Introduction to Cloud-Native Web Applications with Python

Xiaozhou Li, Software Engineer, Ultimaker

About me

Me & 3D printing: https://ultimaker.com/careers/xiaozhou-li

Email: x.li@ultimaker.com / mail@xzli.me
GitHub: https://github.com/xli12
LinkedIn: https://www.linkedin.com/in/xiaozhouli/

- Xiaozhou Li
  - Software Engineer @ Ultimaker
  - Full-stack developer (80% backend/20% frontend)
  - Django Girls volunteer since 2019
  - Live in 🇳🇱 since 2017

  - 🇩🇰 PhD (2017), pharmaceutical sciences (computational pharmaceutical materials science)
  - 🇬🇧 MPhil (2013), computational chemistry
  - 🇨🇳 BEng (2011), applied chemistry

Python (programming language)

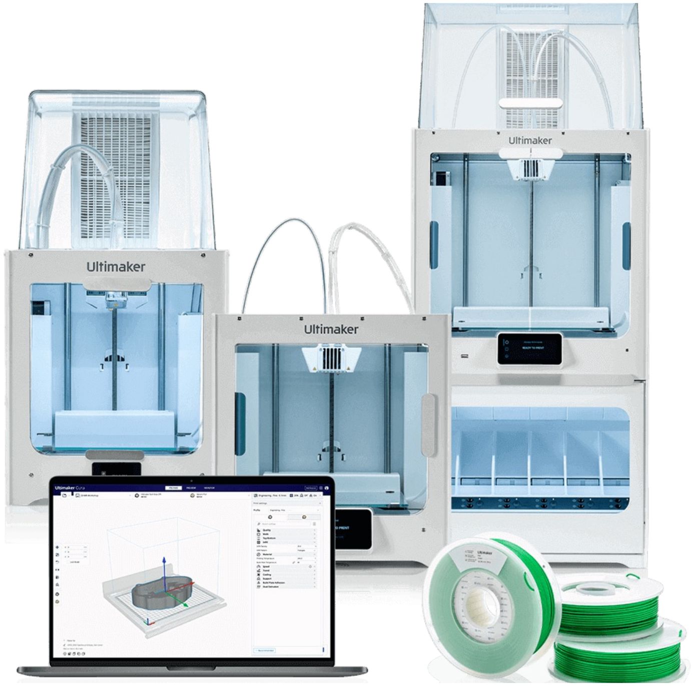
- High-level, interpreted, open-source, general-purpose programming language
- Python 3
  - Python 2 discontinued in 2020
- Widely used
  - Data Science
  - Web development
  - Engineering and research
  - Desktop graphical user interface

Cloud-Native Web Application with Python

Outline

Introduction	Team Stardust (the “cloud” team ☁️) & Ultimaker
Cloud	What is “cloud” & a brief history about cloud platforms
Web application	The components of a simple web application
Cloud platform services	Products that are available through cloud platforms
Summary	Web application + cloud platform

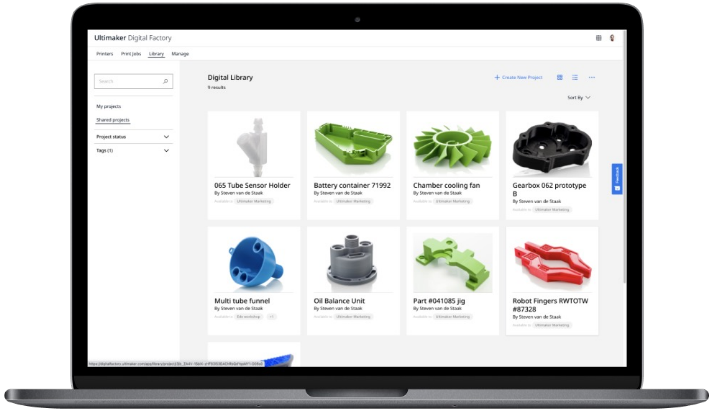
Introduction



- Ultimaker
  - Deliver a 3D printing ecosystem

  - 3D printers (FDM)
  - Software
  - Materials
  - Applications
  - Education

Introduction



- Team Stardust (☁️)
  - Cloud-Native Web Applications
  - Account
  - Digital Factory
    - Printer & Print File Management
  - Marketplace
    - Plugins & Material Profiles

What is Cloud Computing?

Simply put, cloud computing is the delivery of computing services—including servers, storage, databases, networking, software, analytics, and intelligence—over the Internet (“the cloud”) to offer faster innovation, flexible resources, and economies of scale. You typically pay only for cloud services you use, helping you lower your operating costs, run your infrastructure more efficiently, and scale as your business needs change.

--Microsoft Azure “What is Cloud Computing?”

Cloud, still sounds vague? 🤔

Build a website in 2000

1. Plan
   - What do we want to build?
   - Milestones for each phase?
   - Who are the targeted users?
   - How many users do we expect?

Build a website in 2000

1. Plan
2. Submit a budget plan
   - Servers
   - Operation staff
   - Maintenance cost
   - Peak hour/season solutions
   - On-premises data centre

Build a website in 2000

1. Plan
2. Submit a budget plan
3. Implementation, operation, maintenance & “what if”s
    - Number of customers much lower/higher than expected? (capacity planning) – super annoying!
    - Are the features in the app still something that the customers *exactly* want?

Built a website in 2000

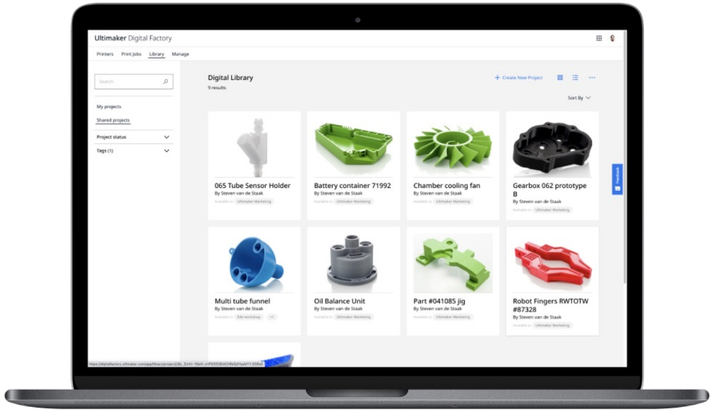
1. Plan
2. Submit a budget plan
3. Implementation & Operation

In general: Overheads!

