

Serving modes

MLOPS DEPLOYMENT AND LIFE CYCLING



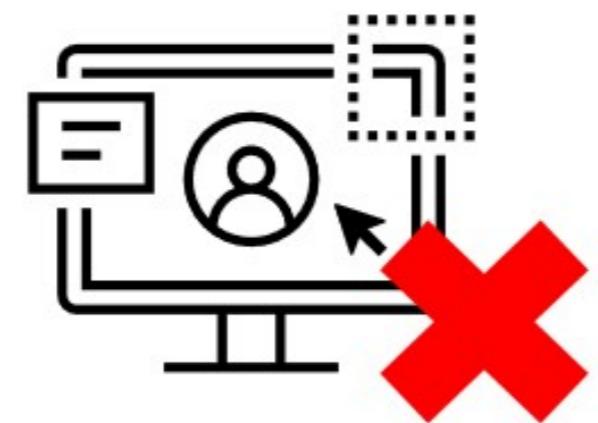
Nemanja Radojkovic

Senior Machine Learning Engineer

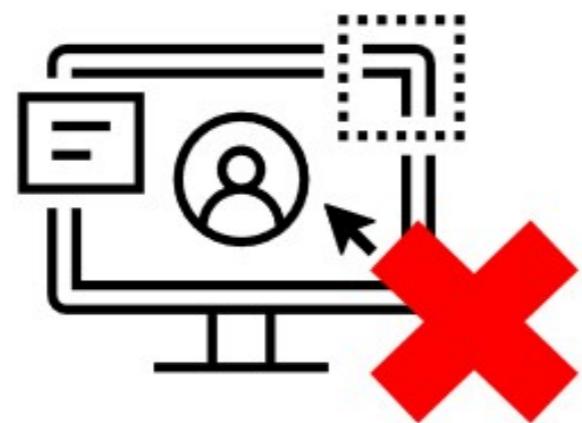
Model as a service



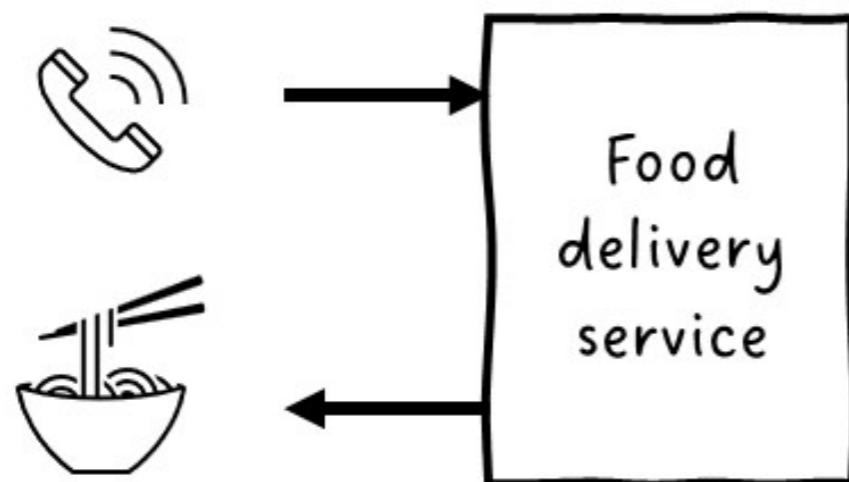
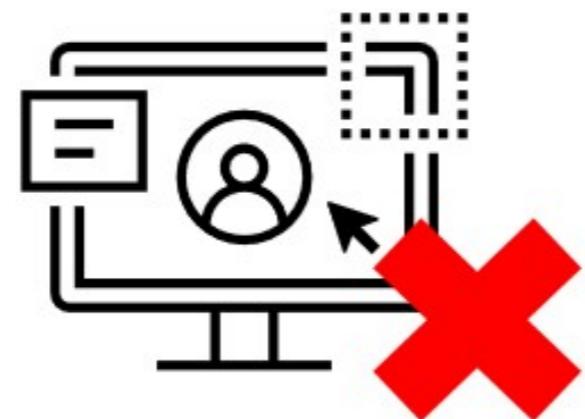
Model as a service



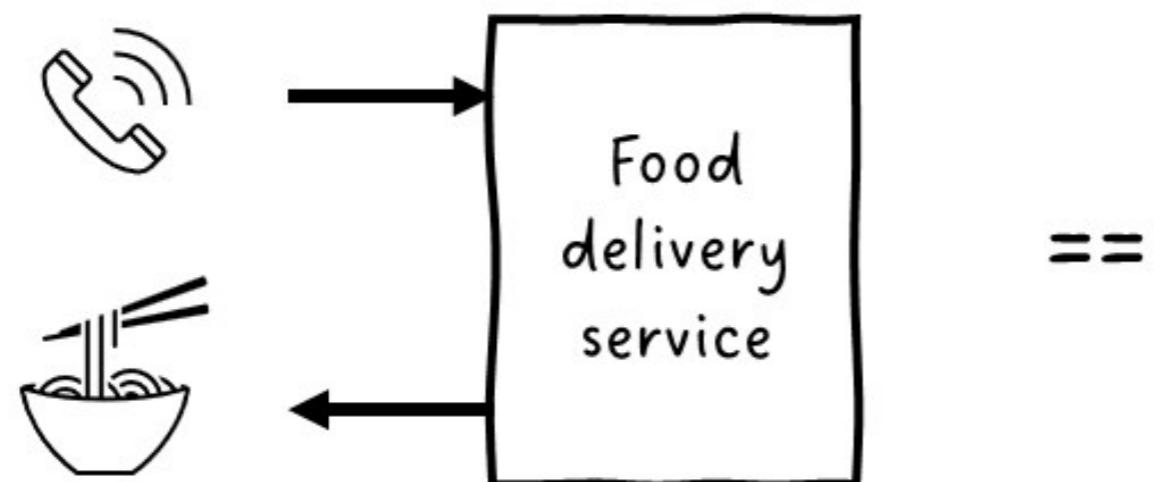
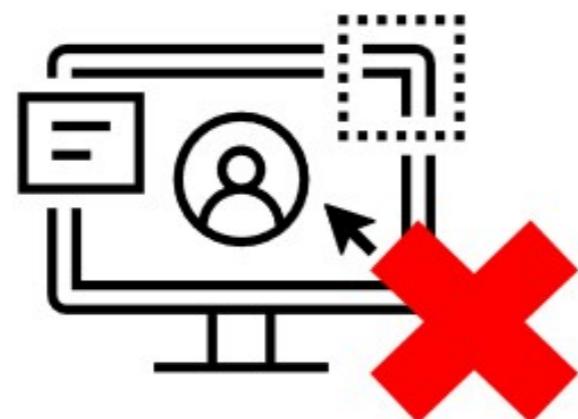
Model as a
service



Model as a service



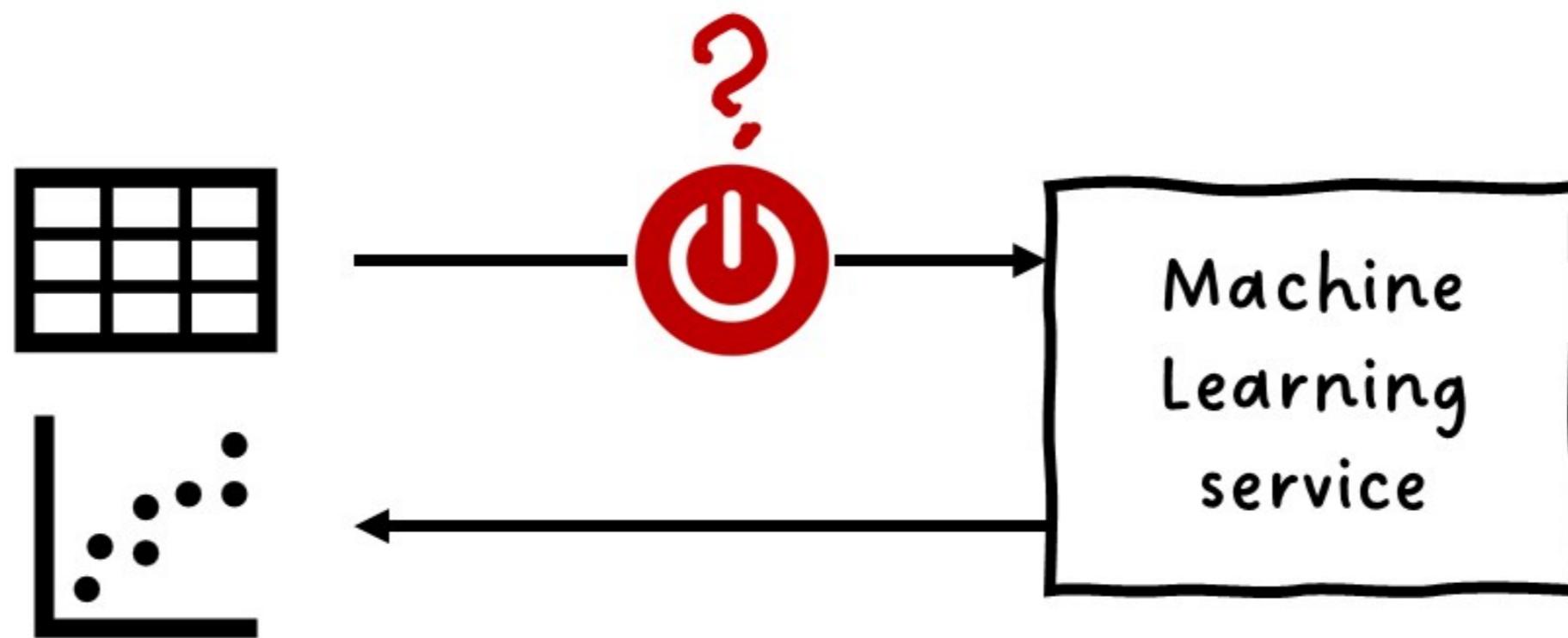
Model as a service

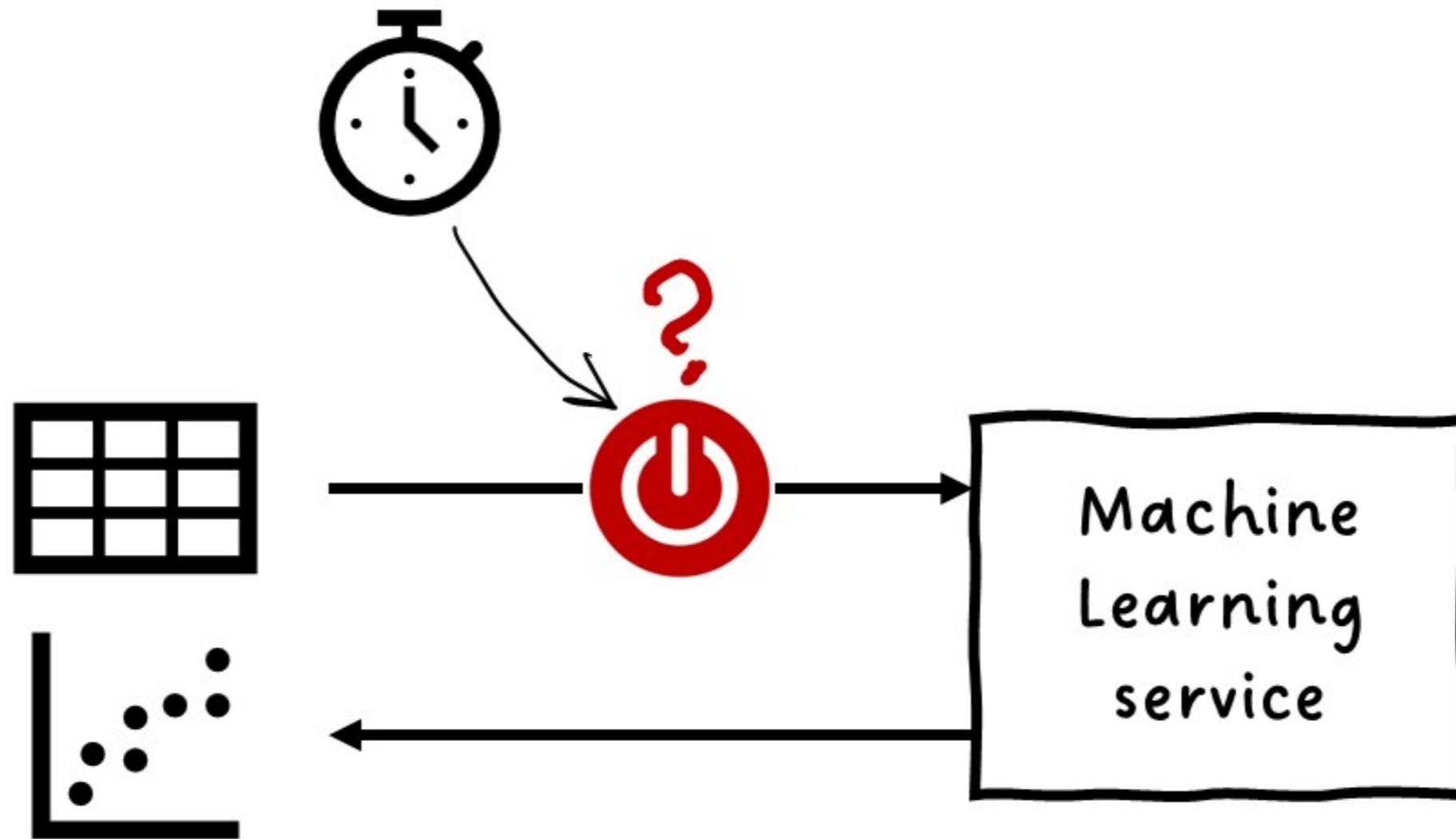


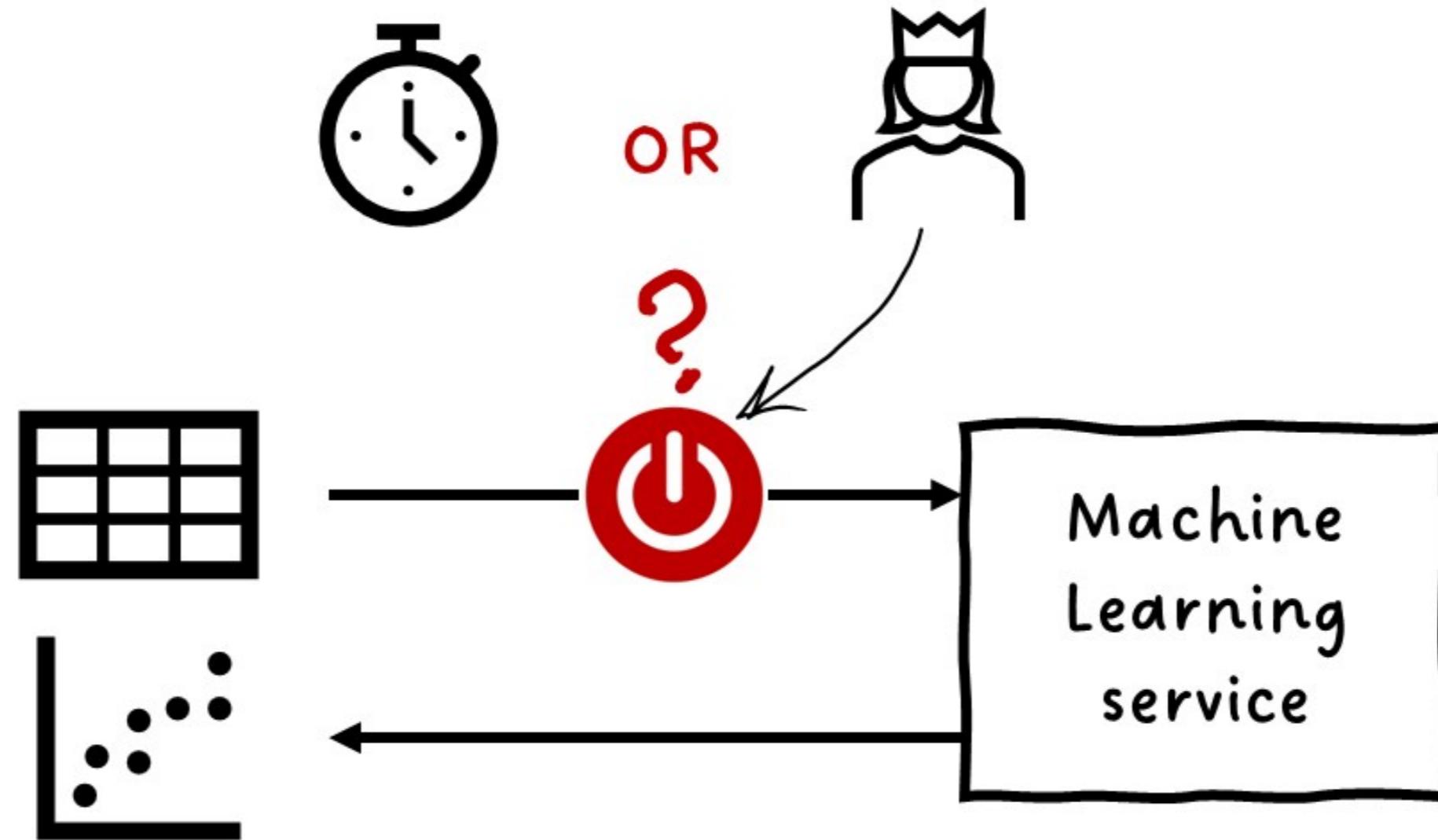
Serving and serving mode

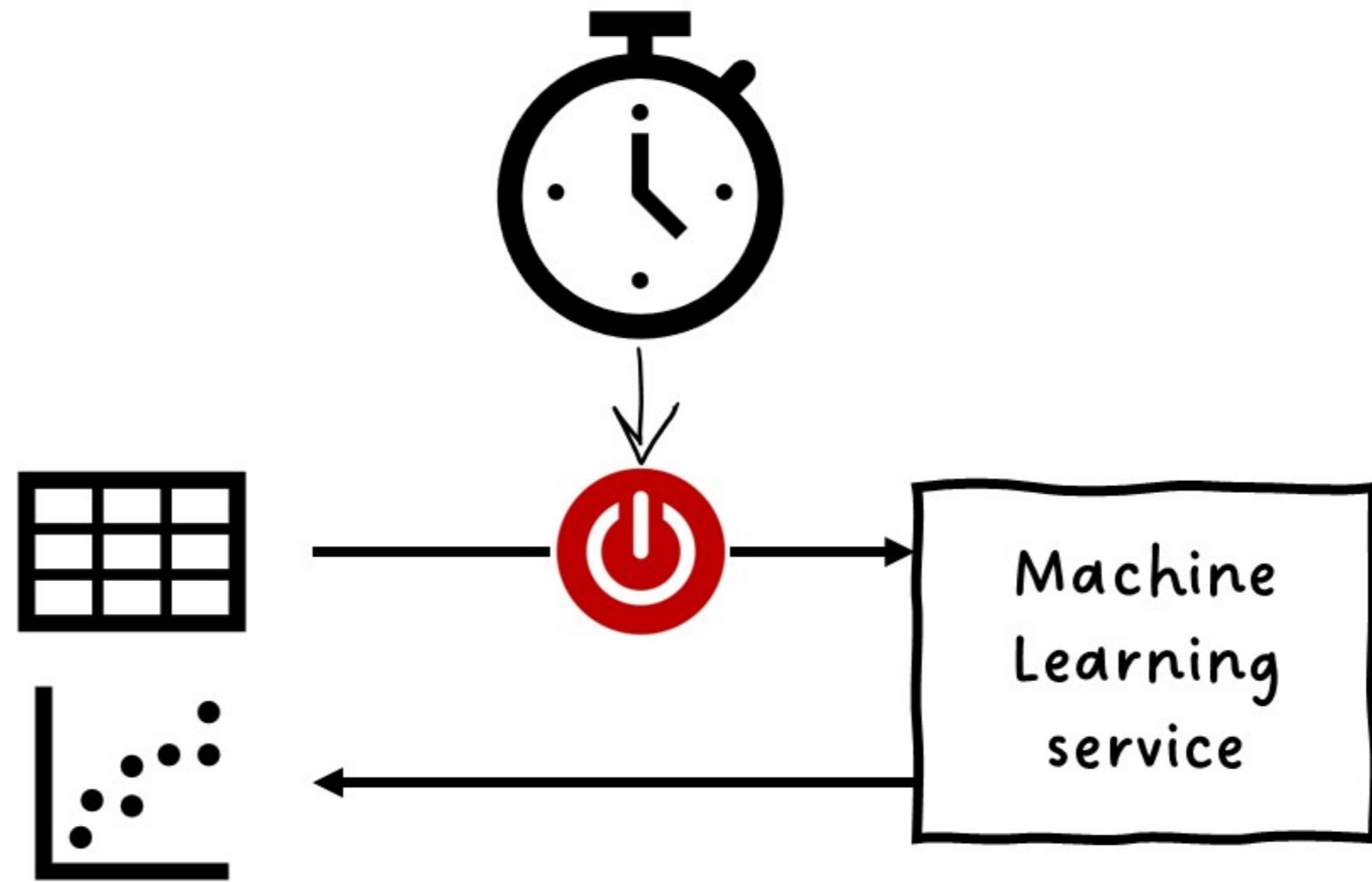
- Providing prediction service == Model serving
- Implementation of a specific type of serving == Serving mode

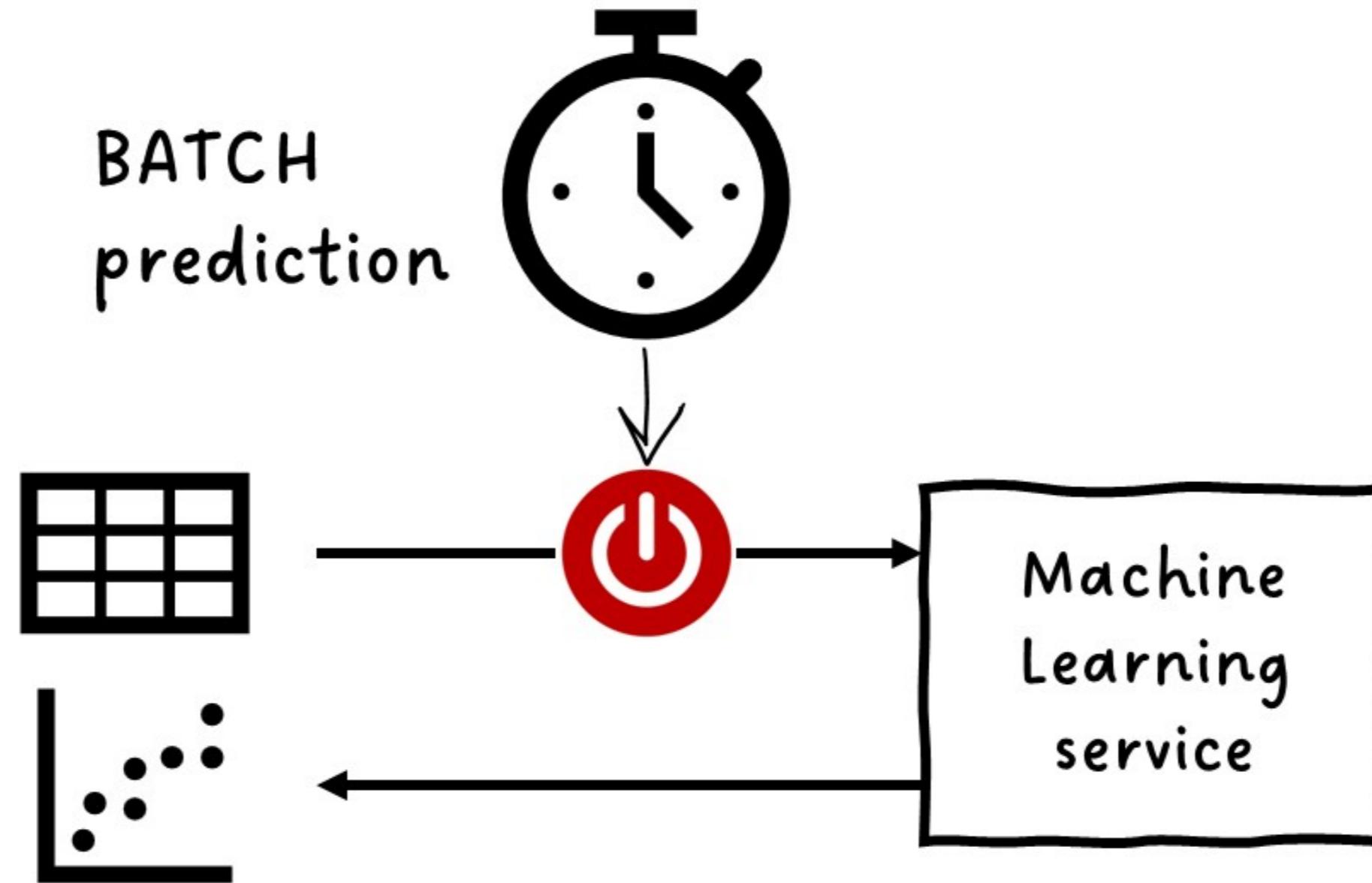
Choose carefully!

