



262  
GFTmeets

# Vertex AI from 10.000 ft

01.12.2022

Pilar Madariaga . Rafael Banzo . Daniel Ruiz

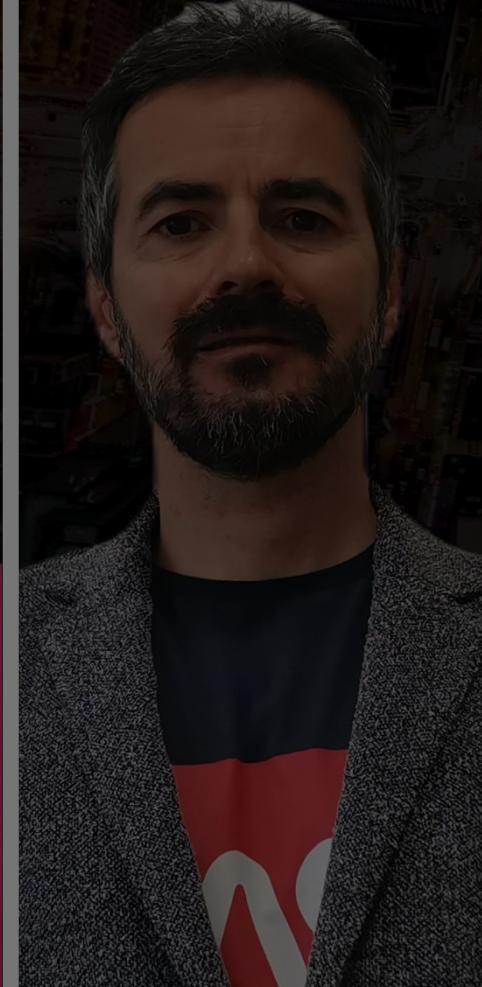
Pilar  
Madariaga

Data Scientist



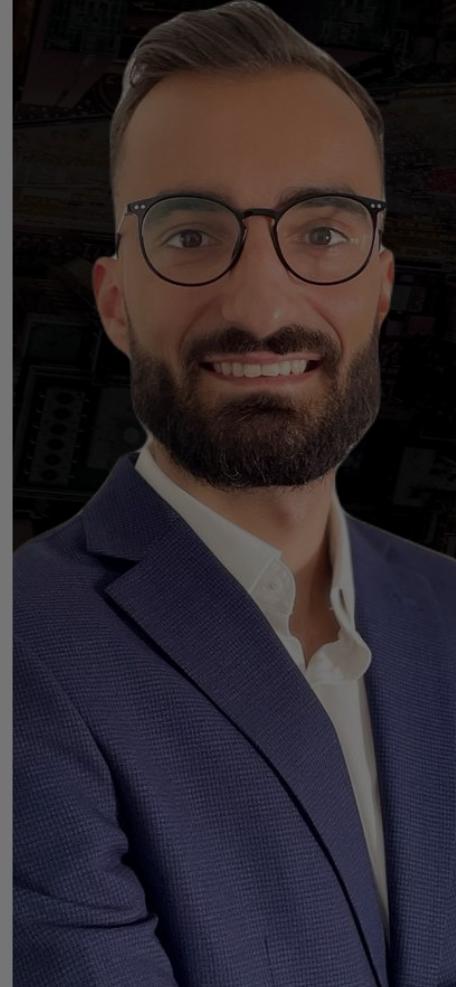
Rafael  
Banzo

Data Scientist



Daniel  
Ruiz

Data Scientist



GFT

> GFTmeets



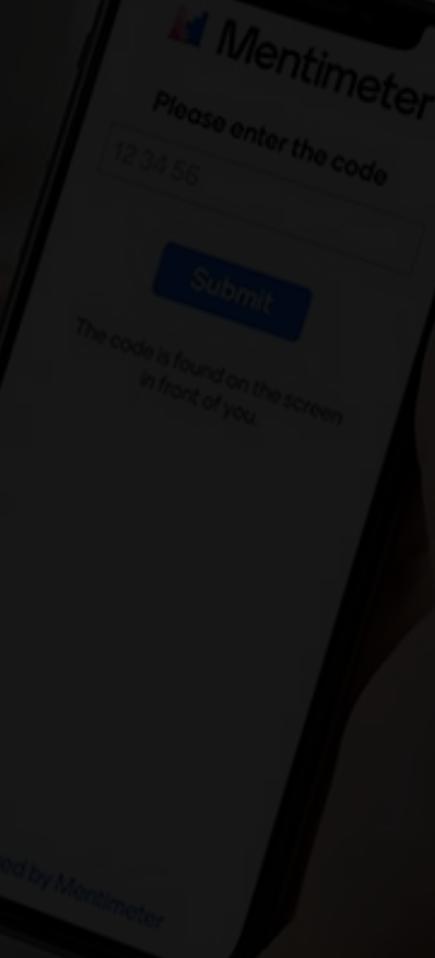
## Vertex AI Introduction

Automated  