- Nothing to do with customer or deliver the actual products

Low-budget, easy-to-start option?

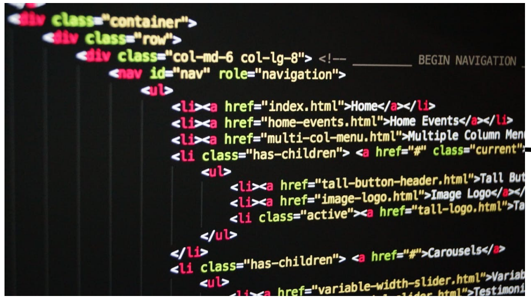
Digital Factory: A cloud-native web application



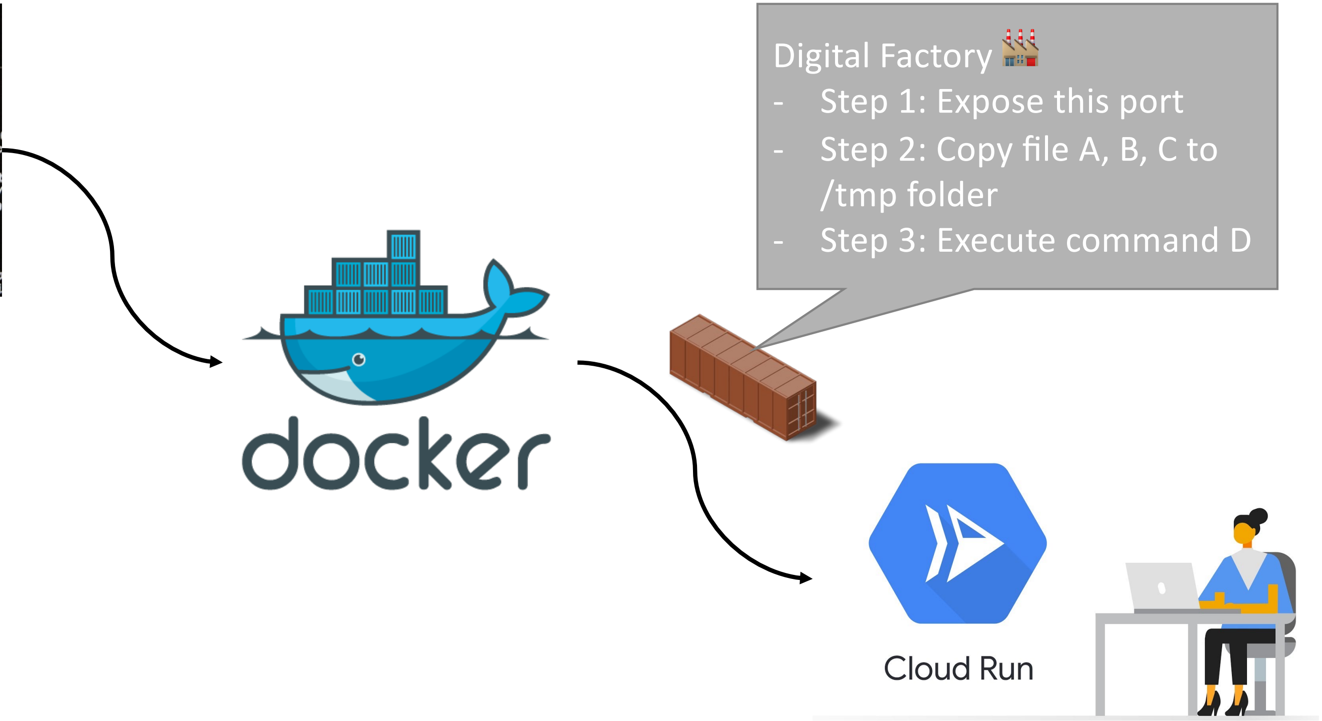
- Cloud-native
  - Utilise cloud computing to "build and run scalable applications in modern, dynamic environments”
- Rent services from cloud providers
  - Pay-as-you-go

Shipping the code

We’ll do the small exercise later!

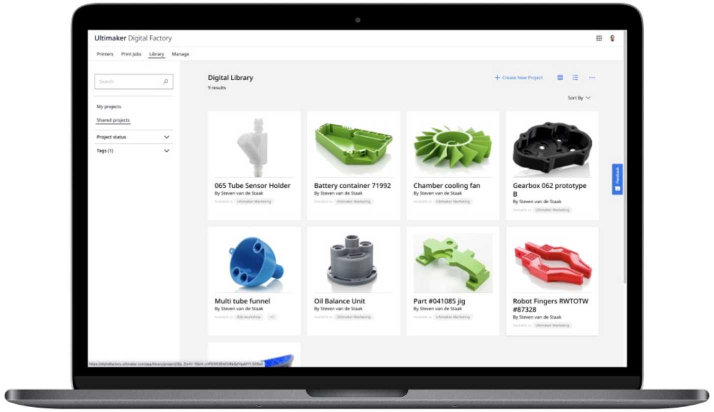


```
<div class="container">
    <div class="row">
        <div class="col-md-6 col-lg-8"> <!-- ________ BEGIN NAVIGATION
            <nav id="nav" role="navigation">
                <ul>
                    <li><a href="index.html">Home</a></li>
                    <li><a href="home-events.html">Home Events</a></li>
                    <li><a href="multi-col-menu.html">Multiple Column Men
                    <li class="has-children"> <a href="#" class="current"
                        <ul>
                            <li><a href="tall-button-header.html">Tall But
                            <li><a href="image-logo.html">Image Logo</a></
                            <li class="active"><a href="tall-logo.html">Ta
                        </ul>
                    </li>
                    <li class="has-children"> <a href="#">Carousels</a>
                        <ul>
                            <li><a href="variable-width-slider.html">Variab
```



Digital Factory 🏭
- Step 1: Expose this port
- Step 2: Copy file A, B, C to /tmp folder
- Step 3: Execute command D

Digital Factory: A cloud-native web application



- Digital Factory up and running!
- https://digitalfactory.ultimaker.com

Summary: Cloud Computing

- Three major Cloud providers:
  - Amazon Web Services (AWS)
  - Microsoft Azure
  - Google Cloud Platform (GCP)

Still have questions? Of course!

