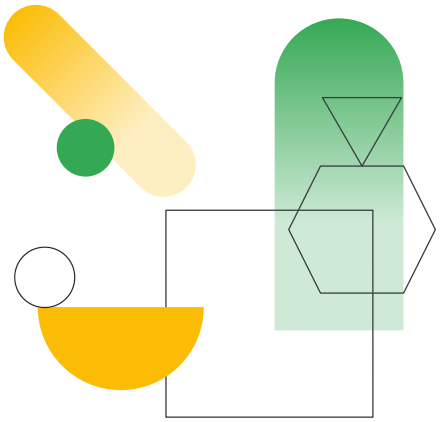
Proprietary + Confidential

Google Cloud

Google Cloud Architect Design and Process Workbook



Proprietary + Confidential

ClickTravel

> Your ticket to the clouds!

Proprietary + Confidential

1b. ClickTravel

Brief description

ClickTravel is a global travel agency that wants to build a scalable e-commerce platform to serve a global customer base.

List a few main features

- Travelers can search and book travel (hotels, flights, trains, cars)
- Pricing will be individualized based on customer preferences and demand
- Strong social media integration with reviews, posts, and analytics
- Suppliers (airlines, hotels, etc.) can upload inventory

List roles of typical users

- Customer
- Traveler
- Inventory supplier
- Manager

Google Cloud

ClickTravel is a global travel agency that wants to build a scalable e-commerce platform to serve a global customer base.

Proprietary + Confidential

2a. Writing user personas

Karen

Karen is a busy businesswoman who likes to take luxury weekend breaks, often booked at the last minute. A typical booking comprises a hotel and flight. Recommendations play a major role in the choice Karen makes, as does customer feedback. Karen likes to perform all operations from her phone.

Google Cloud

Here are a couple of examples of personas for our online travel portal.

Karen is a busy businesswoman who likes to take luxury weekend breaks, often booked at the last minute. A typical booking comprises a hotel and flight. Recommendations play a major role in the choice Karen makes, as does customer feedback. Karen likes to perform all operations from her phone.

Proprietary + Confidential

2a. Writing user personas

Andrew

Andrew is a student who likes to travel home to visit parents and also takes vacations twice yearly. His primary concern is cost, and he will always book the lowest price travel regardless of convenience. Andrew has no loyalty and will use whichever retailer can provide the best deal.

Google Cloud

Andrew is a student who likes to travel home to visit parents and also takes vacations twice yearly. His primary concern is cost, and he will always book the lowest price travel regardless of convenience. Andrew has no loyalty and will use whichever retailer can provide the best deal.

Proprietary + Confidential

2b. Writing user stories

Search for flight and hotel

As a traveler, **I want to** search for a flight-hotel combination to a destination on dates of my choice **so that** I can find the best price.

Proprietary + Confidential

2b. Writing user stories

Supply hotel inventory

As a hotel operator, **I want** to bulk supply hotel inventory **so that** ClickTravel can sell it on my behalf.

Proprietary + Confidential

2b. Writing user stories

Analyze sales performance

As a ClickTravel manager, **I want to** analyze the sales performance data of all our suppliers **so that** I can identify poor performers and help them improve.

Proprietary + Confidential

3. Defining SLIs and SLOs

Based on the requirements of your case study, fill in the table below with SLOs and SLIs.

User story	SLO	SLI
Search hotel and flight	Available 99.95%	Fraction of 200 vs 500 HTTP responses from API endpoint measured per month
Search hotel and flight	95% of requests will complete in under 200 ms	Time to last byte GET requests measured every 15 seconds aggregated per 5 minutes
Supply hotel inventory	Error rate of < 0.00001%	Upload errors measured as a percentage of bulk uploads per day by custom metric
Supply hotel Inventory	Available 99.9%	Fraction of 200 vs 500 HTTP responses from API endpoint measured per month
Analyze sales performance	95% of queries will complete in under 10s	Time to last byte GET requests measured every 60 seconds aggregated per 10 minutes