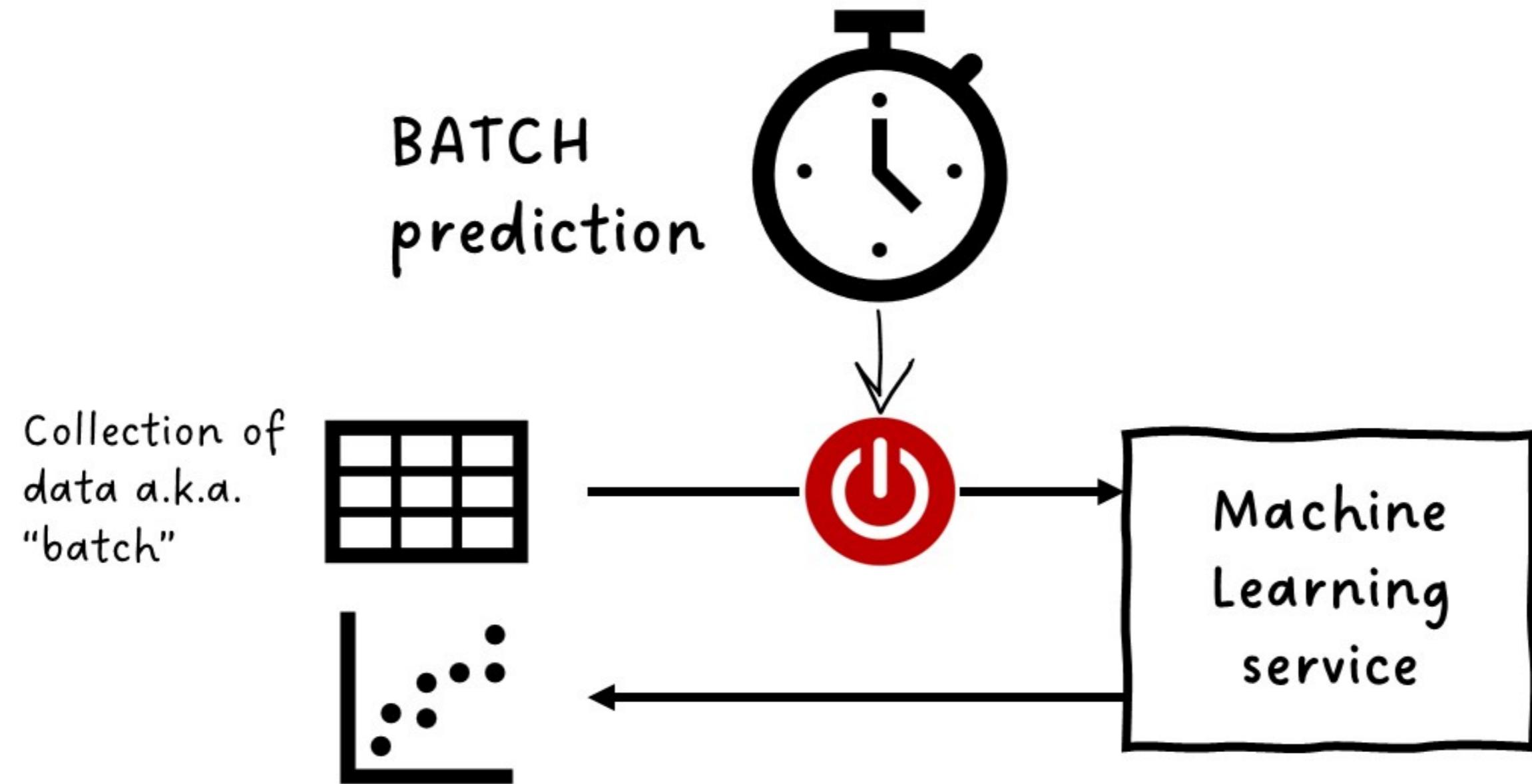


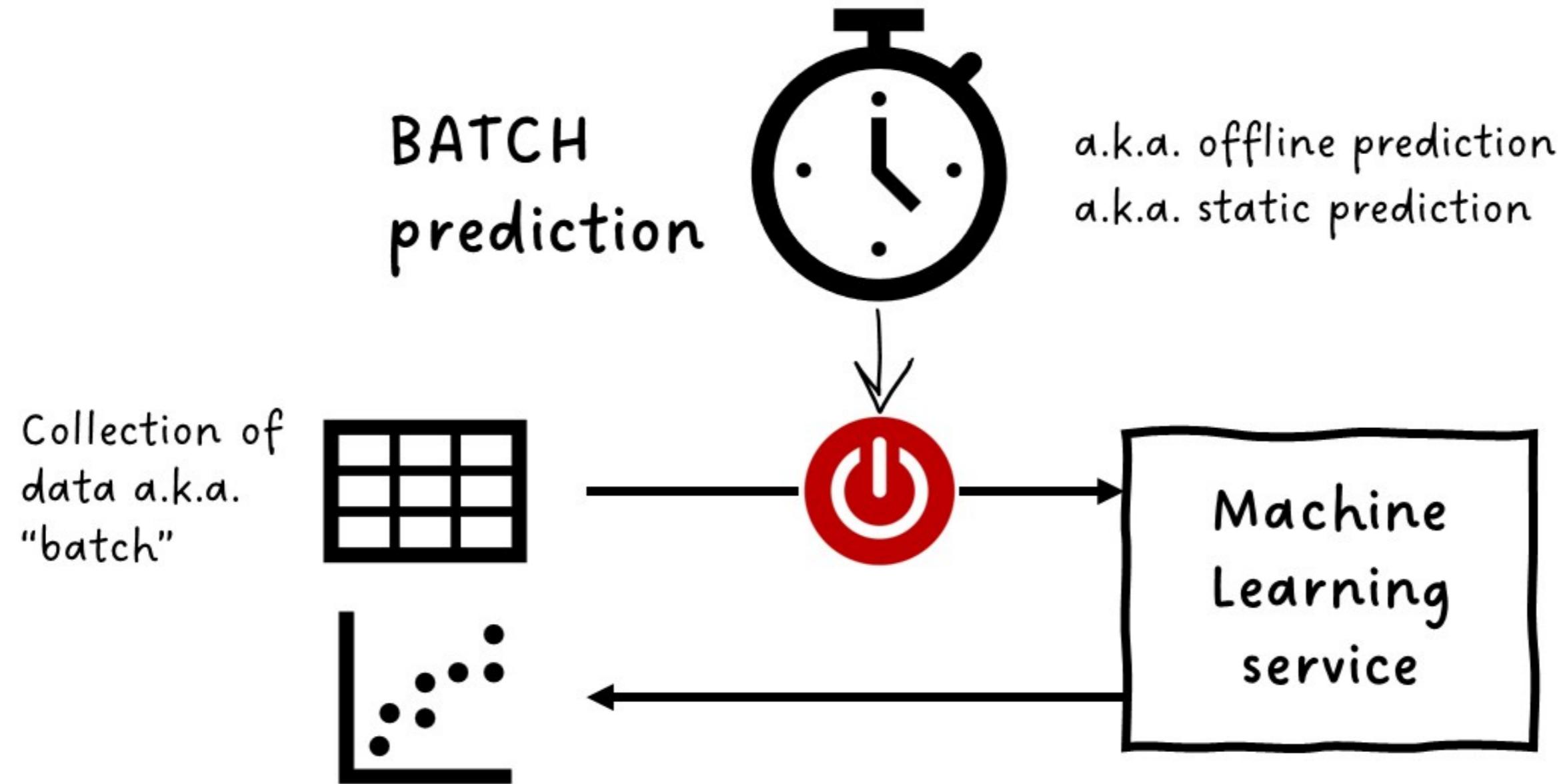






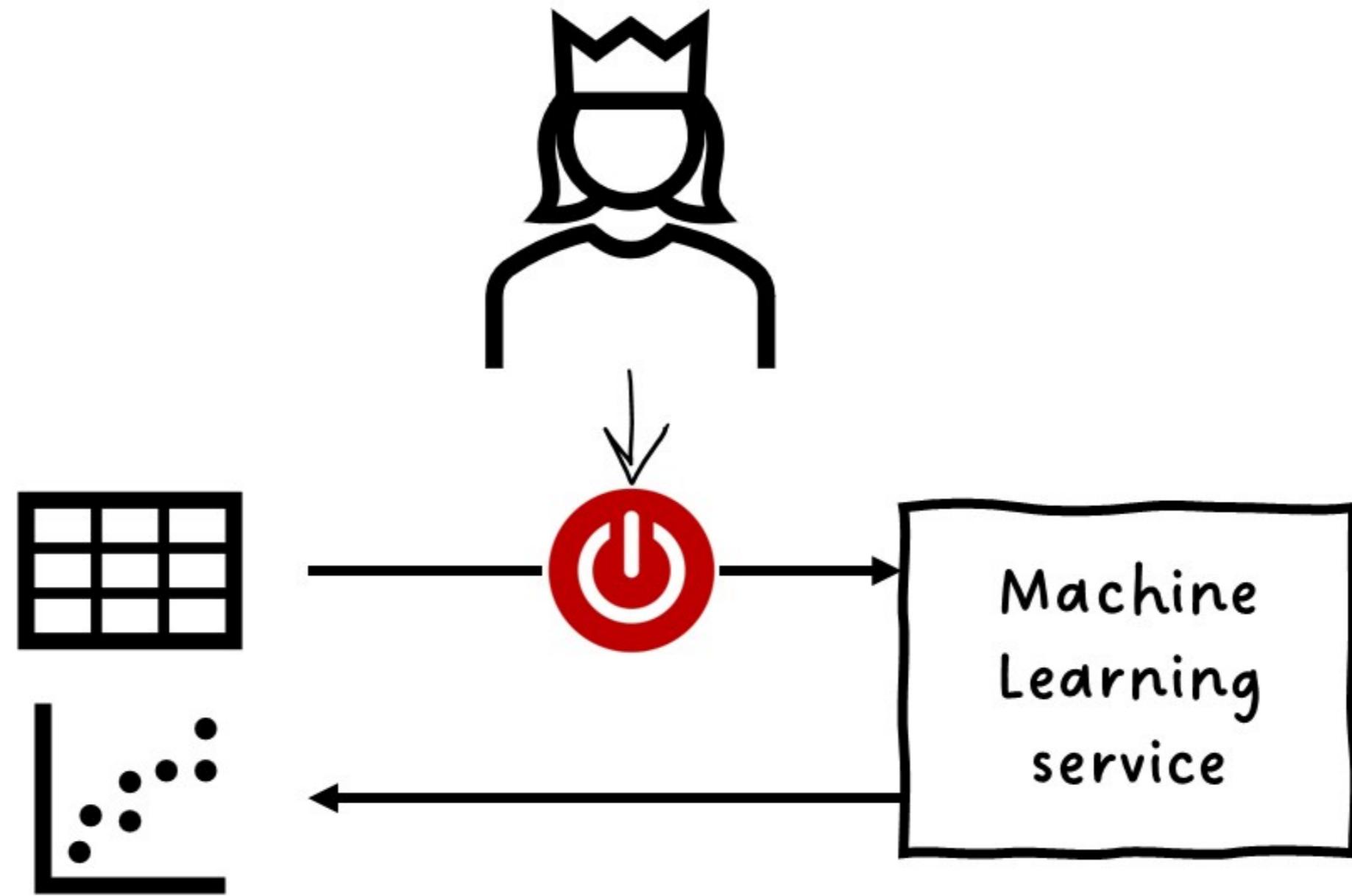


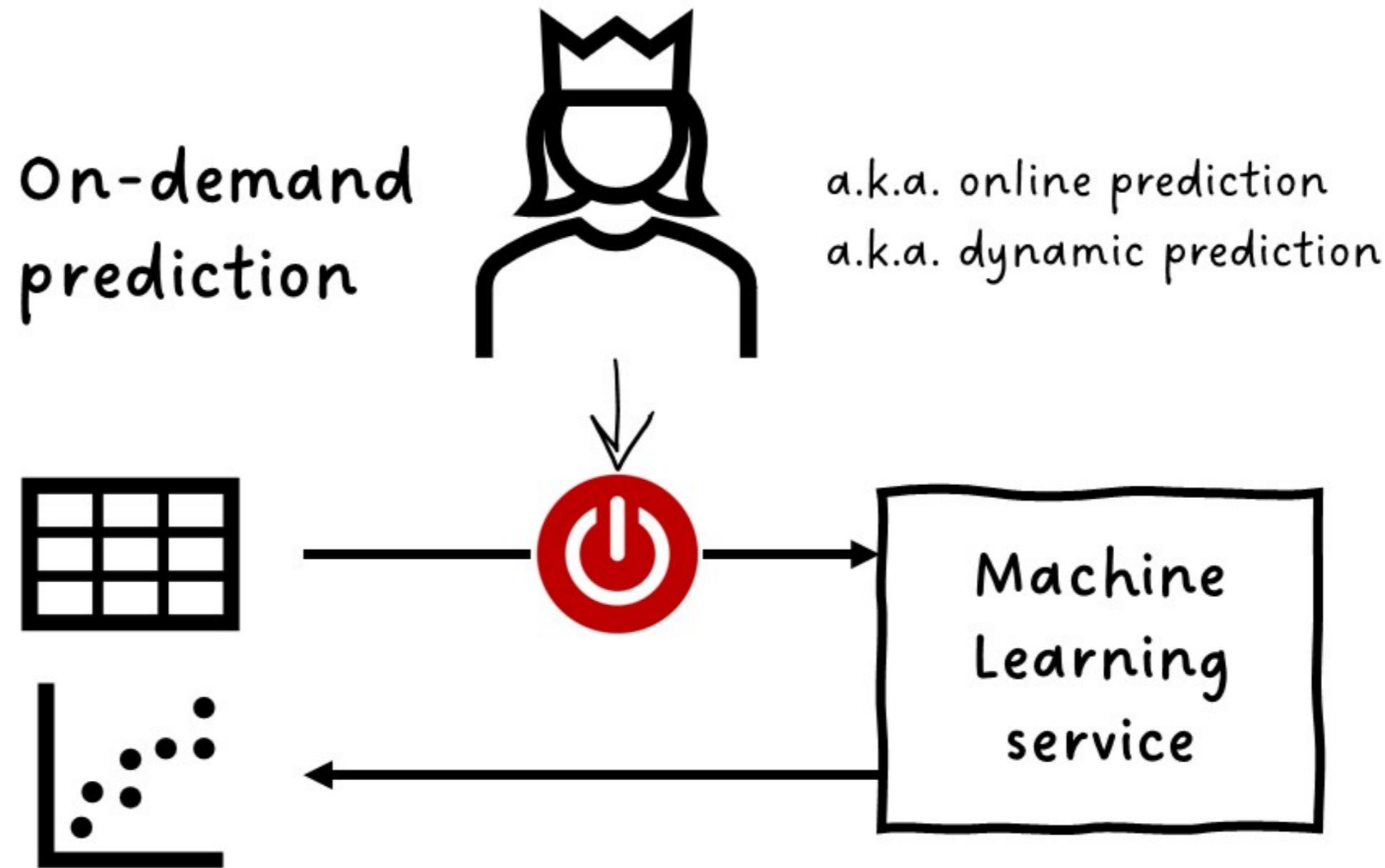


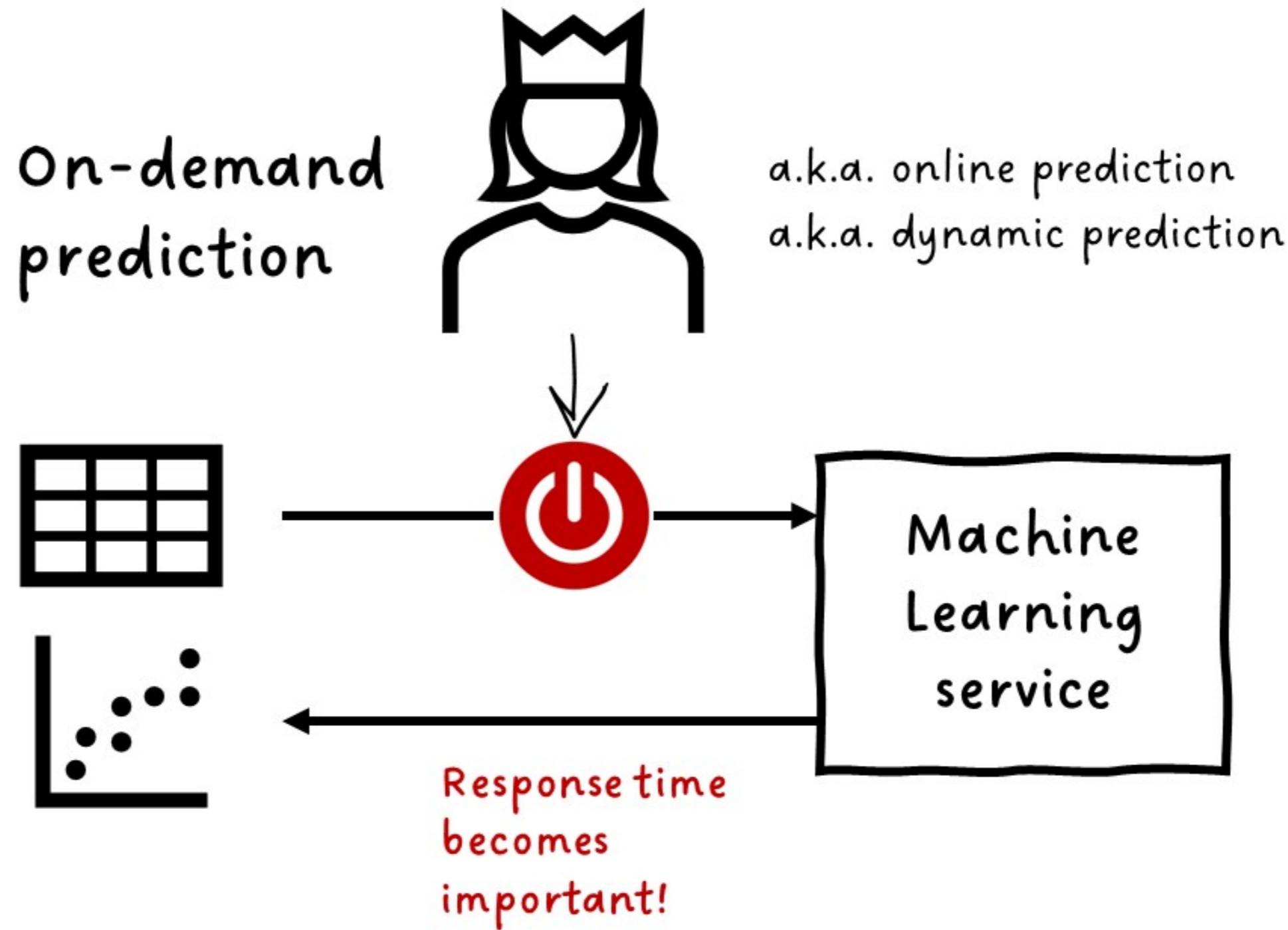


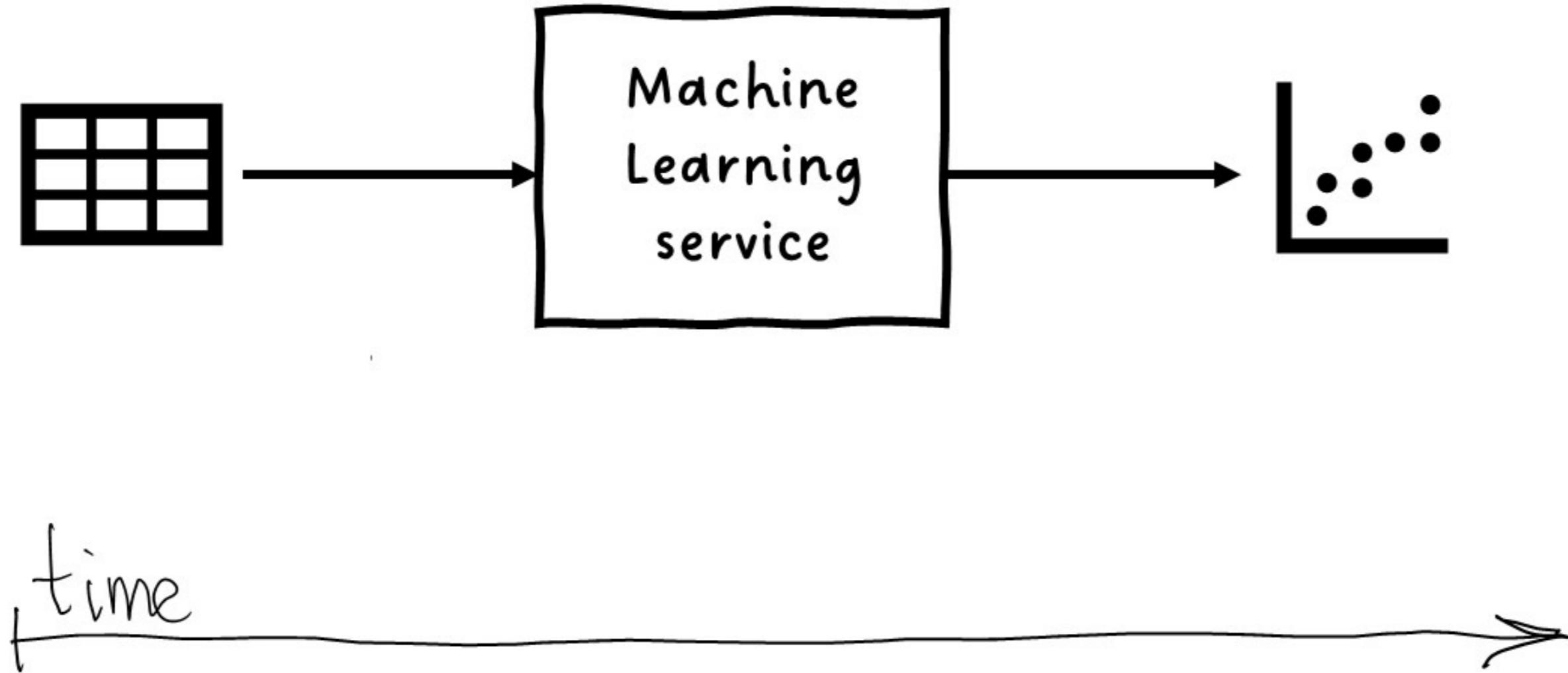
Batch prediction: Keep it simple

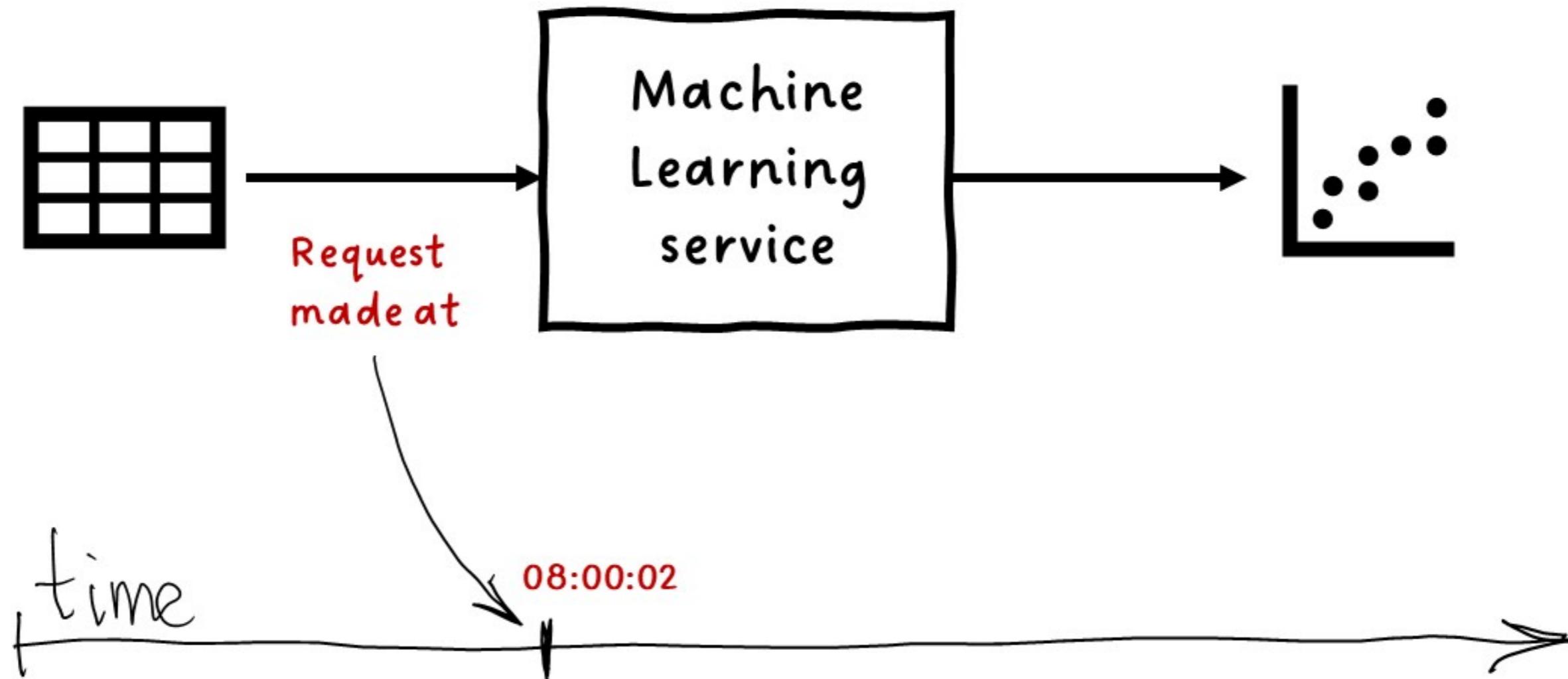
- Batch prediction is the simplest
- If use case allows it, go for it
- Good fit: monthly generation of sales forecasts

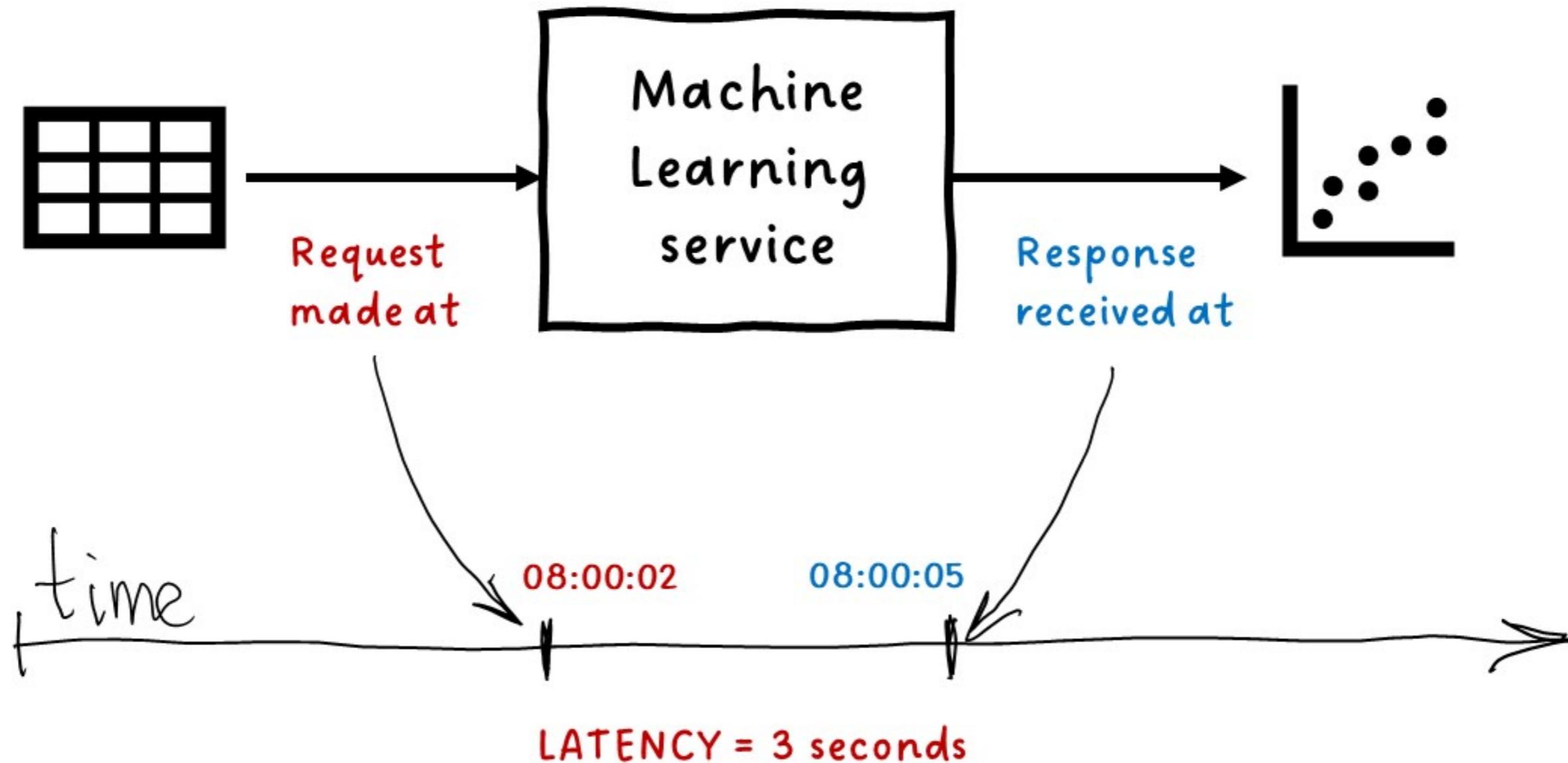












Acceptable latency

What is acceptable?

- < 1 hour?
- < 1 minute?
- < 1 second?
- < 1 millisecond?

Near-real time prediction a.k.a. Stream processing

Acceptable latency $\sim= X$ minutes

Also known as ***stream processing*** (requests and responses form "data streams")

Real-time prediction

Acceptable latency < 1 sec

Example:

- Credit card fraud detection
- Late prediction as good as useless

When latency is a priority

- Weaker, but faster model more valuable than a stronger, but slower one
- Models deployed to end user devices to reduce latency => "edge deployment"
 - ML-infused smartphone apps:
 - navigation apps
 - unlocking via facial recognition
 - image filters

Let's practice!

