

Lecture 17: App Design, Setup & Code Organization

AC215

Shivas Jayaram



Outline

1. Recap
2. Motivation
3. App Design
4. Screenflow & Wireframes
5. Solution Architecture
6. Technical Architecture
7. Setup & Code Organization

Outline

1. **Recap**
2. Motivation
3. App Design
4. Screenflow & Wireframes
5. Solution Architecture
6. Technical Architecture
7. Setup & Code Organization

Recap: 🍄 Cheese App

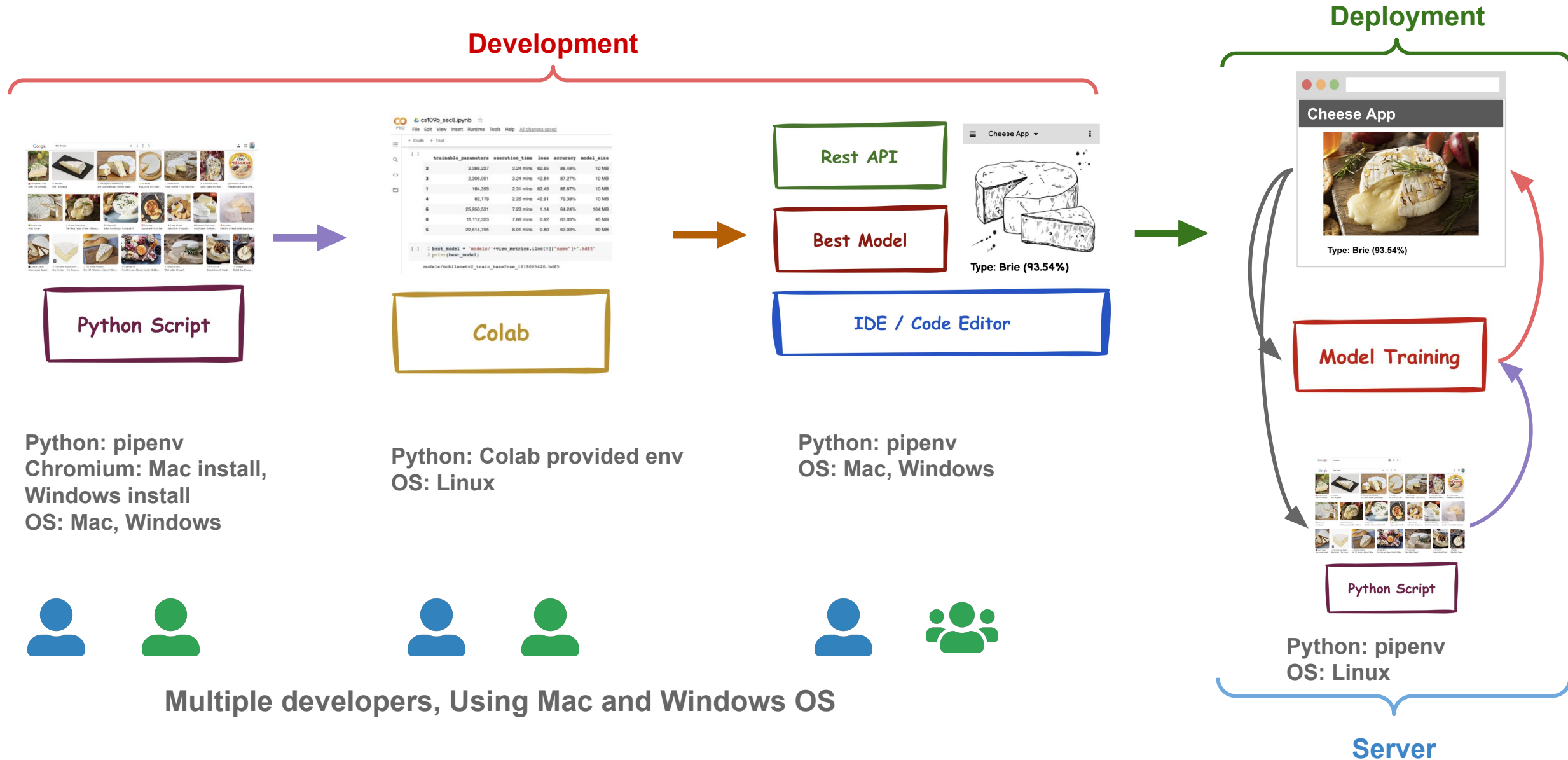
- We want to build an app to identify a cheese by simply taking a photo of it
- Dive deeper into the world of cheese with our interactive chatbot
- How do we build the app?



Recap: How do we build an App?

- Collaborate with team to **design** and **develop**.
- Build a robust **ml pipeline** for **data** and **models**.
- Expose python functions as **backend APIs**.
- Build a **frontend** using HTML & javascript.
- **Deploy** app to a cloud provider.
- <https://formaggio.me/> [Go live]

Recap: How do we build an App?



Recap: Tools

Data:

- Google Cloud Storage
- TensorFlow Data / Records
- Label Studio
- DVC
- ChromaDB

Model:

- Gemini
- Vertex AI Fine Tuning / Training
- Vertex AI Deploy
- W&B

Operations:

- GitHub
- Docker
- Vertex AI Pipelines
- GCP
- Modal

Outline

1. Recap
- 2. Motivation**
3. App Design
4. Screenflow & Wireframes
5. Solution Architecture
6. Technical Architecture
7. Setup & Code Organization

Before you build your App

- Our **ML Pipeline** is ready
- We want to build an app that uses the **ML Components**
- Expose model and python functions as **APIs**
- Identify **user needs** that can fulfilled by APIs
- Design **user interface** needs

How do we do this?

Review: Problem Definition

Imagine being able to identify a cheese by simply taking a photo of it. Our app uses AI-powered visual recognition technology to help you identify the cheese you're looking at, and then provides you with a wealth of information about it.

Take a [photo of the cheese](#), and our app will identify it for you. Then, dive deeper into the world of cheese with our interactive chatbot. Ask questions about the cheese's origin, production process, nutritional information, and history.

Review: Proposed Solution

Key Features:

- Visual cheese identification using AI-powered technology
- Interactive chatbot for asking questions about cheese
- In-depth information on cheese origin, production process, nutritional information, and history
- Expert advice on pairing cheese with wines, crackers, and other accompaniments
- Perfect for cheese enthusiasts, party planners, and anyone looking to explore the world of cheese

Review: Project Scope



Proof Of Concept (POC)

- Scrap cheese images and documents (books etc)
- Verify images and pdfs
- Experiment on some baseline models
- Verify new unseen cheeses are predicted by the model(s)
- Verify ideas using any instruct-LLMs

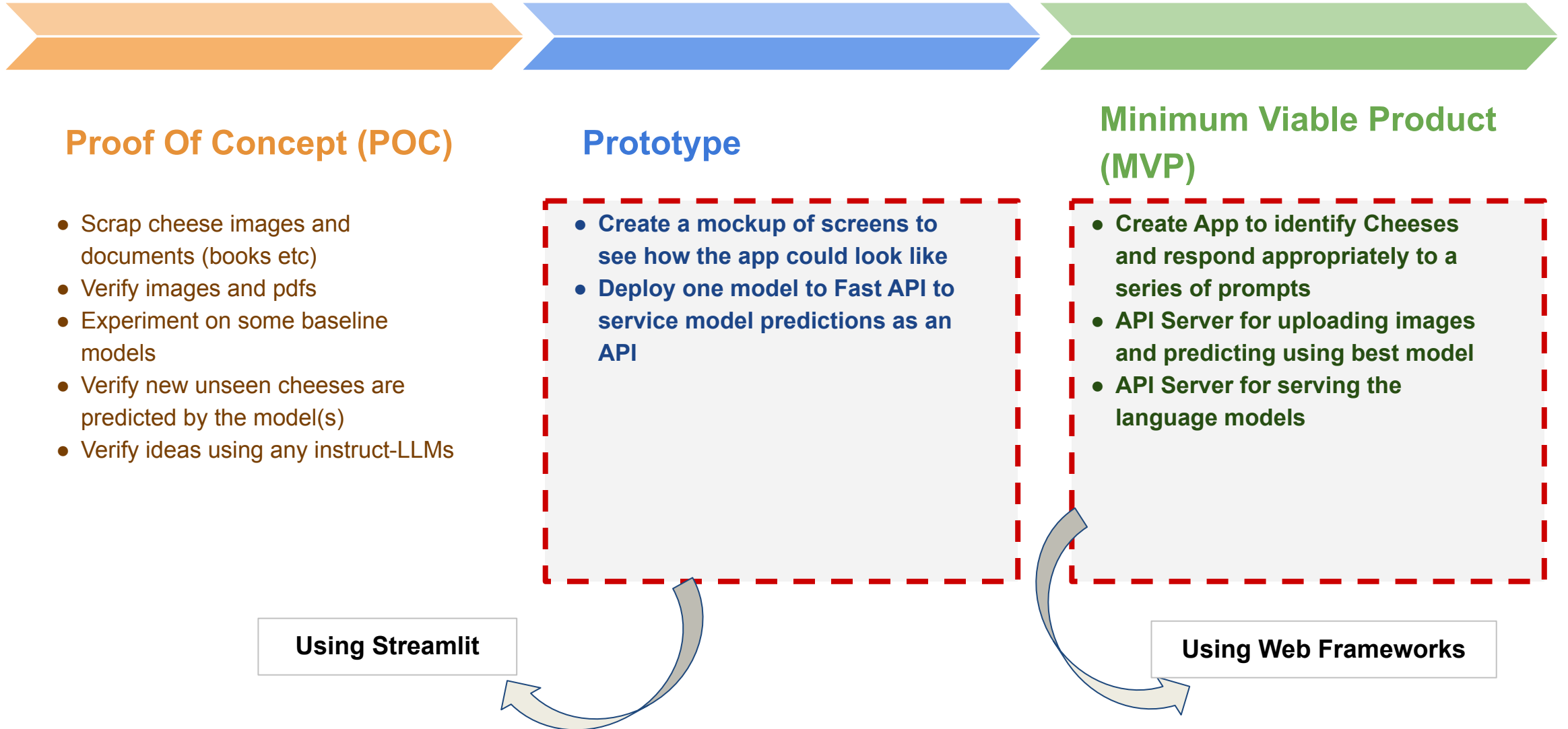
Prototype

- Create a mockup of screens to see how the app could look like
- Deploy one model to Fast API to service model predictions as an API

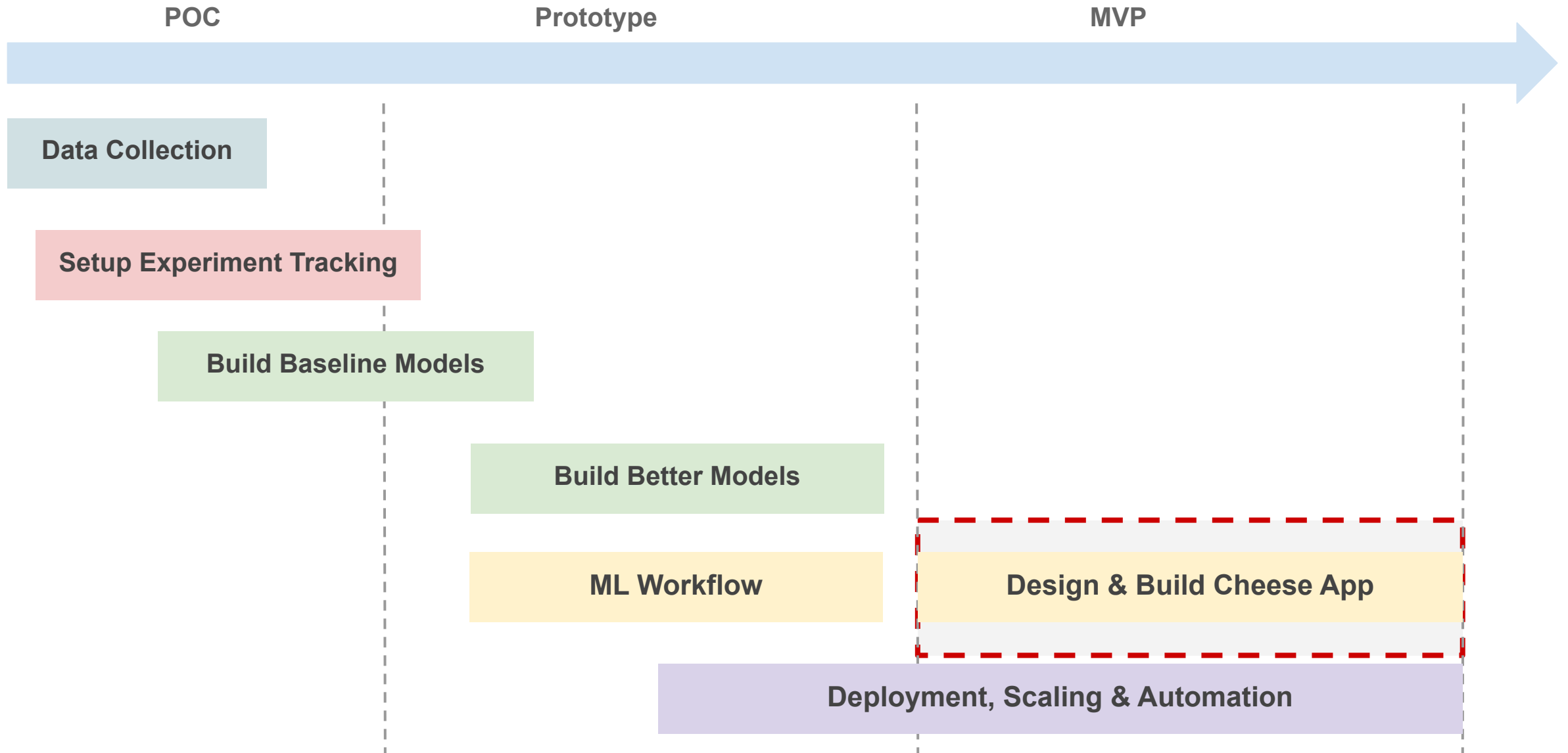
Minimum Viable Product (MVP)

