Deploying an Internal Generative AI App

The information in this presentation is classified:

Google confidential & proprietary

⚠️ This presentation is shared with you under NDA.

- Do **not** record or take screenshots of this presentation.
- Do **not** share or otherwise distribute the information in this presentation with anyone **inside** or **outside** of your organization.

Thank you!

In this module, you learn to ...

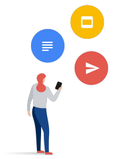
01 Containerize a basic Generative AI app

02 Deploy the app to Cloud Run

03 Protect your app behind a load balancer

04 Grant access to your app through Identity-Aware Proxy

Case Study: Deploy an internal generative AI tool



You can create

- Create a tool that can summarize PDFs or ask a question that can be answered by its text
- Make the tool accessible to non-technical team members
- Control access to the tool
- Put a CICD process in place to easily deploy updates to the tool

What are some options for adding a UI?

Bootstrap Flask

Your engineers know Flask for Python web apps. Bootstrap is an opinionated frontend toolkit. Bootstrap-Flask helps implement Bootstrap within Flask.

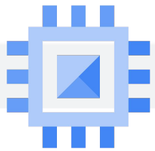
Streamlit

Streamlit allows you to create apps in pure Python, rendering a frontend for you.

Gradio

Gradio is a tool that helps you build a dashboard to control an ML model. You can even share a link to access a model that is being served from your local machine.

What are some options for serving your application?



Compute Engine

You could deploy a persistent VM if you expect few users.

App Engine

You could deploy a scalable app that can scale to zero instances.

Cloud Run

Cloud Run is the latest iteration of Google's serverless application deployments. It allows you to deploy a service with CICD built in.

How can you control authentication?

Identity Platform (Firebase Authentication)

You can grant access using a custom UI with passwords unique to the app or SSO with major auth providers.

Identity-Aware Proxy

You can block people from using the app at all unless they authenticate with their Google Identity which will need to have been pre-authorized.

For this app, you'll use the following:

Bootstrap Flask

Your engineers already know Flask and can customize as much as they want.