MLOPS DEPLOYMENT AND LIFE CYCLING

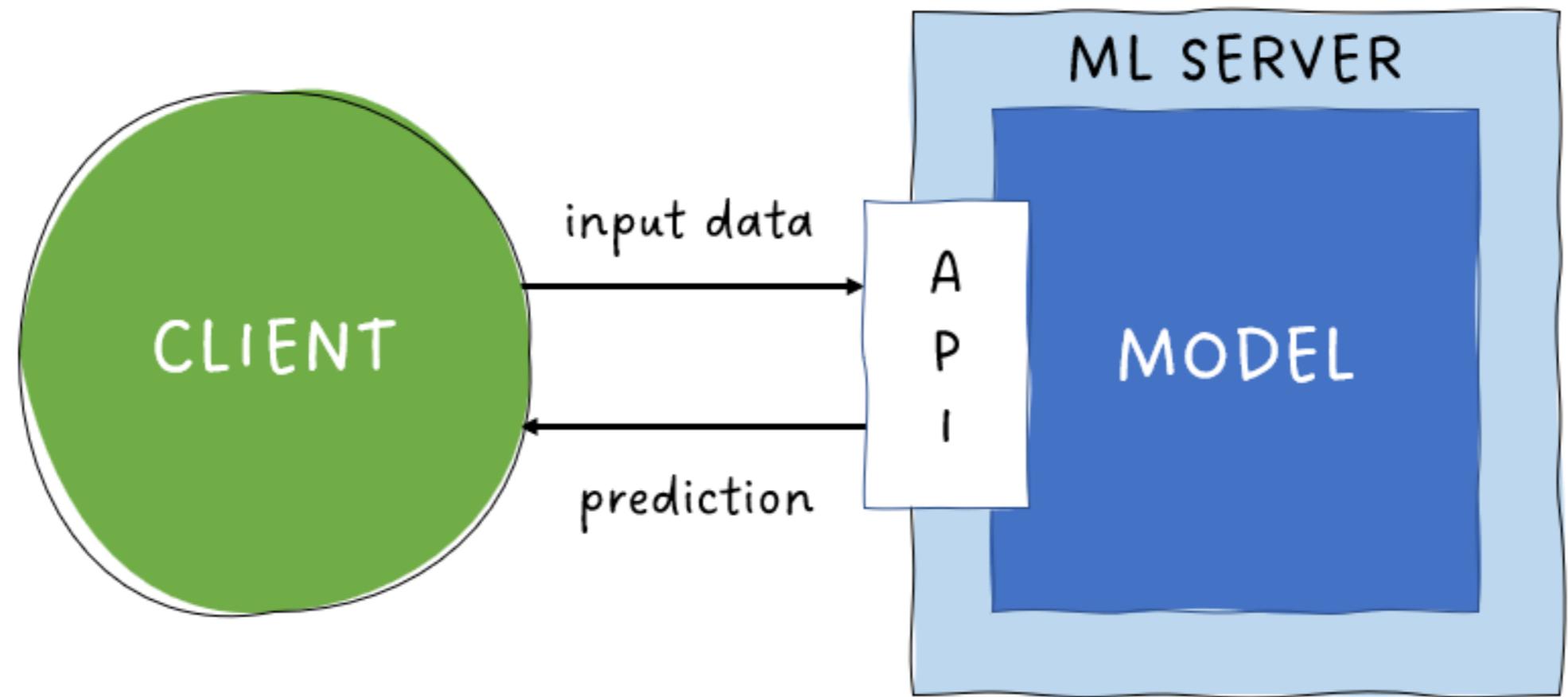
Building the API

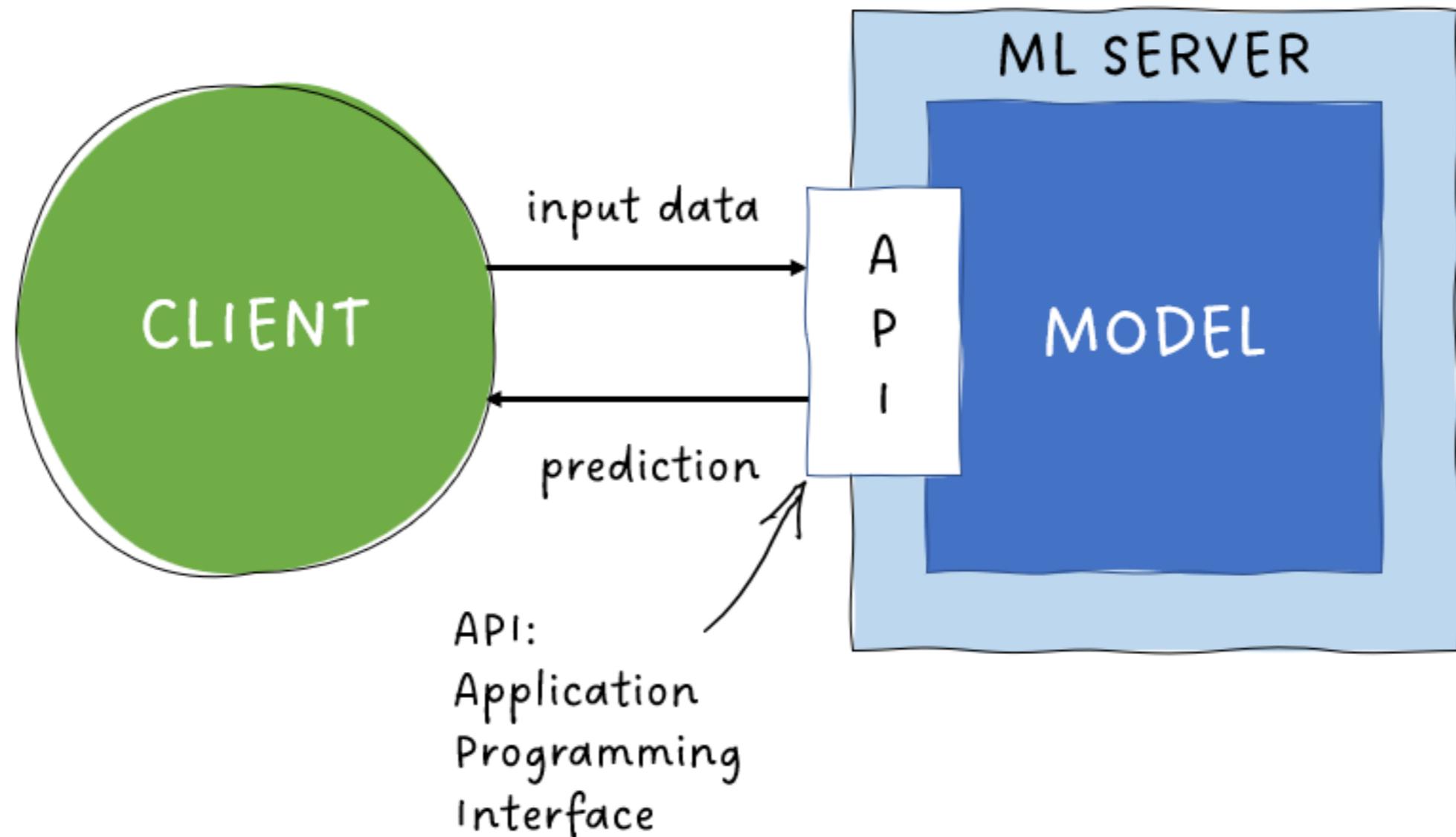
MLOPS DEPLOYMENT AND LIFE CYCLING

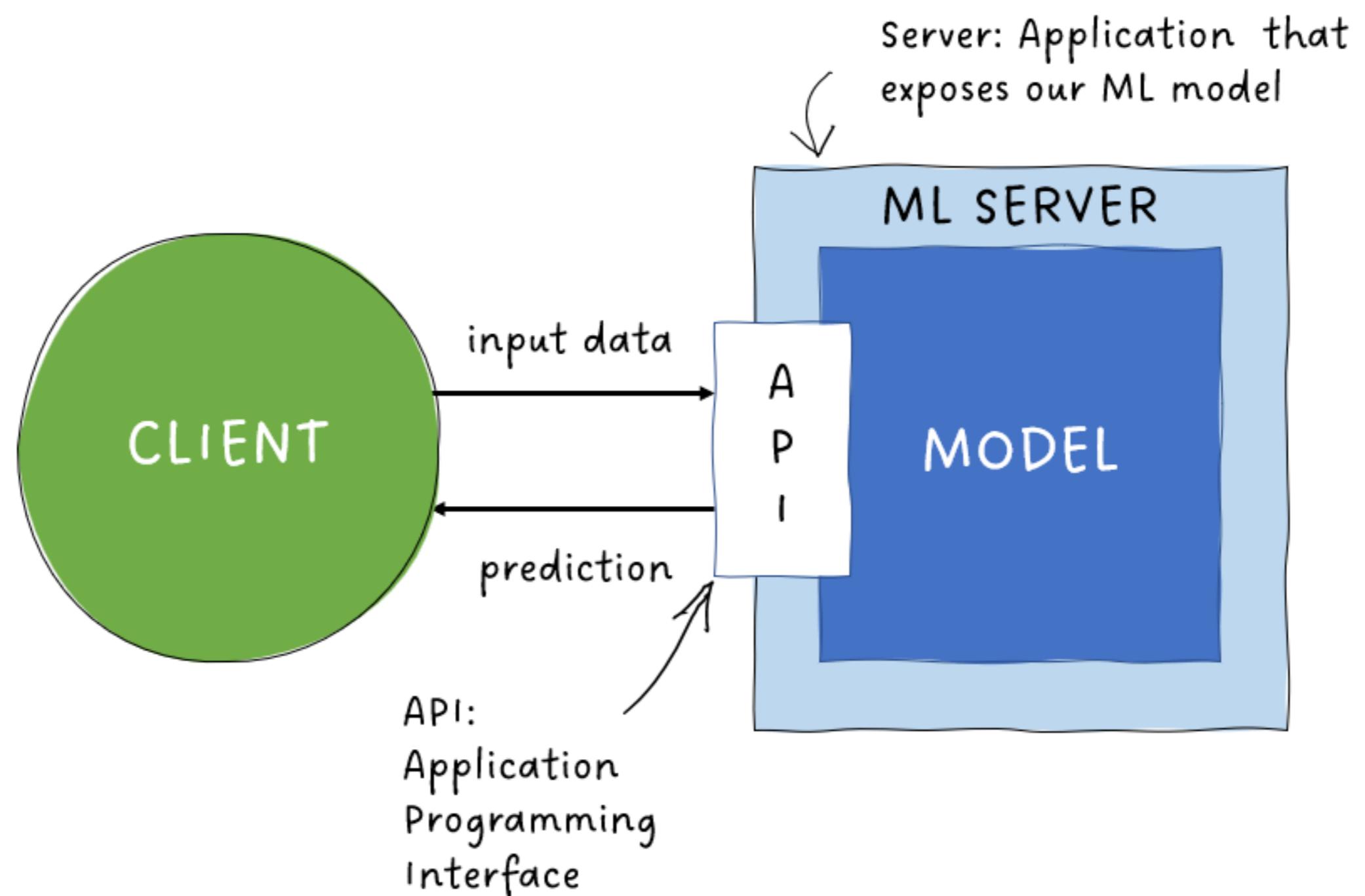


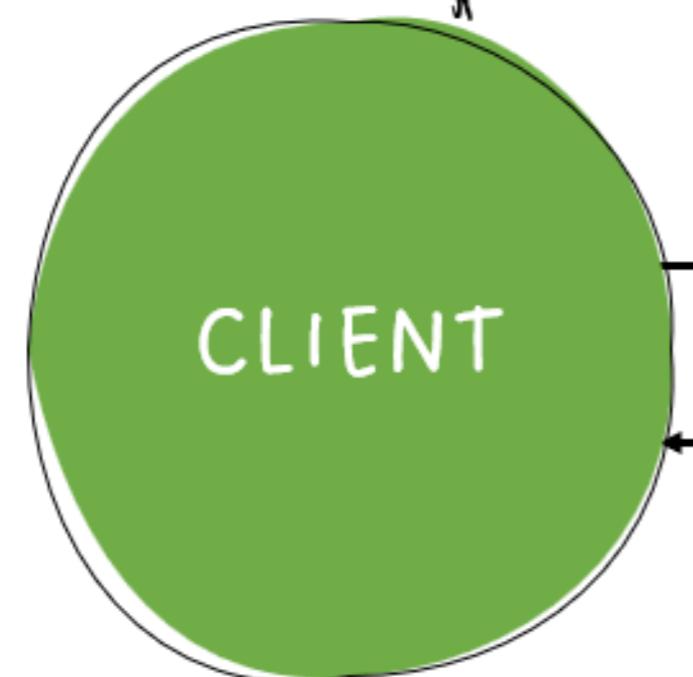
Nemanja Radojkovic

Senior Machine Learning Engineer

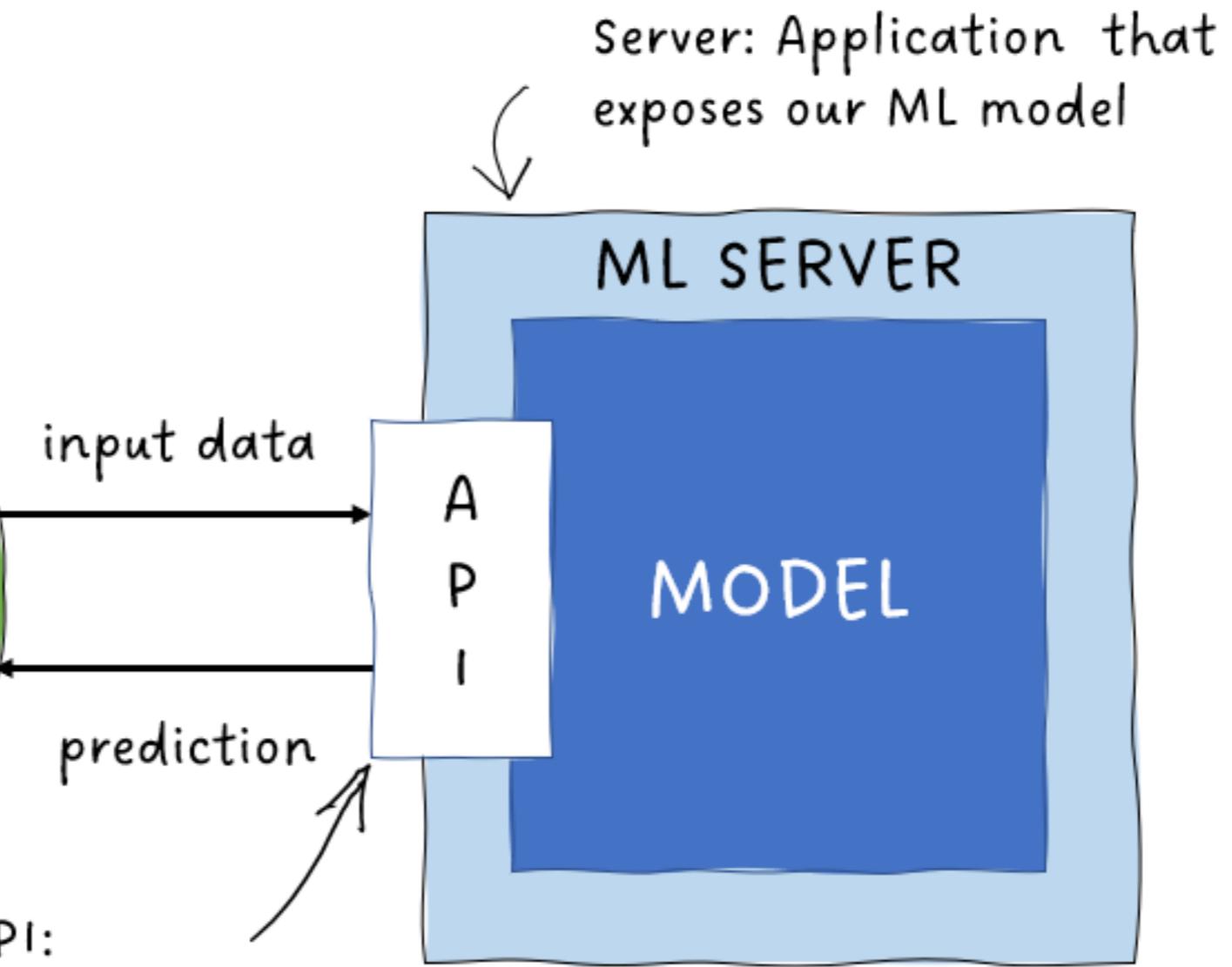








Client: Application that uses our model

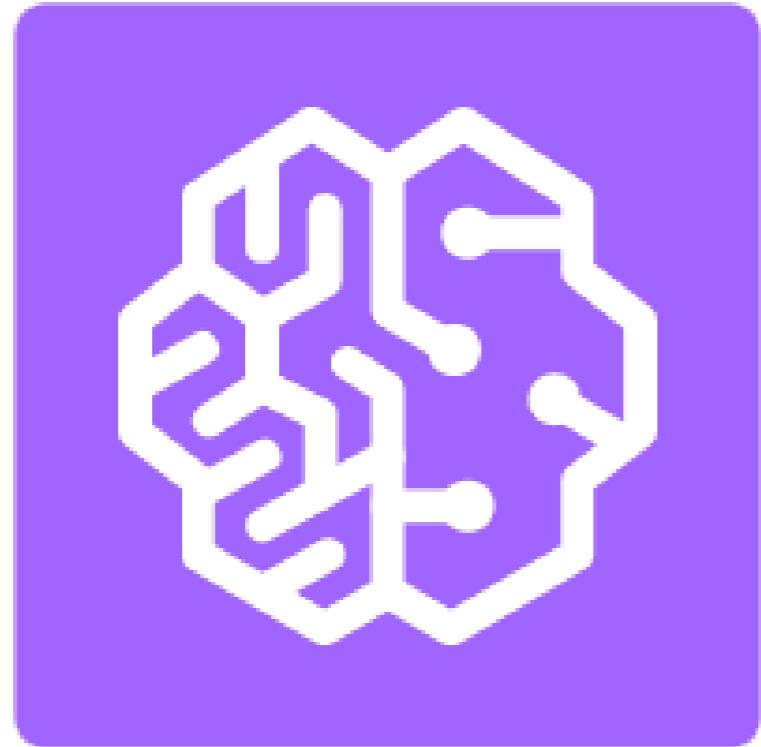


API:
Application
Programming
Interface

Service to the client

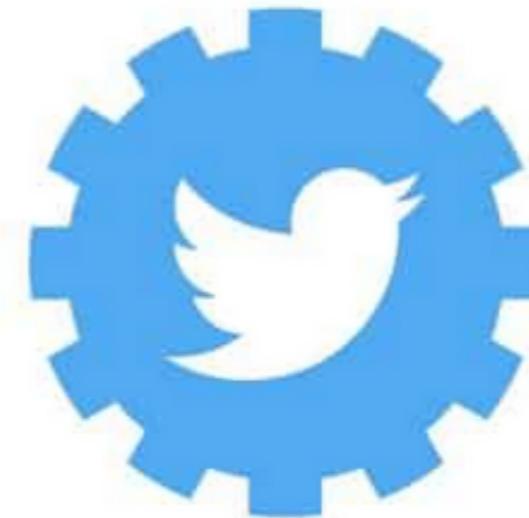
Running procedures on our server