Google Cloud

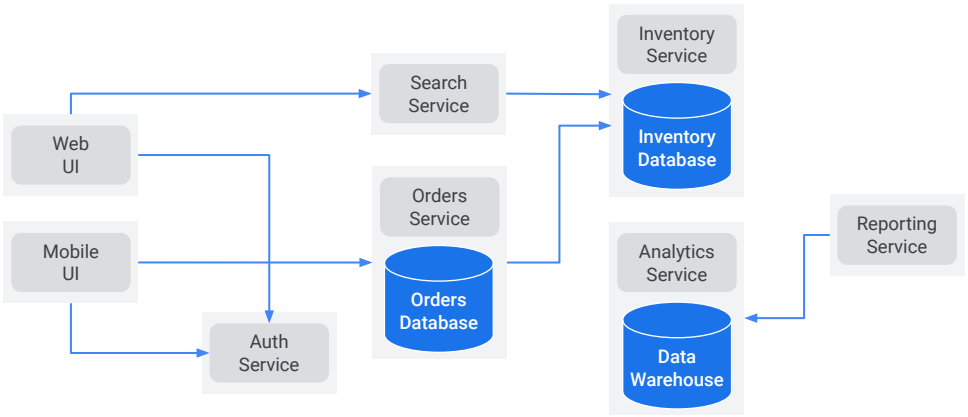
Here are some example SLOs and SLIs for out travel portal application. Notice that the SLI describes what we are going to measure and how: for example, the “Fraction of 200 vs 500 HTTP responses from API endpoint measured per month.” This example is a way of measuring availability.

The SLO represents the goal we are trying to achieve for a given SLI. For example, “Available 99.95%” of the time.

Proprietary + Confidential

4. Design microservices for your application

Draw a diagram showing your application's microservices and their connections.



Google Cloud

Here is a sample diagram depicting the microservices of our online travel portal. I suppose we could lay this out many different ways. There isn't really one and only one right way to design an application.

Notice, we have separate services for our web and mobile UIs. There's a shared authentication service and we have microservices for search, orders, inventory, analytics and reporting. Remember, each of these services will be deployed as a separate application. Where possible we want stateless services, but the orders and inventory services will need databases, and the analytics service will provide a data warehouse.

This might make a good starting point, and we could adjust as needed when we starting implementing the application.

Proprietary + Confidential

5. Designing REST APIs

Fill in the table with your services and their resources and operations.

Service name	Collections	Methods
Search	Trips (flights + hotel combination)	find save
Inventory	Items (flights + hotels)	add search get remove
Analytics	Sales	analyze get list
Order processing	Orders	add get list update

Google Cloud

Here's an example for our online travel portal. Obviously, our API would be larger than this, but in a way the APIs are all more of the same. Each service manages and makes available some collection of data. For any collection of data there are a handful of typical operations we do with that data.

This is similar to Google Cloud APIs. For example in Google Cloud, we have a service called Compute Engine, which is used to create and manage virtual machines, networks, and the like. The Compute Engine API has collections like instances, instanceGroups, networks, subnetworks, and many more. For each collection, various methods are used to manage the data.

Proprietary + Confidential

6. Defining storage characteristics

Fill in the required storage features.

Service	Structured or Unstructured	SQL or NoSQL	Strong or Eventual Consistency	Amount of Data (MB, GB, TB, PB, ExB)	Read only or Read/Write
Inventory	Structured	NoSQL	Strong	GB	Read/Write
Inventory uploads	Unstructured	N/A	N/A	GB	Read only
Orders	Structured	SQL	Strong	TB	Read/Write
Analytics	Structured	SQL	Eventual	TB	Read only

Google Cloud

Here's an example for our online travel portal, ClickTravel. We focussed on the inventory, inventory uploads, ordering, and analytics services. As you can see, each of these services has different requirements that might result in choosing different Google Cloud services.

Proprietary + Confidential

7. Choosing Google Cloud Storage and Data Services

Choose the Google Cloud storage products for each service.

Service	Persistent Disk	Cloud Storage	Cloud SQL	Firestore	Cloud Bigtable	Cloud Spanner	BigQuery
Inventory				X			
Inventory uploads		X					
Orders			X				
Analytics							X

Google Cloud

Here's an example for our online travel portal, ClickTravel.