- All we have done was delivering a Docker image to the Cloud platform… and that was it?
  - No, that’s not the case of course!
- Okay, what else?
  - Databases
  - Message queues
  - Logging
  - Alerting
  - Load balancer
  - Code repositories
  - Continuous integration/continuous delivery (CI/CD)
  - And more…

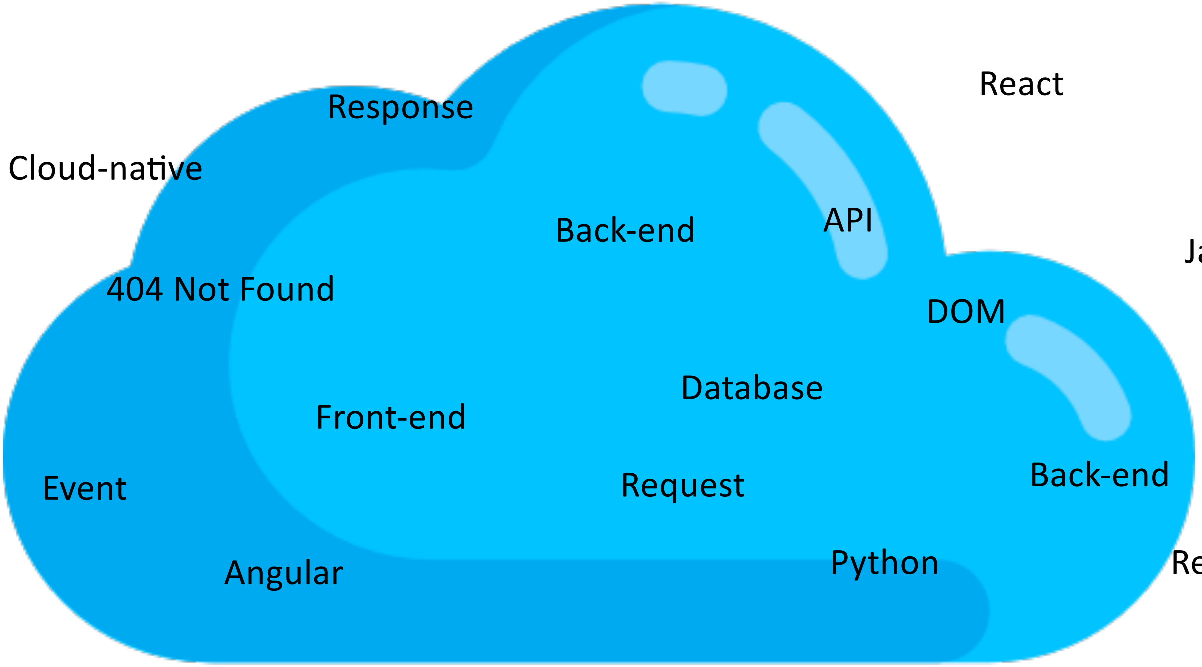
Direction?

1. What is cloud computing? ✅
2. What is a web application? 🐍
3. Combine cloud computing with a web application ☁️

The boring definition of a web application

A web application (or web app) is application software that runs on a web browser. Web applications are delivered on the World Wide Web to users with an active network connection.