- Create App to identify Cheeses and respond appropriately to a series of prompts
- API Server for uploading images and predicting using best model
- API Server for serving the language models

Review: Project Scope

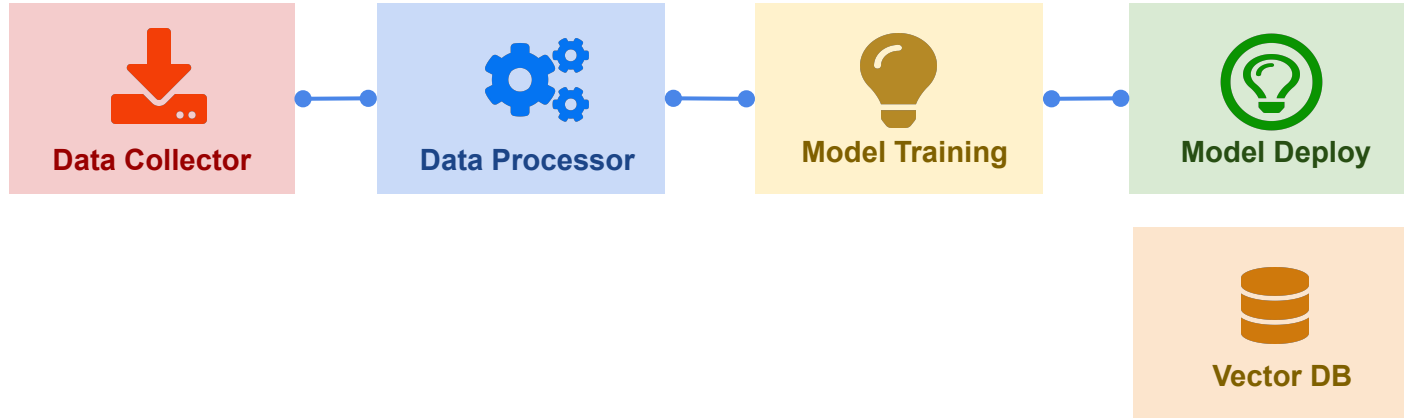


Review: Cheese App Status



Cheese App Development

ML Pipeline



App Dev



Google Cloud Platform



Cloud Storage



Vertex AI



Compute Engine

Outline

1. Recap
2. Motivation
- 3. App Design**
4. Screenflow & Wireframes
5. Solution Architecture
6. Technical Architecture
7. Setup & Code Organization

App Design

- In a traditional software app you have code and data.
- In an **AI App**, in addition you have models to perform tasks
- We will follow a structured approach to design and develop an AI App
- The design will consist of the following components:
 - Screenflow & Wireframes
 - Solution Architecture
 - Technical Architecture

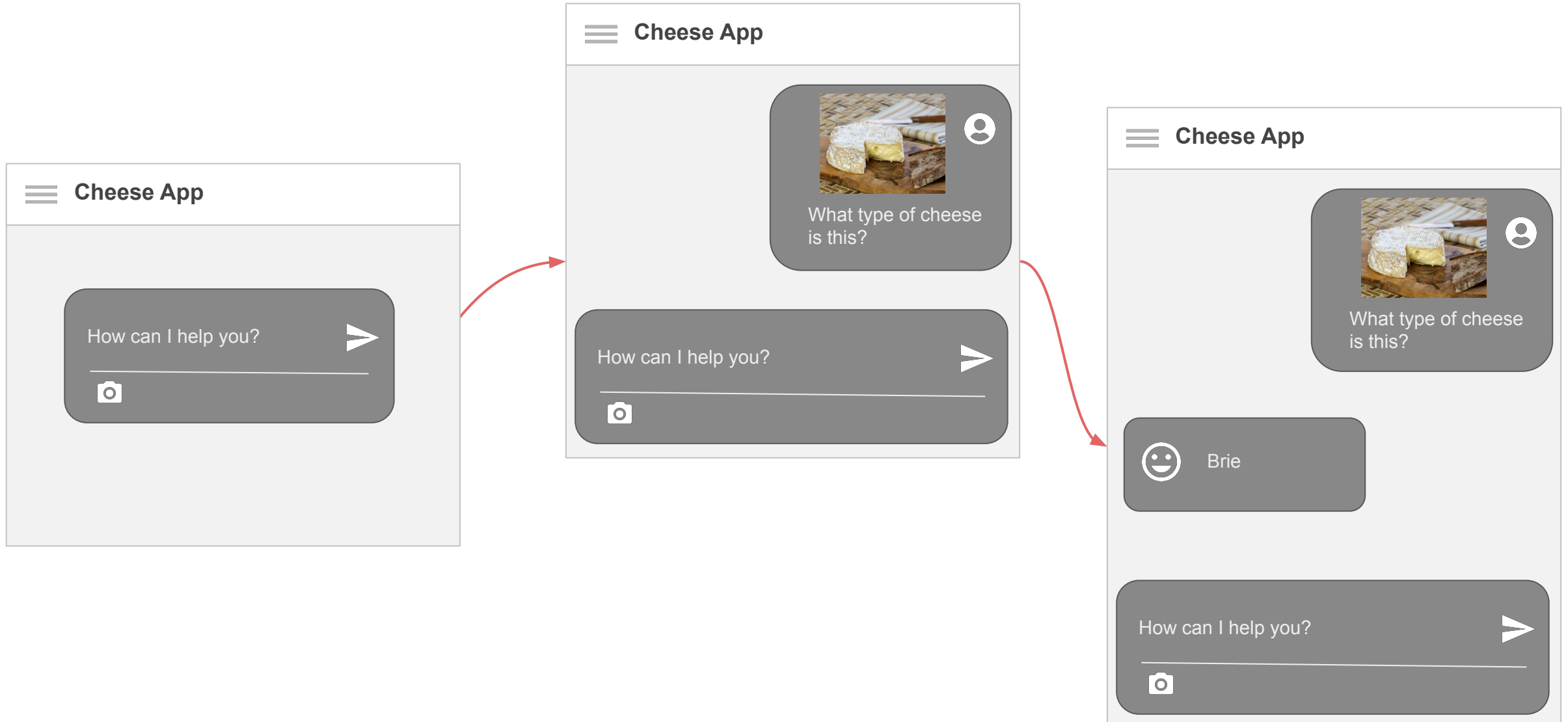
Outline

1. Recap
2. Motivation
3. App Design
- 4. Screenflow & Wireframes**
5. Solution Architecture
6. Technical Architecture
7. Setup & Code Organization

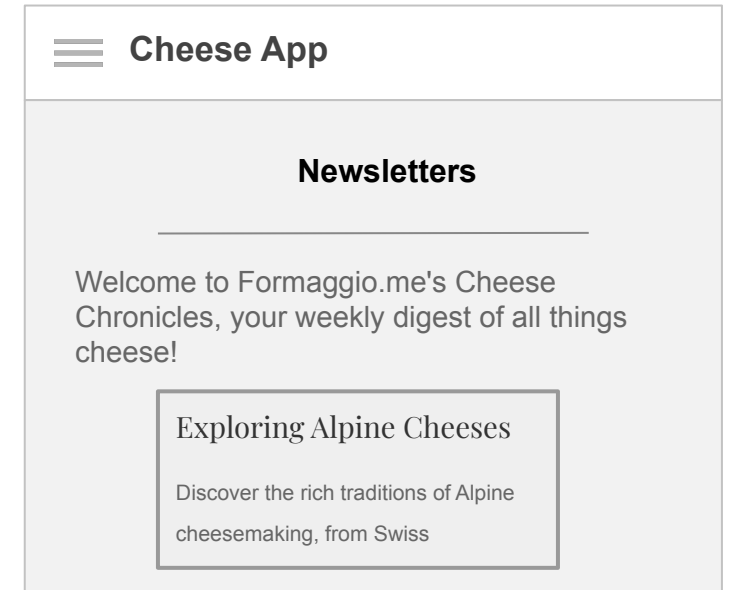
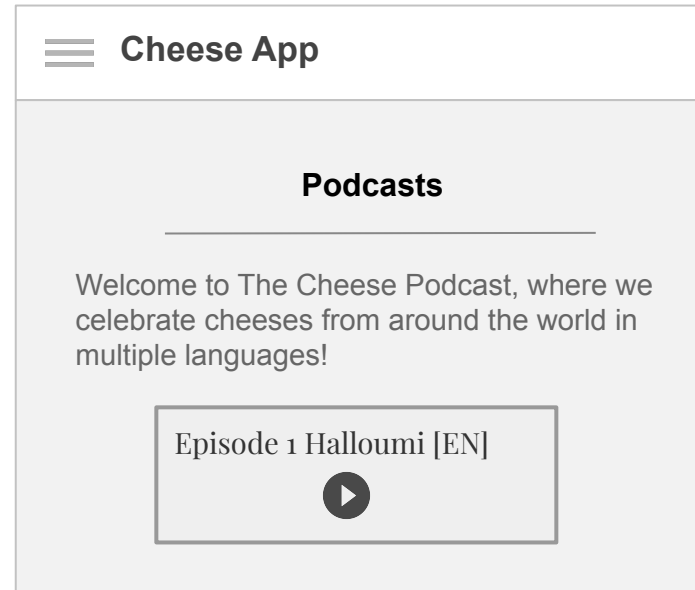
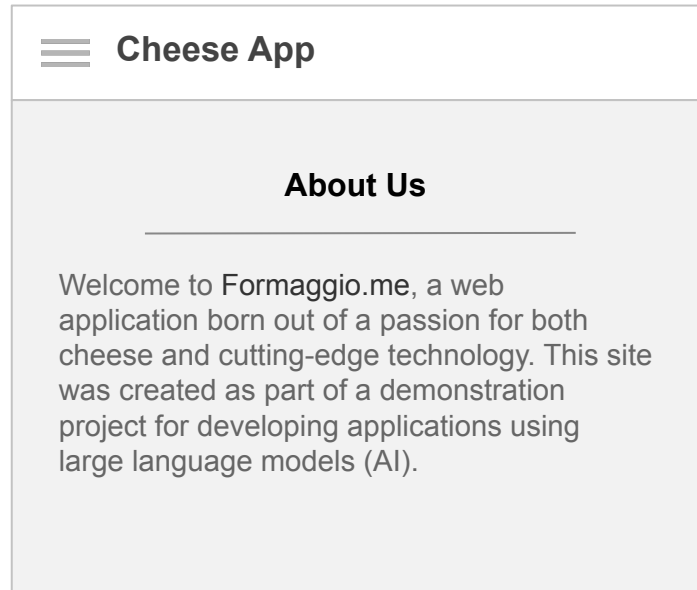
Screenflow & Wireframes

Start with brainstorming ideas on whiteboard/paper

Screenflow & Wireframes



Screenflow & Wireframes



Outline

1. Recap
2. Motivation
3. App Design
4. Screenflow & Wireframes
- 5. Solution Architecture**
6. Technical Architecture
7. Setup & Code Organization

Solution Architecture

- Helps to identify the building **blocks** in an App
- Start by asking how will your **App** address the **Problem Statement**
- Identifying the following:
 - The **Process** being performed by the user
 - The code **Execution** blocks required to fulfil the **Process**
 - The **State** required during the life cycle of the App

Solution Architecture

Process (People)

- User actions
- Admin tasks
- Data Scientist tasks
- Developer tasks

Execution (Code)