For the inventory service we will use Cloud Storage for the raw inventory uploads. Suppliers will upload their inventory as JSON data stored in text files. That inventory will then be imported into a Firestore database.

The orders service will store its data in a relational database running in Cloud SQL.

The analytics service will aggregate data from various sources into a data warehouse, for which we'll use BigQuery.

Proprietary + Confidential

8a. Defining network characteristics for your services

Fill in the required network features.

Service	Internet facing or Internal only	HTTP	TCP	UDP	Multi-Regional?
Search	Internet facing	X			Yes
Inventory	Internal		X		No
Analytics	Internet facing	X			No
Web UI	Internet facing	X			Yes
Orders	Internal		X		No

Google Cloud

Here's a completed example for our online travel portal, ClickTravel.

The inventory and orders service are internal and regional using TCP. The other services need to be facing the internet using HTTP. We decided to deploy these to multiple regions for lower latency, higher performance, and high availability to our users who are in multiple countries around the world.

Proprietary + Confidential

8b. Select the load balancers for your services

Choose the Google Cloud load balancer product(s) for each service.

Service	HTTP	TCP	UDP
Search	X		
Inventory		X	
Analytics	X		
Web UI	X		
Orders		X	

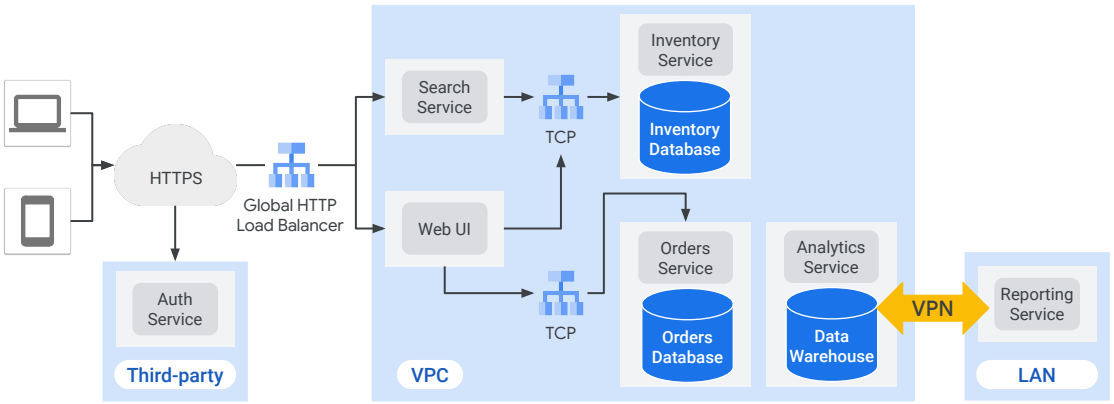
Google Cloud

Based on those network characteristics, we chose the global HTTP load balancer for our public-facing services and the internal TCP load balancer for our internal-facing services.

Proprietary + Confidential

9. Diagramming your network

Draw a diagram that depicts how your services will communicate over the network. Include regions, zones, load balancers, CDN, and DNS if applicable.



Google Cloud

Here's an example for our online travel portal, ClickTravel.

User traffic from mobile and web will first be authenticated using a third-party service. Then a Global HTTP Load Balancer directs traffic to our public facing Search and web UI services.

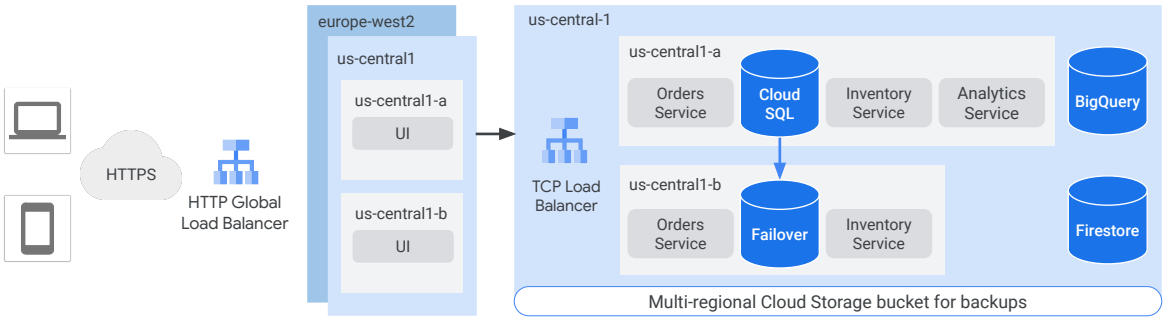
From there, regional TCP load balancers direct traffic to the internal inventory and orders services.

