## AI4Europe

## **Technical Architecture Preview**

#### **Overview**

#### Al4Europe Architecture

Ecosystem of services, registered with uniform descriptions

All services can offer their own interfaces that other services can build on (layers)

Services can be hosted anywhere, can have their own infrastructure (e.g. databases), can have their own monetization

Services can be designed for different audiences (e.g. academia, industry, experts, novices,...), showing/filtering all AloD resources in different ways

Services from other EU projects Matchmaking services

Trustworthiness checks

SME-focused services

Other services

AutoML services Data quality services

Build on AloD clients/API directly

#### Interfaces

Build on core services

#### Websites

Search / Gallery of resources Share resources (drag and drop) Project websites

Chatbots / NLP Semantic search

#### Core (1st tier) services

Experiment engines (e.g. Al4Experiments) **HPC / Compute Services** Hosted Jupyter notebooks Benchmarking services ETL jobs, e.g. meta-data improvement

**Data Platforms** Other Platforms OpenML.org European Language Grid HuggingFace European Open Science Cloud Zenodo PapersWithCode

Users can store resources directly in AloD (if infrastructure allows) or choose to use integrated existing platforms

AloD offers aggregated view/search over all available resources

Interact via connector 3

#### AloD Connectors

#### Handle integration of existing platforms (with own APIs)

Map external resources to AloD schema and register them

L Build on AloD clients or REST API

#### Periodically indexes/updates existing resources

Users uploading resources to these platforms will also find them in AloD

One connector per external platform

#### AloD Nodes

#### Locally installed mini-platform

Makes local resources visible in AloD Handles communication with AloD API

E.g. for companies, universities, hospitals that want to keep data local

Build on AloD clients or REST API

#### Al-on-demand Core API

Enables sharing and organisation of Al resources (directly, or via services or existing platforms)

#### Unified interface to all Al resources

for developing new services, interfaces, and applications

#### Easy to use libraries for development

to accelerate research, applications, automate processes

AloD Clients Language bindings (e.g. Python, Java, Command-line,...) to easily talk to the platform Easy querying + conversion to different formats Easy installation (e.g. `pip install aiod`)

#### Search Engine

Indexes all assets for fast search (e.g. website)

Catalog of all Al assets and all crucial metadata

Large, scalable, secure file storage

#### AIoD REST API

REST Interface (implemented in FastAPI, potentially also GraphQL)

GET/POST/DELETE, Query, and Edit all Al Assets

-> datasets, models, code, containers, benchmarks, media (news, courses, projects), services

API definition defines uniform data schema for all Al assets (e.g. all metadata for a dataset)

Build on existing standards as much as possible to facilitate interoperability Enables powerful search/filtering for interesting / compatible assets

Authentication service (OAuth). Provides an API key.

Access Control (who is allowed to see which resources), can be linked to monetization

Hosted centrally (e.g. in Kubernetes) with mirrors for redundancy/scalability

S3 Storage references in meta-data Git LFS

Binder (Jupyter)

Infrastructure

Meta-data store

(Relational DB)

Versioned code and data

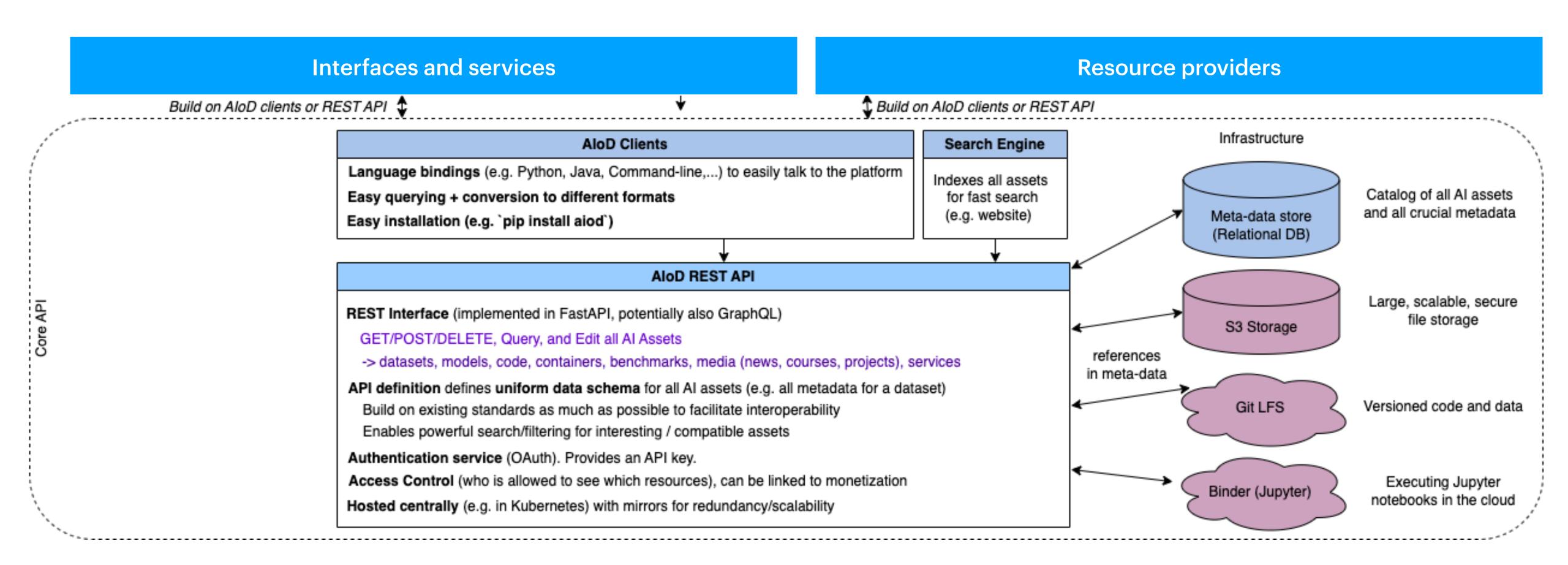
Executing Jupyter notebooks in the cloud