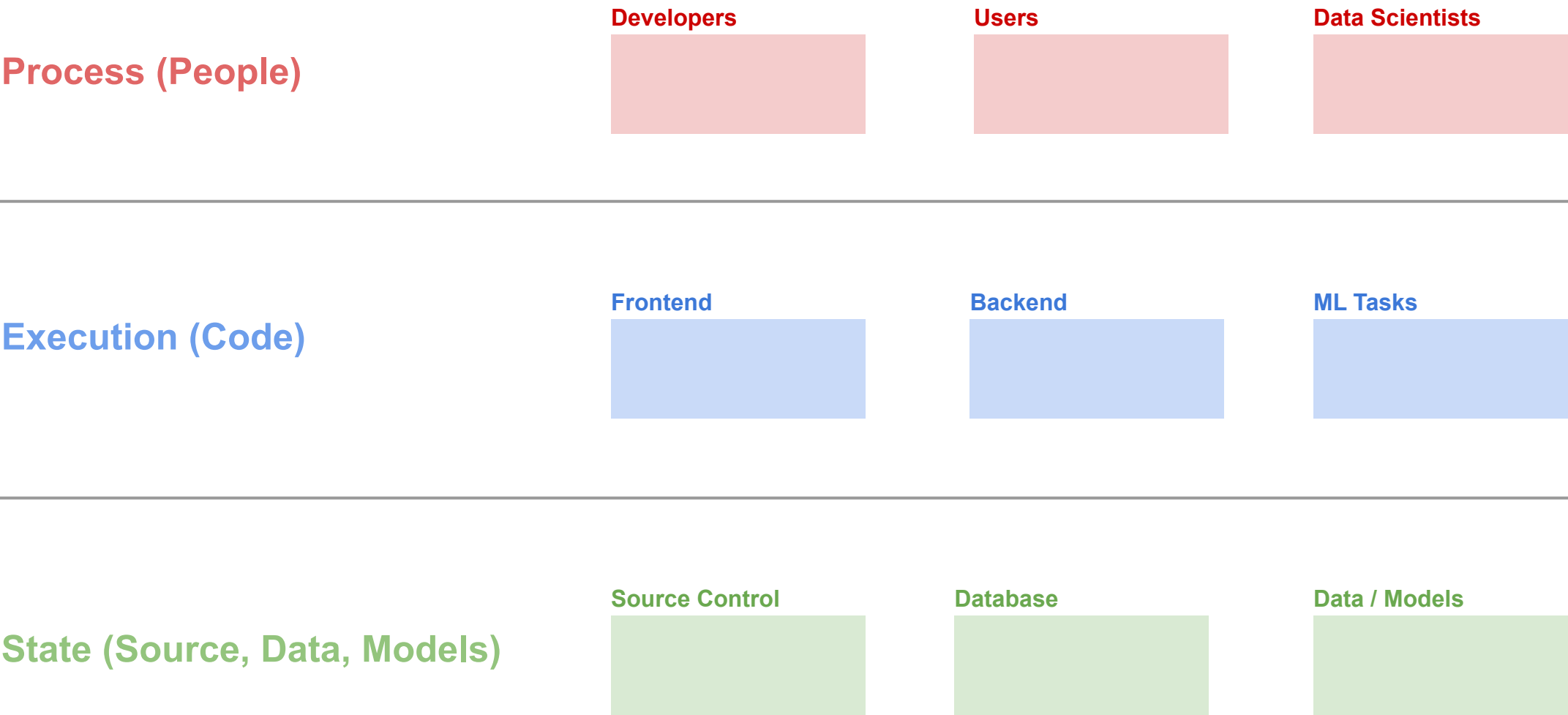
- Frontend apps
- Backend services
- Data science notebooks
- ML tasks
- LLM APIs
- Agents

State (Source, Data, Models)

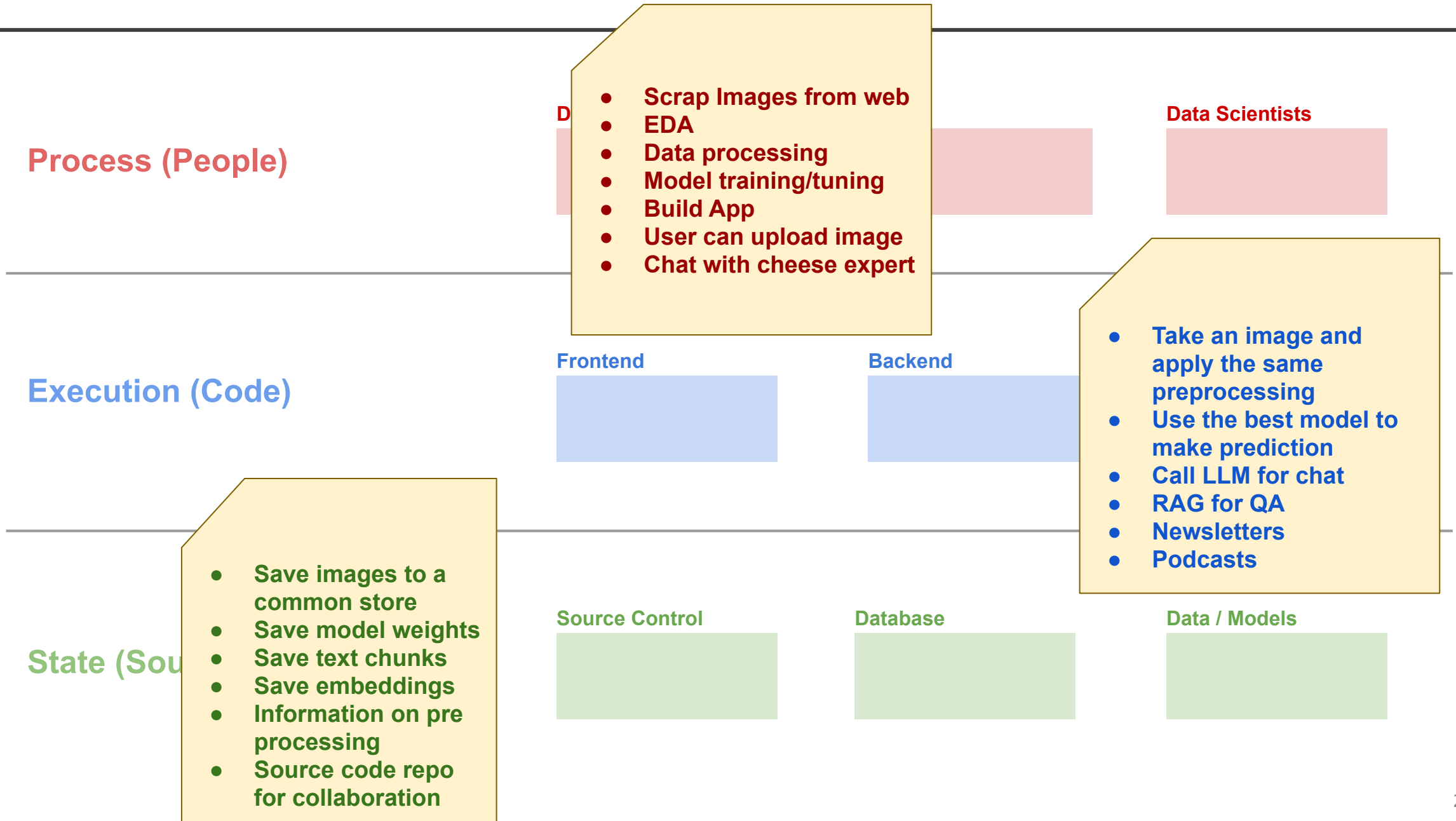
- Source control
- Artifact registry
- Data store
- Model registry
- Knowledge base