Welcome to the jargon cloud of web applications



Bootstrap

React

Response

Cloud-native

Back-end

API

Java

404 Not Found

DOM

JavaScript

Database

Front-end

Back-end

Request

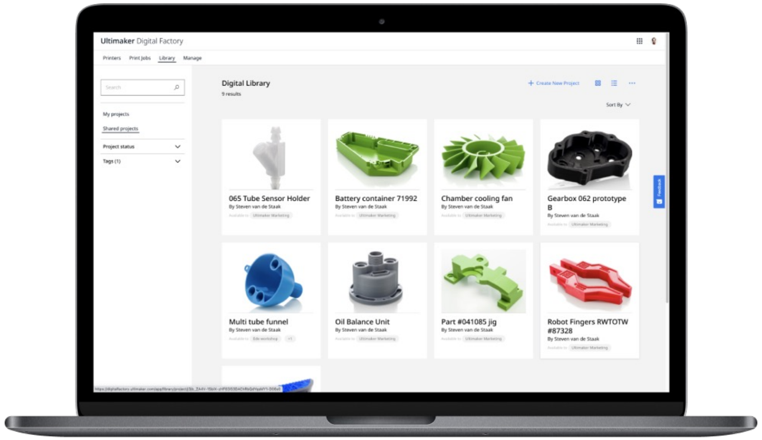
Event

Python

Responsive design

Angular

Use an example to understand a web application

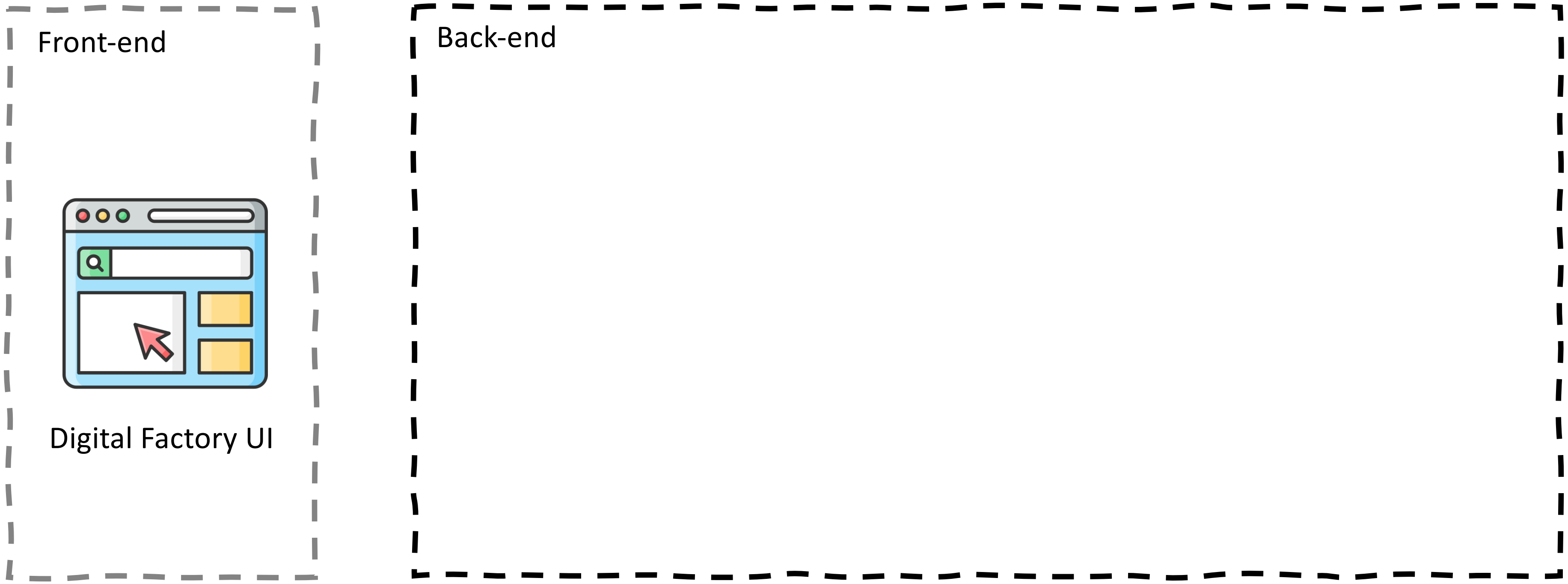


digitalfactory.ultimaker.com

Front-end

- An interface that users directly interact with
- Web application: Browsers
    - HTML (contents)
    - CSS (styling)