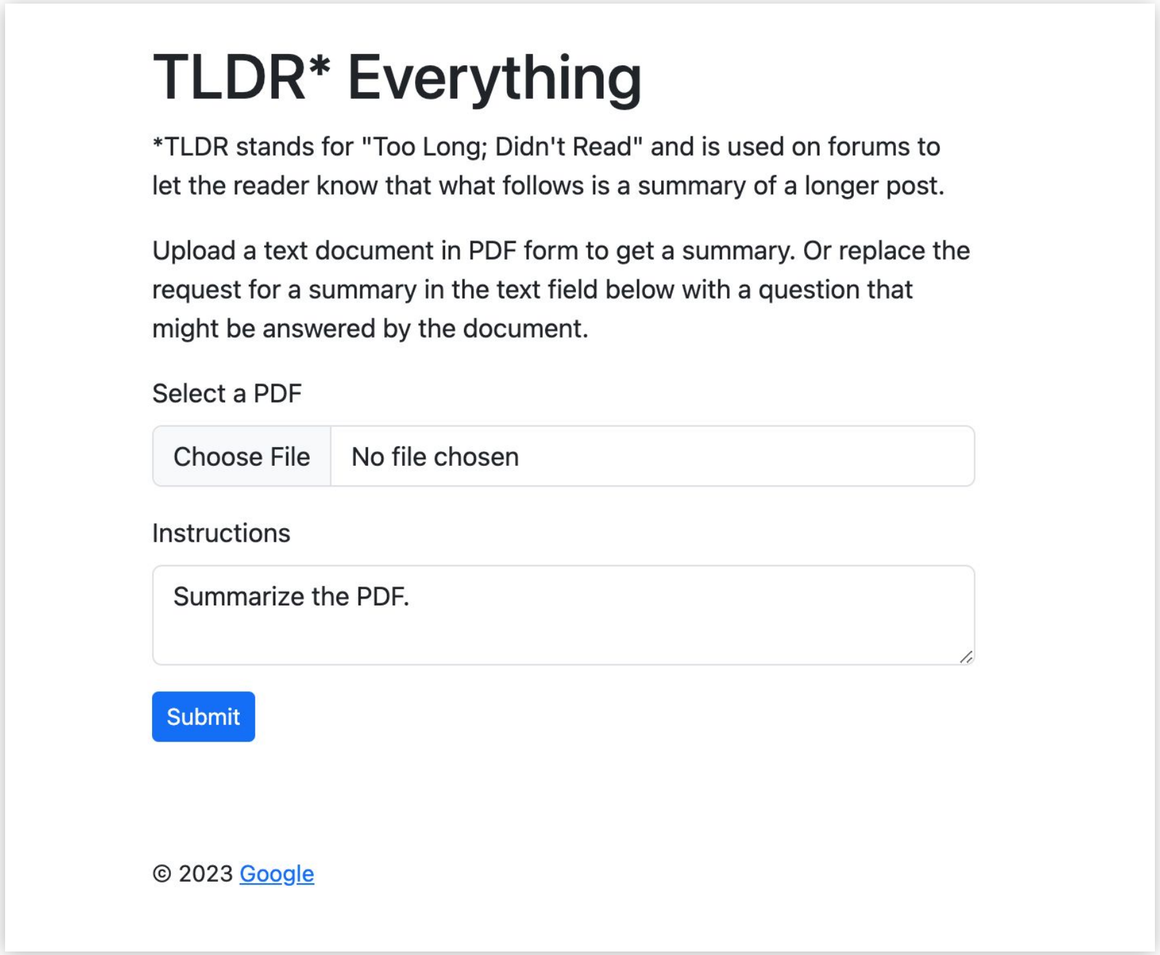
Cloud Run

You like working from containers and the CICD deployment.