Solution Architecture

AI App

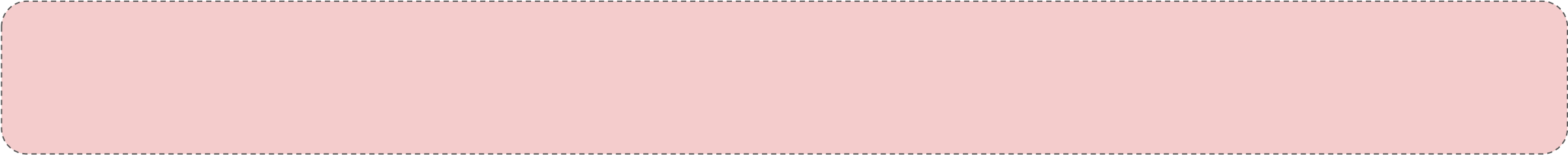


Solution Architecture



Solution Architecture

Process



Execution

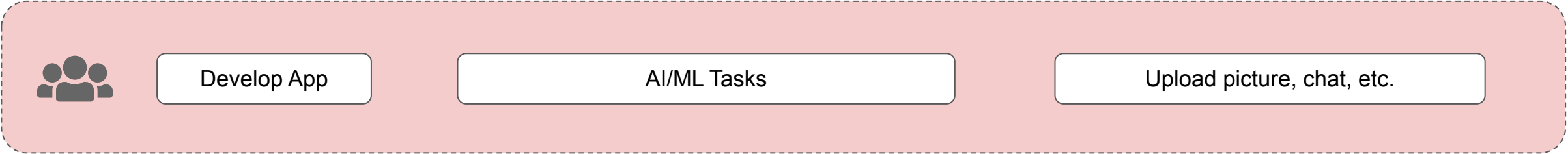


State



Solution Architecture

Process



Execution

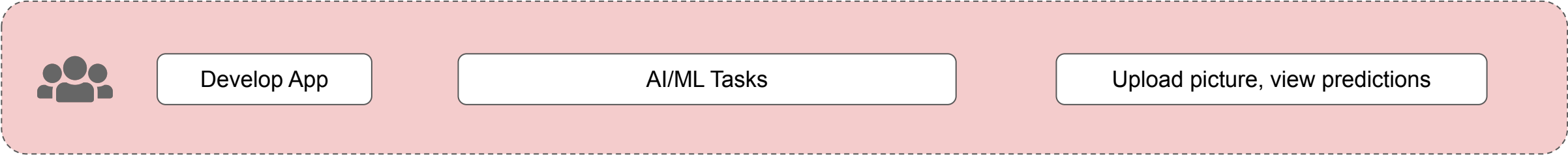


State



Solution Architecture

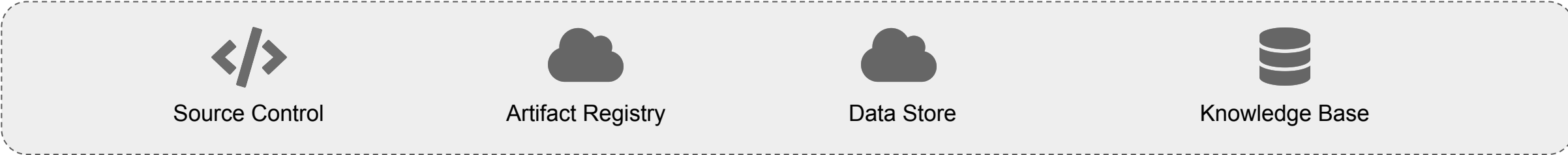
Process



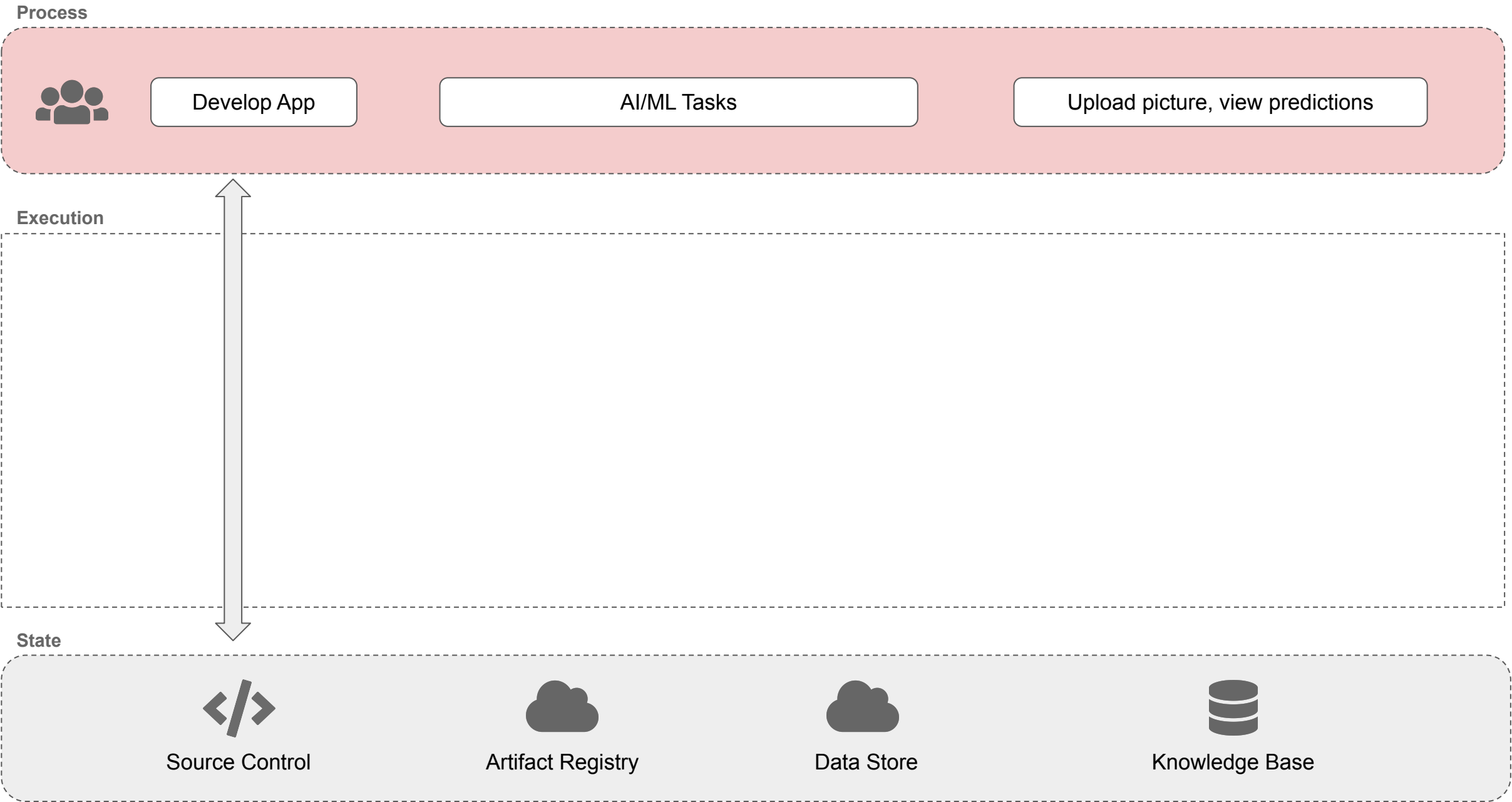
Execution



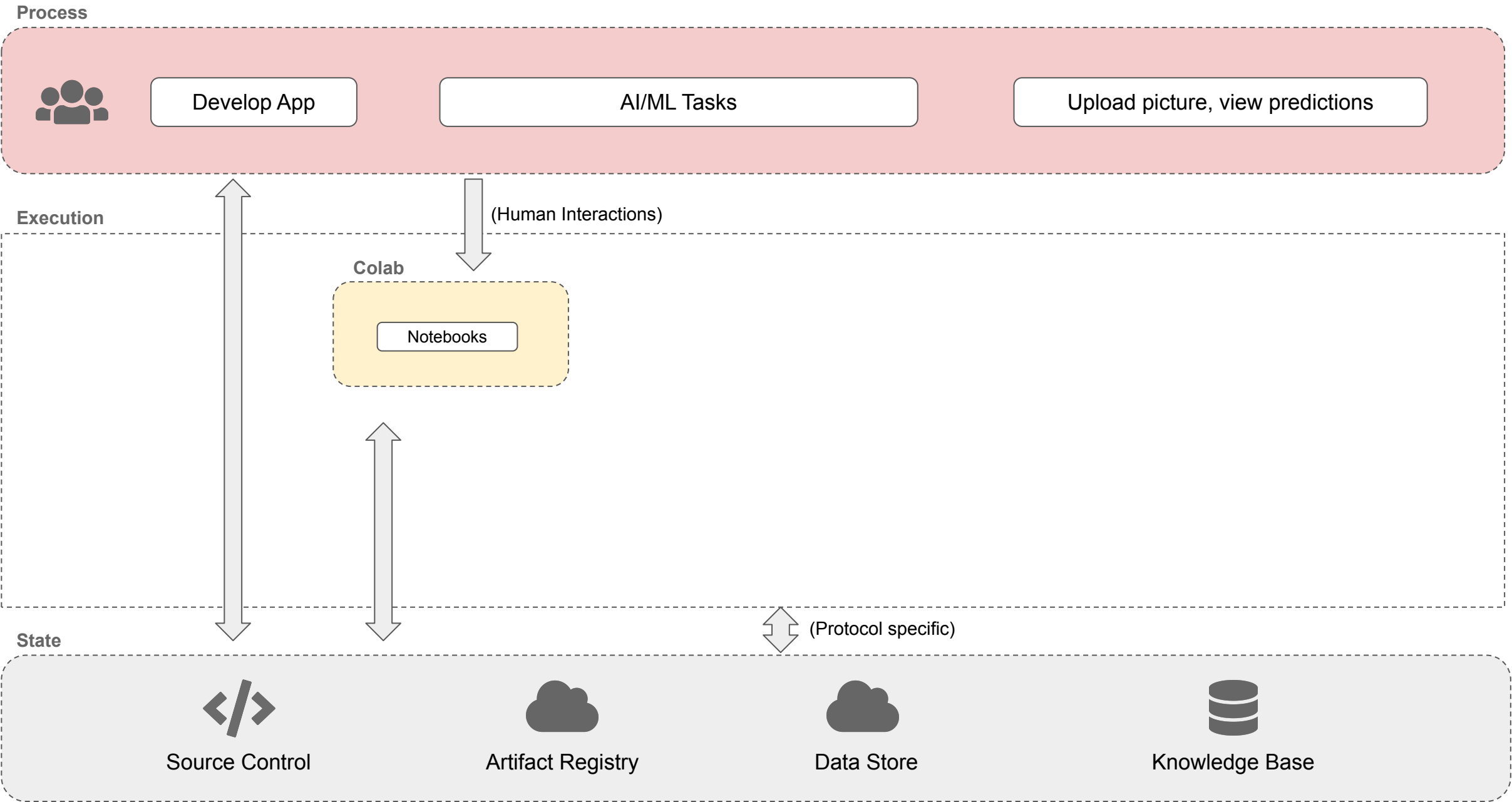
State



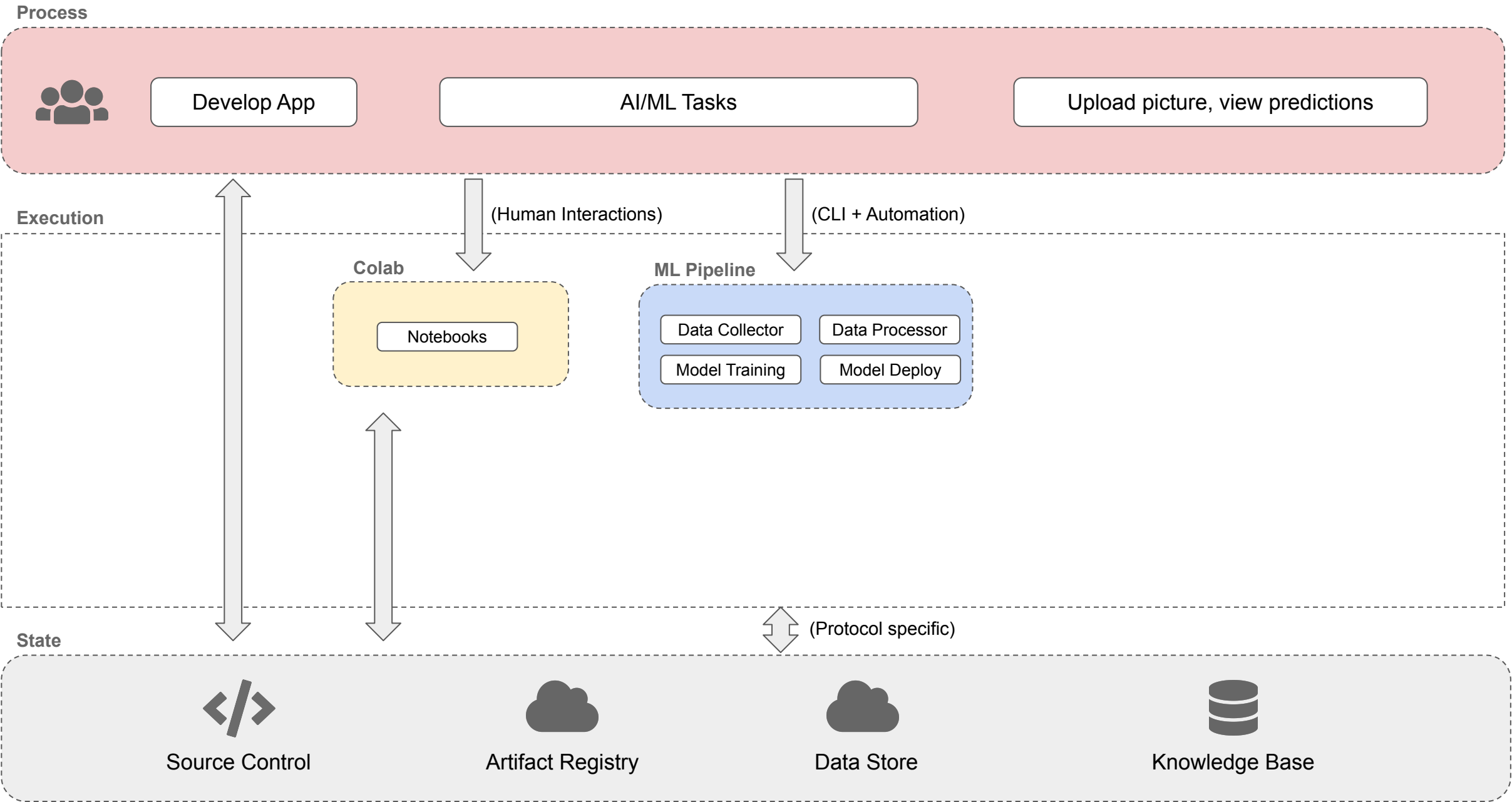
Solution Architecture



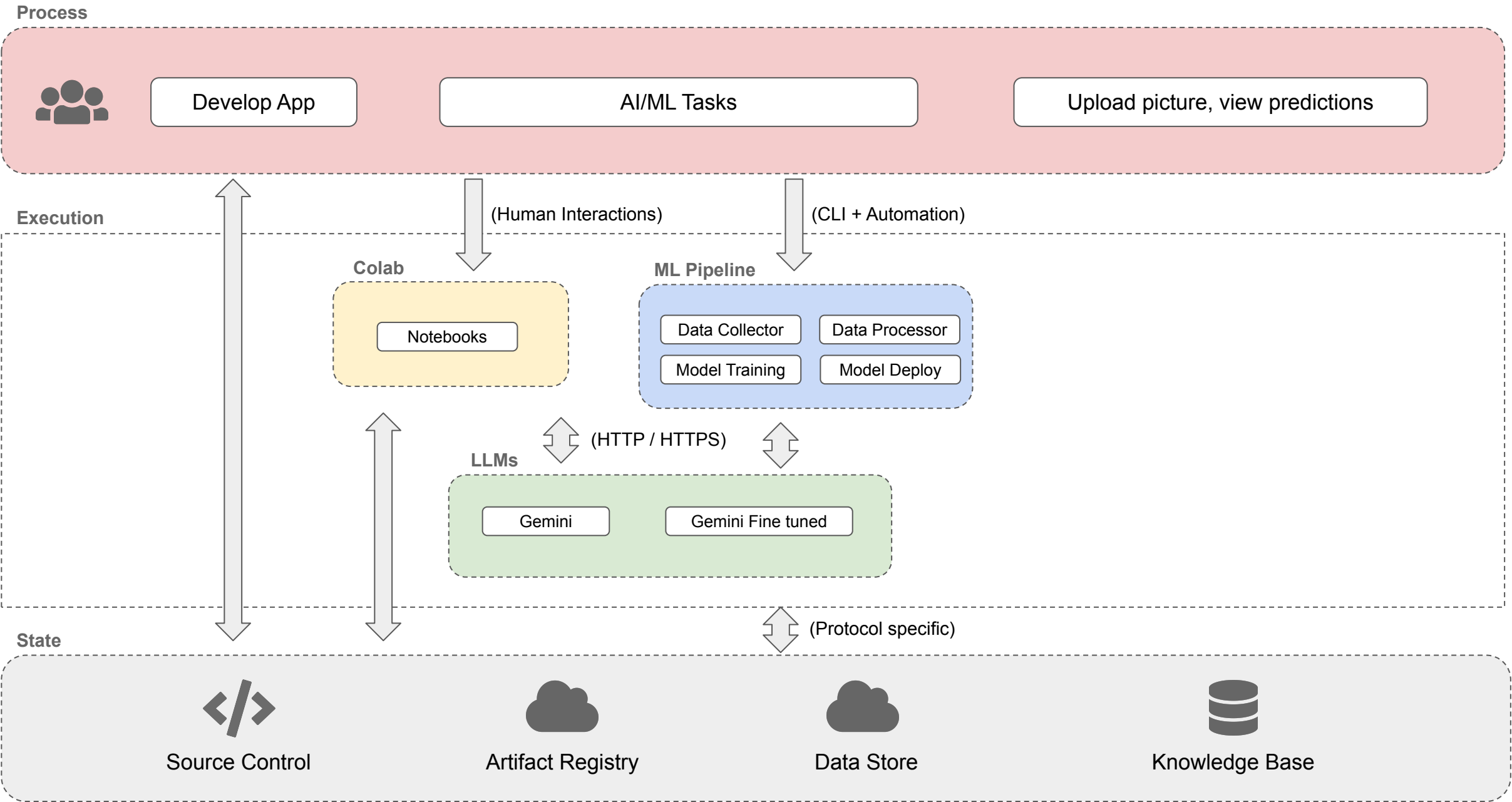
Solution Architecture



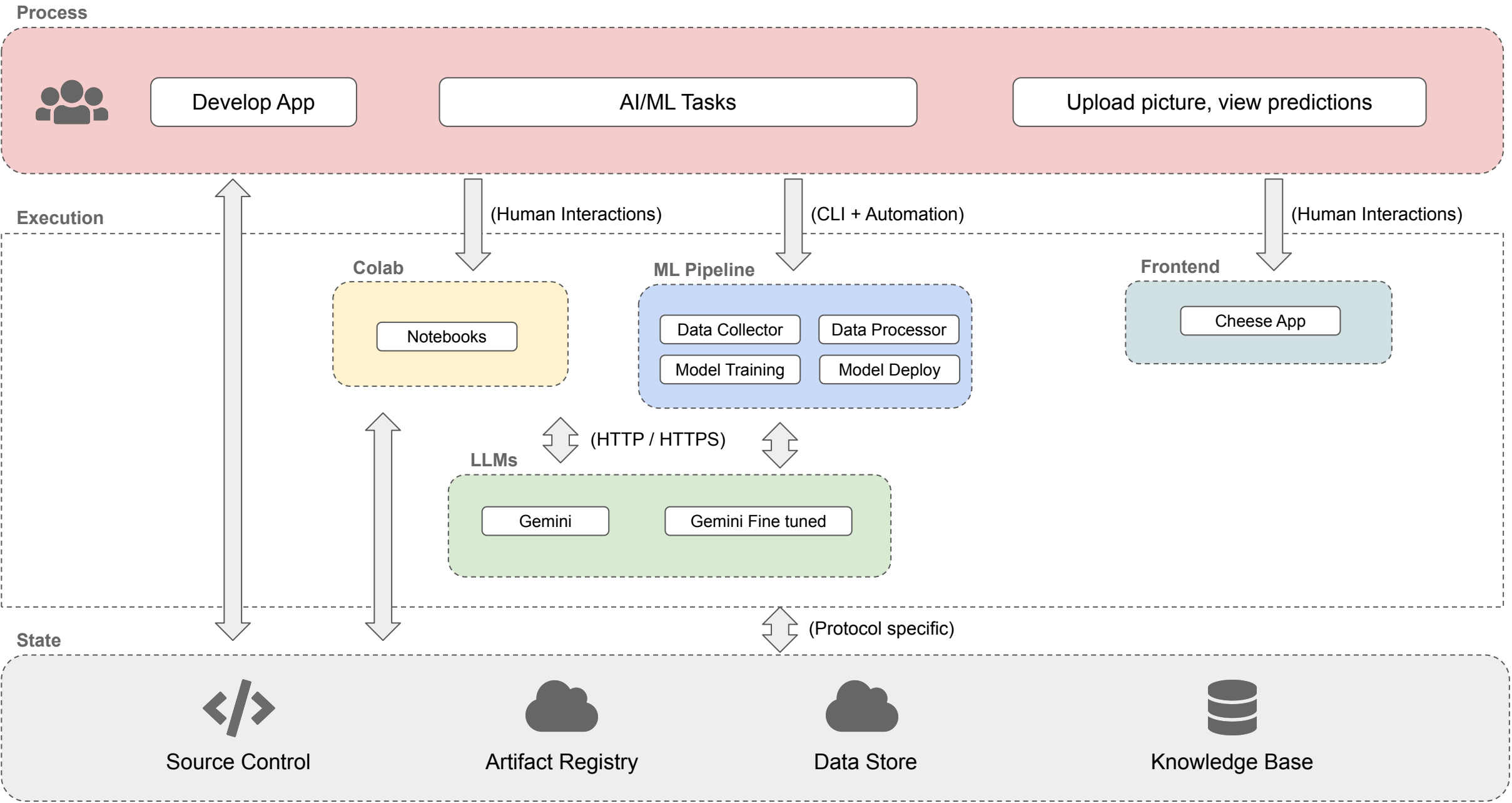
Solution Architecture



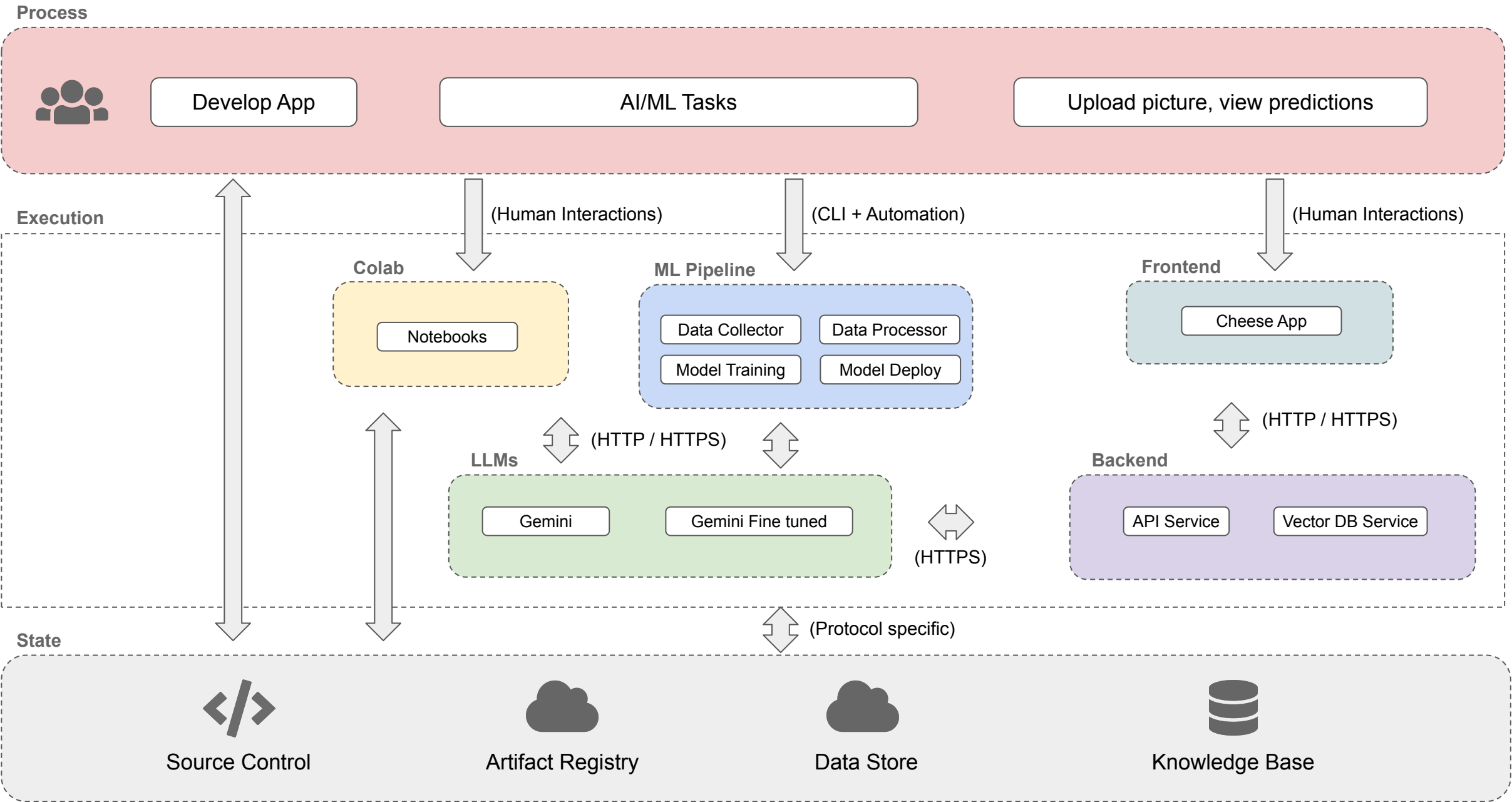
Solution Architecture



Solution Architecture



Solution Architecture



Solution Architecture Summary

- **Process**

- Data Scientists perform ML Tasks
- Developers build App
- Users can upload pictures and have a chat conversation

- **Colab**

- Web based hosted notebook solution from Google to experiment ML task

- **ML Pipeline**

- Containerized ML components
- Helps to automate and run ML tasks

- **Frontend**

- User friendly single page app with capabilities to upload and chat with backend