    - JavaScript (Effects)

Front-end & back-end: Request

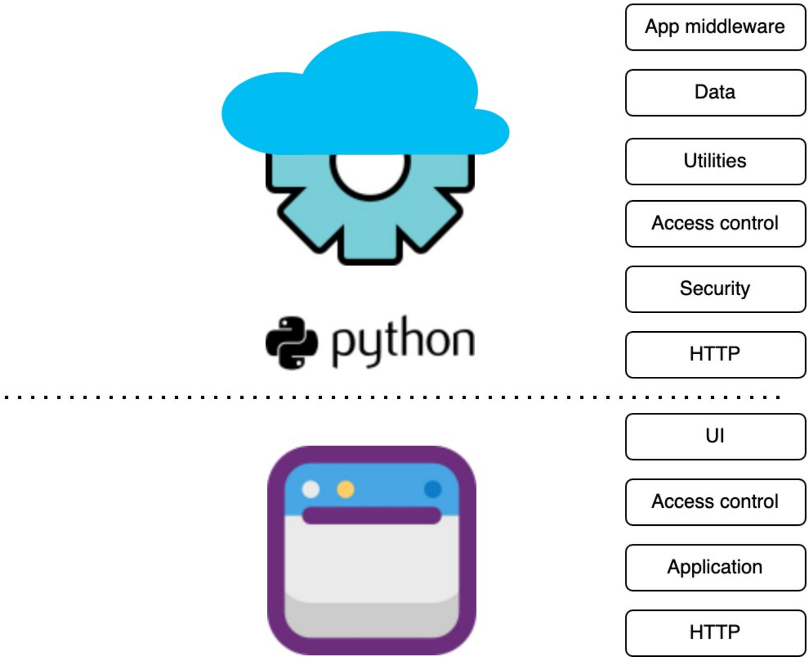


Front-end

Digital Factory UI

Back-end

Re-inventing the wheels?



App middleware

Data

Utilities

Access control

Security

HTTP

UI

Access control

Application

HTTP

User registration

Routes:
/projects
/printers

Password validation and encryption

Web framework

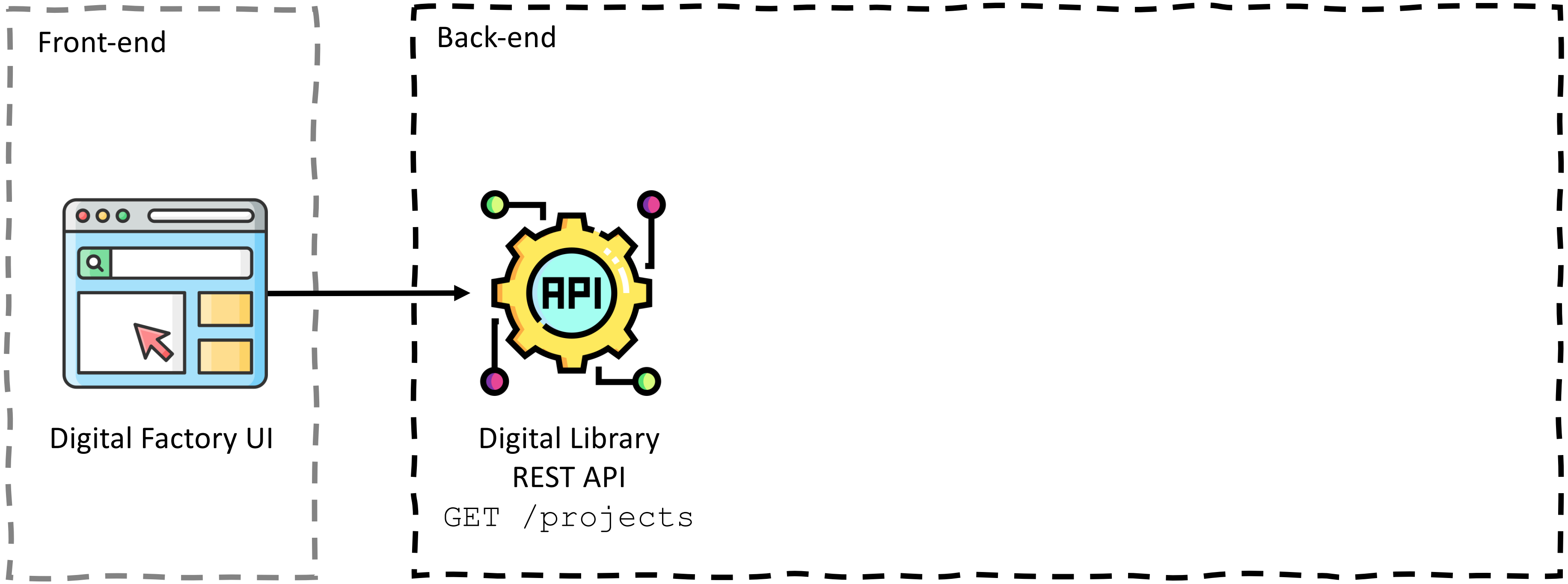
Definition

- Software framework designed for developing web applications
  - Web services
  - Web resources
  - Web APIs