Machine  
Learning

Manual AI  
Development

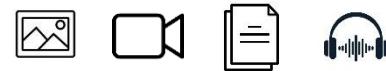
AI Model  
Deployment

Success Cases

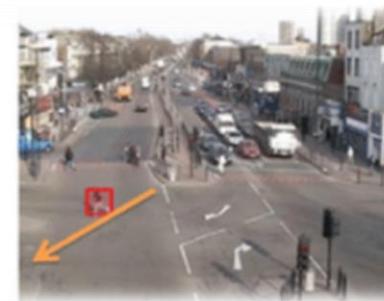
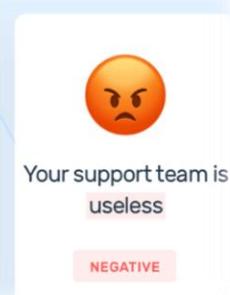
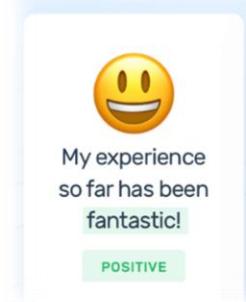
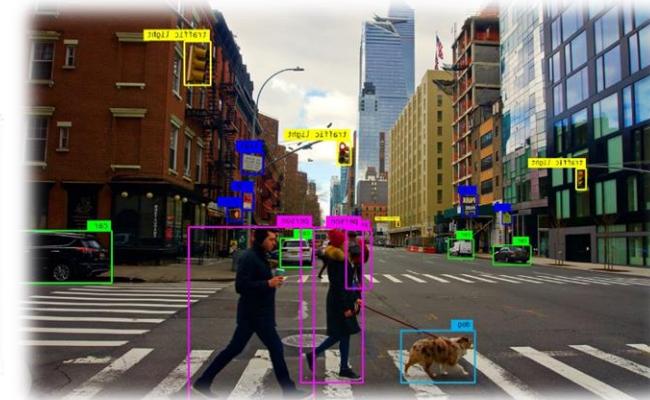
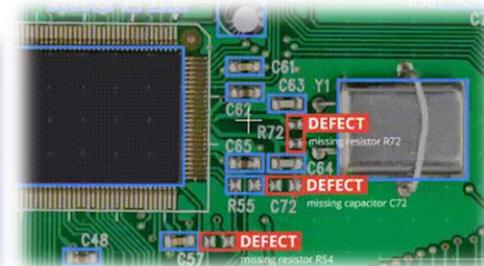
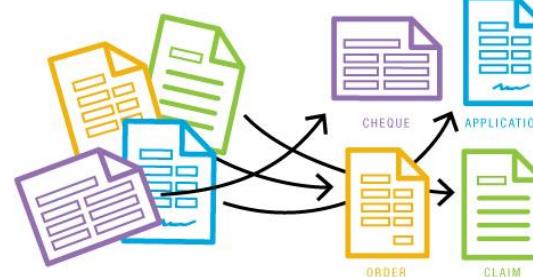