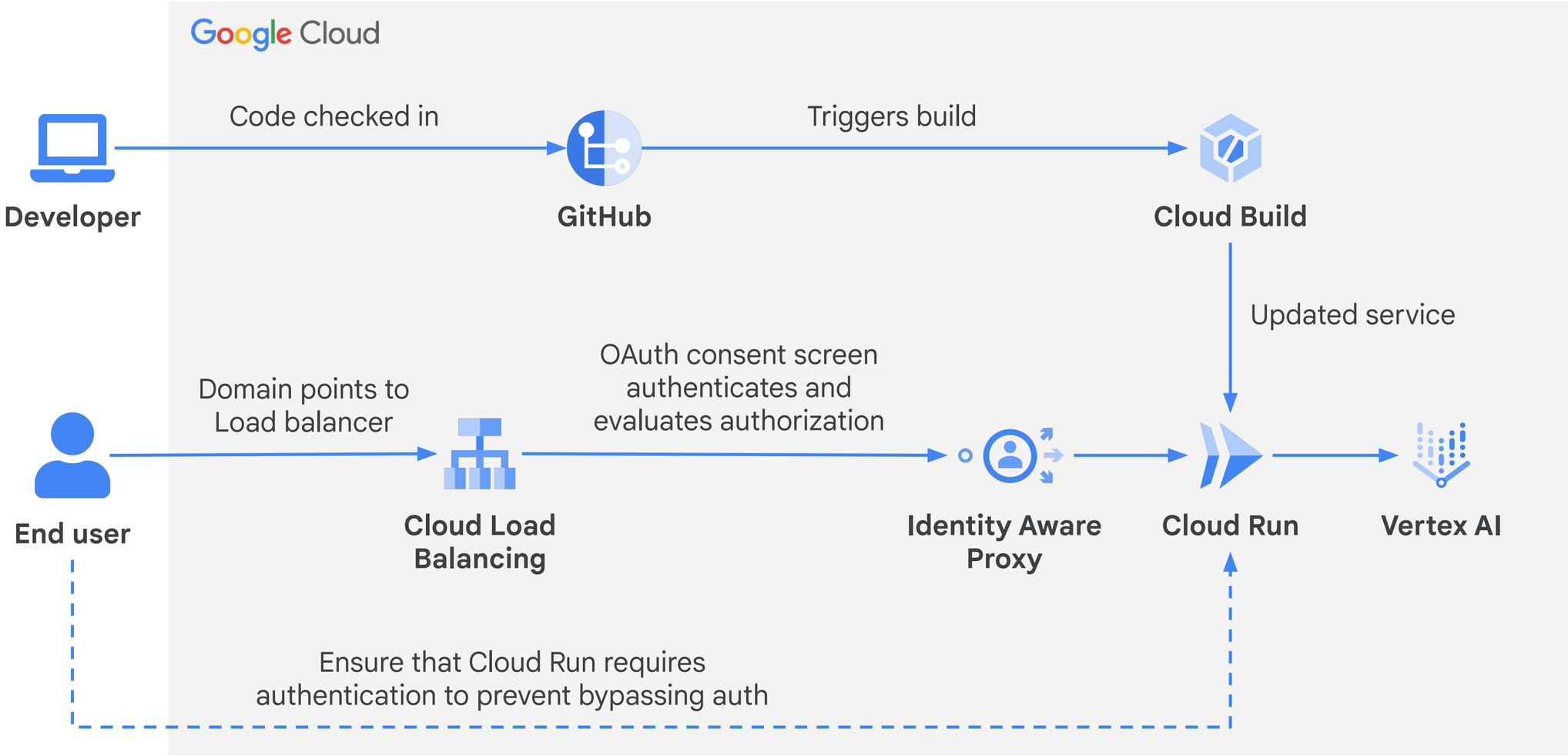
Identity-Aware Proxy

You don't want to mess with Firebase.