Examples of web frameworks

- Server-side
  - Java: Spring, Grails etc.
  - Ruby: Ruby on Rails, Sinatra etc.
  - Python: Django, Flask, Tornado etc.
- Client-side (single-page application)
  - React
  - Vue.js
  - Angular

Front-end & back-end: Request



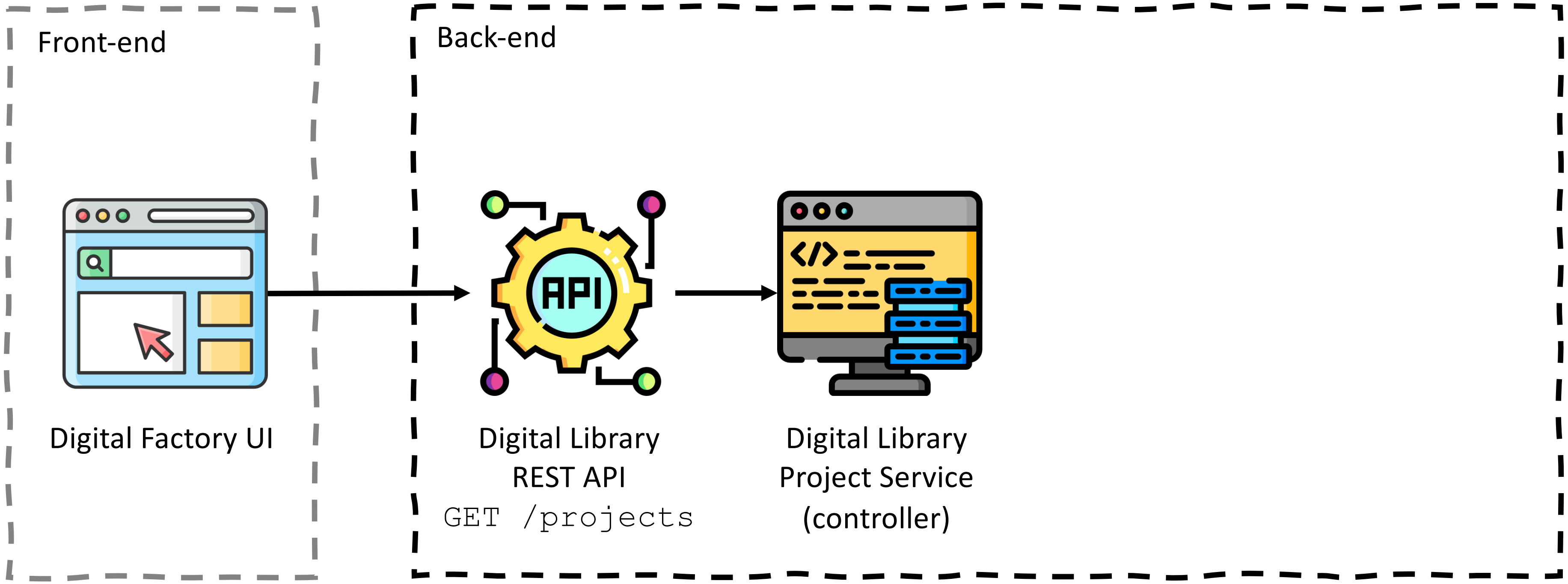
Front-end

Digital Factory UI

Back-end

Digital Library
REST API
`GET /projects`

Front-end & back-end: Request



Front-end

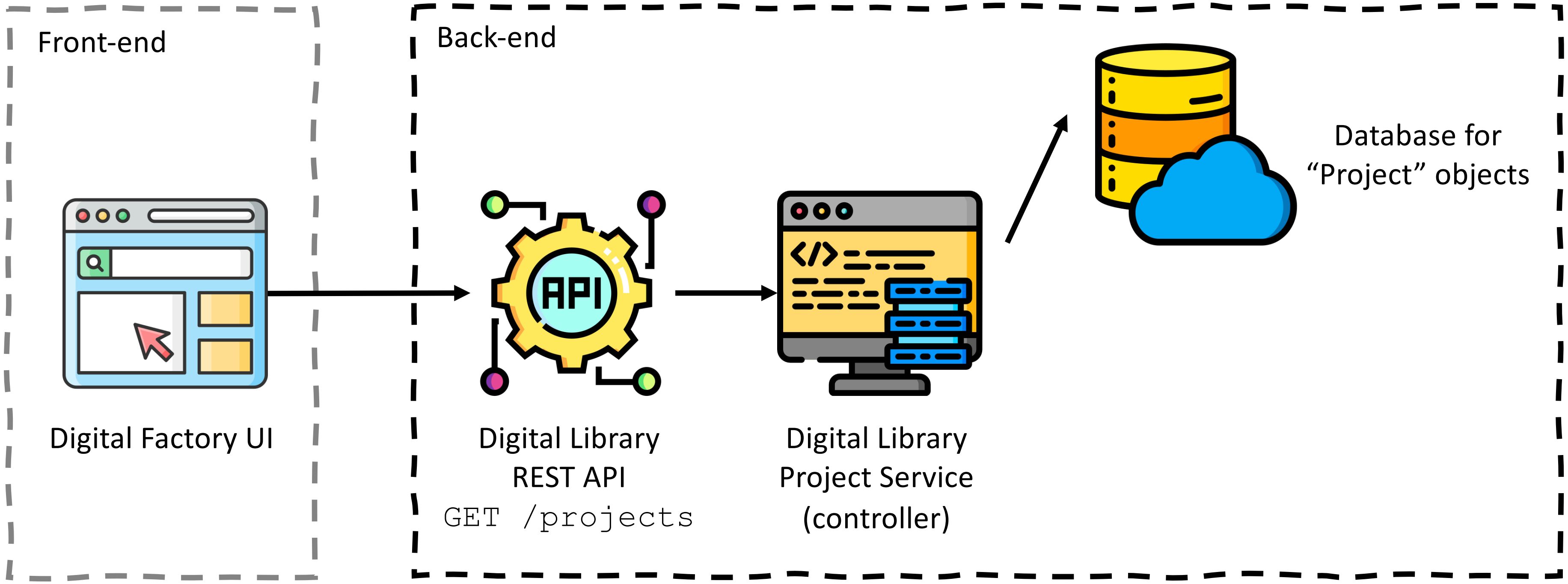
Digital Factory UI

Back-end

Digital Library
REST API
`GET /projects`

Digital Library
Project Service
(controller)

Front-end & back-end: Request



Front-end

Digital Factory UI

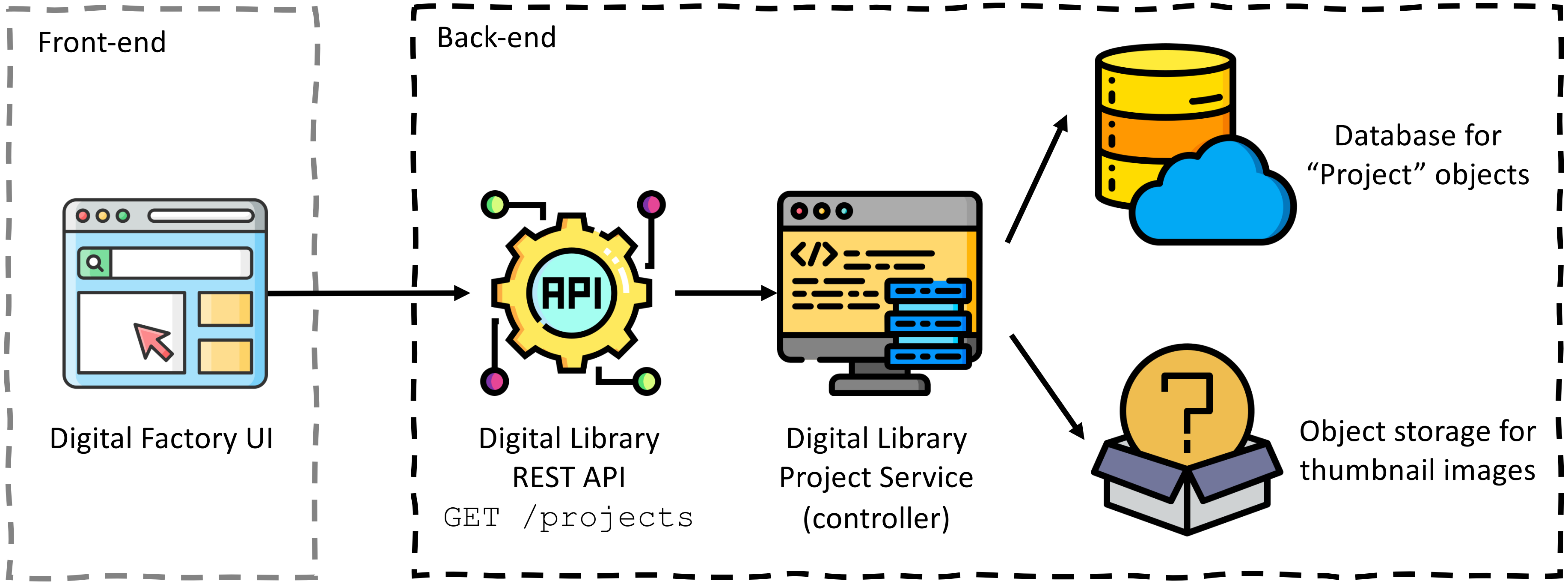
Back-end

Digital Library
REST API
`GET /projects`

Digital Library
Project Service
(controller)