#### **Core API**

#### **Al-on-demand Core API**

Enables sharing and organisation of Al resources (directly, or via services or existing platforms) Unified interface to all Al resources for developing new services, interfaces, and applications

Easy to use libraries for development to accelerate research, applications, automate processes



## Interfaces and services

Ecosystem of services, registered with uniform descriptions

All services can offer their own interfaces that other services can build on (layers)

Services can be hosted anywhere, can have their own infrastructure (e.g. databases), can have their own monetization

Services can be designed for different audiences (e.g. academia, industry, experts, novices,...), showing/filtering all AloD resources in different ways d party service: (App Store)

#### Services from other EU projects

Matchmaking services SME-focused services Trustworthiness checks Other services

AutoML services Data quality services

١.

Build on core services

Build on AloD clients/API directly

#### Interfaces

#### Websites

Search / Gallery of resources Share resources (drag and drop) Project websites

Chatbots / NLP Semantic search

#### Core (1st tier) services

Experiment engines (e.g. Al4Experiments)

**HPC / Compute Services** 

Hosted Jupyter notebooks

Benchmarking services

ETL jobs, e.g. meta-data improvement

Build on AloD clients or REST API

**Core API** 

## Resource providers

#### **Data Platforms**

OpenML.org HuggingFace Zenodo

PapersWithCode

#### Other Platforms

European Language Grid European Open Science Cloud

•••

Users can store resources directly in AloD (if infrastructure allows) or choose to use integrated existing platforms

AloD offers aggregated view/search over all available resources

Interact via connector \$\forall \$



#### AloD Connectors

Handle integration of existing platforms (with own APIs)

Map external resources to AloD schema and register them

Periodically indexes/updates existing resources

Users uploading resources to these platforms will also find them in AloD

One connector per external platform

#### AloD Nodes

#### Locally installed mini-platform

Makes local resources visible in AloD Handles communication with AloD API

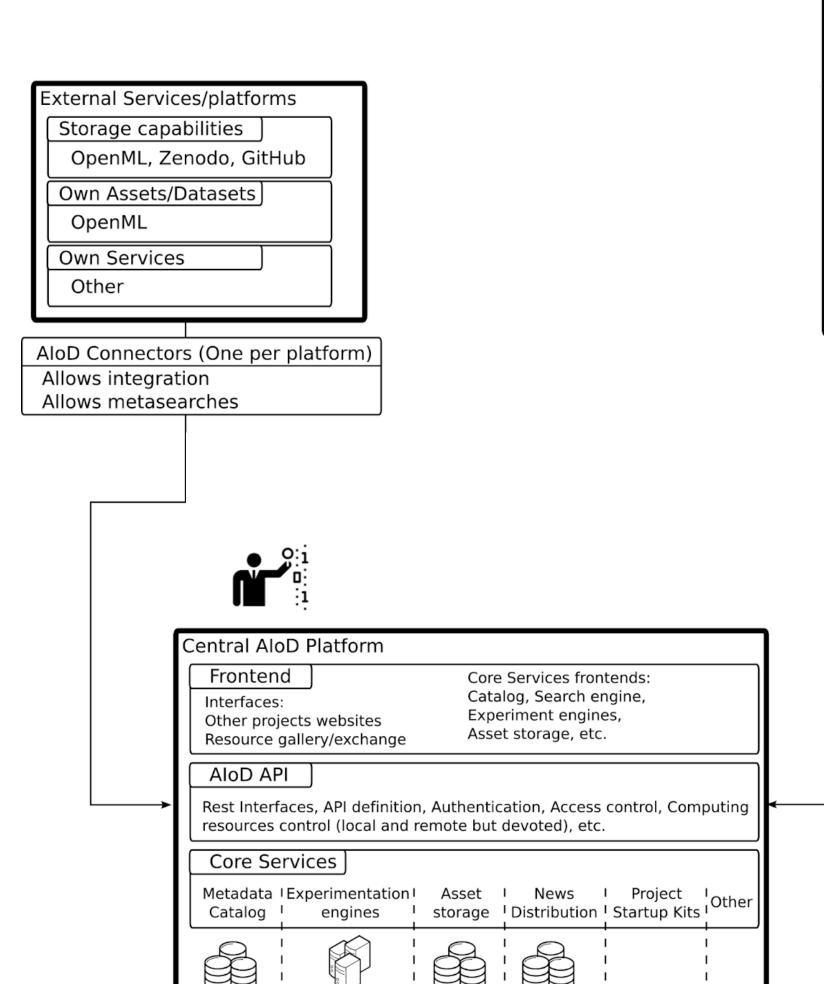
E.g. for companies, universities, hospitals that want to keep data local