- **Backend**

- API server to expose python functions to frontend

- **State**

- Source control to store/version code
- Container registry for docker images
- Image store for data
- Models and model artifacts store

Outline

1. Recap
2. Motivation
3. App Design
4. Screenflow & Wireframes
5. Solution Architecture
- 6. Technical Architecture**
7. Setup & Code Organization

Technical Architecture

- Helps design and develop an **AI App**
- High level view from **development** to **deployment**
- Illustrates **interactions** between components/**containers**
- **Blueprint** of the system
 - Helps team members understand the big picture
 - Helps onboarding new team members

Building a Technical Architecture

Developers / Data Scientists

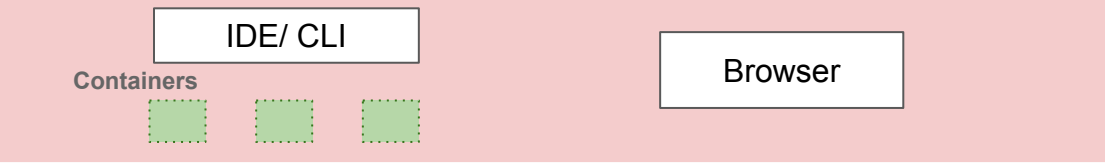


Users

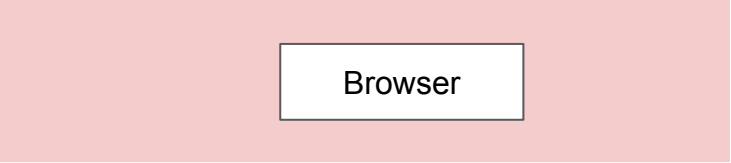


Building a Technical Architecture

Developers / Data Scientists

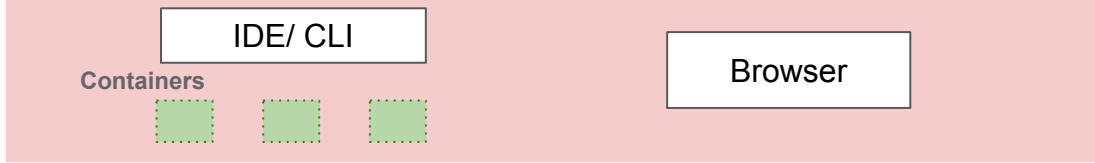


Users

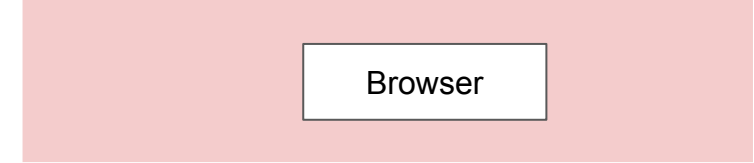


Building a Technical Architecture

Developers / Data Scientists



Users



Developers:

- Use IDE (VSCode), CLI to build app components
- All development is containerized

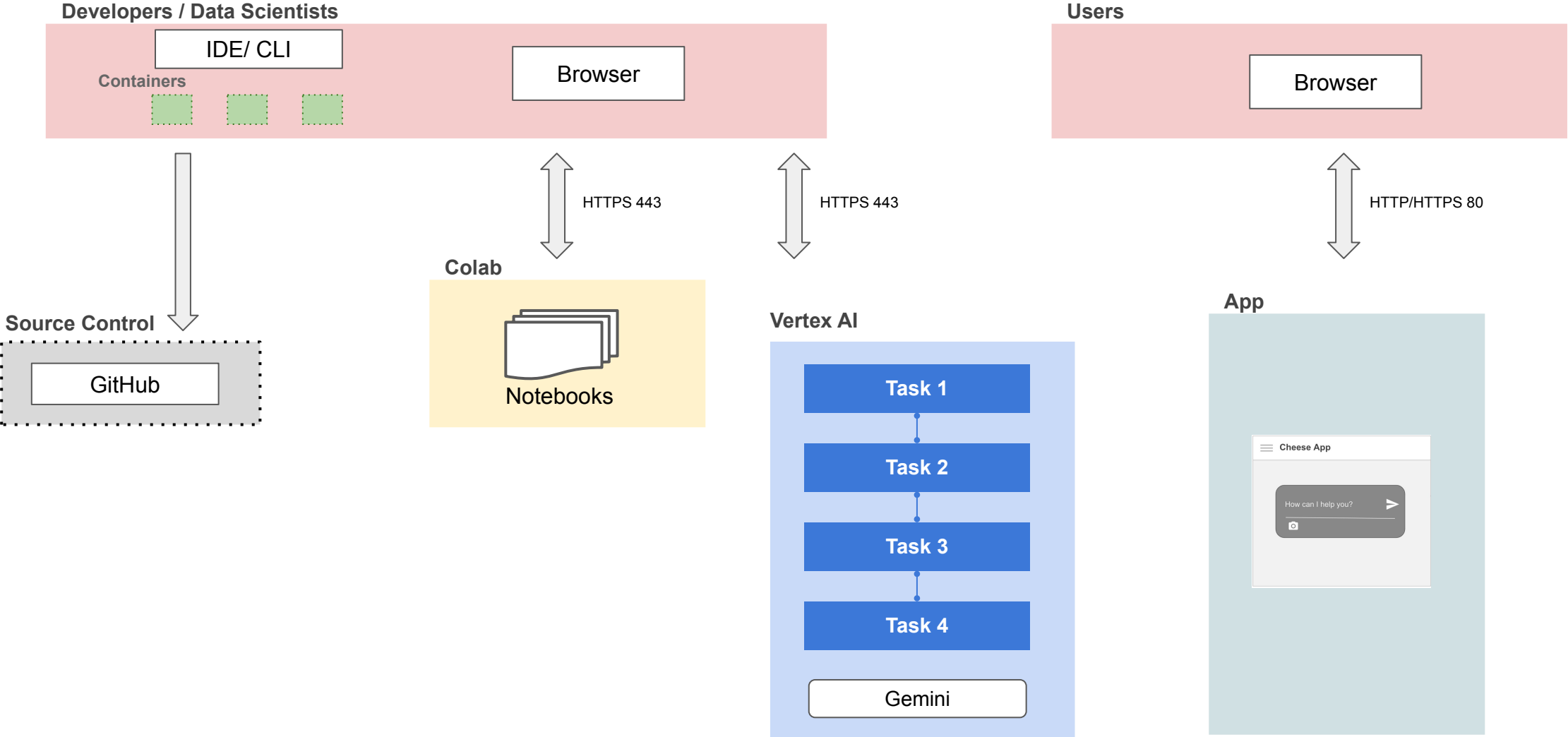
Data Scientists:

- Use Colab/JupyterHub
- EDA on notebooks
- Data & Model experimentation on notebooks
- Use IDE (VSCode), CLI to build ML Tasks
- All development is containerized
- Access LLMs

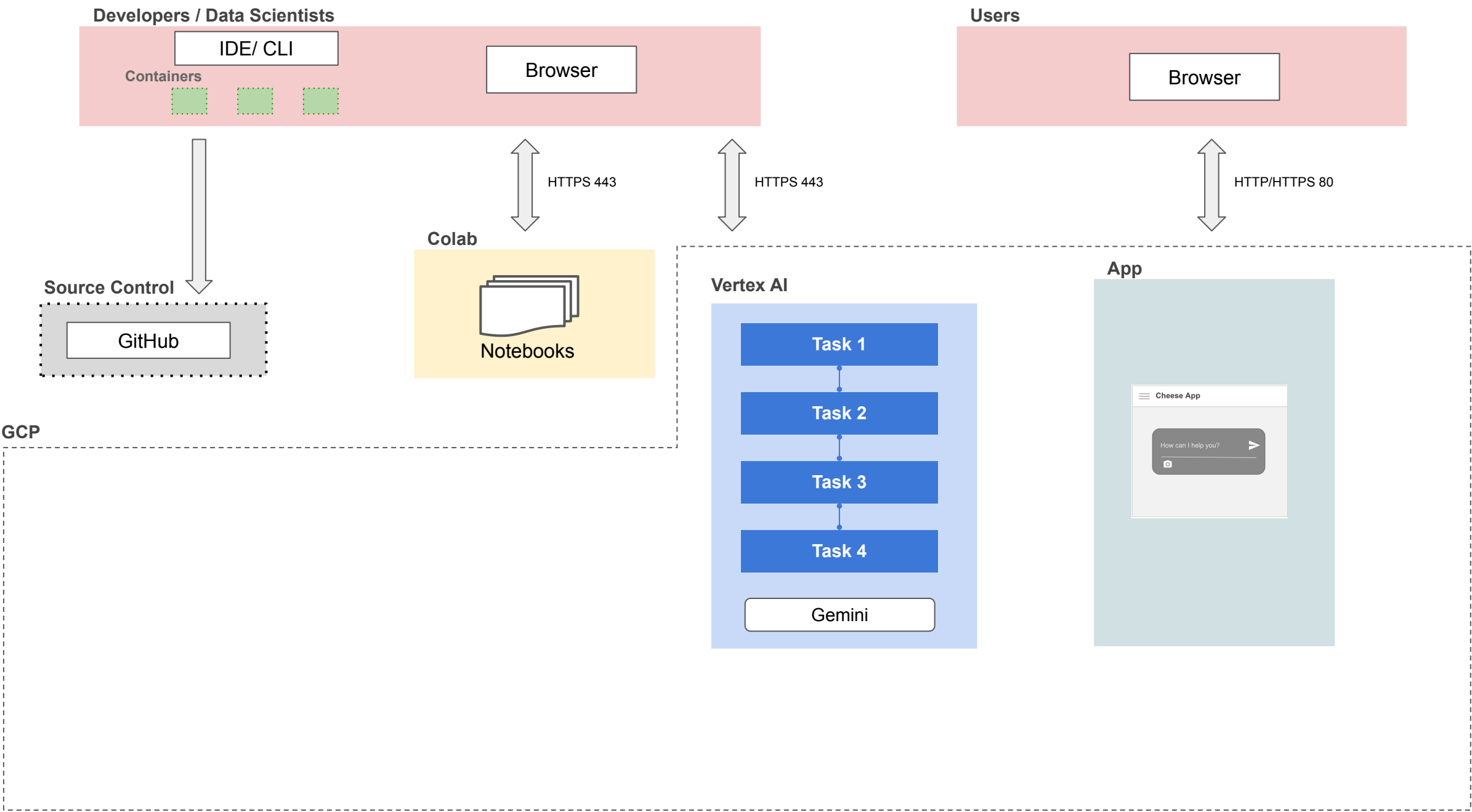
Users:

- Access the App using a browser
- Upload images and view prediction results
- Have a chat conversation