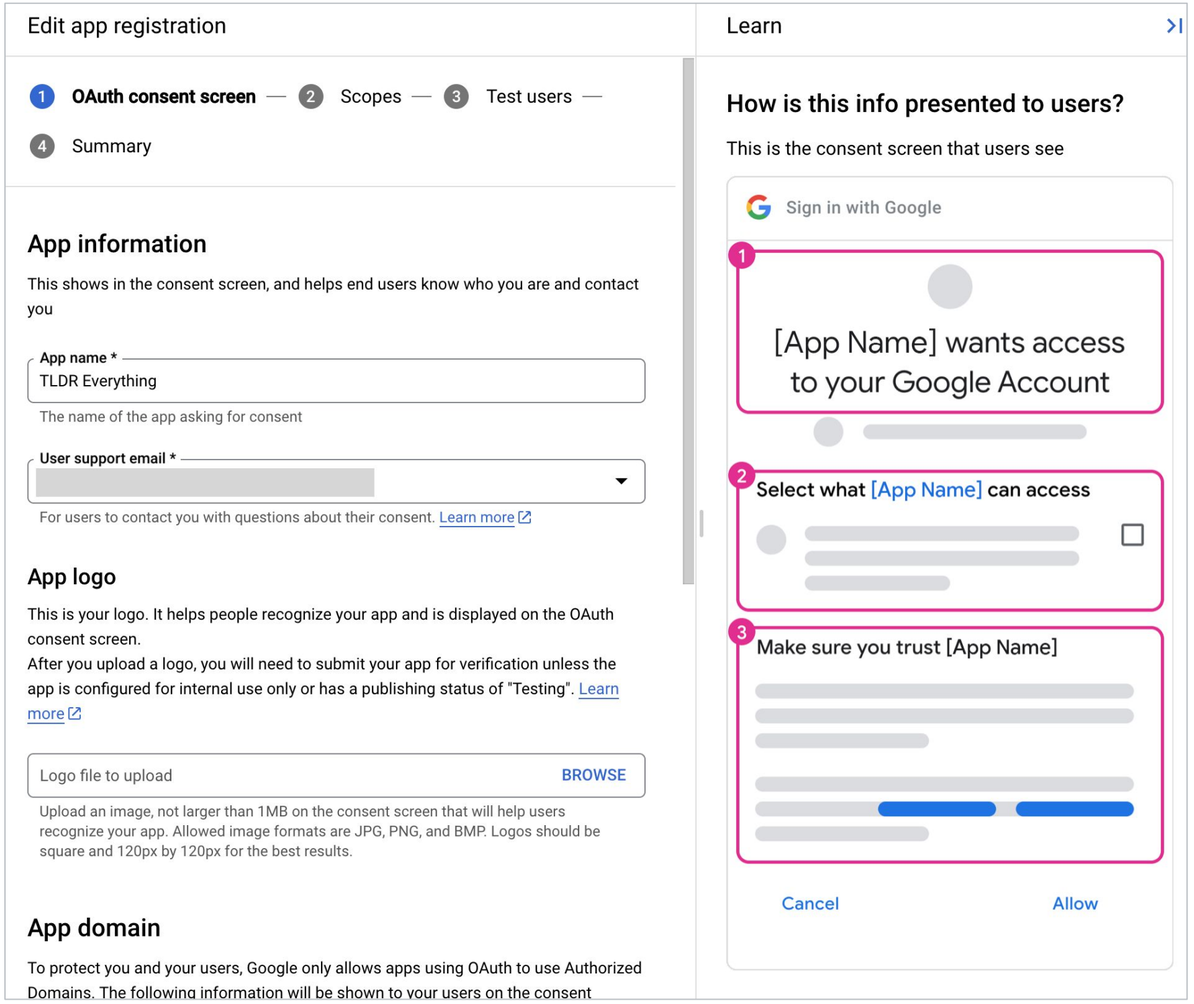
This is the frontend you will build



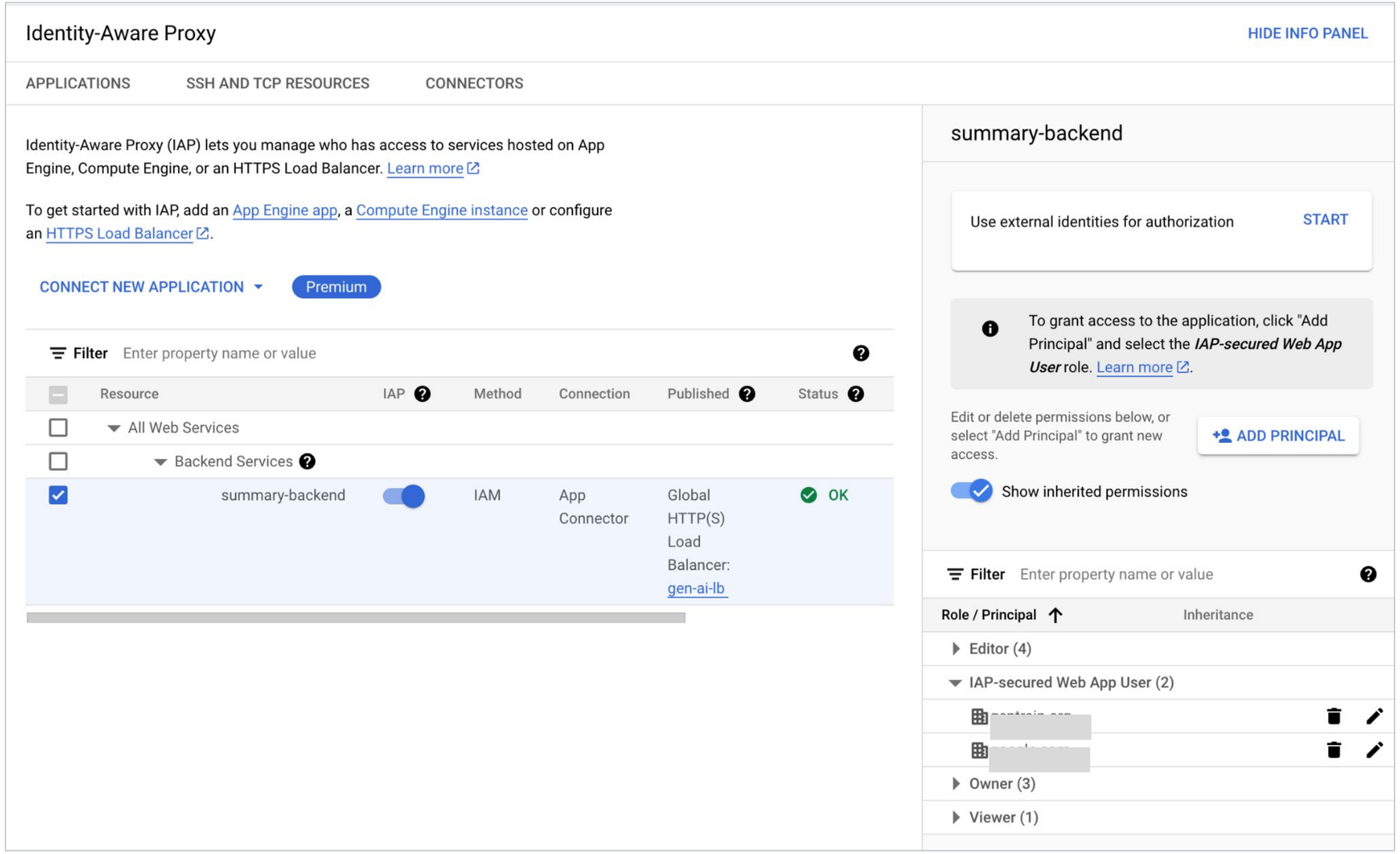
The architecture allows for CICD updates and authenticating with Google Identities