The analytics service could leverage BigQuery as the data warehouse with an on-prem reporting service that accesses the analytics service over a VPN. This might be good enough to start, and we could refine this once we start implementing it.

Proprietary + Confidential

10. Designing reliable, scalable applications

Even if some service is down, we want the web frontend of our application to be available nearly all the time. We also want the website to be fast with very low latency to users all over the world. Draw a diagram that depicts how we can achieve this using Google Cloud services.



Google Cloud

For our online travel application, ClickTravel, I'm assuming that this is an American company, but that I have a large group of customers in Europe.

I want the UI to be highly available, so I've placed it into us-central1 and europe-west2 behind a global HTTP Load Balancer. This load balancer will send user requests to the region closest to the user, unless that region cannot handle the traffic.

I could also deploy the backends globally but if I'm trying to optimize cost, I could start by just deploying those in us-central1.This will create latency for our European users but I can always revisit this later and have a similar backend in europe-west2.

To ensure high availability, I've decided to deploy the Orders and Inventory services to multiple zones. Because the Analytics service is not customer-facing, I can save some money by deploying it to a single zone. I again have a failover Cloud SQL database, and the Firestore database and BigQuery data warehouse are multi-regional, so I don't need to worry about a failover for those.

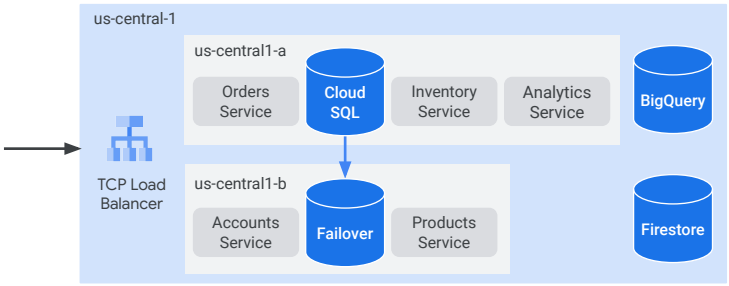
The Cloud SQL database needs a failover for high availability. Because BigQuery and Firestore are managed, we don't have to worry about that.

In case of a disaster, I'll keep backups in a multi-regional Cloud Storage bucket. That way if there is a regional outage, I can restore in another region.

Proprietary + Confidential

11a. Disaster recovery scenario

You've deployed for high availability by replicating resources in multiple zones. However, to meet regulatory requirements, you need a plan to recover from a disaster that brings down the entire region. Create a plan to bring up your application in another region if your main region is down.



Proprietary + Confidential

11b. Service disaster recovery scenarios

Write a high-level list of possible scenarios.

Service	Scenario	Recovery Point Objective	Recovery Time Objective	Priority
Inventory Firestore	Programmer deleted all inventory accidently	1 hour	1 hour	High
Orders Cloud SQL database	Orders database crashed	0 minutes	5 minutes	High
Analytics BigQuery database	User deletes table	0 minutes	24 hours	Medium

Google Cloud

Here's an example of some disaster scenarios for our online travel portal, ClickTravel.

Each of our services uses different database services and has different objectives and priorities. All of that affects how we design our disaster recovery plans.

Proprietary + Confidential

11c. Resource disaster recovery plans

For each scenario, fill in the table.