Database for
“Project” objects

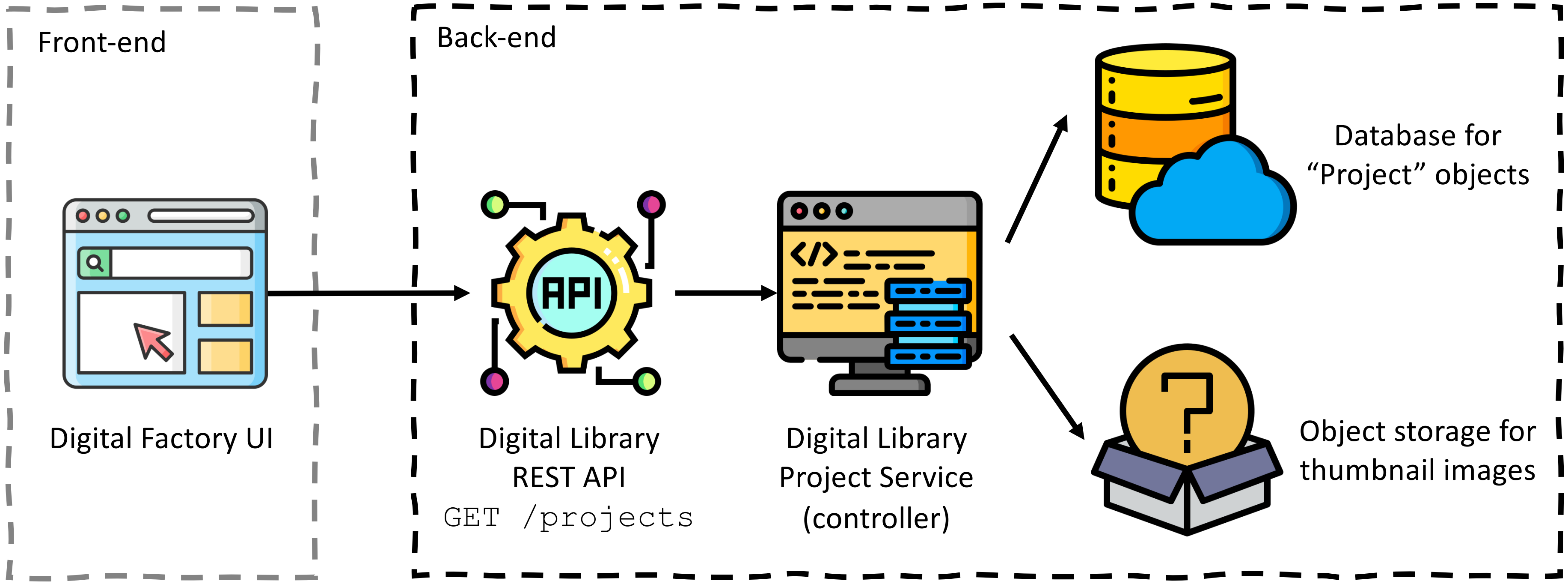
Front-end & back-end: Request



Cloud object storage

- Each object has a unique name
- The object name is stored in the database
- Based on the unique object name, we find it in the Cloud Storage

A simple cloud-native web application



Front-end

Digital Factory UI

Back-end

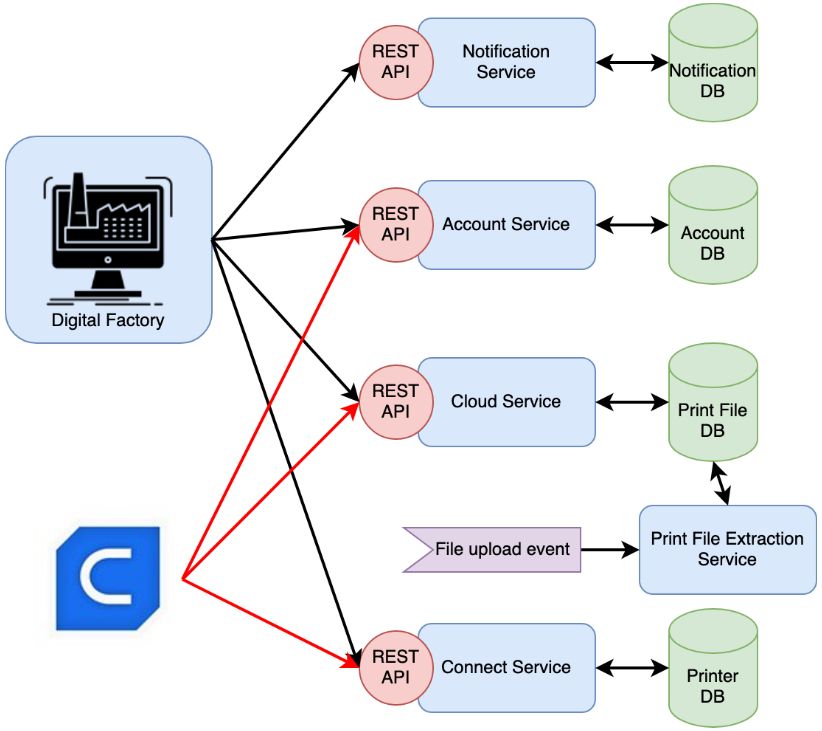
Digital Library
REST API
`GET /projects`

Digital Library
Project Service
(controller)