Build on AloD clients or REST API

**Core API** 

Providers Resource

## **API-Resource connectors**



ı RAM, FPGAs,... ı storage, ı

Map APIs

Translate between data schema

AloD nodes

Frontend

Local view, customizations

AloD API

Local-only, or merge global/local

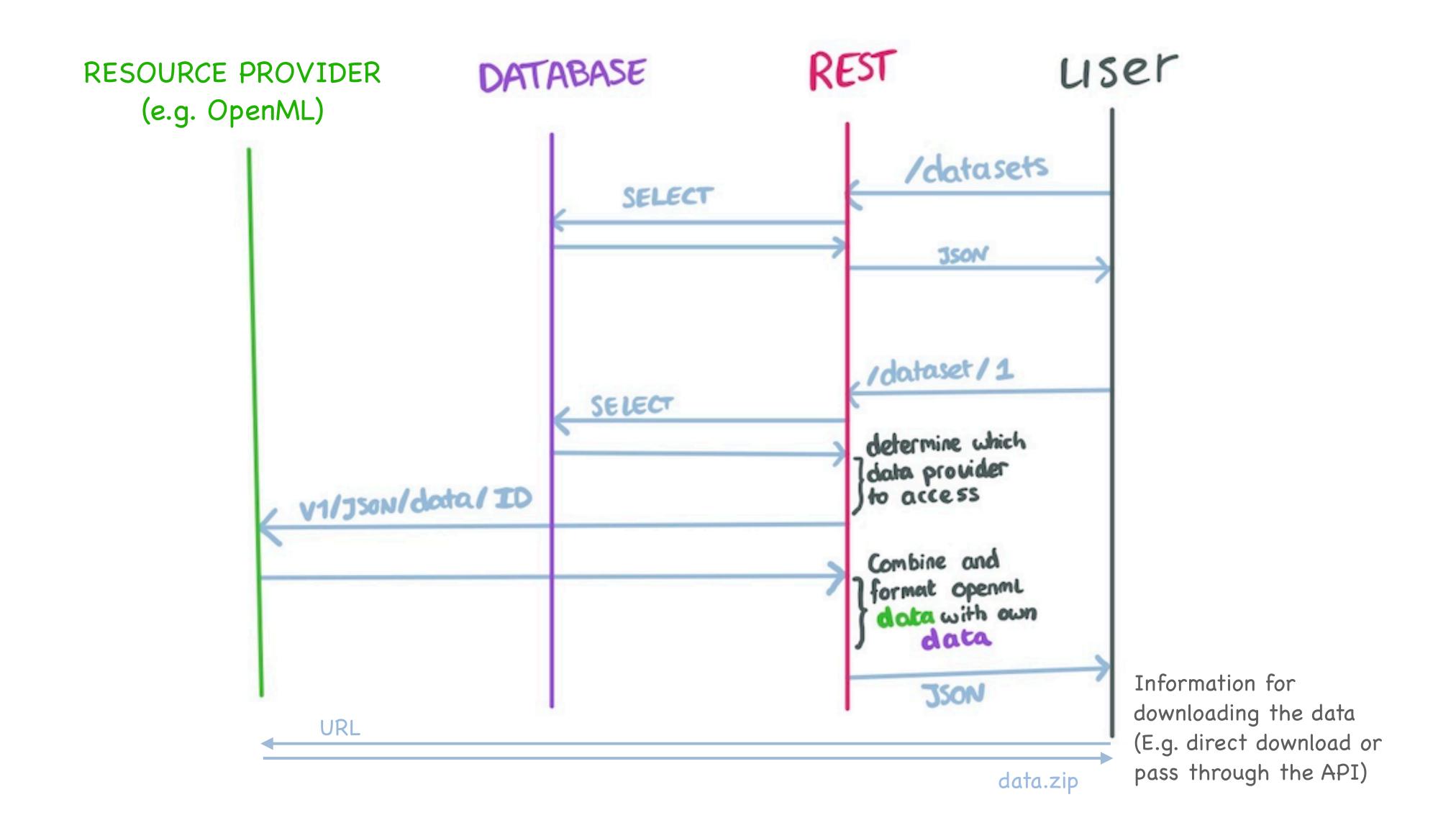
Core Services

Metadata | Experimentation | Asset Catalog | engines | storage

SQL DBs? | CPUs, GPUs, | S3-like storage, git

Synchronize (universal connector)