Example: Sentiment analysis API for Amazon Comprehend



Access remote databases

Example: Twitter API, New York Times API



Twitter API

API architectures

1. REST: REpresentational State Transfer
2. RPC: Remote Procedure Call
3. SOAP: Simple Object Access Protocol

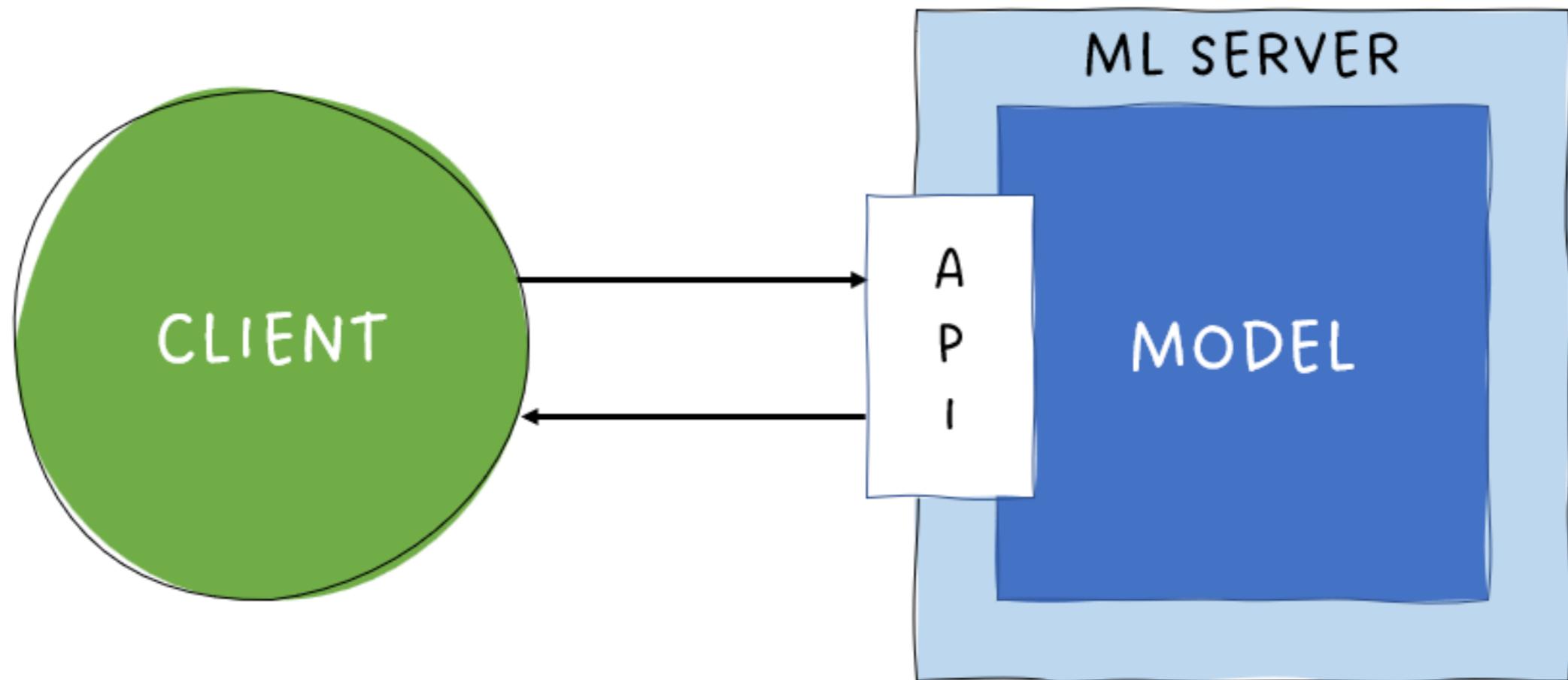
In this course: Architecture-agnostic view

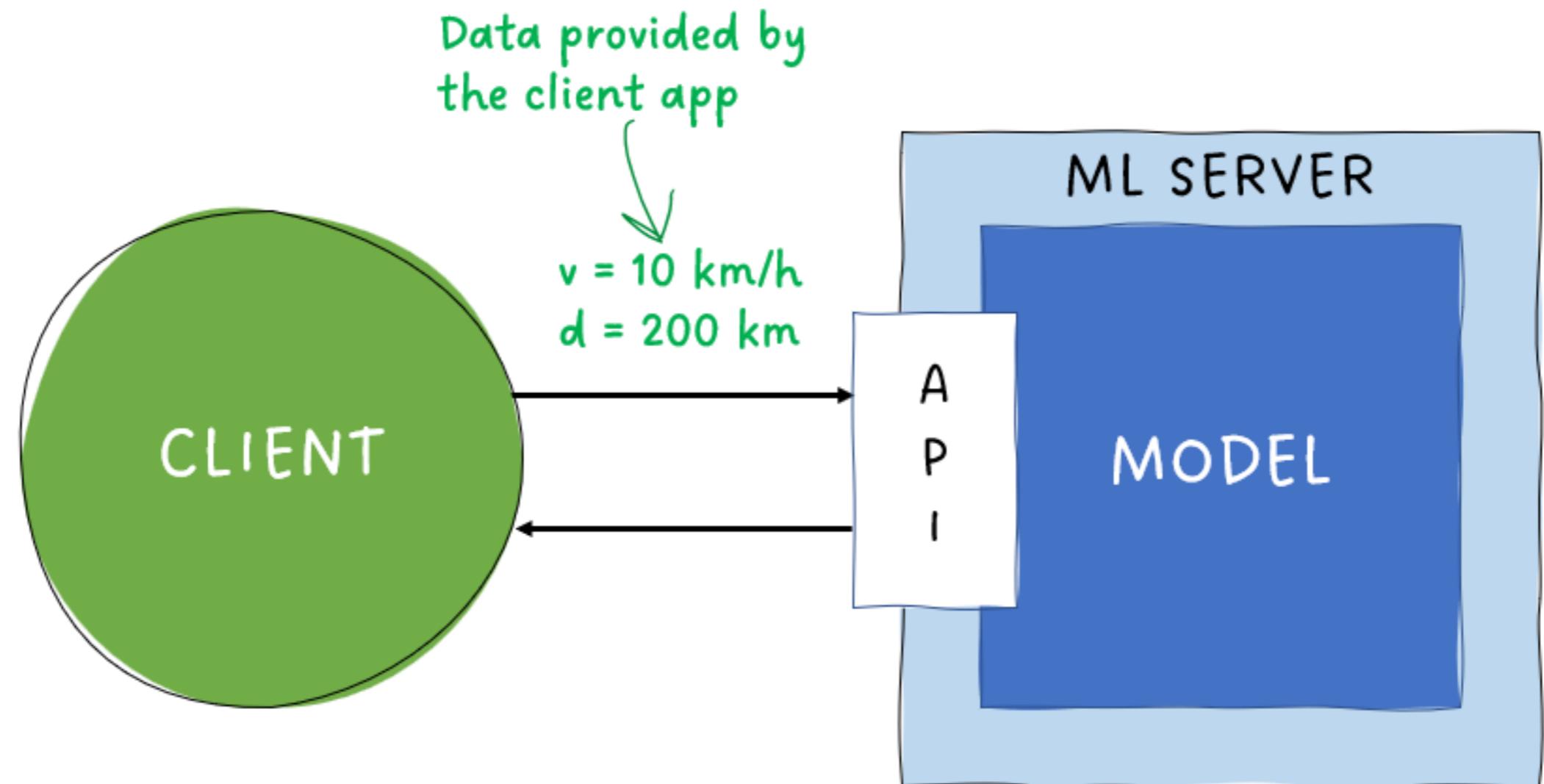
Example: Navigation app

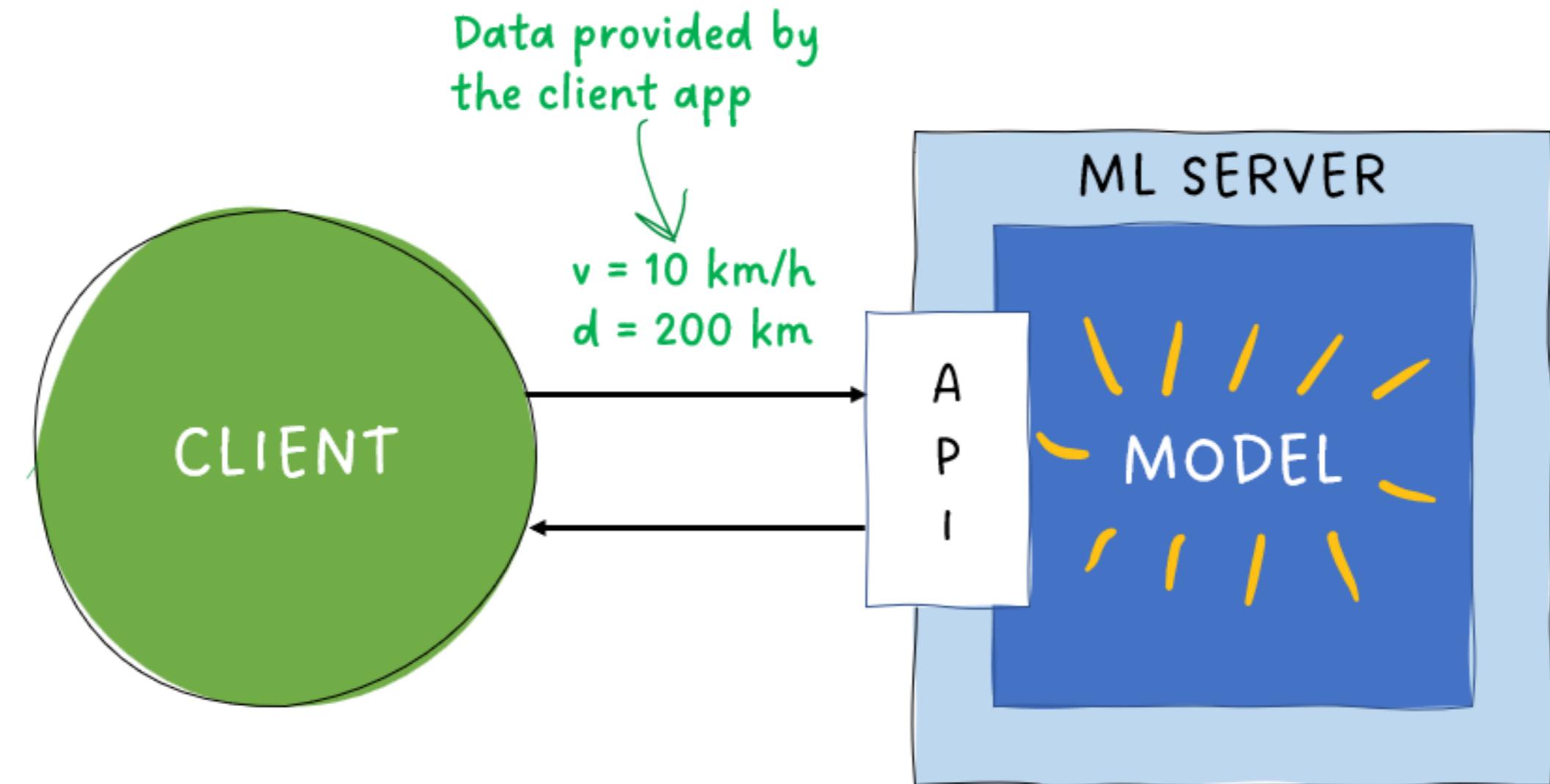


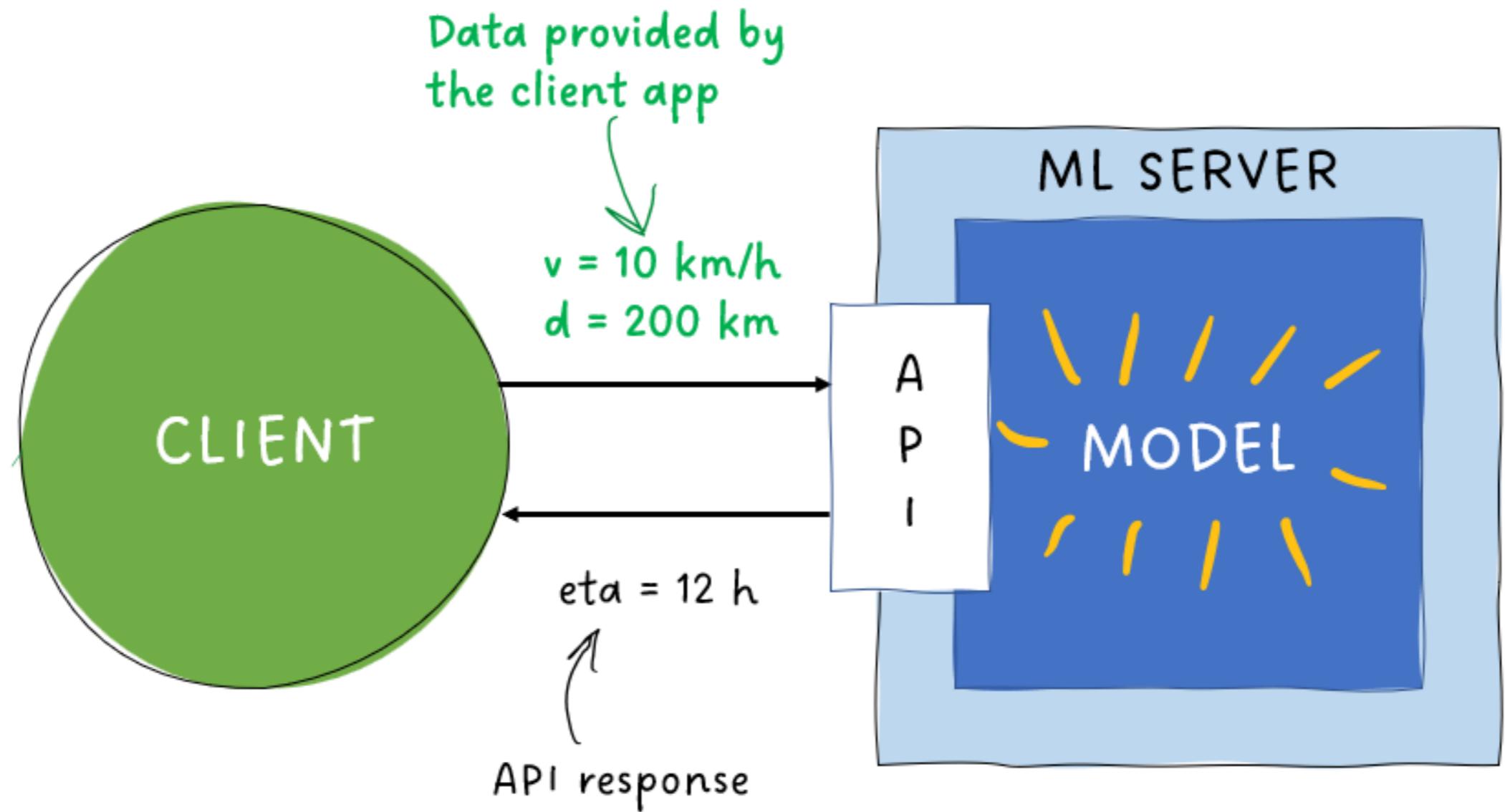
API purpose: Enable client apps to calculate the estimated time of arrival