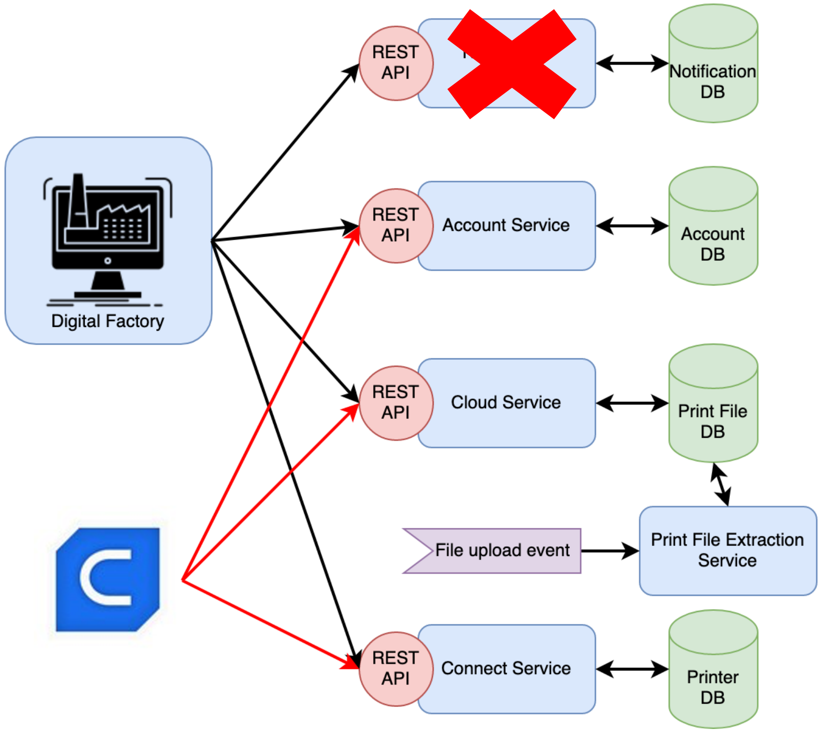
Database for
“Project” objects

Object storage for
thumbnail images

Microservices & Event-driven architecture

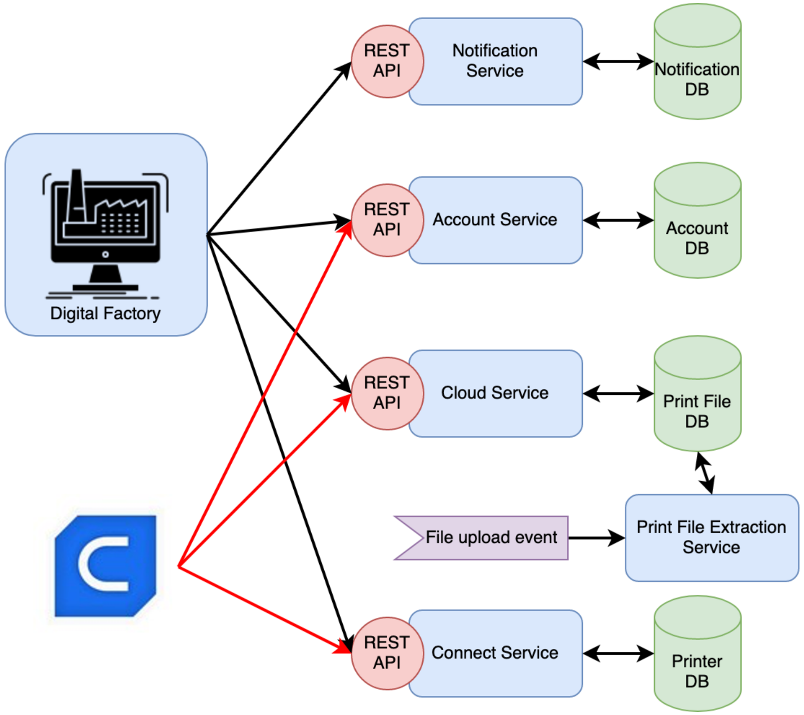


Microservices & Event-driven architecture



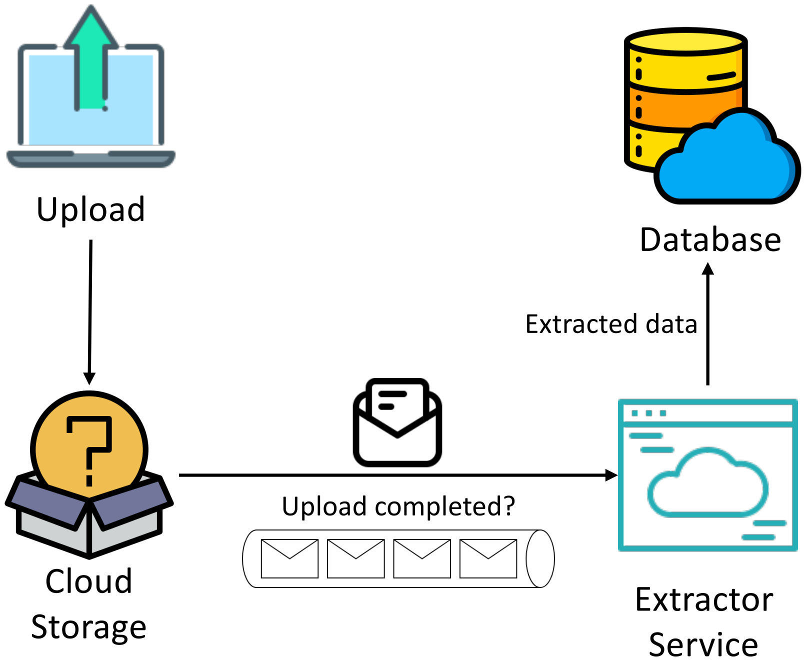
Microservices

- A microservices architecture is a type of application architecture where the application is developed as a collection of services. It provides the framework to develop, deploy, and maintain microservices architecture diagrams and services independently.



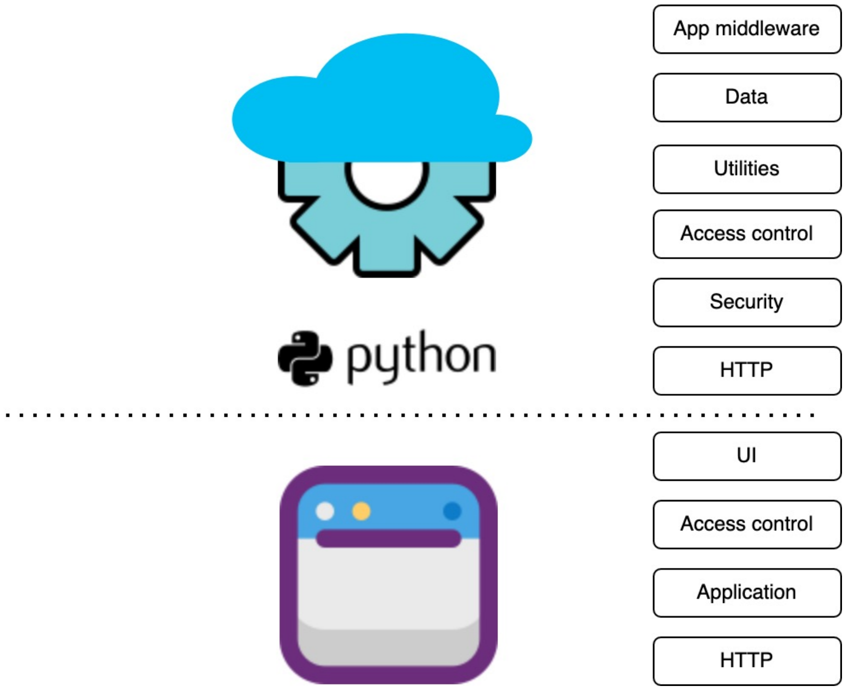
https://cloud.google.com/learn/what-is-microservices-architecture

Event-driven architecture & message queue



- Upload a file to Cloud Storage
- Once the upload is completed, Cloud Storage sends a message to the message queue
- The Extractor service picks up the message from the queue, and process the message (extract the data from the uploaded file)
- The Extrator service stores the extracted data to the database

Summary



App middleware

Data

Utilities

Access control

Security

HTTP

UI

Access control

Application

HTTP

Pros & Cons: Cloud-native web applications

Pros

- Cost efficient
- Scalability
- Automation and flexibility
- Faster release
- No overhead burdens

Cons

- Management can be complex
    - Container orchestration
    - Terraform configuration
- Cloud provider reliability

Demo

- Deploy a Django App to Google Cloud Run
  - Google’s managed compute platform which lets the users to run containers directly on Google’s scalable infrastructure

Ultimaker