The OAuth Consent screen is what users will be shown when granting the app access to their Google Identity

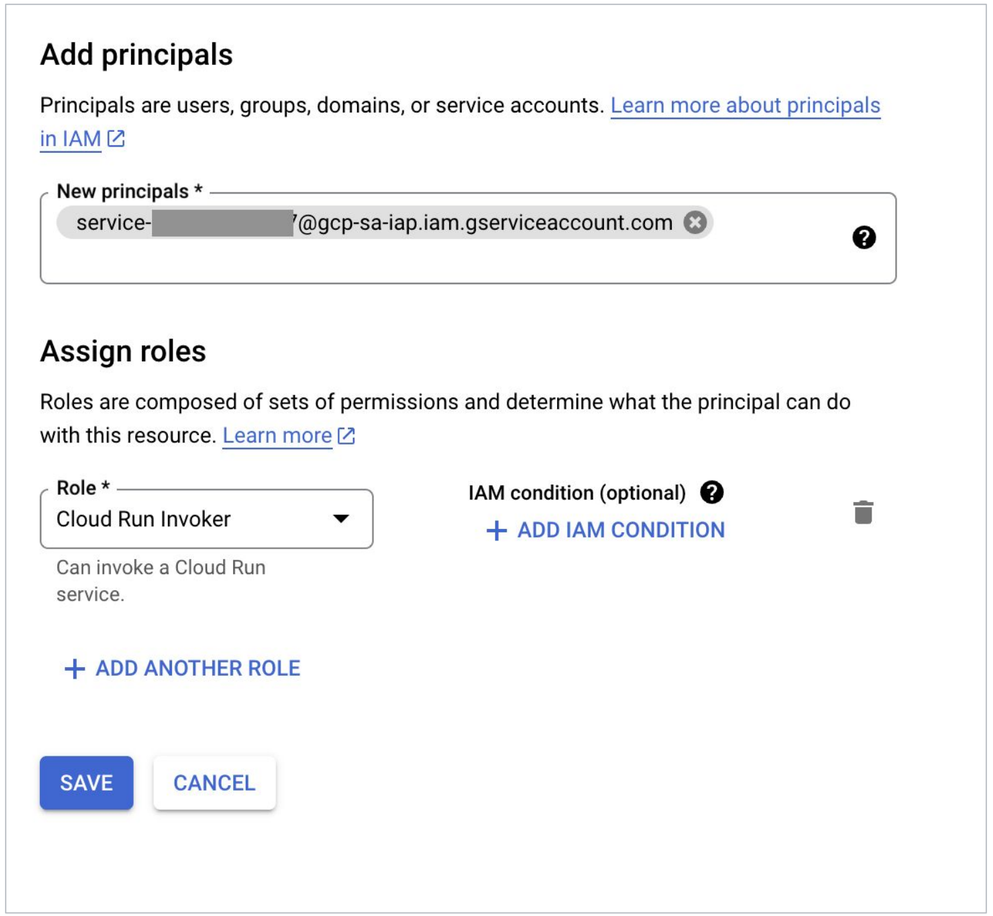


Users or whole email domains (i.e. example.com) can be granted access to IAP as IAP-Secured Web App Users