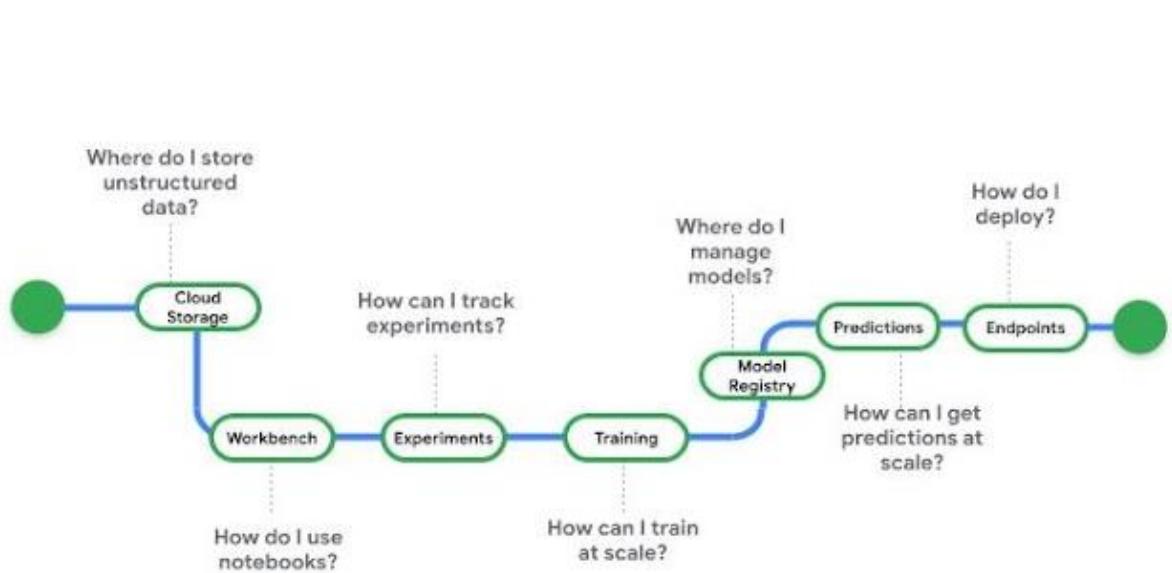
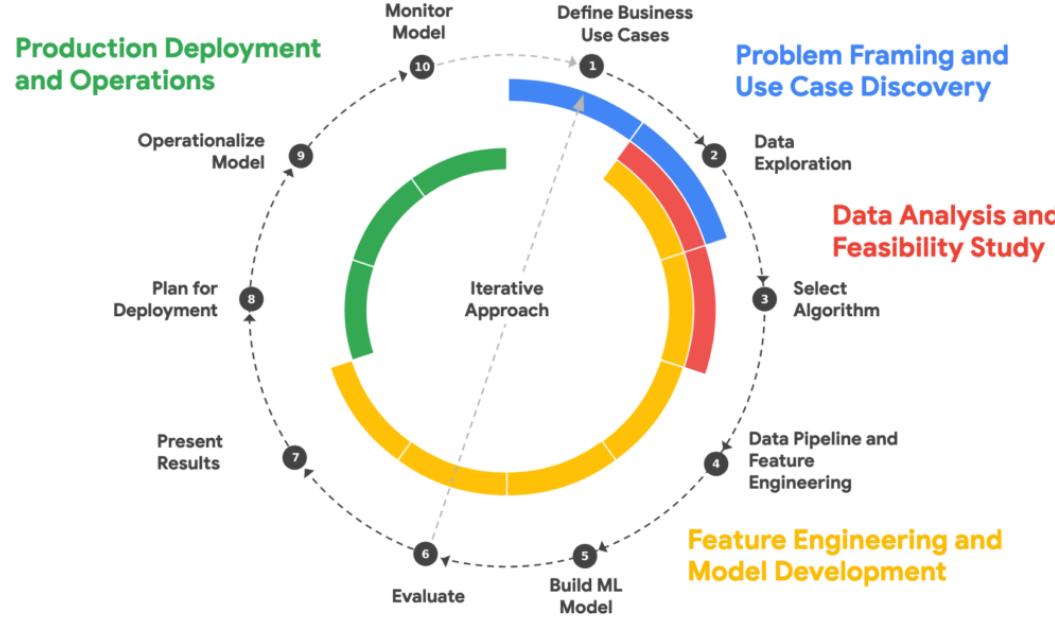
# Vertex AI Introduction



## > Hello world



## > The Prototype to Production Journey





## > What is Vertex AI & Why does a unified platform matter

### AI and Machine Learning →



#### Vertex AI

Unified platform for training, running, and managing ML models.



#### Vertex AI Workbench

Single interface for the entire Data Science workflow.



#### AI Infrastructure

Options for training deep learning and ML models cost-effectively.



#### AutoML

Custom machine learning model development, with minimal effort.



#### Natural Language AI

Sentiment analysis and classification of unstructured text.



#### Speech-to-Text

Speech recognition and transcription across 125 languages.



#### Text-to-Speech

Speech synthesis in 220+ voices and 40+ languages.



#### Translation AI

Language detection, translation, and glossary support.



#### Video AI

Video classification and recognition using machine learning.



#### Vision AI

Custom and pre-trained models to detect emotion, text, and more.



#### Dialogflow

Lifelike conversational AI with state-of-the-art virtual agents.