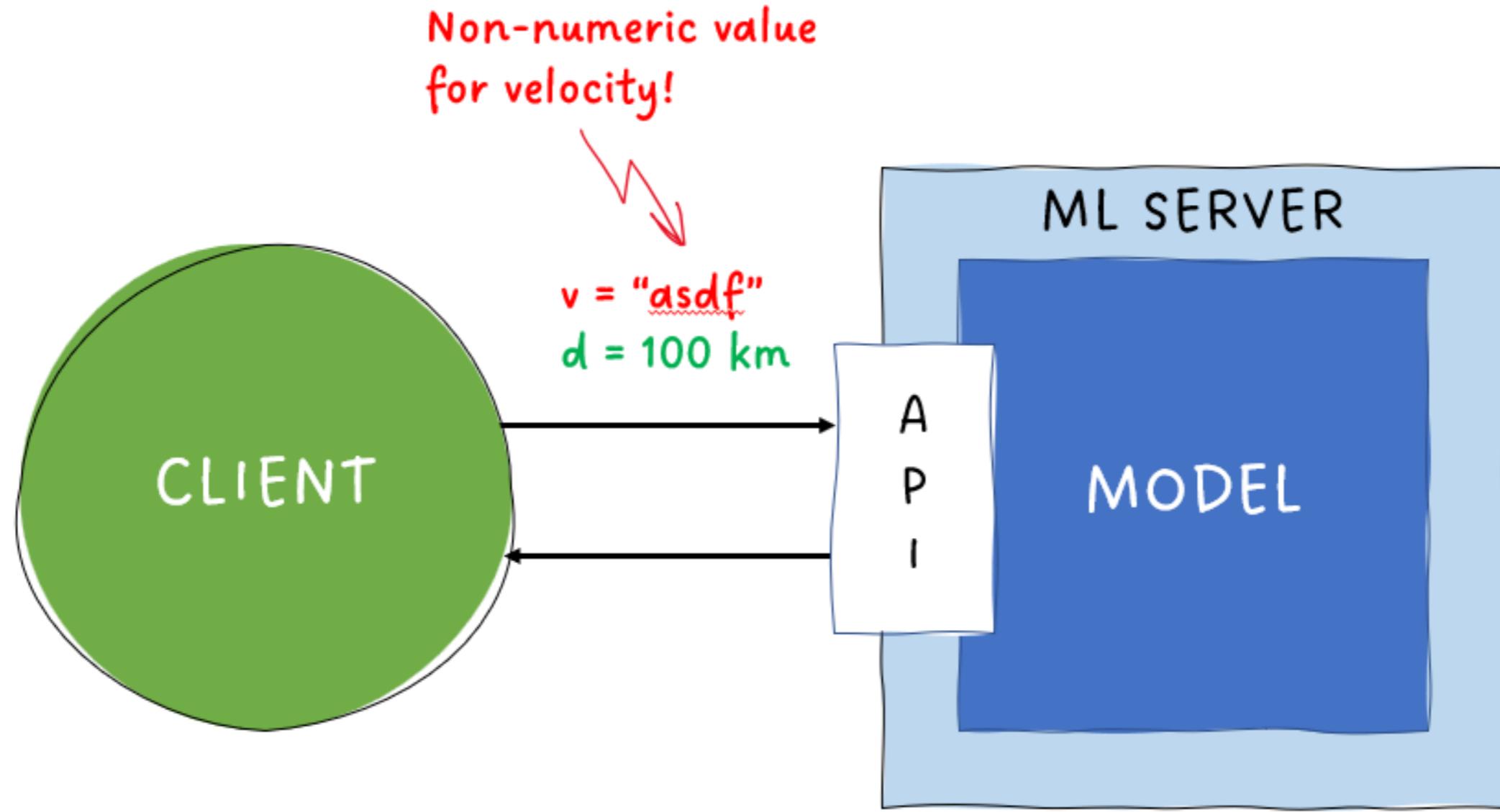
¹ Image: Pixabay, <https://www.stockvault.net/user/profile/161904>

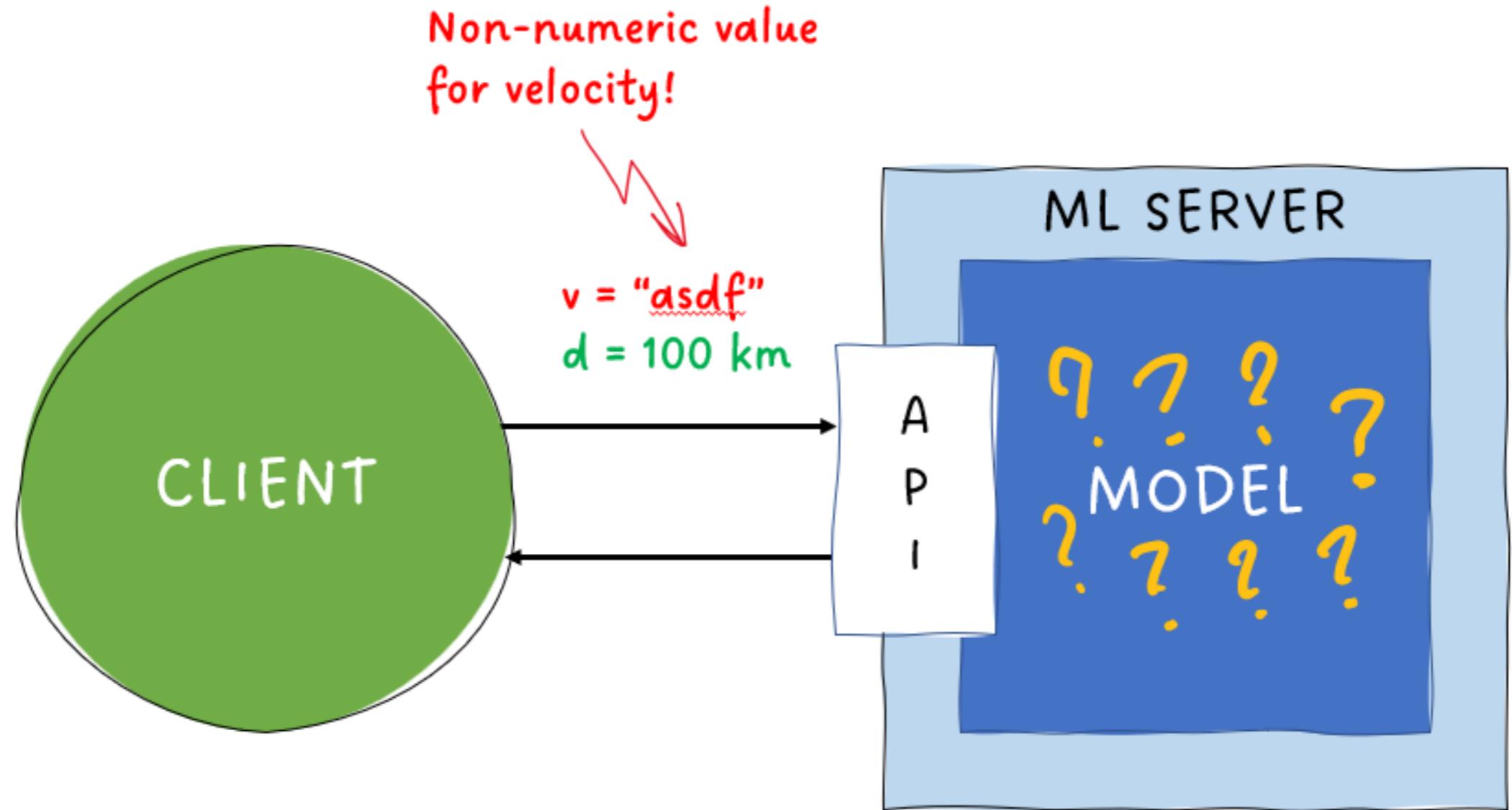


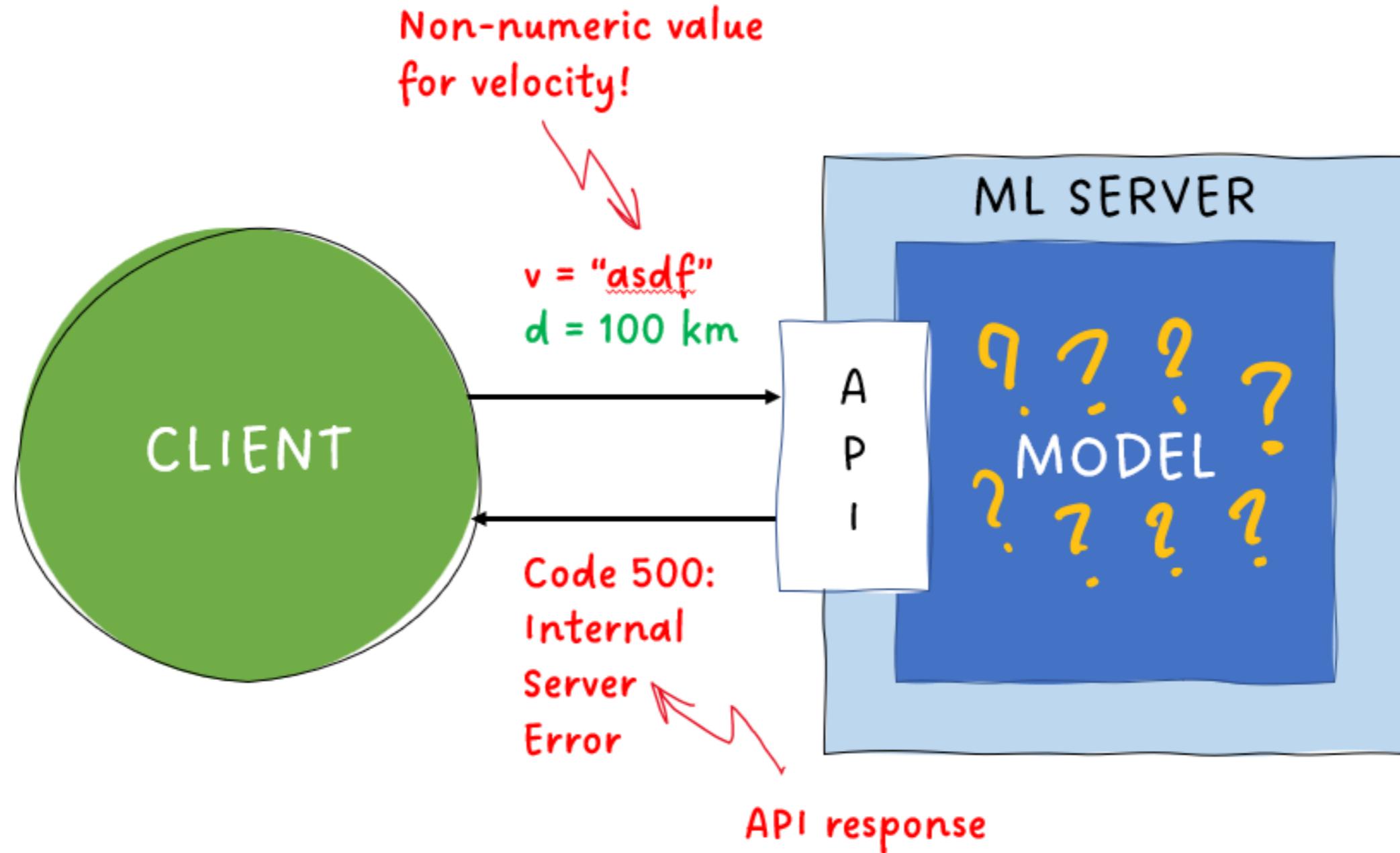


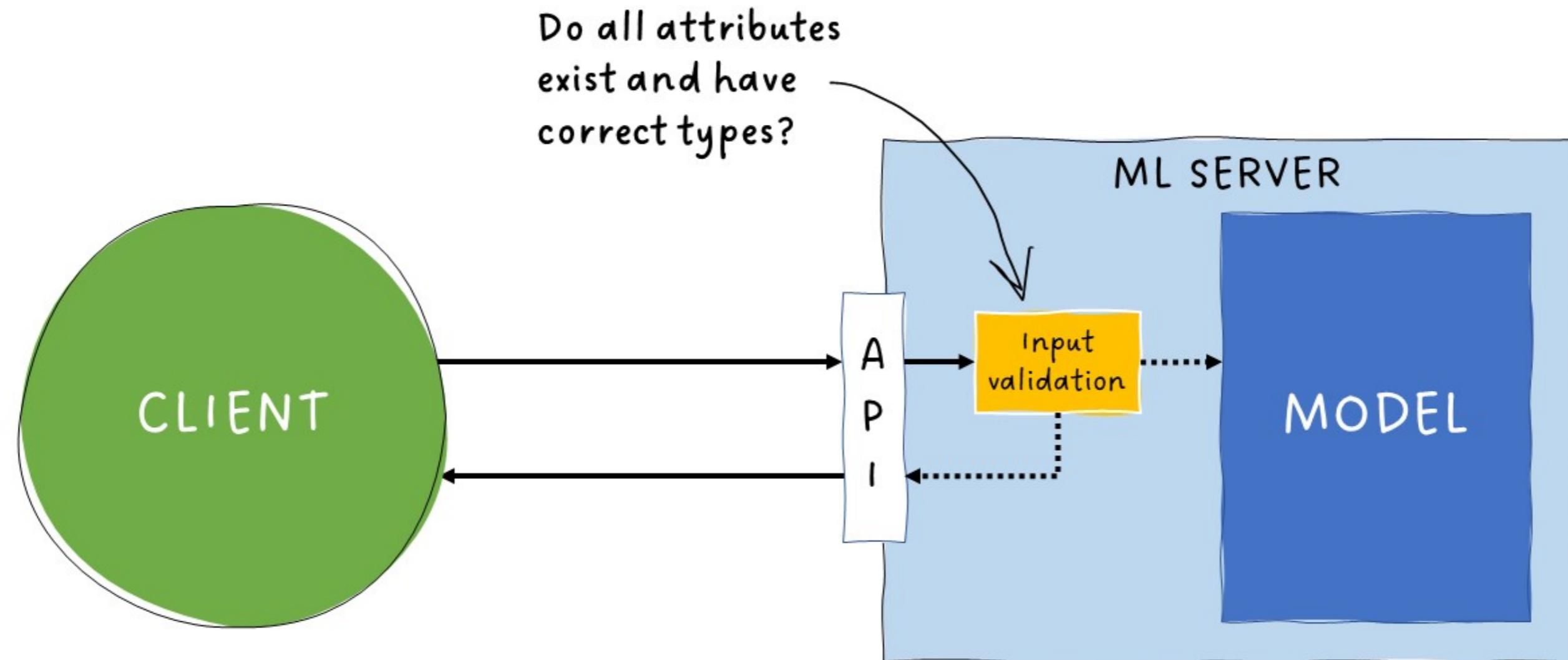








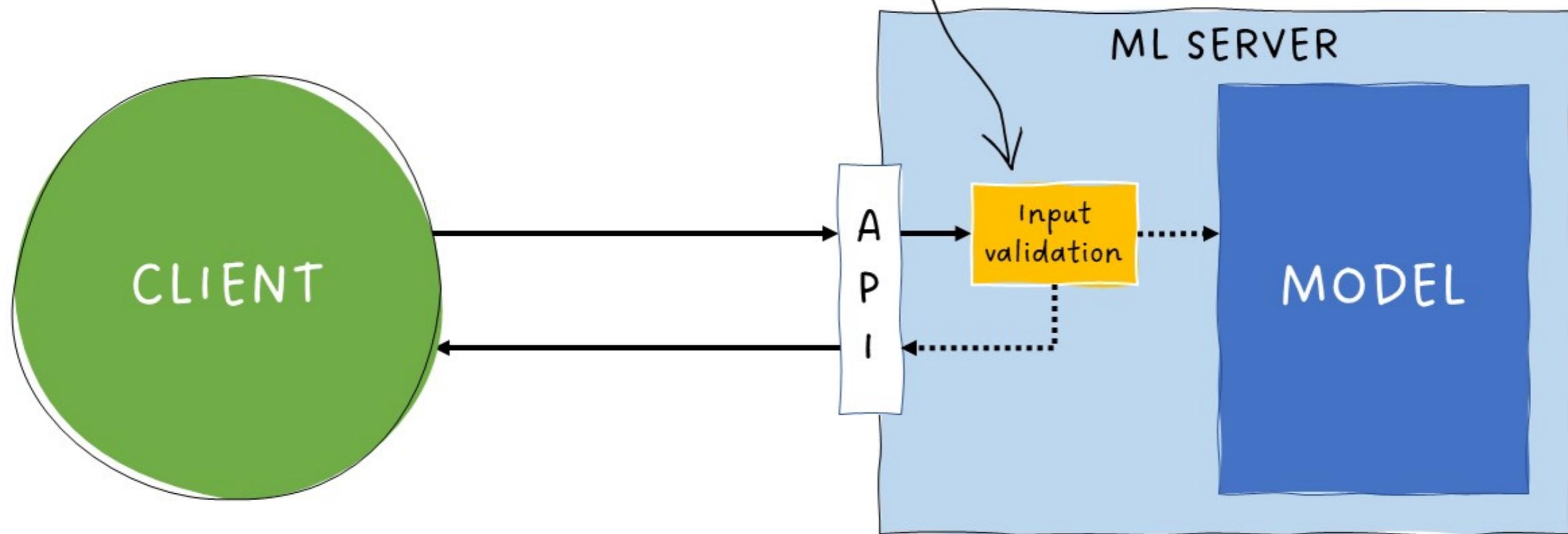


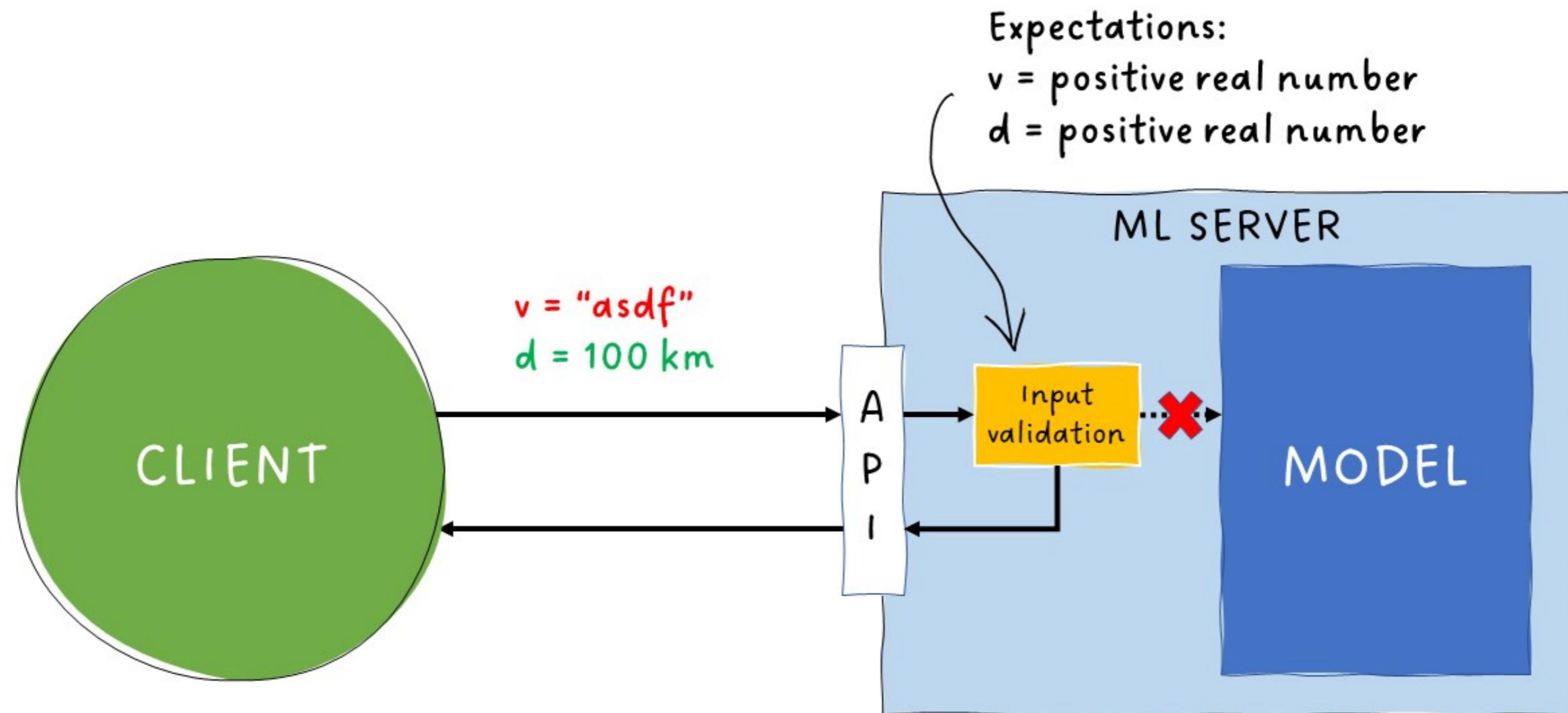


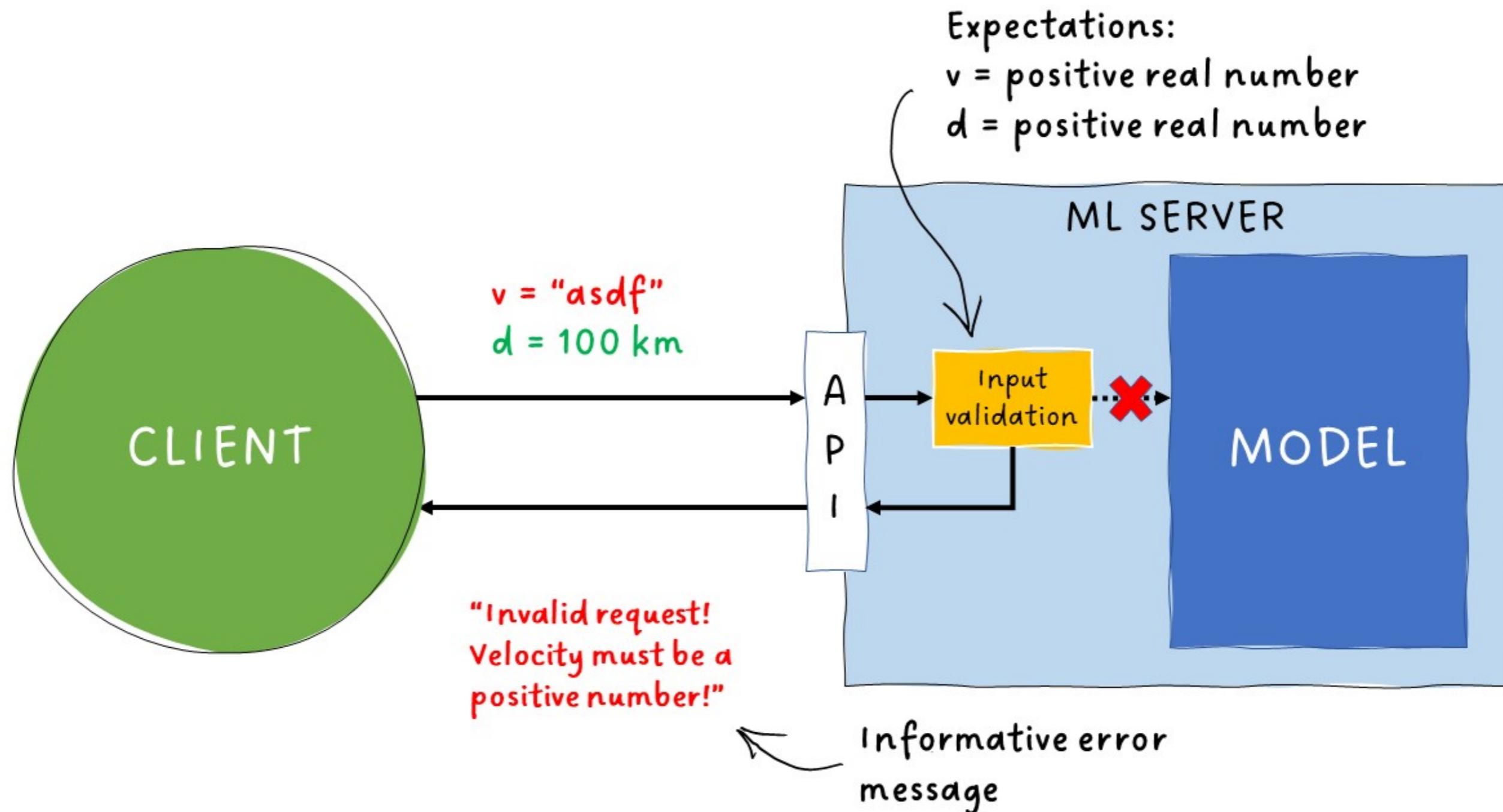
Expectations:

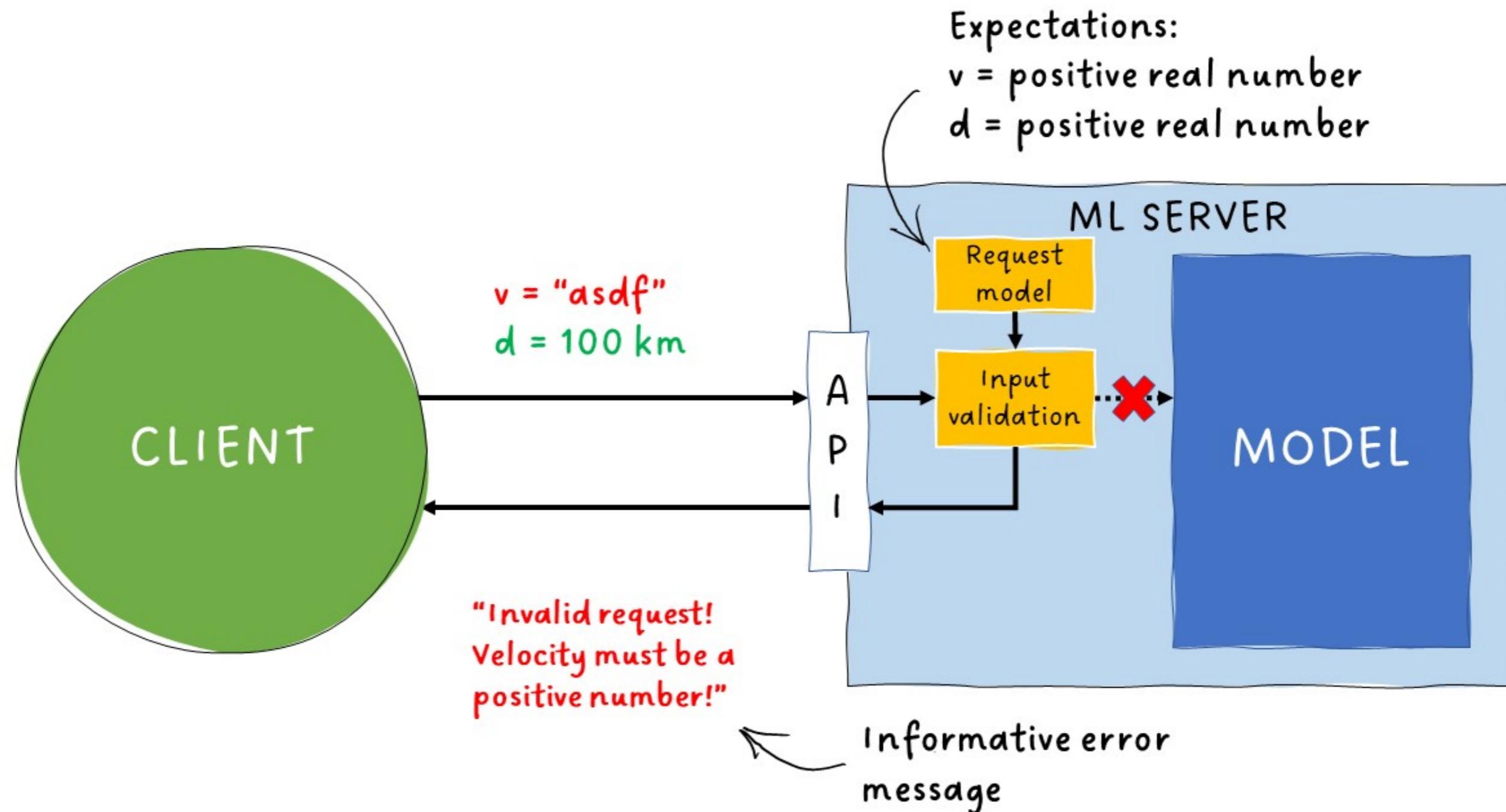
v = positive real number

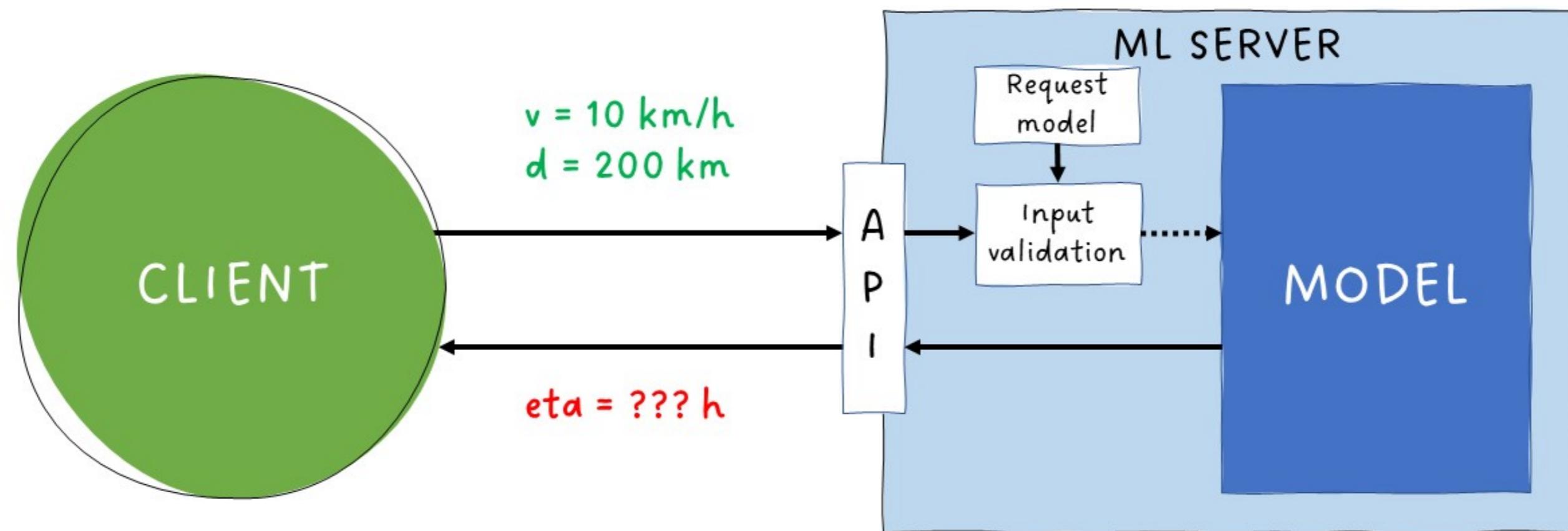
d = positive real number

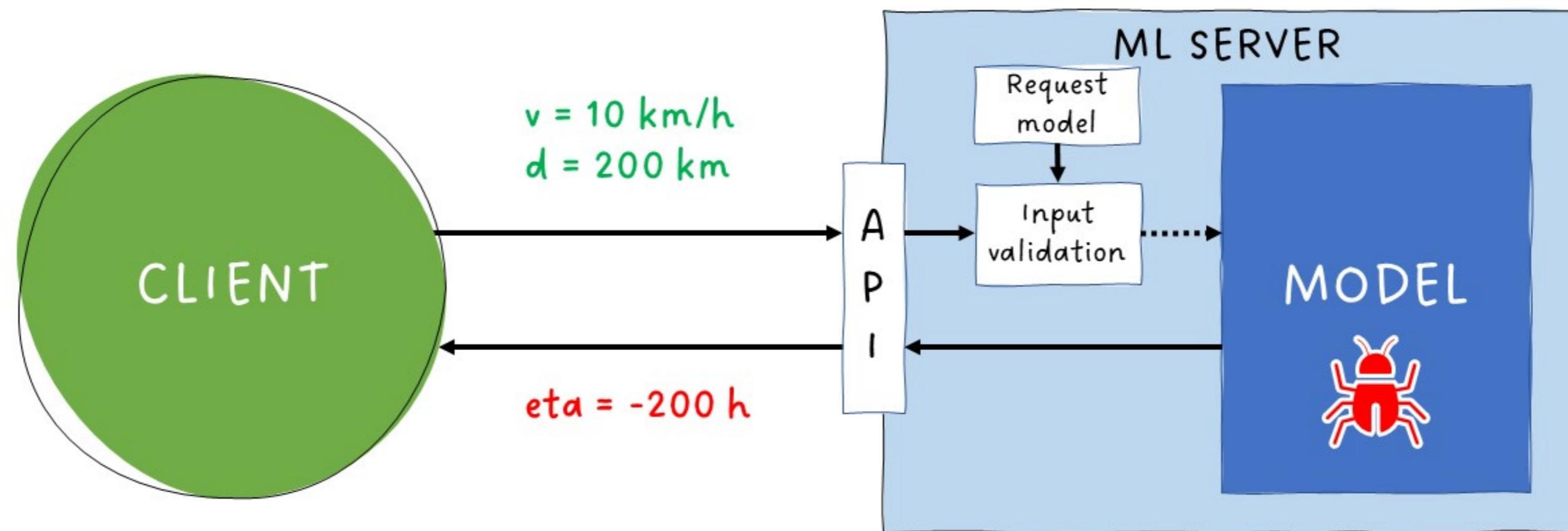


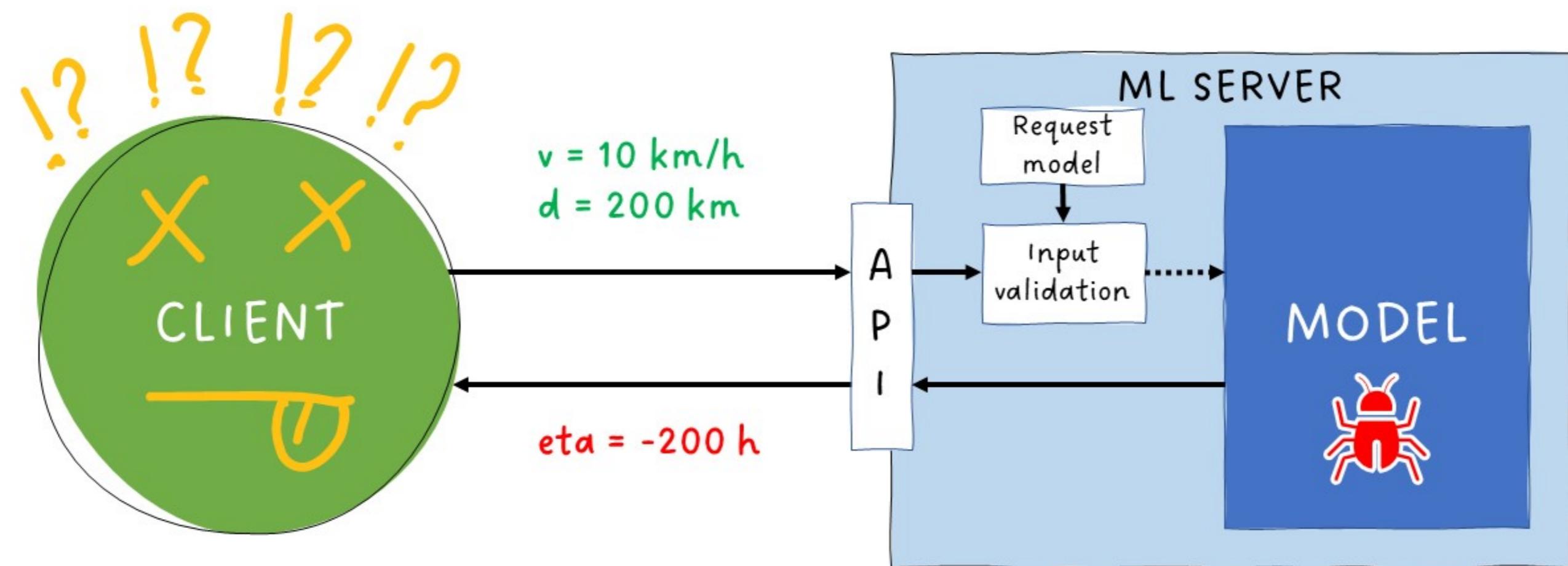


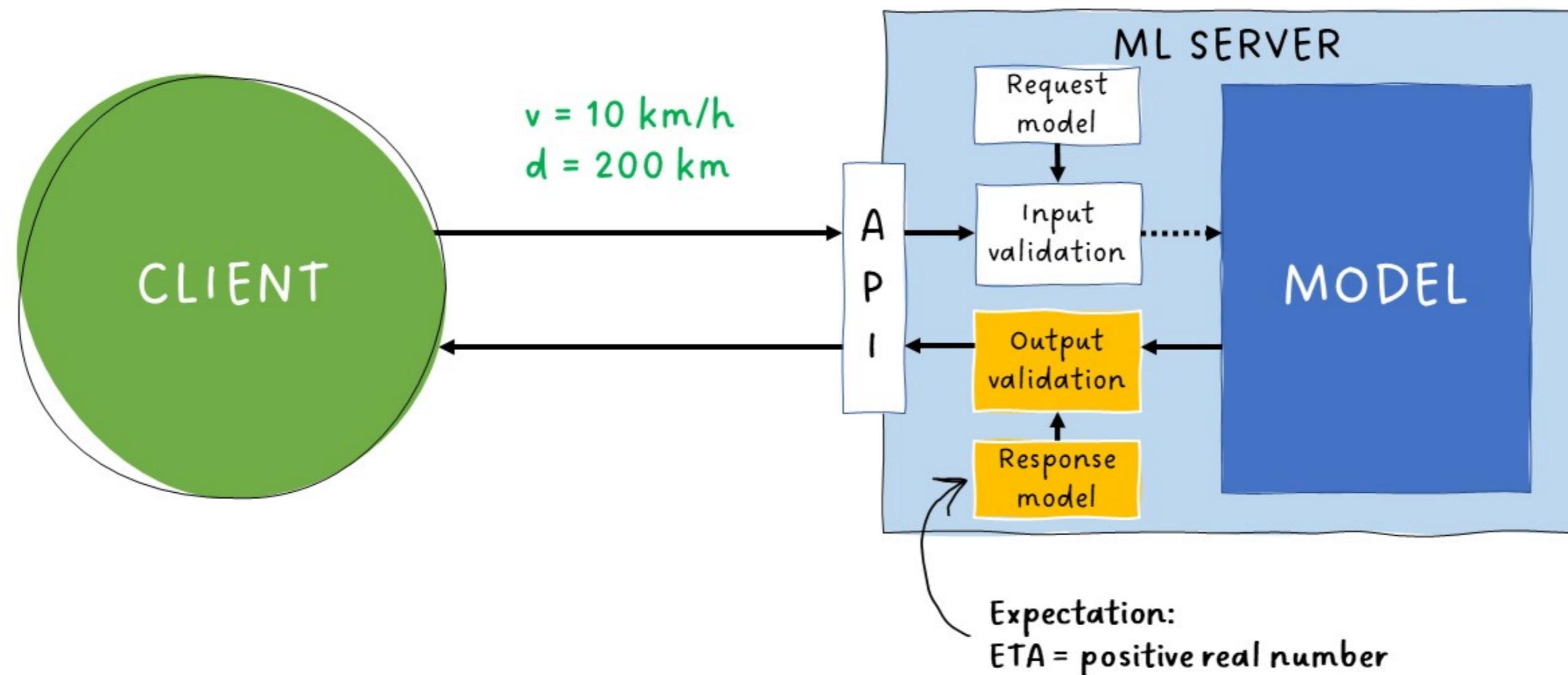


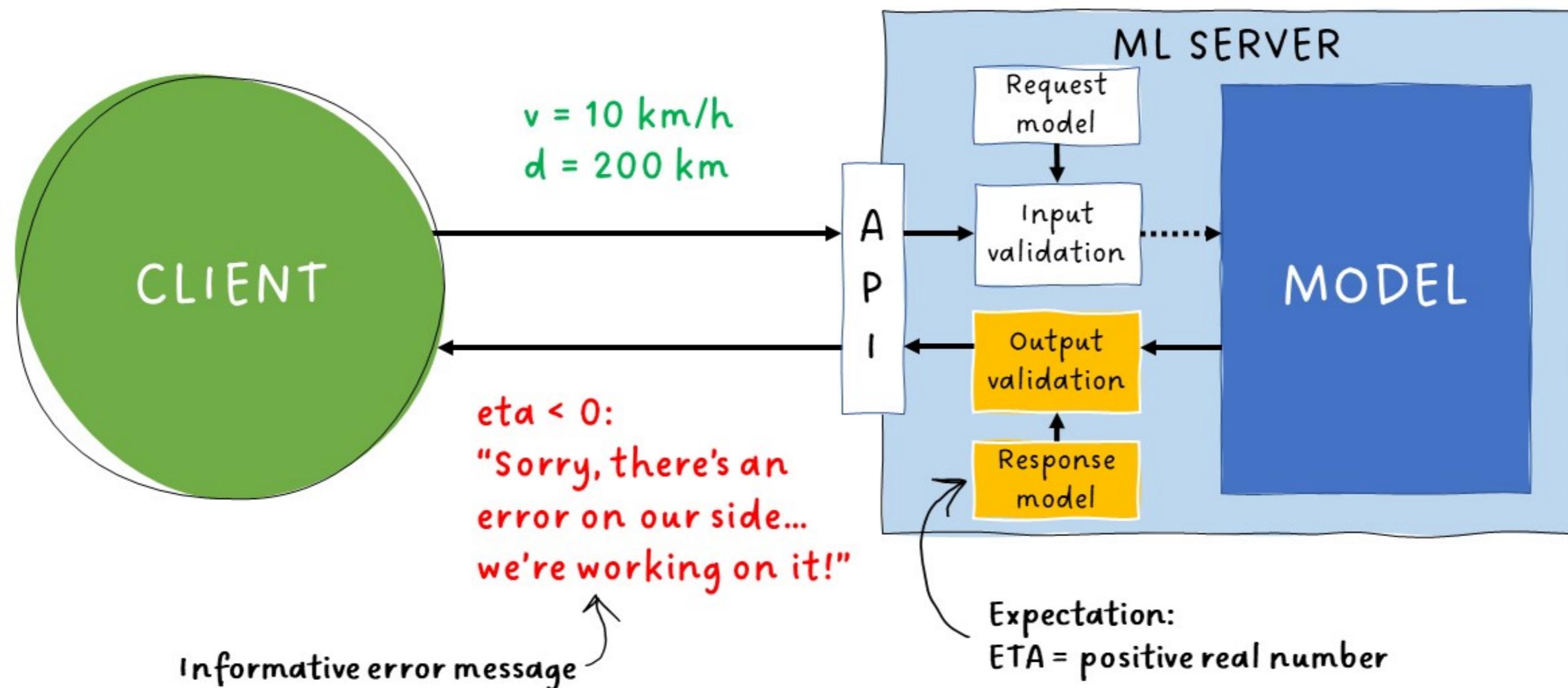


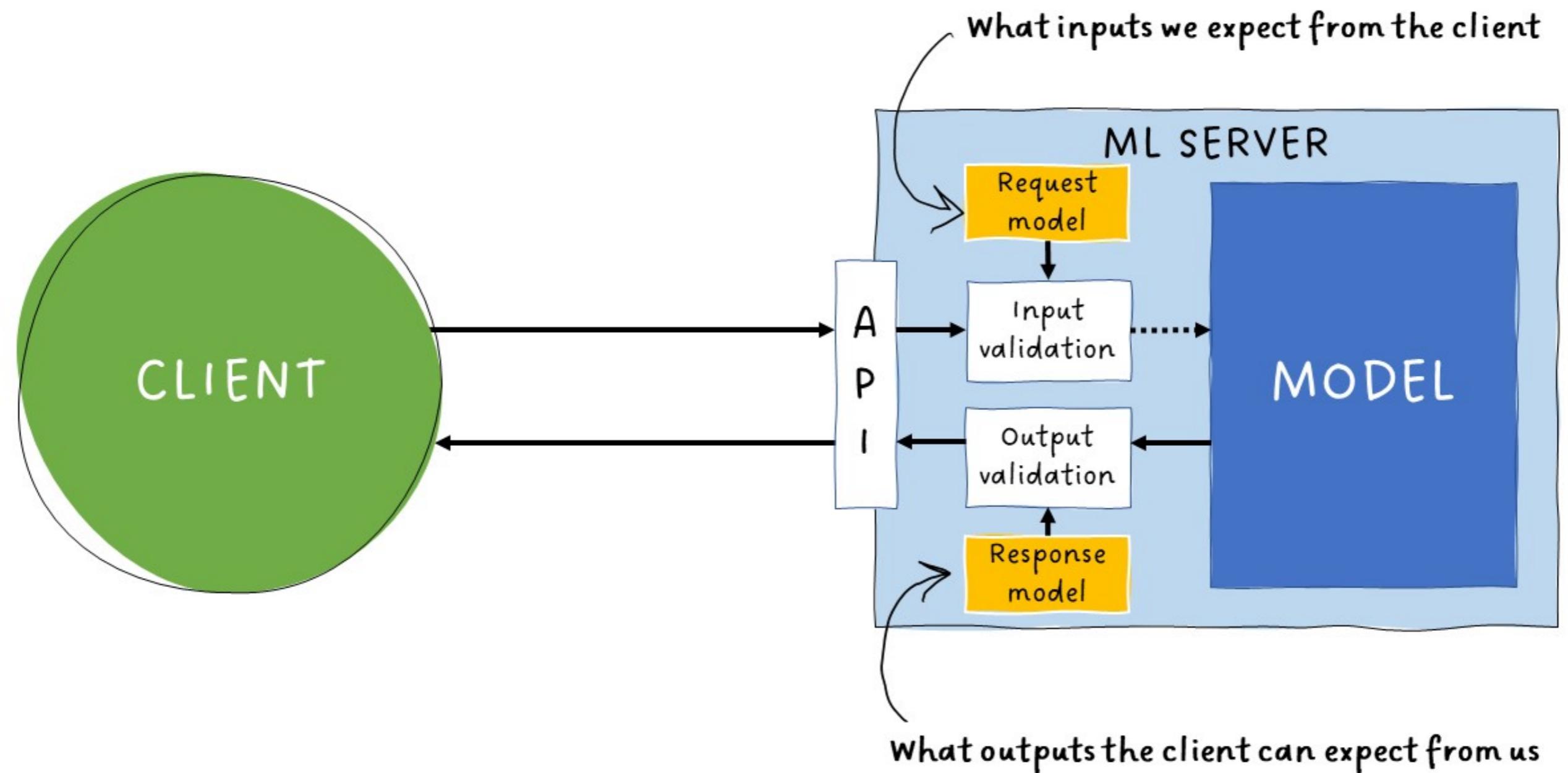


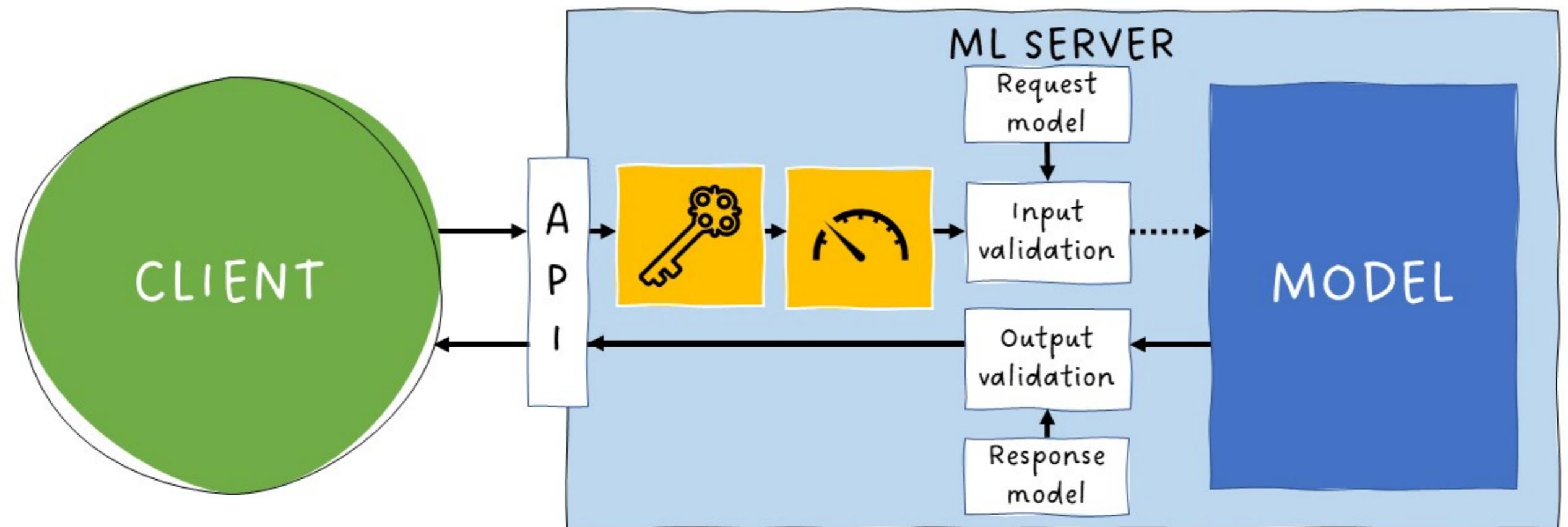


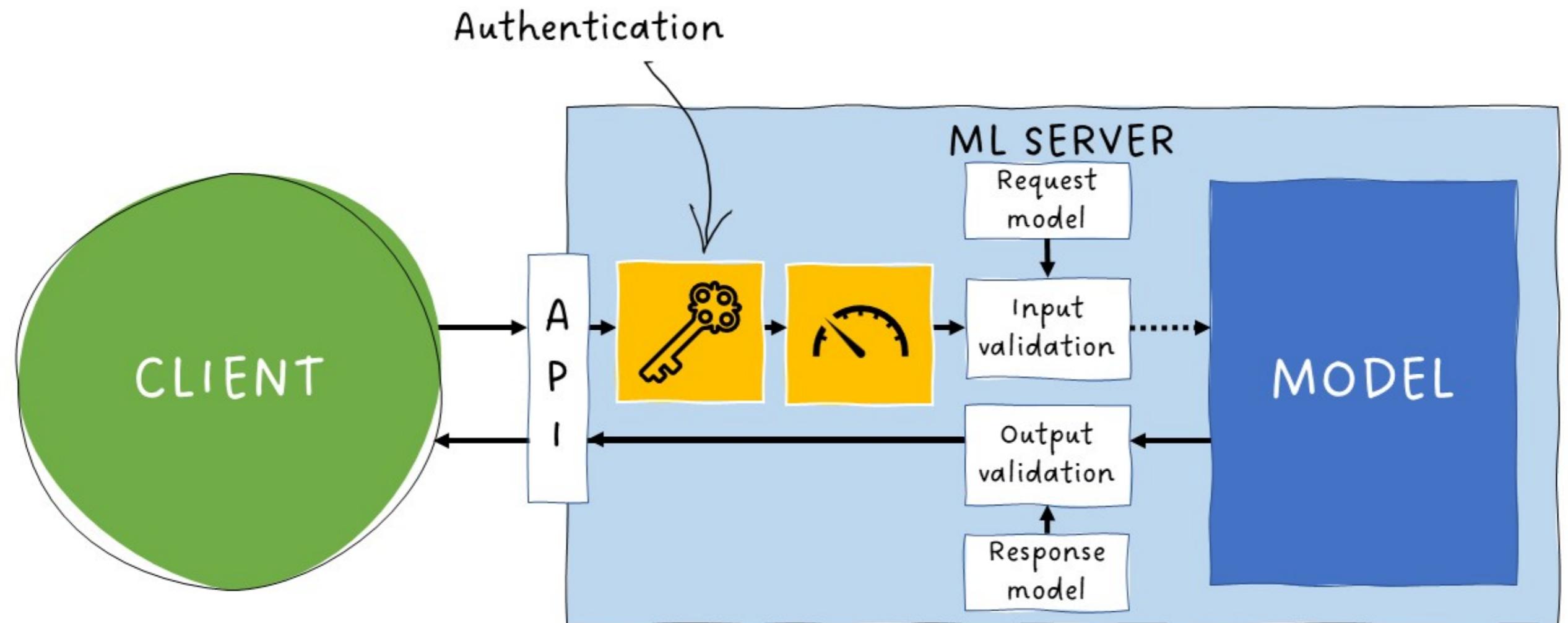


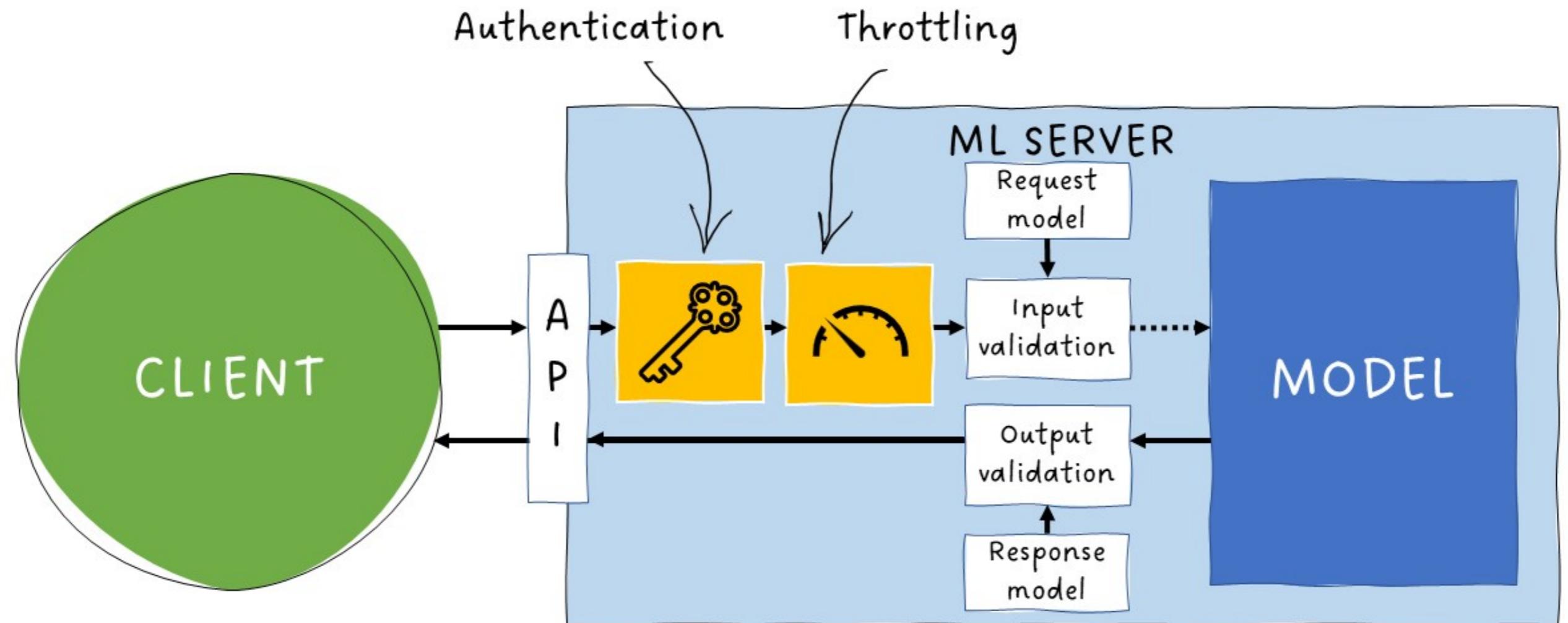














FastAPI

- open-source API framework for Python
- all essential features out of the box
- build and launch REST APIs in no time

¹ <https://fastapi.tiangolo.com>

Let's practice!

MLOPS DEPLOYMENT AND LIFE CYCLING

Deployment progression and testing