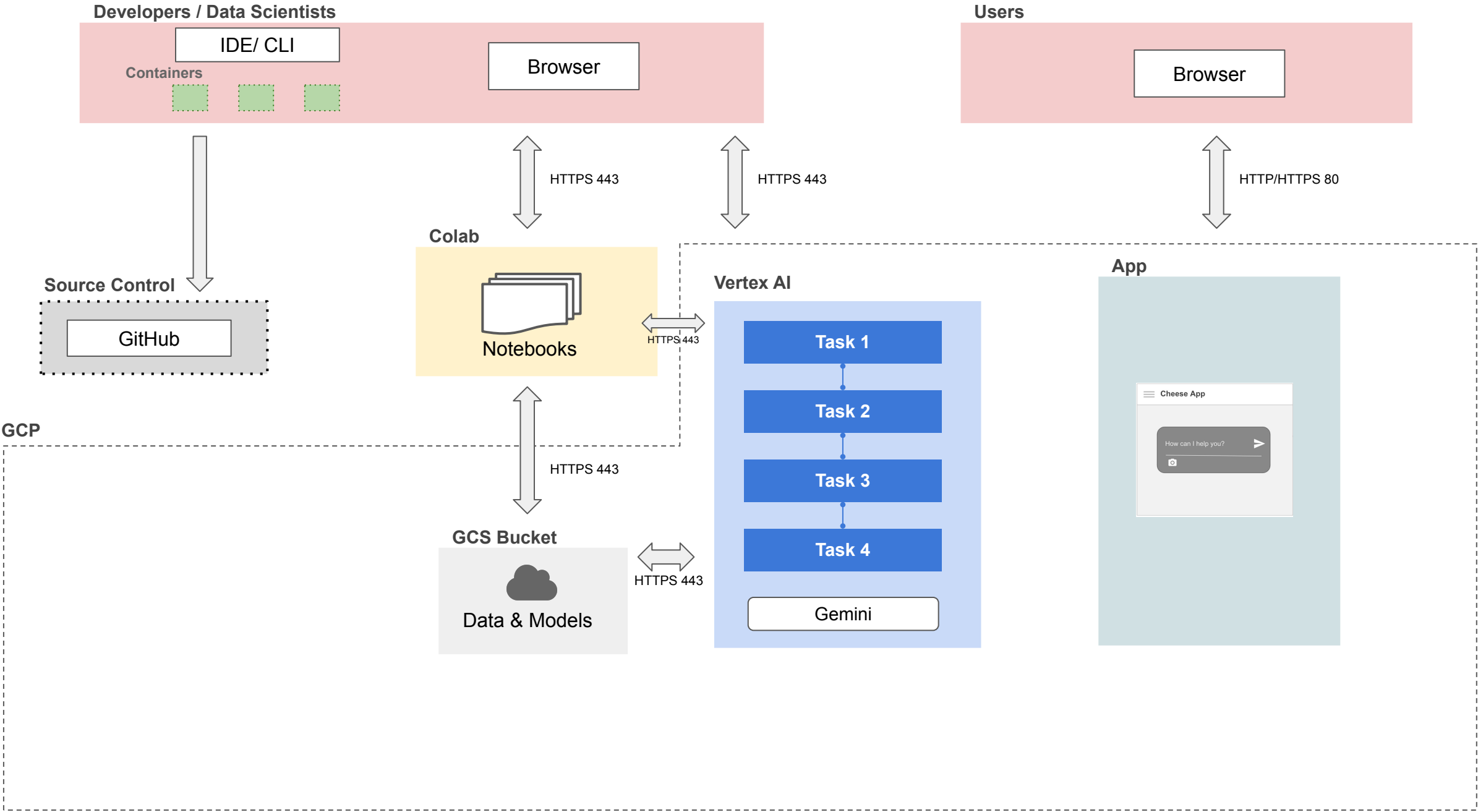
Building a Technical Architecture



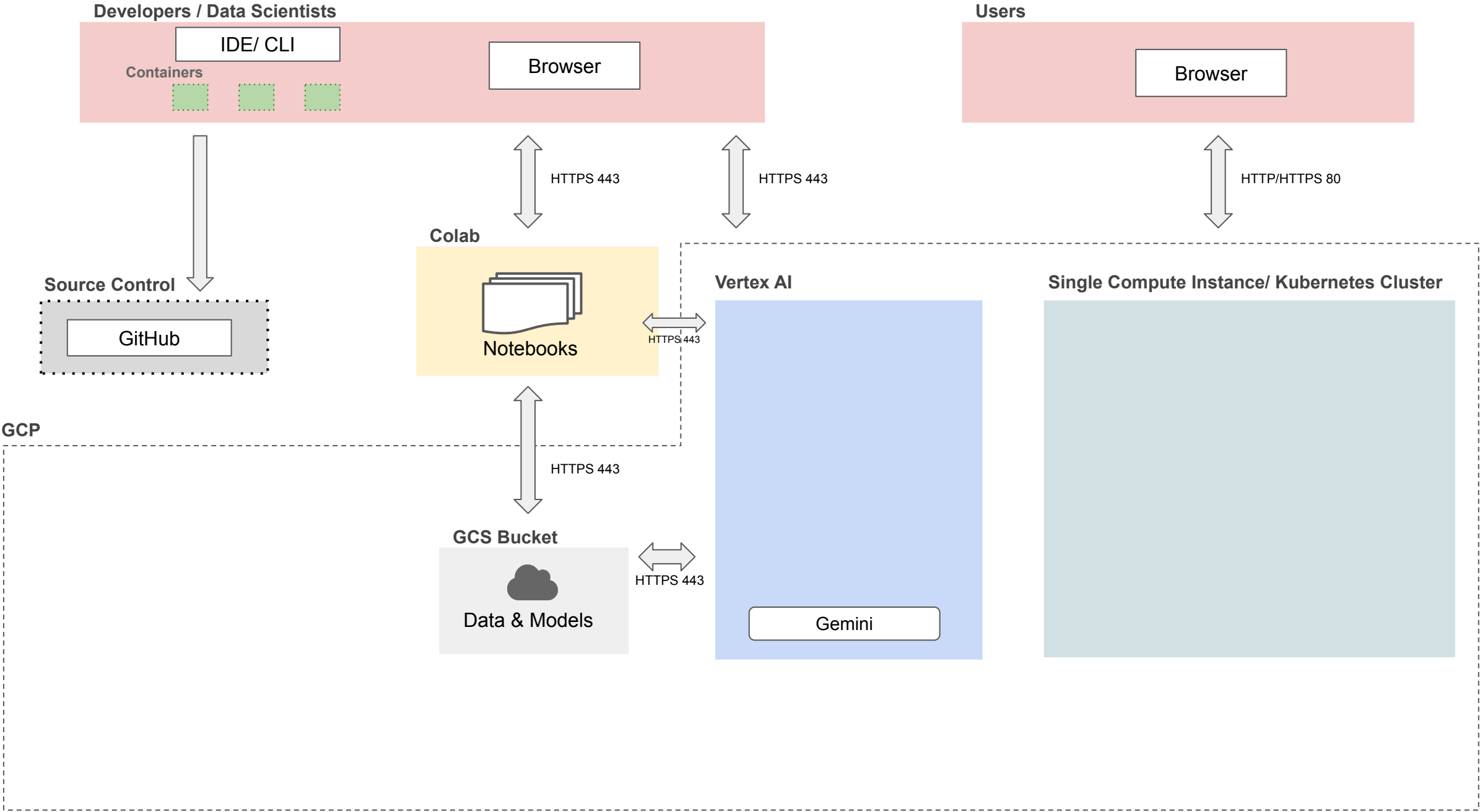
Building a Technical Architecture



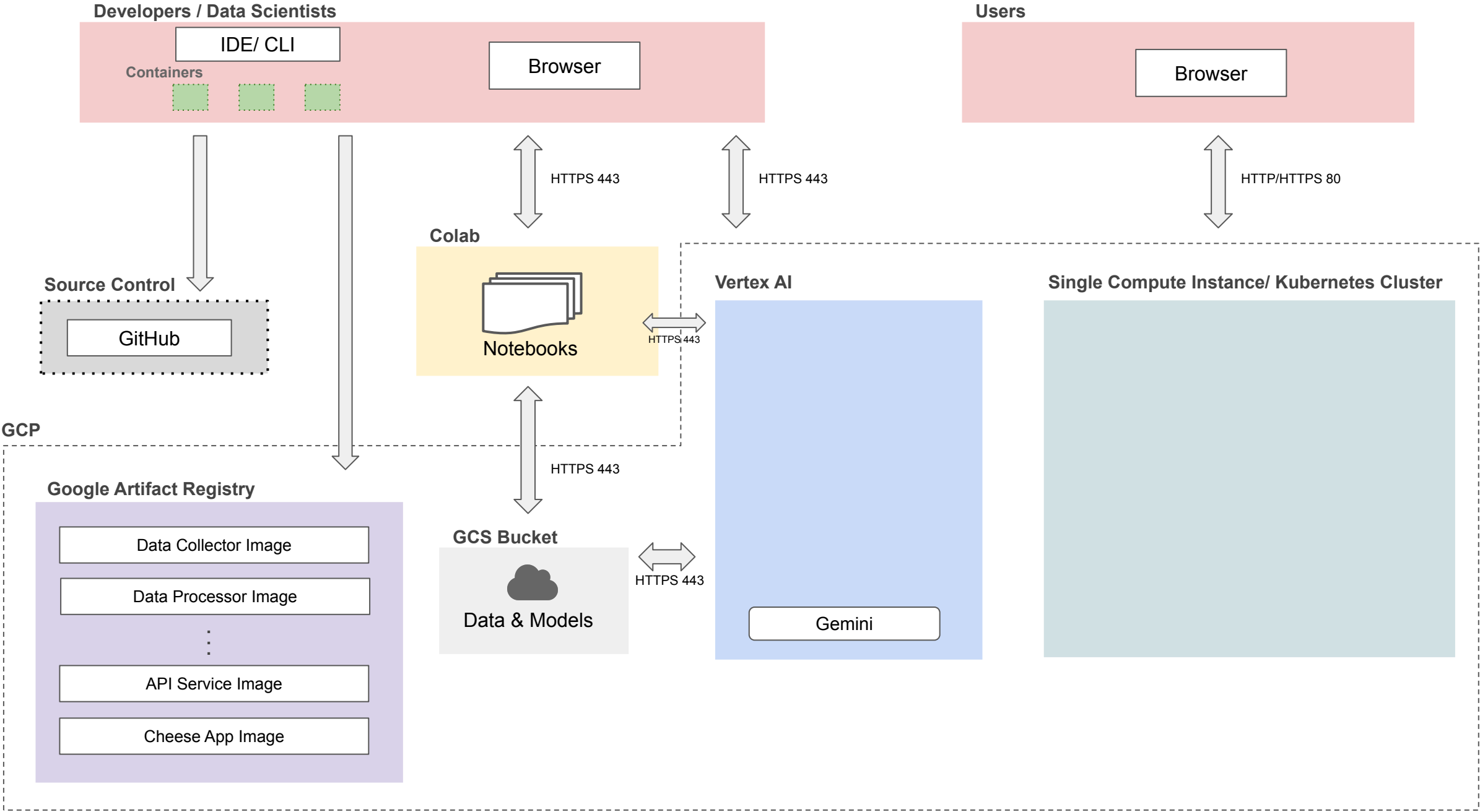
Building a Technical Architecture



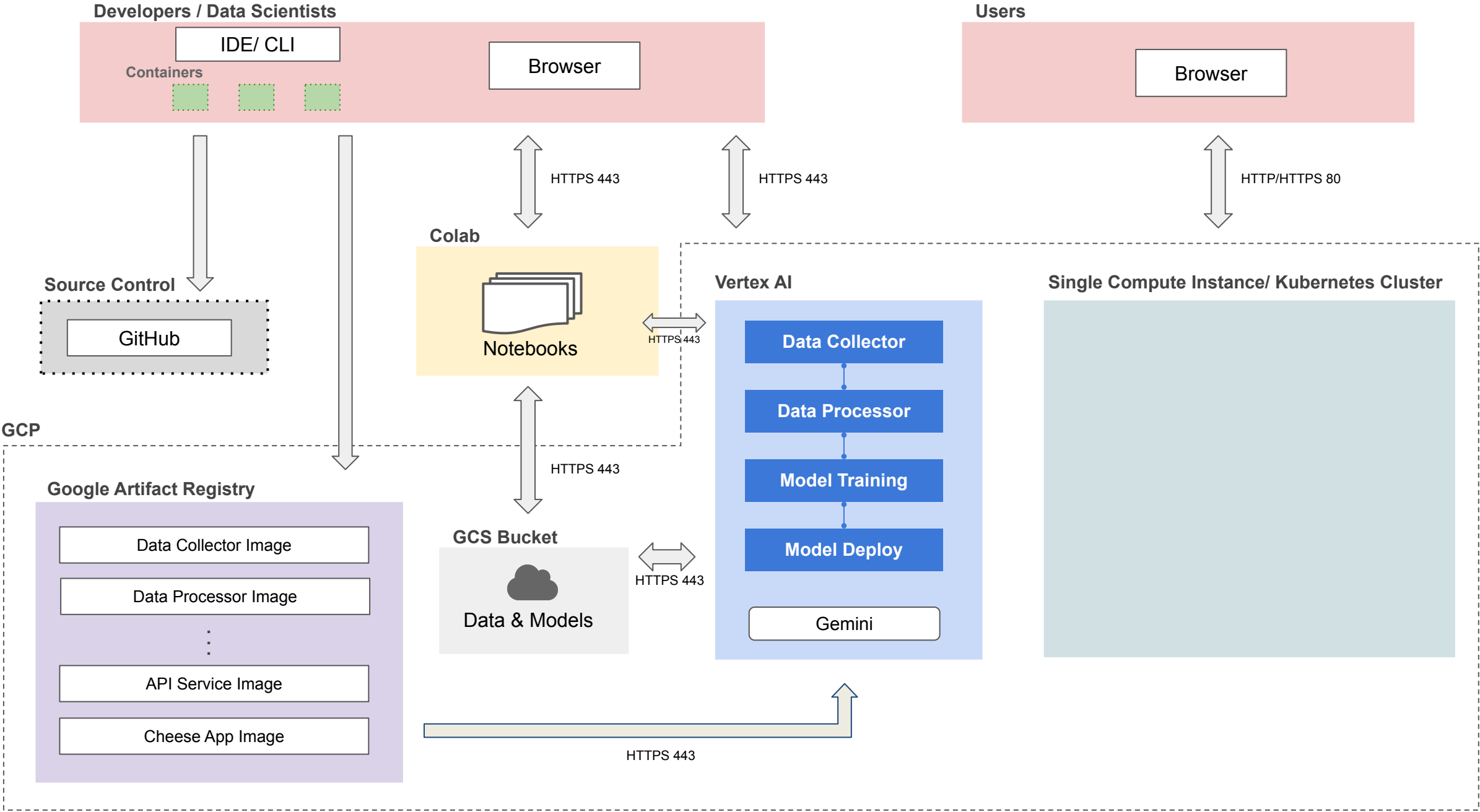
Building a Technical Architecture



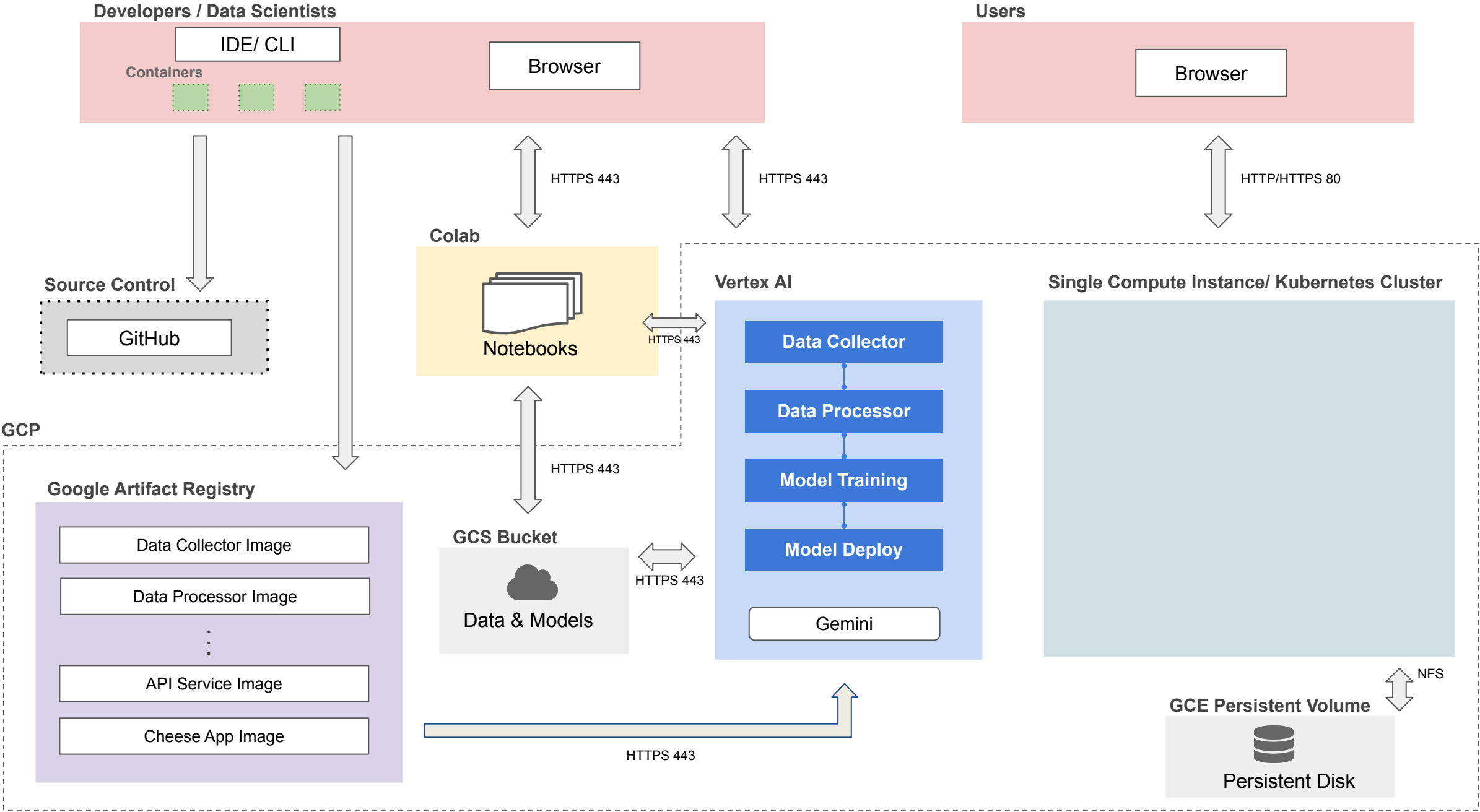
Building a Technical Architecture



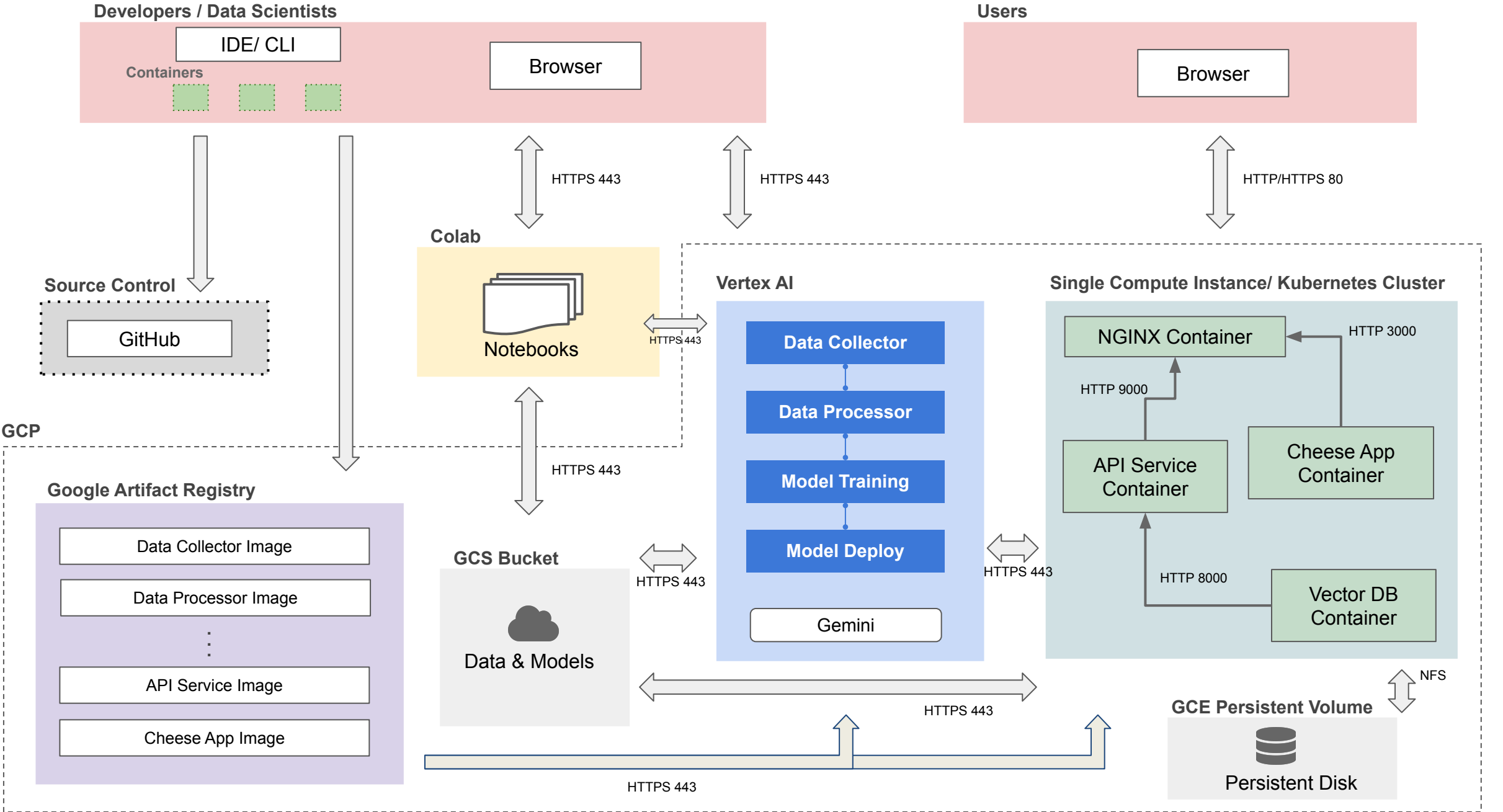
Building a Technical Architecture



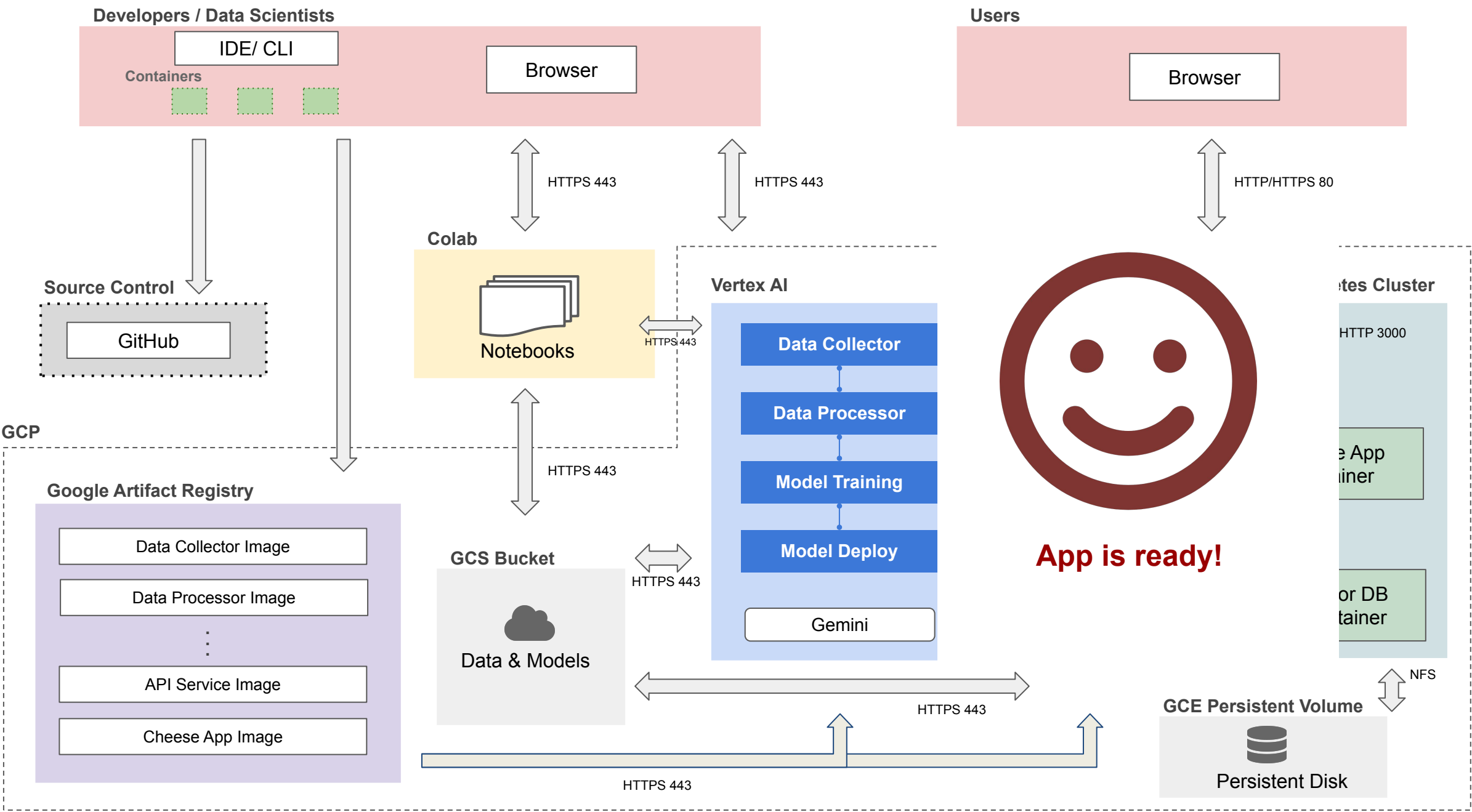
Building a Technical Architecture



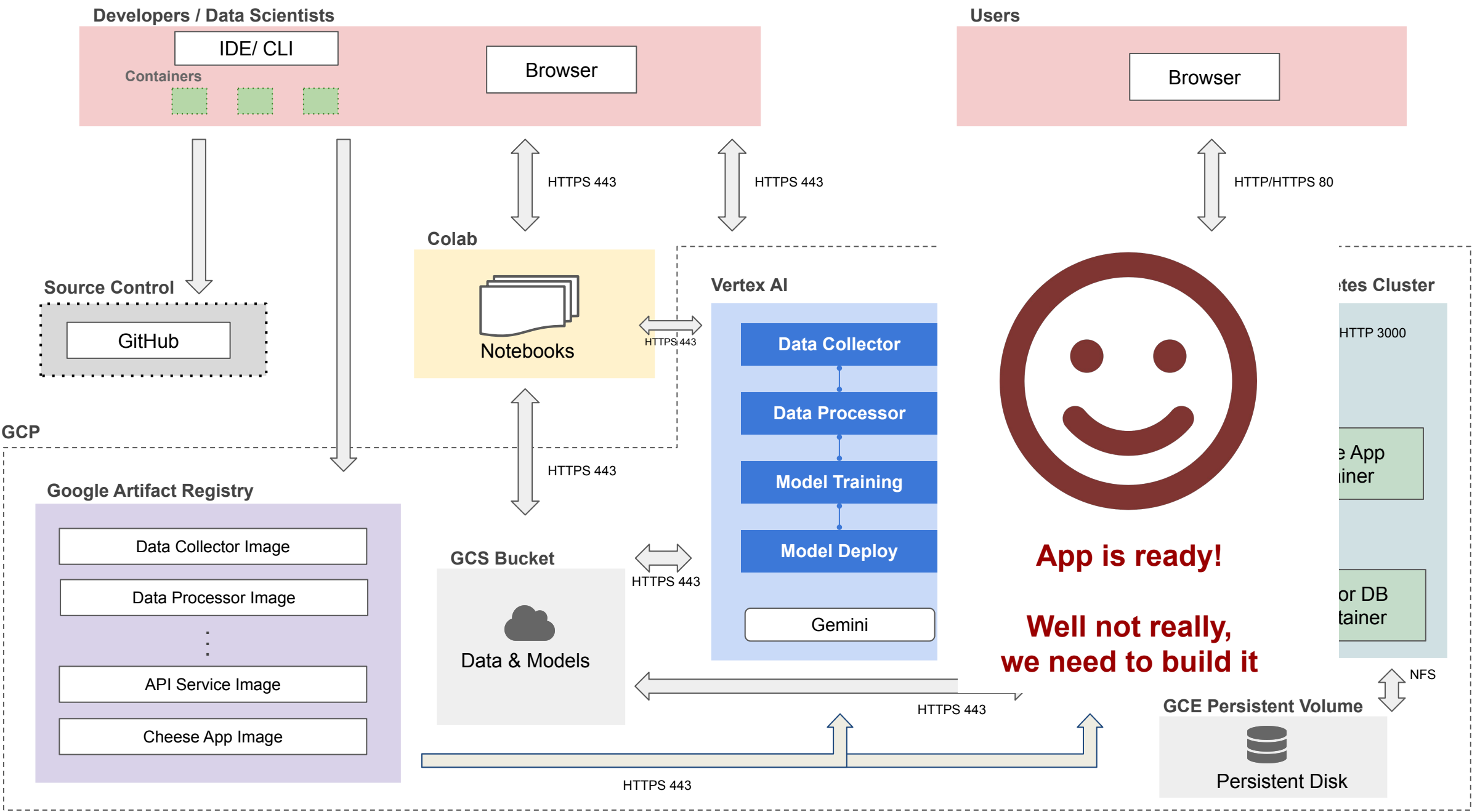
Technical Architecture



Technical Architecture



Technical Architecture



Technical Architecture Summary

- **Source Control**
 - GitHub
- **Google Cloud Platform (GCP)**
 - GCP for deployment
- **Google Artifact Registry**
 - Host all the container images
- **GCS Buckets**
 - Storage buckets for models and model artifacts
 - Data store
- **Vertex AI**
 - Serverless ML Tasks
 - Gemini LLM
- **GCE Persistent Volume**