# Automated Machine Learning



## > Demo An image classifier “in few minutes”

**Vertex AI**

**TOOLS**

- Dashboard
- Workbench
- Pipelines**

**DATA**

- Feature Store
- Datasets
- Labeling tasks

**MODEL DEVELOPMENT**

- Training
- Experiments
- Metadata

**DEPLOY AND USE**

- Model Registry
- Endpoints
- Batch predictions
- Matching Engine

**hello-vertex-ai-pipeline-20220721065514**

Runtime Graph | 4/4 steps completed | Expand Artifacts | 100% |

**image-dataset-create**  
gcr.io/...ine-components:1.0.14

**automl-image-training-job**  
gcr.io/...ine-components:1.0.14

**endpoint-create**  
gcr.io/...ine-components:1.0.14

**model-deploy**  
gcr.io/...ine-components:1.0.14

**Execution Info** Completed

**VIEW JOB** **VIEW LOGS**

**Pipeline run analysis**

**SUMMARY** **NODE INFO**

**Display name** image-dataset-create  
**Name** image-dataset-create  
**Type** system.ContainerExecution  
**Duration** 26 min 4 sec  
**Started** Jul 21, 2022, 8:55:19 AM  
**Completed** Jul 21, 2022, 9:21:23 AM

**Input Parameters**

Parameter	Type	Value
data_item_labels	string	{}
display_name	string	vertex_pipeline_binary_classif_birdsw
gcs_source	string	gs://oi_labimages/birdswing_lab_bin
import_schema_uri	string	gs://google-cloud-aiplatform/schema/dataset/ioformat
labels	string	{}
location	string	us-central1
project	string	visual-inspection-ic

**Output Parameters**

Parameter	Type
No parameters to display	



# Manual AI Development



## > The Development Environment Vertex AI Workbench

The screenshot shows the Vertex AI Workbench interface with a Jupyter Notebook open. The left sidebar displays a file tree with an 'ipynb' search bar, showing files like 'src', 'test', 'tutorials', 'demo\_2.ipynb' (selected), and 'demo.ipynb'. The main area has three tabs: 'demo.ipynb', 'Terminal 1', and 'demo\_2.ipynb' (active). The notebook content includes:

- Library installation & imports**:

```
[9]: import pandas as pd
import plotly.express as px

pd.options.plotting.backend = 'plotly'
```
- Overview of the data**:

```
[6]: df['Machine'].unique().tolist()
```

```
[6]: ['P12', 'P1', '1_1', 'P11', '2_3', None, 'PressCrown']
```
- Rows with part of asset name null**:

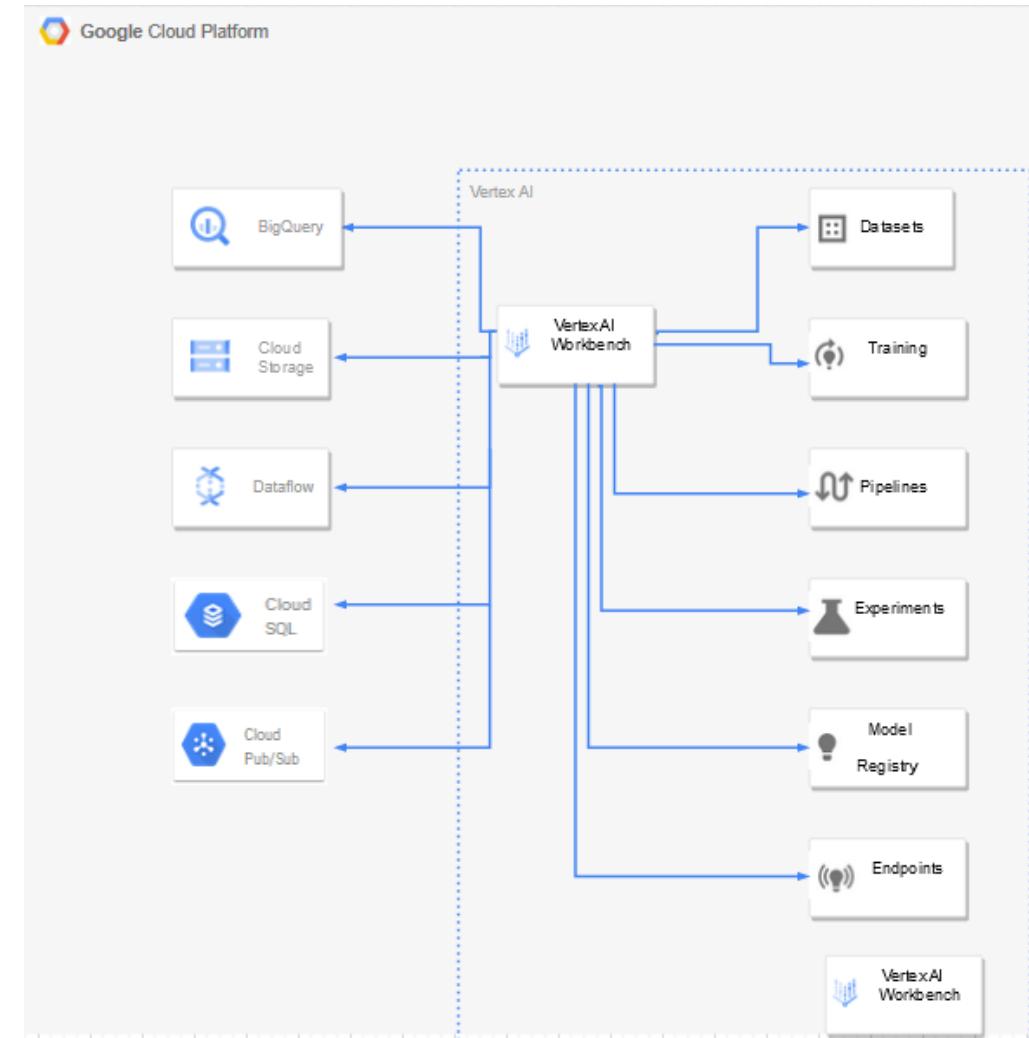
```
[7]: # Machine null
print (f"Rows with Area = None: {len(df[df['Area'].isnull()])}")
print (f"Rows with Department = None: {len(df[df['Department'].isnull()])}")
print (f"Rows with Line = None: {len(df[df['Line'].isnull()])}")
print (f"Rows with Cell = None: {len(df[df['Cell'].isnull()])}")
print (f"Rows with Machine = None: {len(df[df['Machine'].isnull()])}")
print (f"Rows with Station = None: {len(df[df['Station'].isnull()])}")

df_name_part_none = df.loc[(df['Department'].isnull() | df['Area'].isnull() | df['Line'].isnull() | df['Cell'].isnull() | df['Machine'].isnull() | df['Station'].isnull())]
print (f"Rows with part of asset name = None: {len(df_name_part_none)}")
```