MLOPS DEPLOYMENT AND LIFE CYCLING



Nemanja Radojkovic

Senior Machine Learning Engineer

Not so fast!



- Don't rush to deploy without thorough testing!

Agenda

TEST TYPES

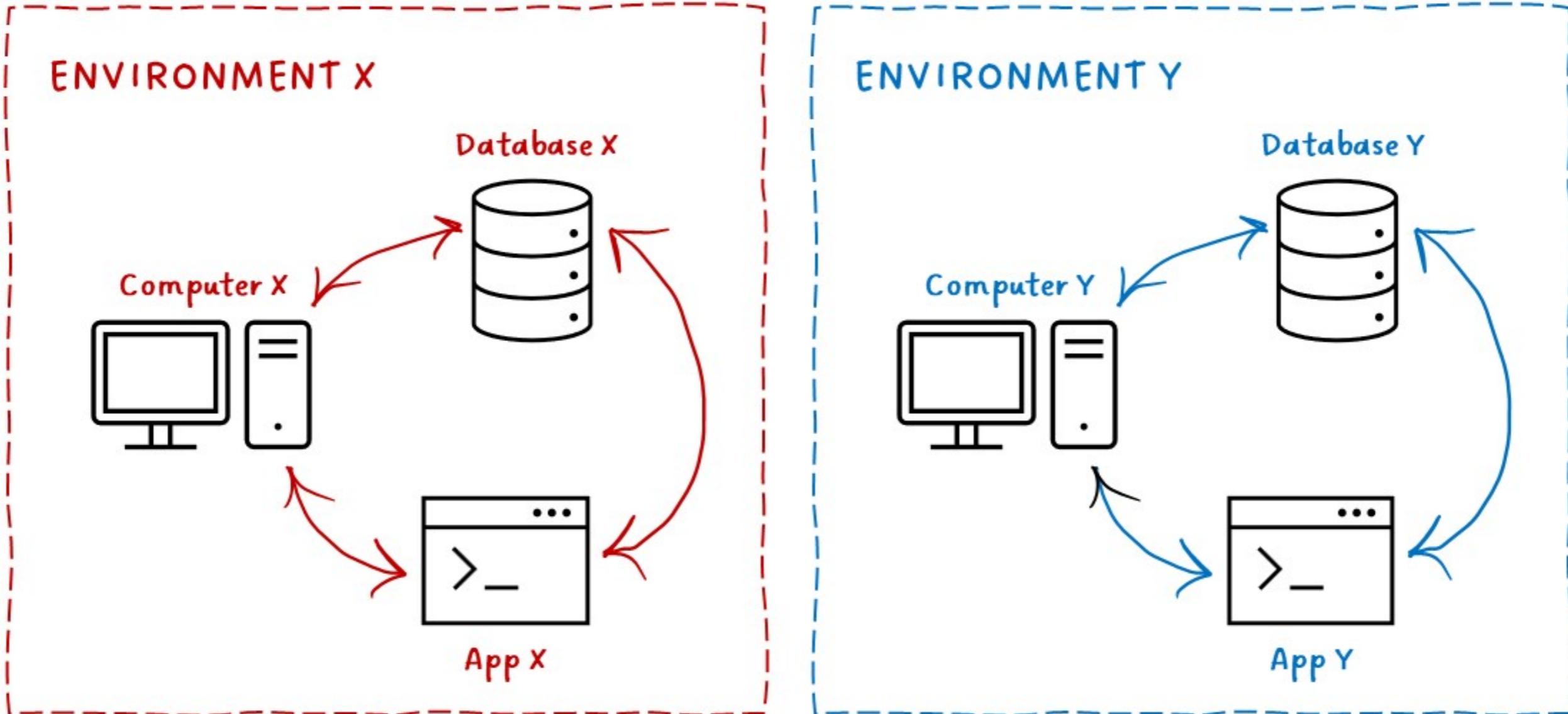
- Unit tests
- Integration tests
- Smoke tests
- Load tests
- Acceptance tests

DEV, DEPLOY, TEST ENVIRONMENTS

- Development
- Test
- Staging
- Production

OUT OF SCOPE: Predictive model performance

IN SCOPE: Testing the overall ML app



Unit testing

Checking if independent code units behave as expected

EXAMPLE

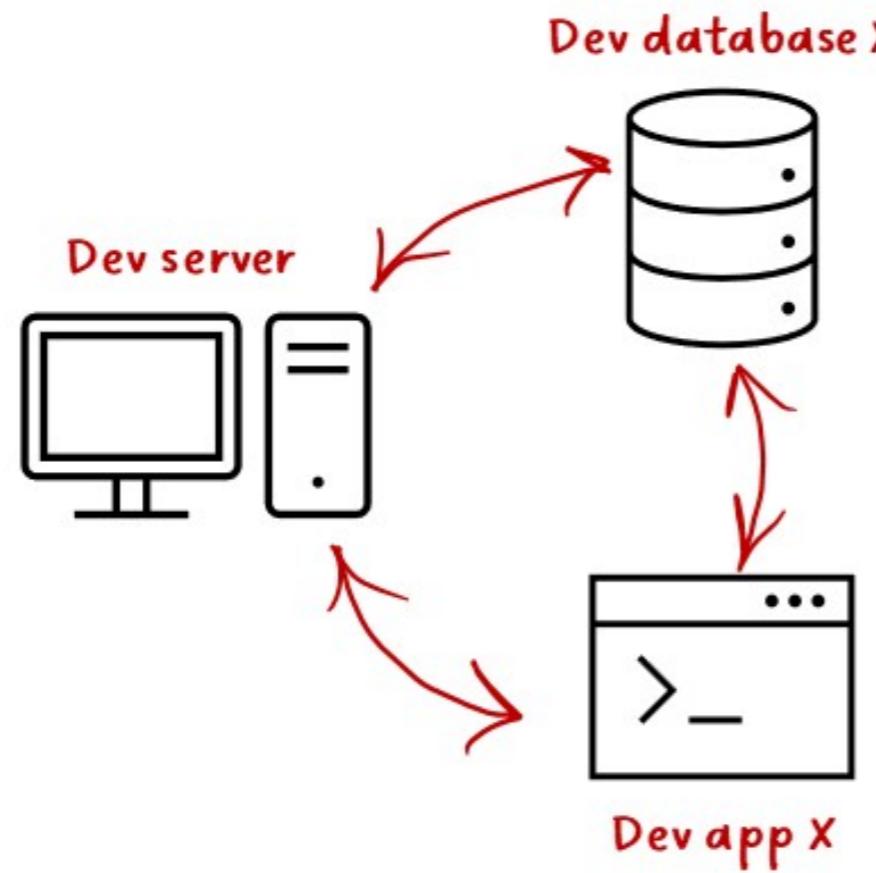
Function to test:

```
def add_two_numbers(x, y):  
    return x + y
```

Unit test:

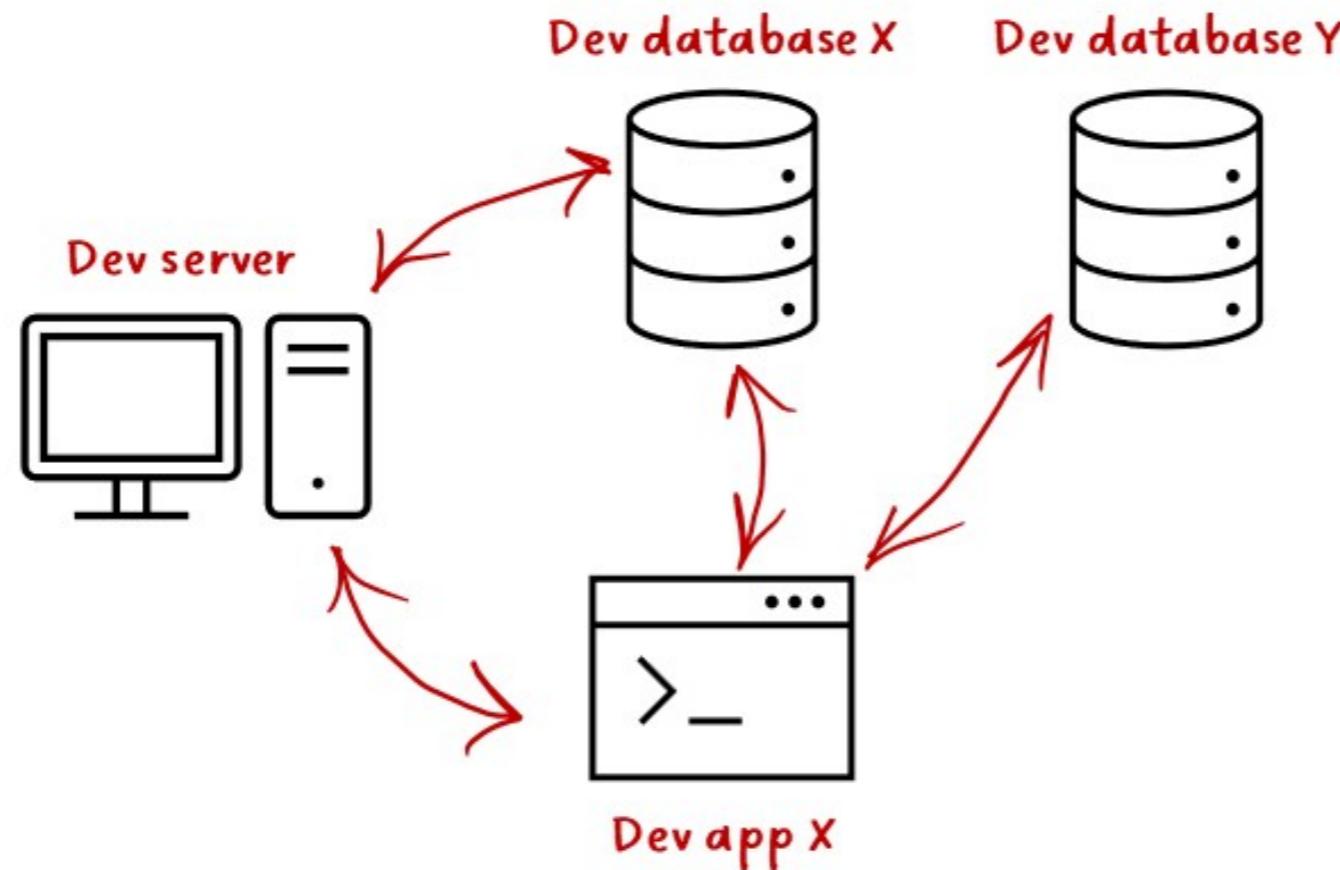
```
add_two_numbers(1, 1) == 2  
add_two_numbers(2, 3) == 5
```