Resource	Backup Strategy	Backup Location	Recovery Procedure
Inventory Firestore	Daily automated backups	Multi-Regional Cloud Storage bucket	Cloud Functions and Cloud Scheduler
Orders Cloud SQL database	Binary logging and backups Failover replica in another zone	N/A	Automated failover Run backup script if needed
Analytics BigQuery database	No specific backup required	N/A	Re-import data to rebuild analytics tables

Google Cloud

Our Analytics service in BigQuery had the lowest priority; therefore, we should be able to re-import data to rebuild analytics tables if a user deletes them.

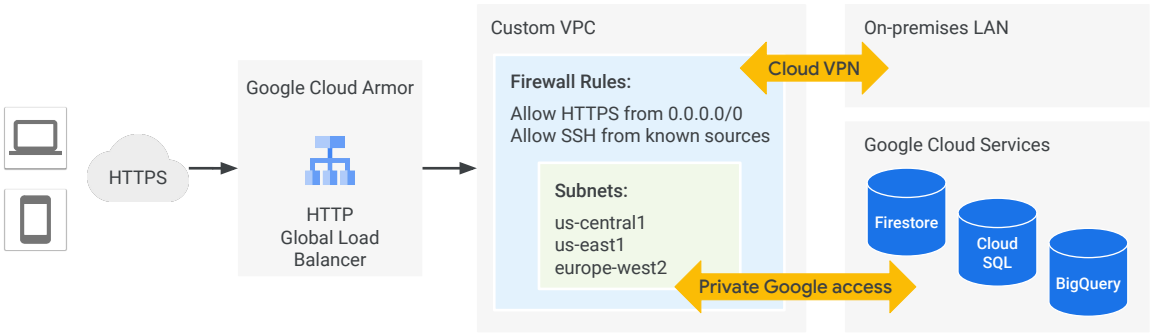
Our Orders service can't tolerate any data loss and has to be up and running almost immediately. For this we need a failover replica in addition to binary logging and automated backups.

Our Inventory service uses Firestore, and for that we can implement daily automated backups to a multi-regional Cloud Storage bucket. Cloud Functions and Cloud Scheduler can help with the recovery procedure.

Proprietary + Confidential

12. Modeling secure Google Cloud services

Draw a diagram that depicts how you will secure your services. Include firewalls, IAM roles, service accounts and network resources as appropriate.



Google Cloud

First, I configured Google Cloud Armor on a global HTTP Load Balancer to block any denied IP addresses. My custom VPC network has subnets in us-central1 for my American customers, and a backup subnet in us-east1 and a subnet in europe-west2 for my European customers.

My firewall rules only allow SSH from known sources, and although I allow HTTPS from anywhere, I can always deny IP addresses with Google Cloud Armor at the edge of Google Cloud's network. I also configured Cloud VPN tunnels to securely communicate with my on-premises network for my reporting service.

Now, while my load balancer needs a public IP address, I can secure my backend services by creating them without external IP addresses. In order for those instances to communicate with the Google Cloud database services, I enable Private Google Access. This enables the inventory, orders, and analytics services' traffic to remain private, while reducing my networking costs.

Proprietary + Confidential

13. Cost estimating and planning

Use the **pricing calculator** to determine and record the cost of your microservices.

Service name	Google Cloud Resource	Cost
Orders	Cloud SQL	$1,264.44
Inventory	Firestore	$215.41
Inventory	Cloud Storage	$1,801.00
Analytics	BigQuery	$214.72

Google Cloud

Here's a rough estimate for the database applications of my online travel portal, ClickTravel.

I adjusted my orders database to include a failover replica for high availability and came up with some high-level estimates for my other services. My inventory service uses Cloud Storage to store JSON data stored in text files. Because this is my most expensive service, I might want to reconsider the storage class or configure object lifecycle management.

Again, this is just an example, and your costs would depend on your case study.

Cloud SQL based on 1 instance of PostgreSQL in Iowa, 30 GB SSD storage Backup 1,000 GB, 12 cores, 16 GB RAM

Firestore based on US multi regional, 200,0000 reads per day 10000 writes 0 deletes and 1,000 GB stored

Cloud Storage Iowa - us central 50 TB storage 200,000 class A operations and 200,0000 class B operations

BigQuery pricing based on location US multi-regional, 10 TB storage, 100 GB per month streaming, 2 TB queries


Google Cloud