At the bottom, the status bar shows 'Simple' mode, 'Python 3 | Idle', 'Mode: Command', 'Ln 1, Col 1', and 'demo\_2.ipynb'.



## > Vertex AI Workbench Use other GCP services





## > Creating Vertex AI Workbench

The screenshot shows the Google Cloud Vertex AI Workbench creation interface. On the left, there's a sidebar with various options like Dashboard, Datasets, Feature Store, Labeling tasks, Pipelines, Training, Experiments, Model Registry, Endpoints, Batch predictions, Metadata, Matching Engine, and Marketplace. The 'Workbench' option is currently selected. The main area is titled 'Create a user-managed notebook'. It has fields for 'Environment' (set to 'TensorFlow Enterprise 2.10 (Intel® MKL-DNN/MKL)'), 'Select a script to run after creation' (with a 'BROWSE' button), and 'Custom metadata' (with a '+ ADD ITEM' button). The 'Machine configuration' section is open, showing a dropdown menu for 'Machine type'. The 'a2-ultragpu-8g' option is highlighted with a red box. Other visible machine types include N1 standard, A2 highgpu, A2 megagpu, A2 ultragpu, C2D high-CPU, C2D high-memory, and C2D standard. At the bottom of the configuration section, there are checkboxes for 'Turn on Integrity Monitoring' (which is checked) and 'Disk(s)'.



# > Creating Vertex AI Workbench

## Google Cloud CLI Command line

```
gcloud notebooks instances create INSTANCE_NAME
  --vm-image-project=deeplearning-platform-release
  --vm-image-family=VM_IMAGE_FAMILY
  --machine-type=MACHINE_TYPE
  --location=LOCATION
```

## REST Request

### HTTP request

POST <https://notebooks.googleapis.com/v1/{parent}/instances>

```
1 {
2   "machineType": "n1-standard-4",
3   "instanceOwners": [
4     "rbanzo@ford.com"
5   ],
6   "vmImage": {
7     "project": "deeplearning-platform-release",
8     "imageFamily": "tf-ent-2-8-cu113-notebooks"
9   },
10  "metadata": {
11    "framework": "TensorFlow:2.8",
12    "notebooks-api": "PROD",
13    "report-system-health": "true",
14    "title": "TensorFlow2.8/Keras.CUDA11.3.GPU"
15  }
16}
```

## Terraform

```
1 resource "google_notebooks_instance" "test-notebook" {
2   name          = "test-notebook"
3   project       = "test-project"
4   location      = "us-central1-a"
5   machine_type  = "n1-standard-1"
6   install_gpu_driver = false
7   instance_owners = ["rlbo@gft.com"]
8   vm_image {
9     project      = "deeplearning-platform-release"
10    image_family = "pytorch-latest-cpu"
11  }
12   metadata = {
13     terraform = "true"
14   }
15 }
```