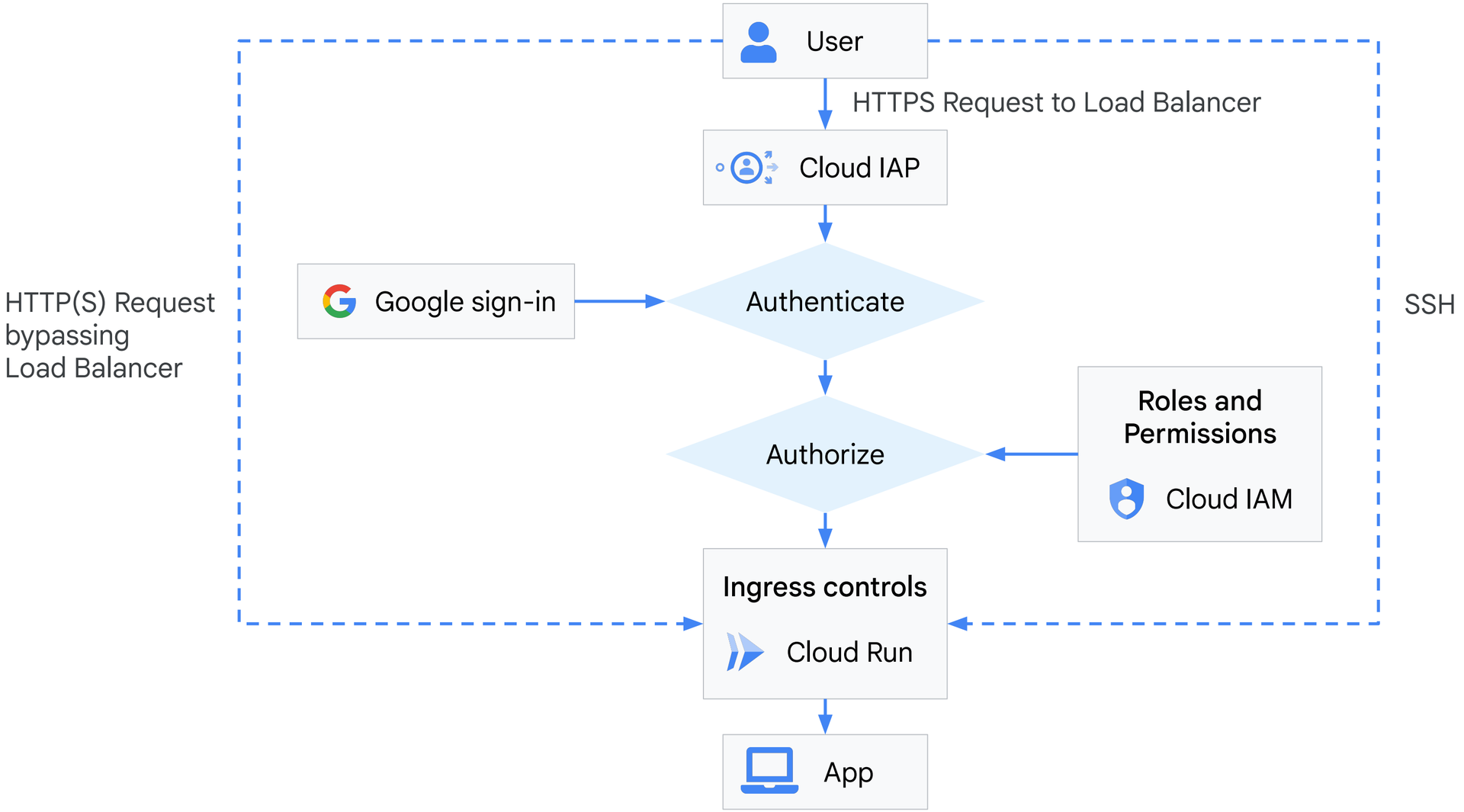


The IAP service account (which takes some time to be created) must be given the role