 - Any files that need to be persisted when container images are updated
- **Compute Instance**
 - Hosting single instance of all containers
- **Kubernetes Cluster**
 - Kubernetes cluster will be used to scalable the app on GCP

Tutorial: Building Solution Architecture

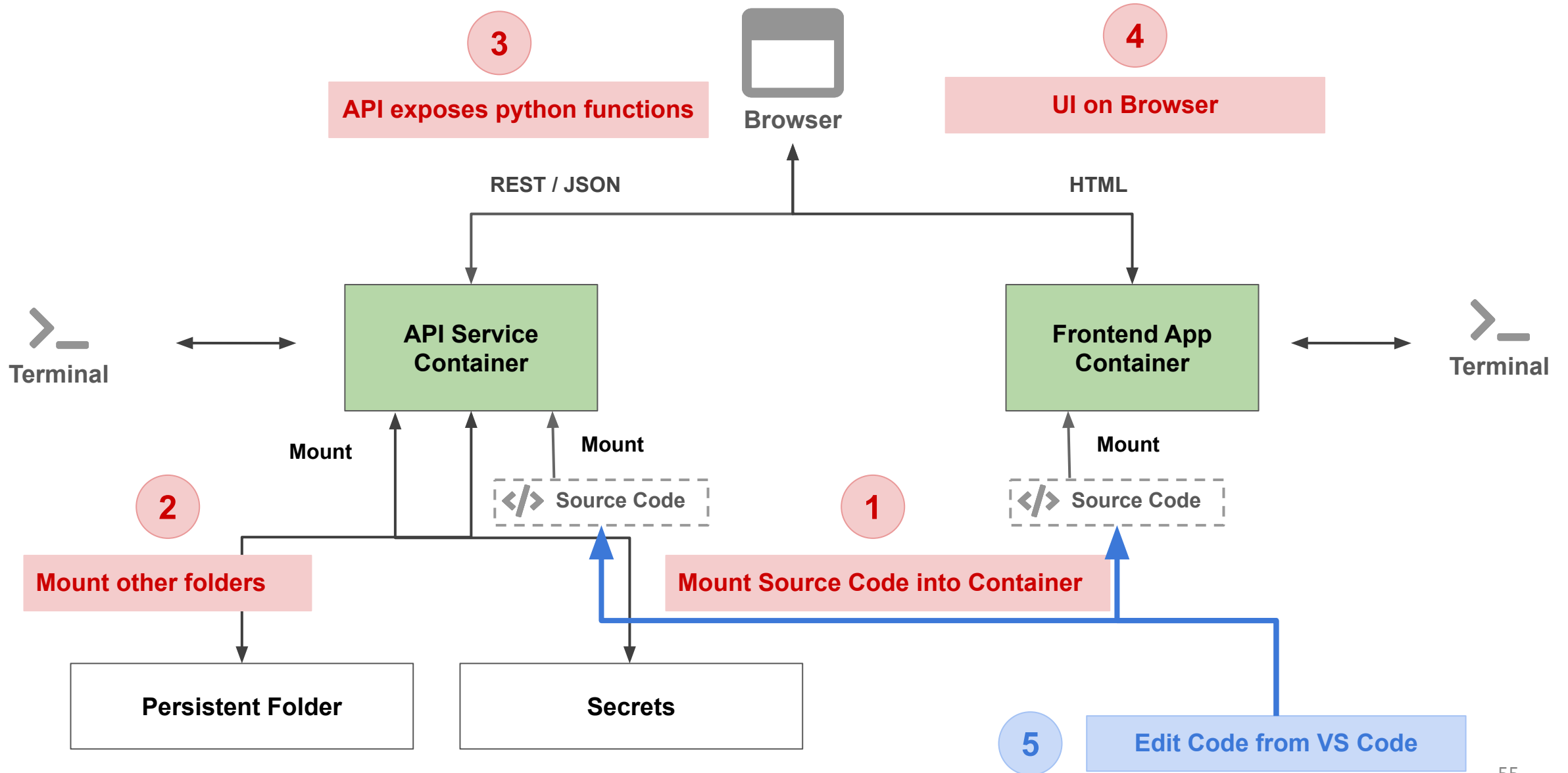
Steps to build a **Solution Architecture**

- You will work with your project group
- Go to
<https://docs.google.com/presentation/d/1lBnVcjT4tIshJThe-yhfGgGuvOMmZNyPuSQJ-b7WUnE/edit?usp=sharing> .
- Duplicate Slides 2,3 to the end of the slides.
- Put your group name in the slides.
- Identify **Process**, **Execution**, **State** for your project.
- For later: Complete **Solution Architecture** slide for your project.

Outline

1. Recap
2. Motivation
3. App Design
4. Screenflow & Wireframes
5. Solution Architecture
6. Technical Architecture
7. **Setup & Code Organization**

Setup & Code Organization



Tutorial: Setup & Code Organization

[Cheese App - Setup & Code Organization](#)

THANK YOU