## gRPC

### CreateInstance

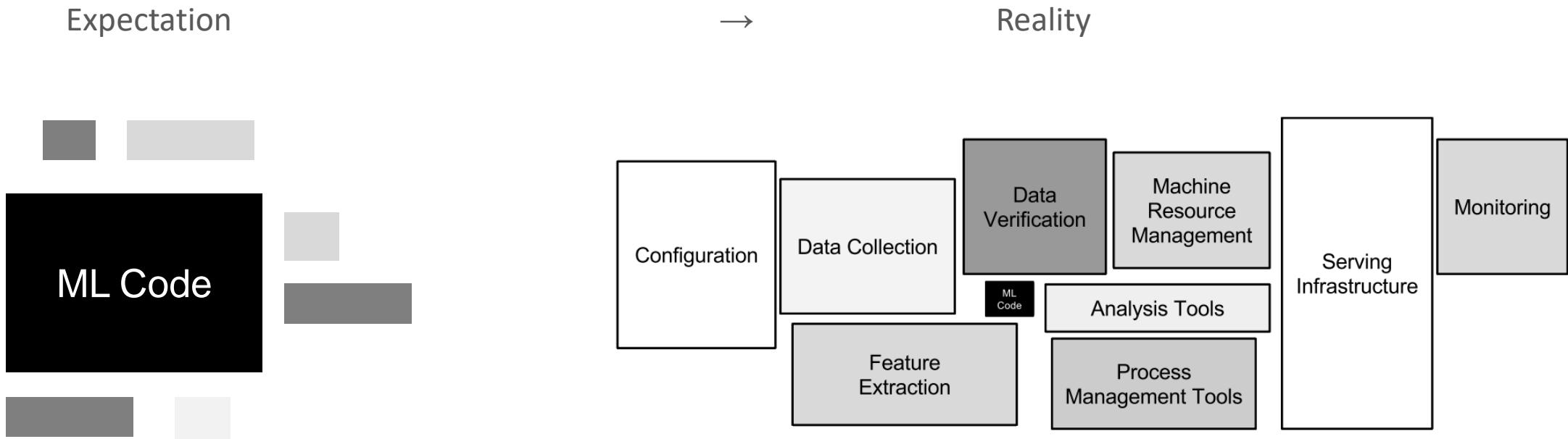
rpc CreateInstance(CreateInstanceRequest) returns (Operation)

Creates a new Instance in a given project and location.