### **API-Resource connectors**



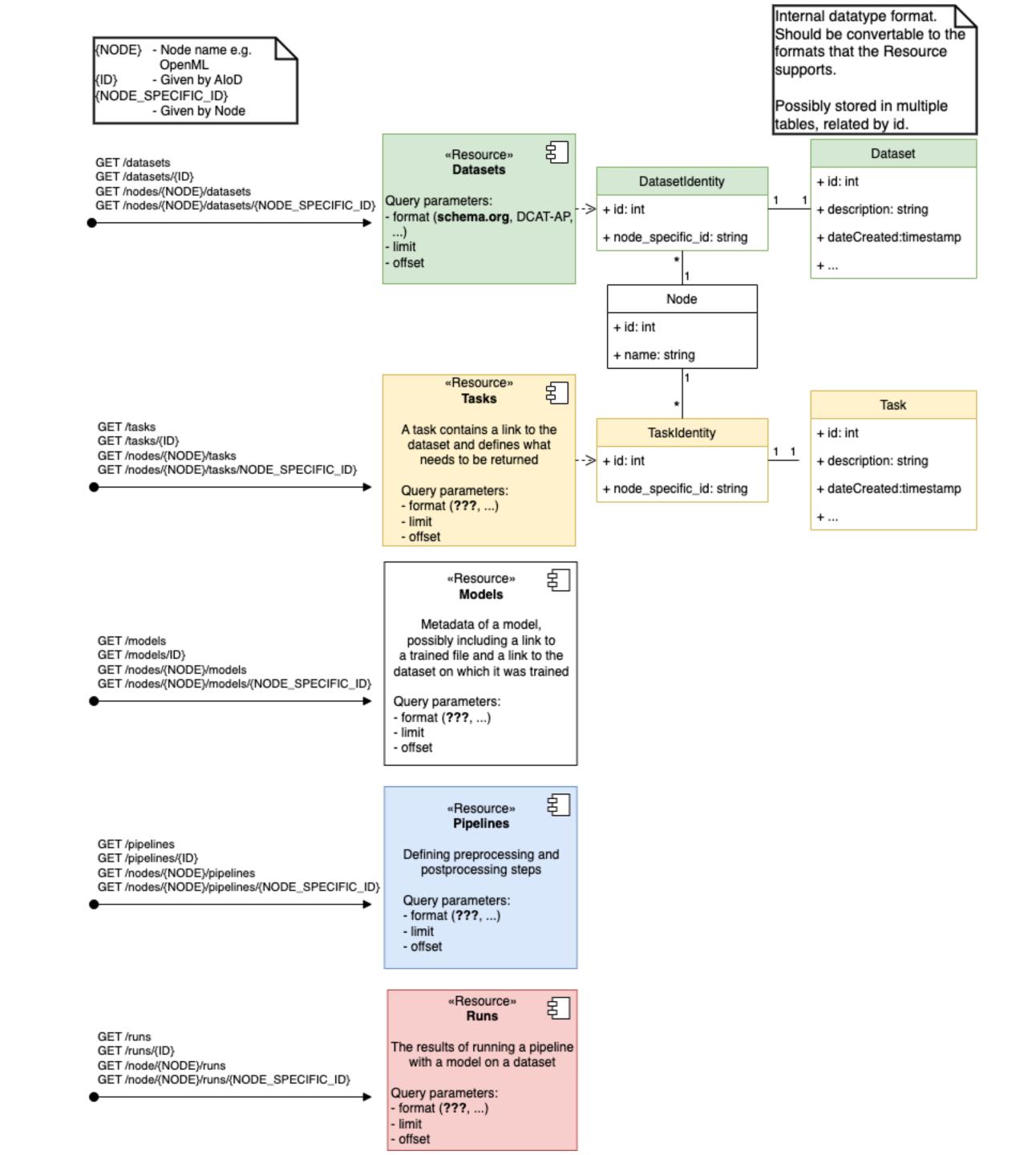
# API spec (rough sketch)



#### **Prototype implemented**

(only datasets and papers)

https://github.com/openml-labs/server-demo



## API spec (sketch)