of “Cloud Run Invoker”

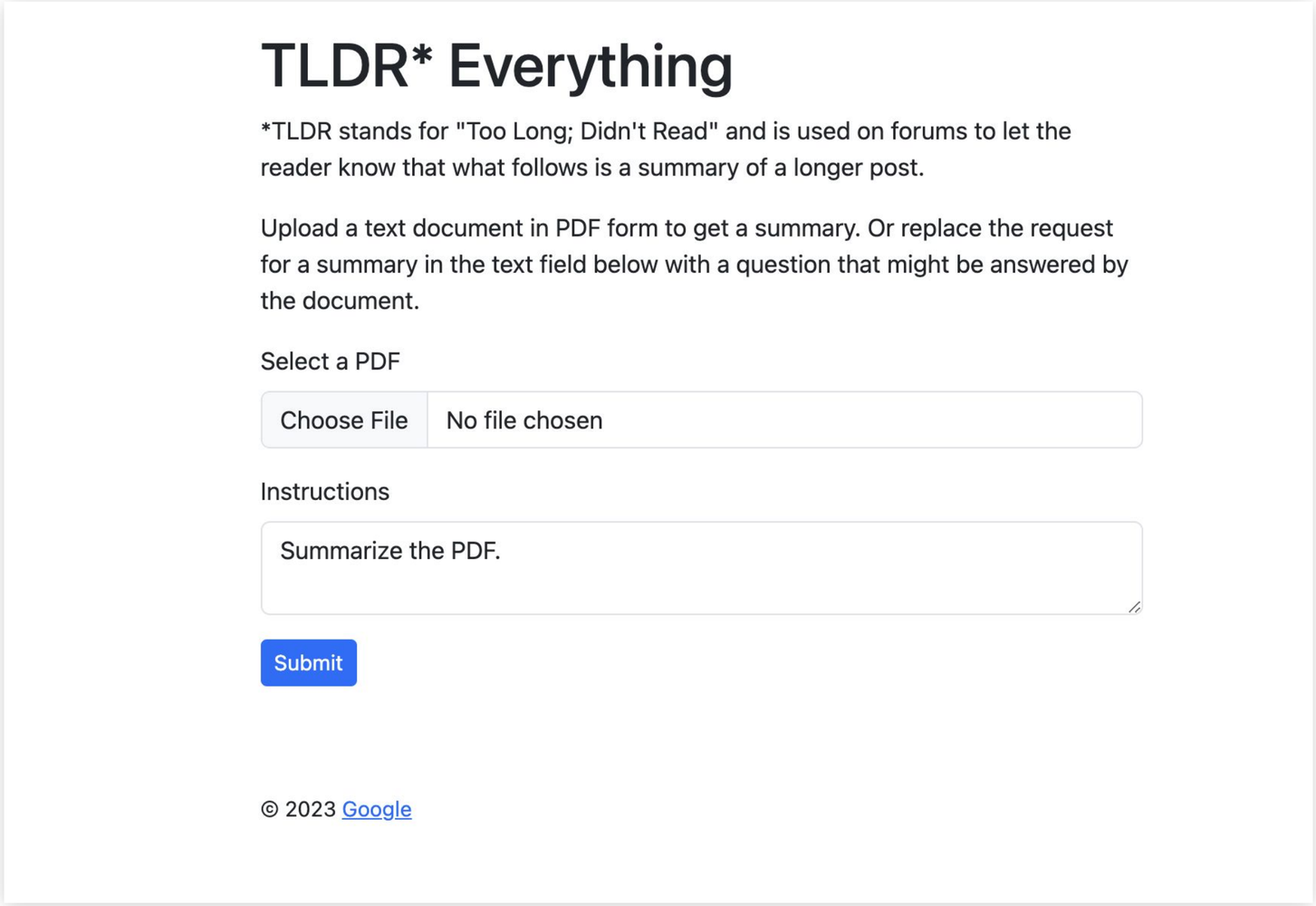
`service-[PROJECT-NUMBER]@gcp-sa-iap.iam.gserviceaccount.com`