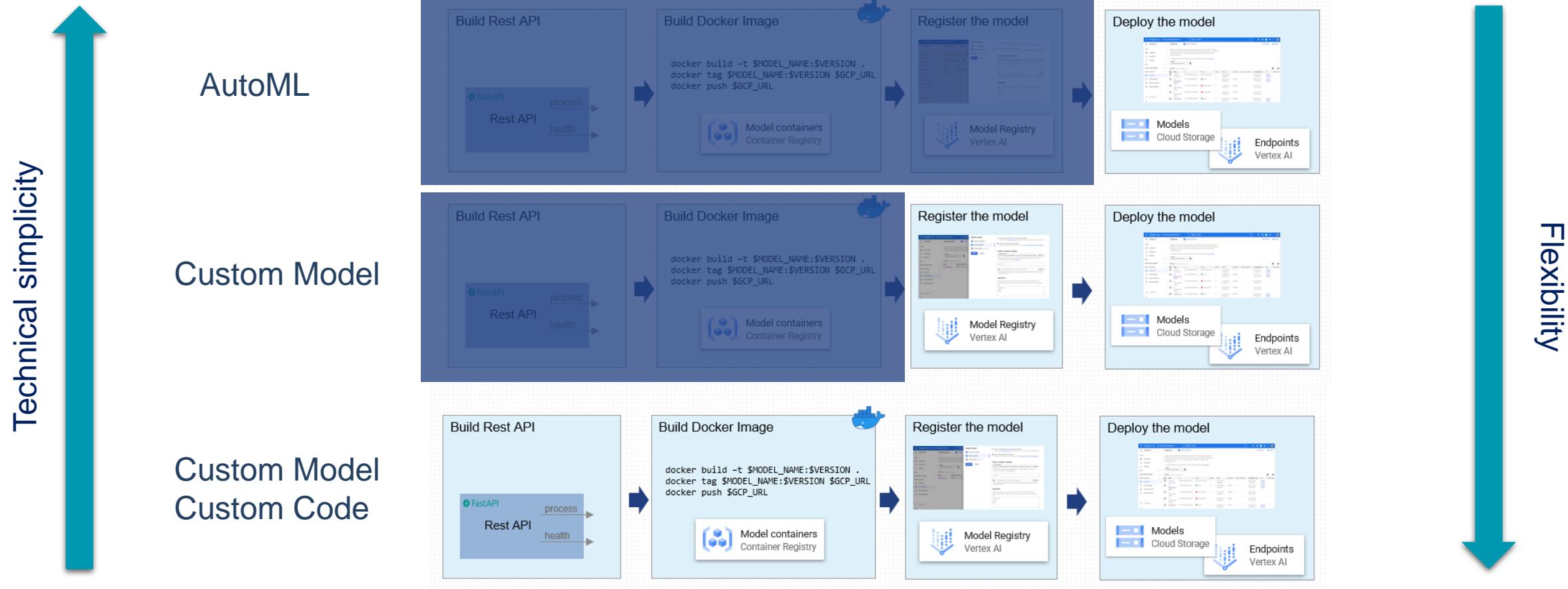


# AI Model Deployment

## > AI model deployment in Vertex AI



## > AI model deployment in Vertex AI



# > AI model deployment in Vertex AI

The screenshot shows the Google Cloud Vertex AI Model Registry interface. On the left, there's a sidebar with various tools like Dashboard, Workbench, Pipelines, and Model Registry. The Model Registry section is active, showing a table with columns Name and Deployment status. One row is selected, showing 'ai-segmentation...' and 'Deployed on Vertex AI'. The main area is titled 'Import model' with three steps: 1. Name and region (selected), 2. Model settings, and 3. Explainability (optional). Step 1 has a sub-section 'Custom container settings' where a Docker image URL is specified: 'Container image: gcr.io/cam-seatingweldhealth/ai-segmentation-model@sha256:6e2908f'. Below this are fields for 'Command' and 'Model artifact location (Cloud storage path)'. Step 2 is titled 'Arguments' with a text input containing '-flag\_a=xxx -flag2 flag3'.

→ Docker image URL

The screenshot shows the Google Cloud Vertex AI Endpoints interface. The sidebar includes Dashboard, Workbench, Pipelines, Model Registry, Batch predictions, Matching Engine, and Marketplace. The Endpoints section is active, showing a table with rows for 'ai-segmentation-model', 'ai-segmentation-model', 'ai-segmentation-model', and 'endpoint-start'. A 'CREATE ENDPOINT' button is visible. The main area shows a 'New endpoint' form with two steps: 1. Define your endpoint and 2. Model settings. Step 1 is active, showing fields for 'Region' (set to 'northamerica-northeast2'), 'Name' (set to 'ai-segmentation-model'), and 'ID'. Step 2 is partially visible.

← Resources

This part of the screenshot highlights the 'New endpoint' configuration page. It shows the 'New endpoint' form with 'Define your endpoint' and 'Model settings' steps. The 'Model settings' step is expanded, showing 'Advanced scaling options' with fields for 'Machine type' (set to 'n1-standard-4, 4 vCPUs, 15 GiB memory') and 'Service account' (set to 'Compute Engine default service account'). An 'ADVANCED SCALING OPTIONS' section is also shown. To the right, a large 'Scalability' heading with an upward arrow is positioned above the scaling settings, and a large 'Resources' heading with a leftward arrow is positioned below the machine type and service account fields.



# Success Cases



> We used Vertex AI and... It was a success!



LAZARD  
ASSET MANAGEMENT



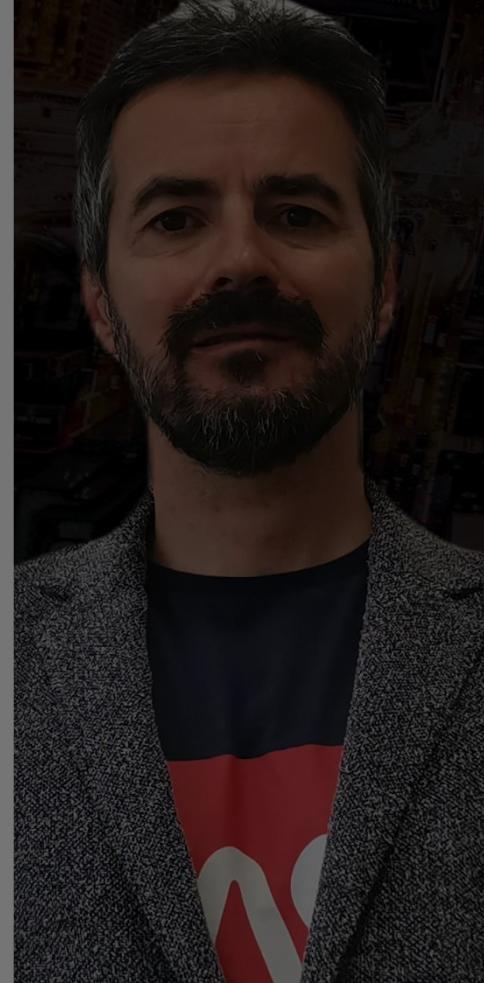
Pilar  
Madariaga

Data Scientist



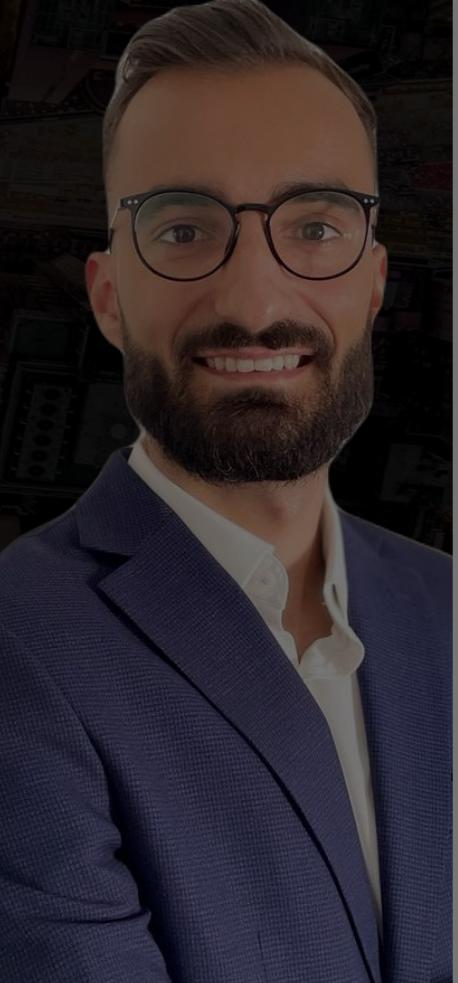
Rafael  
Banzo

Data Scientist



Daniel  
Ruiz

Data Scientist



GFT □ > GFTmeets



Live  
Q&A



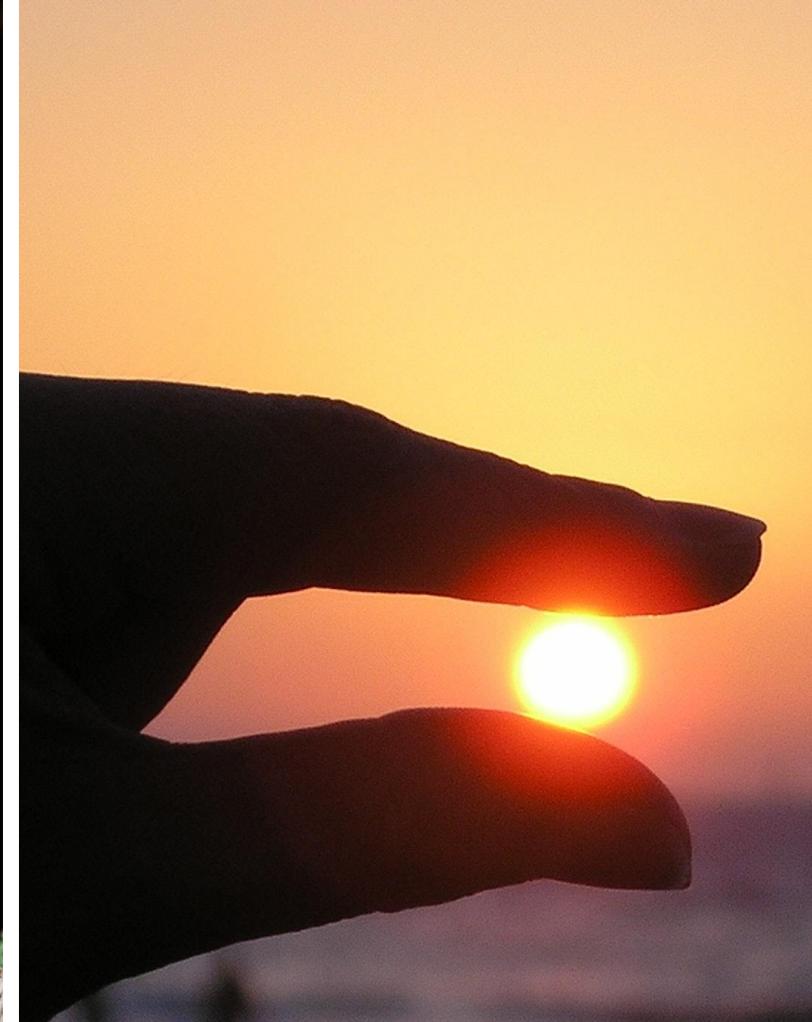
## Inclusive Coding

Monday, Dec 12 - 9:30



SPECIAL XMAS EDITION 2022

Friday, Dec 16 - 12:30



La dificultad en perspectiva

Tuesday, Jan 17 - 12:30