How IAP works for a Cloud Run app



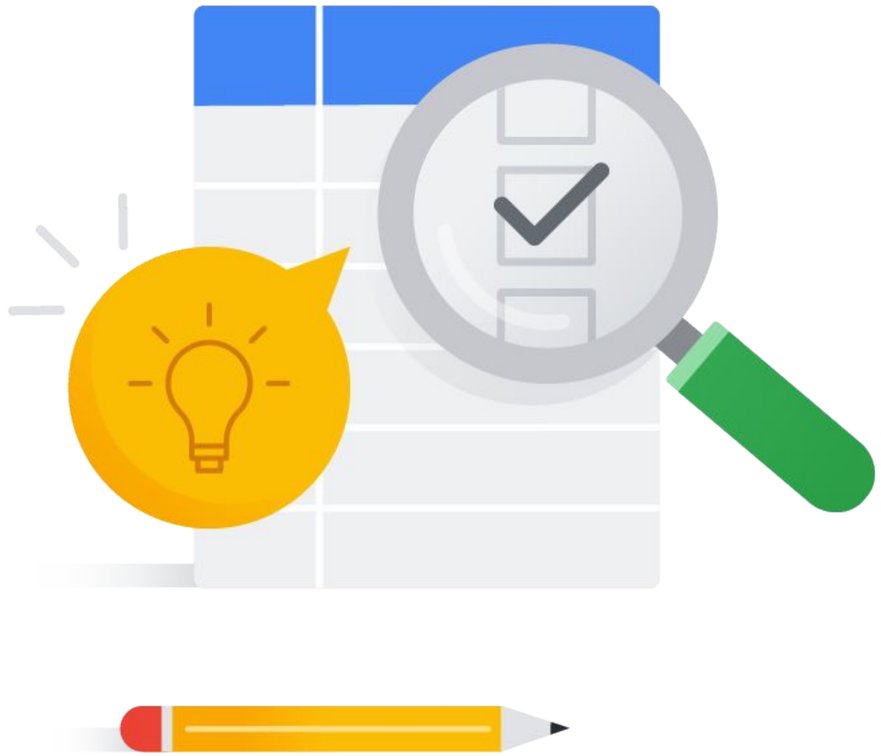
Now your users can authenticate and grant access through the OAuth Consent Screen to access the app



Lab

2 hours

Lab: Deploy and Secure a Gen AI Web Application



In this module, you learned to ...

01 Containerize a basic Generative AI app

02 Deploy the app to Cloud Run

03 Protect your app behind a load balancer

04 Grant access to your app through Identity-Aware Proxy

Questions and answers


Google Cloud