DEV ENVIRONMENT



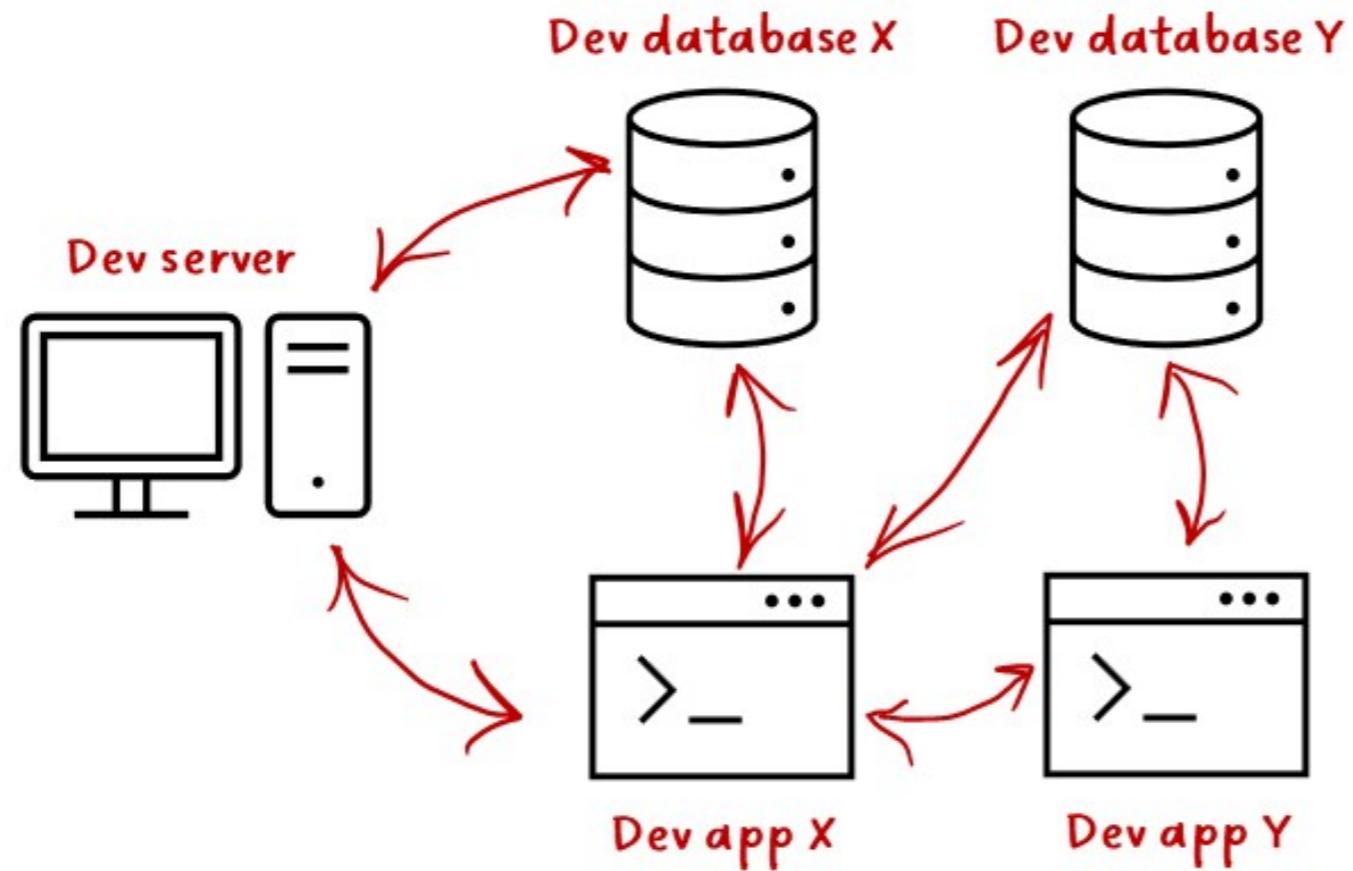
- Experimentation
- Constantly changing

DEV ENVIRONMENT



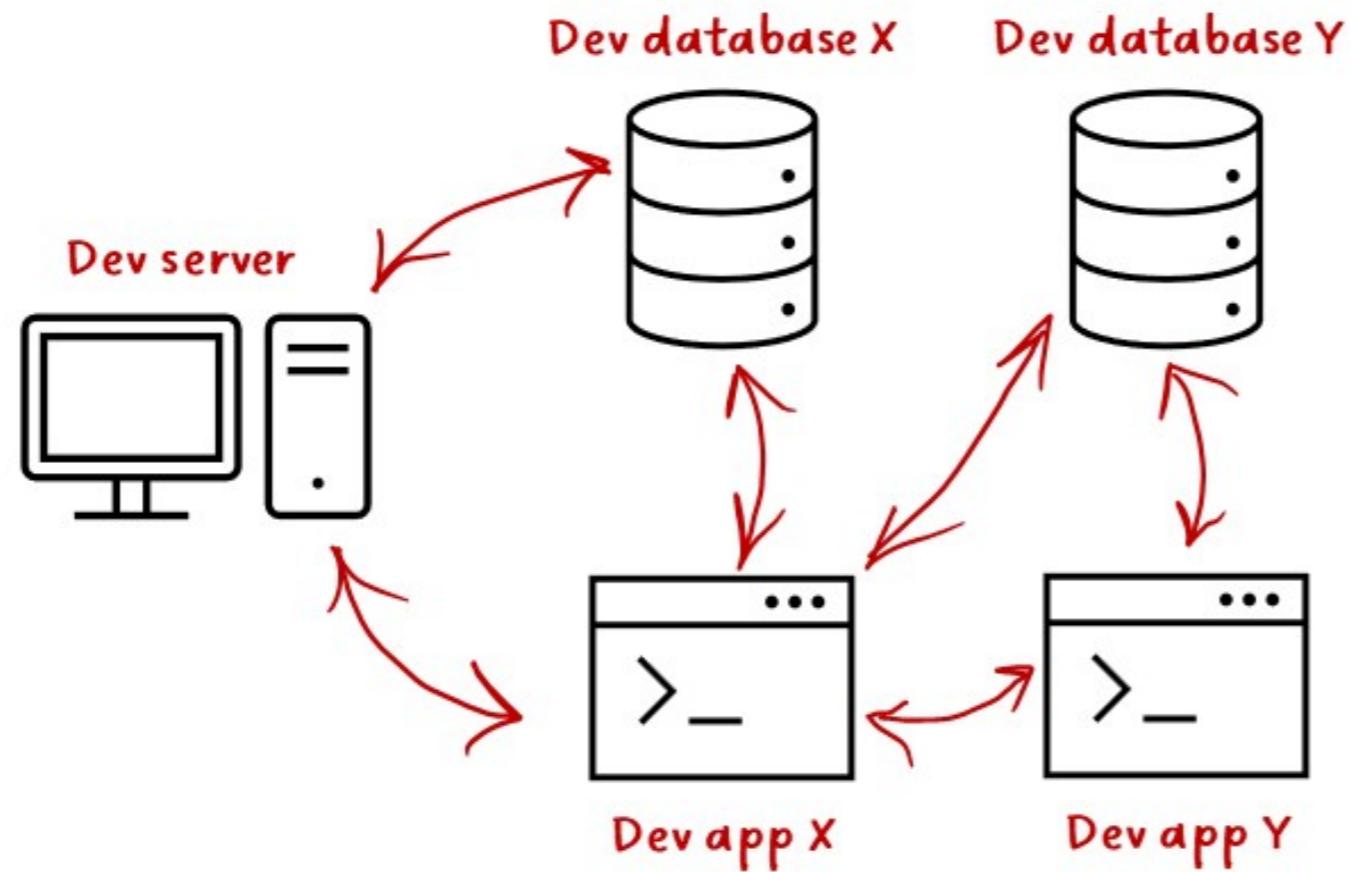
- Experimentation
- Constantly changing

DEV ENVIRONMENT

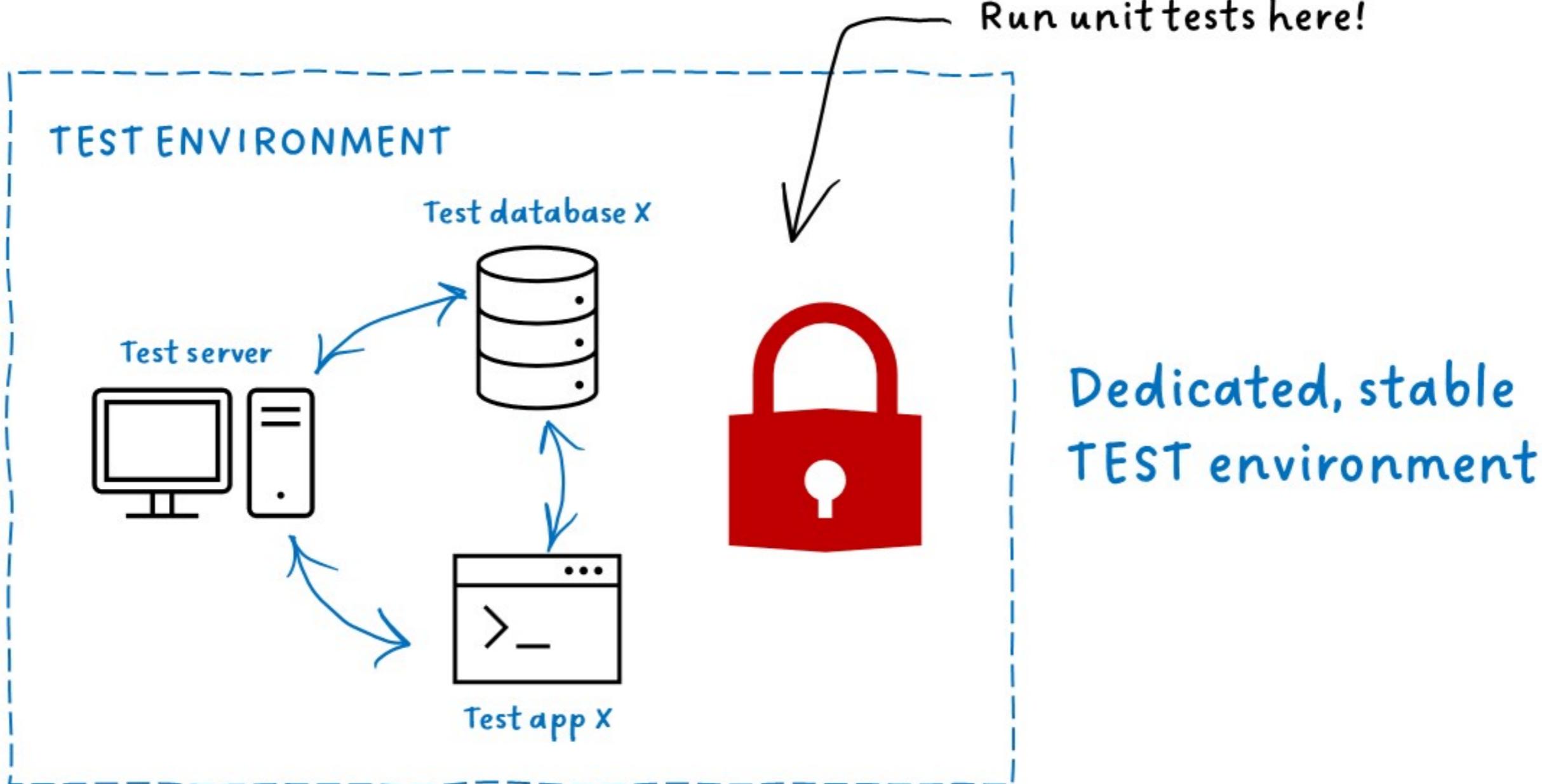


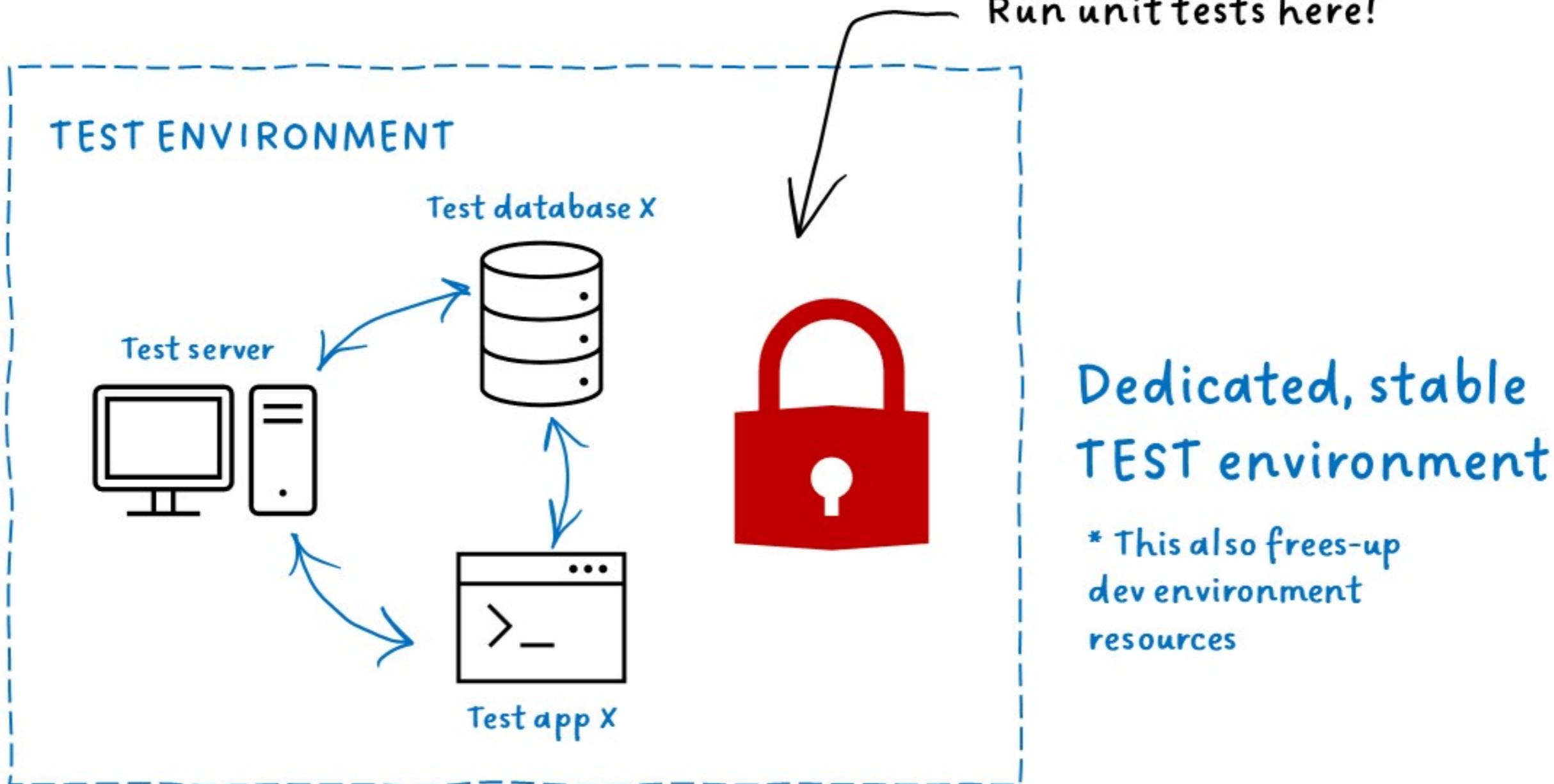
- Experimentation
- Constantly changing

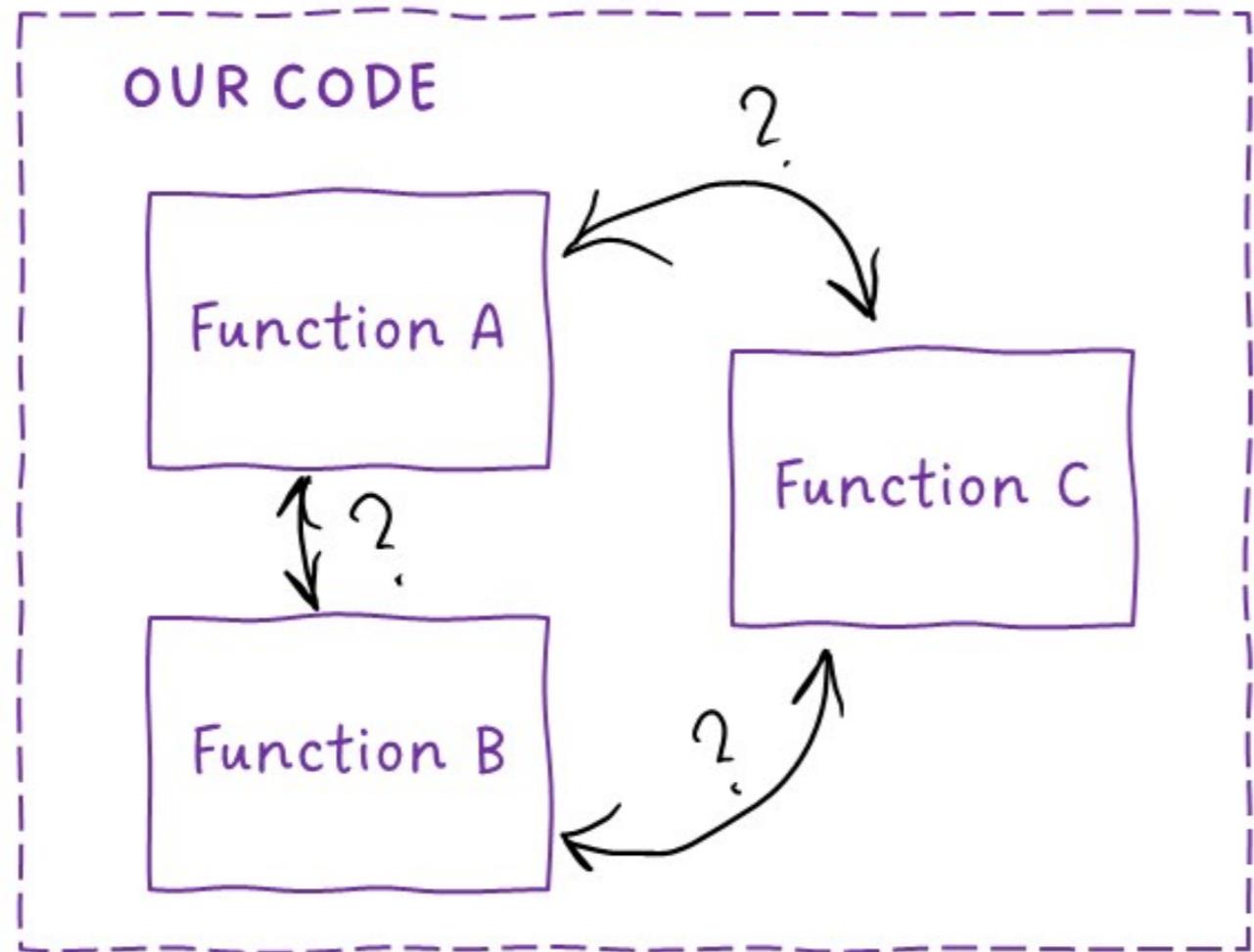
DEV ENVIRONMENT

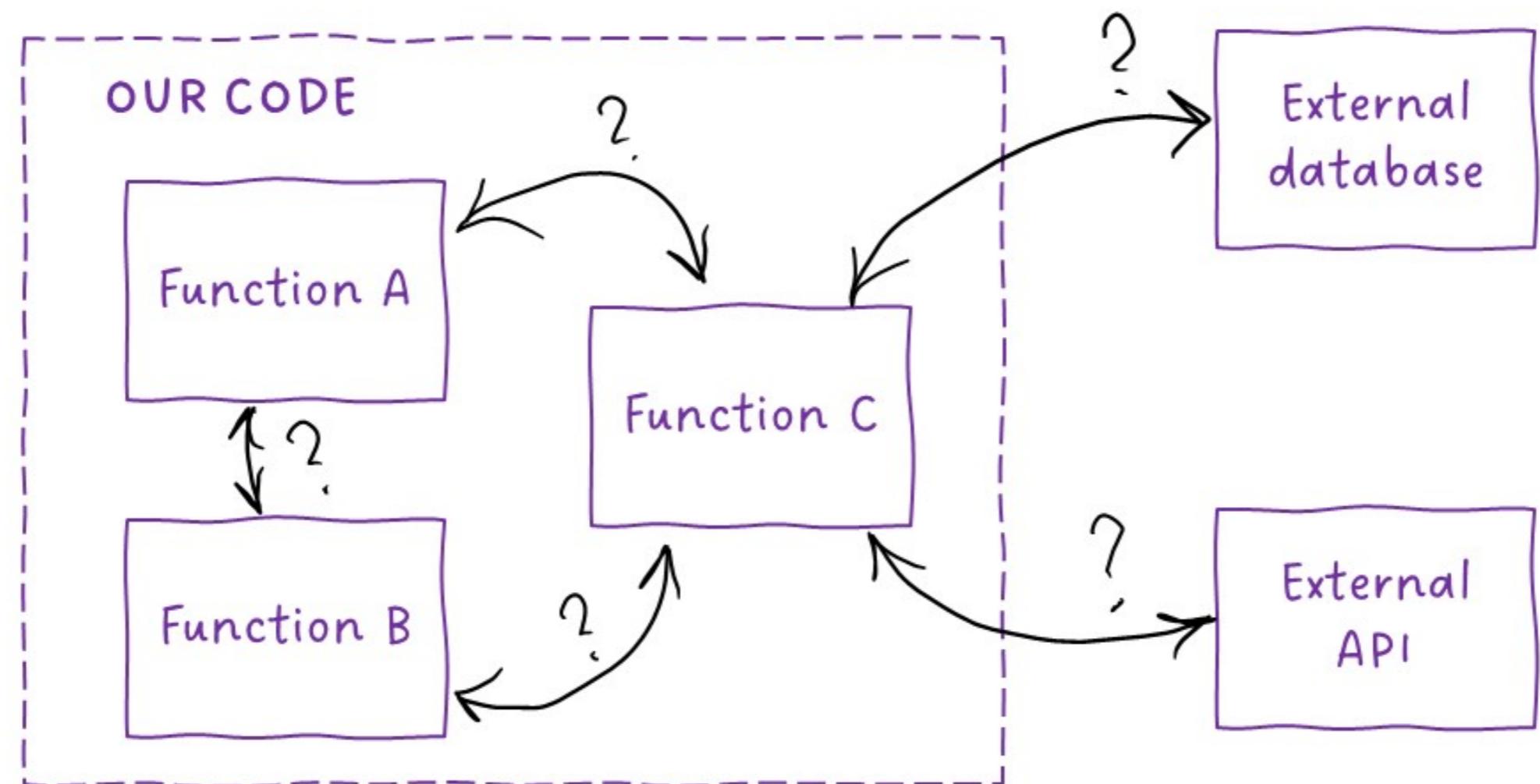


- Experimentation
- Constantly changing
- Tests require stability

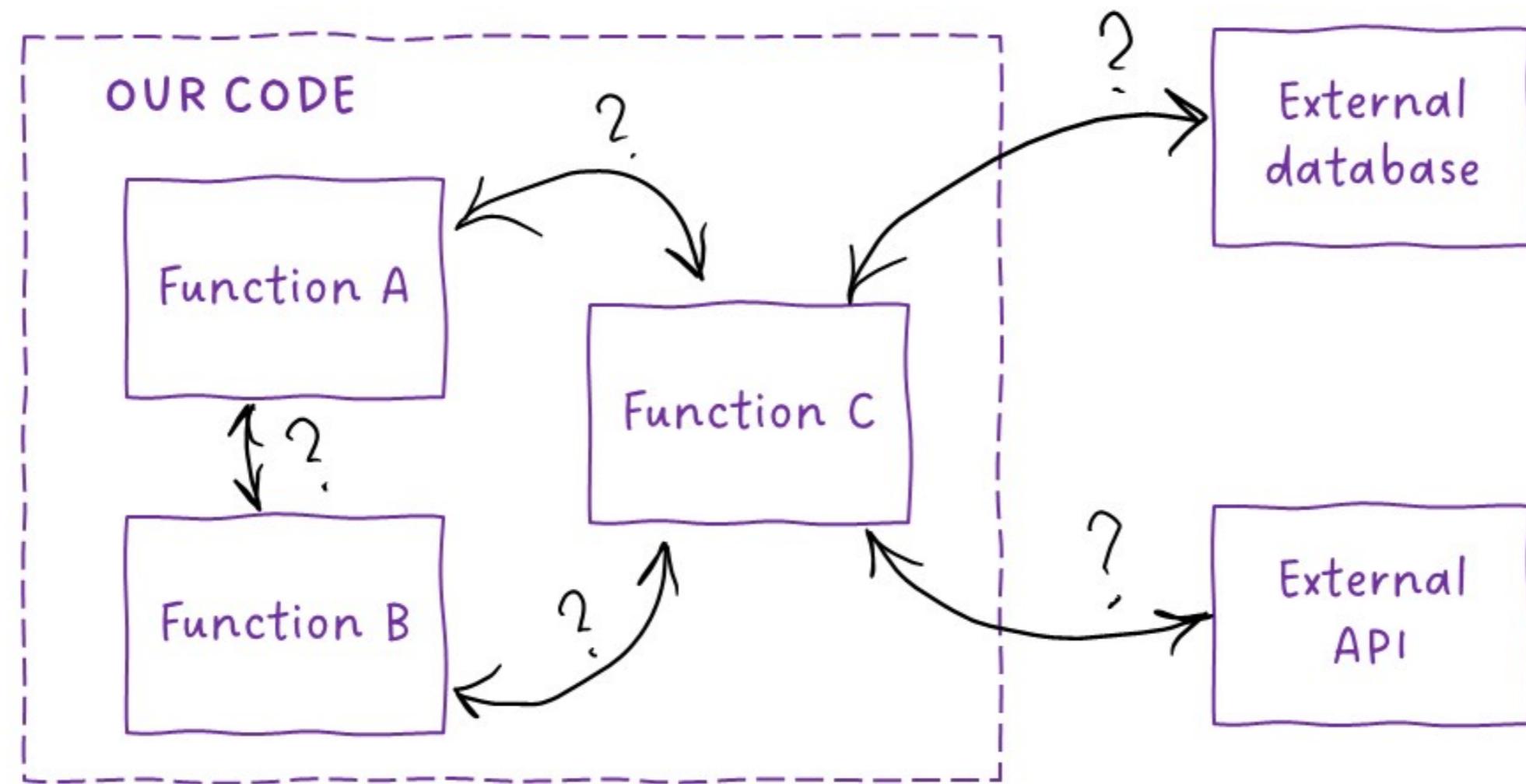




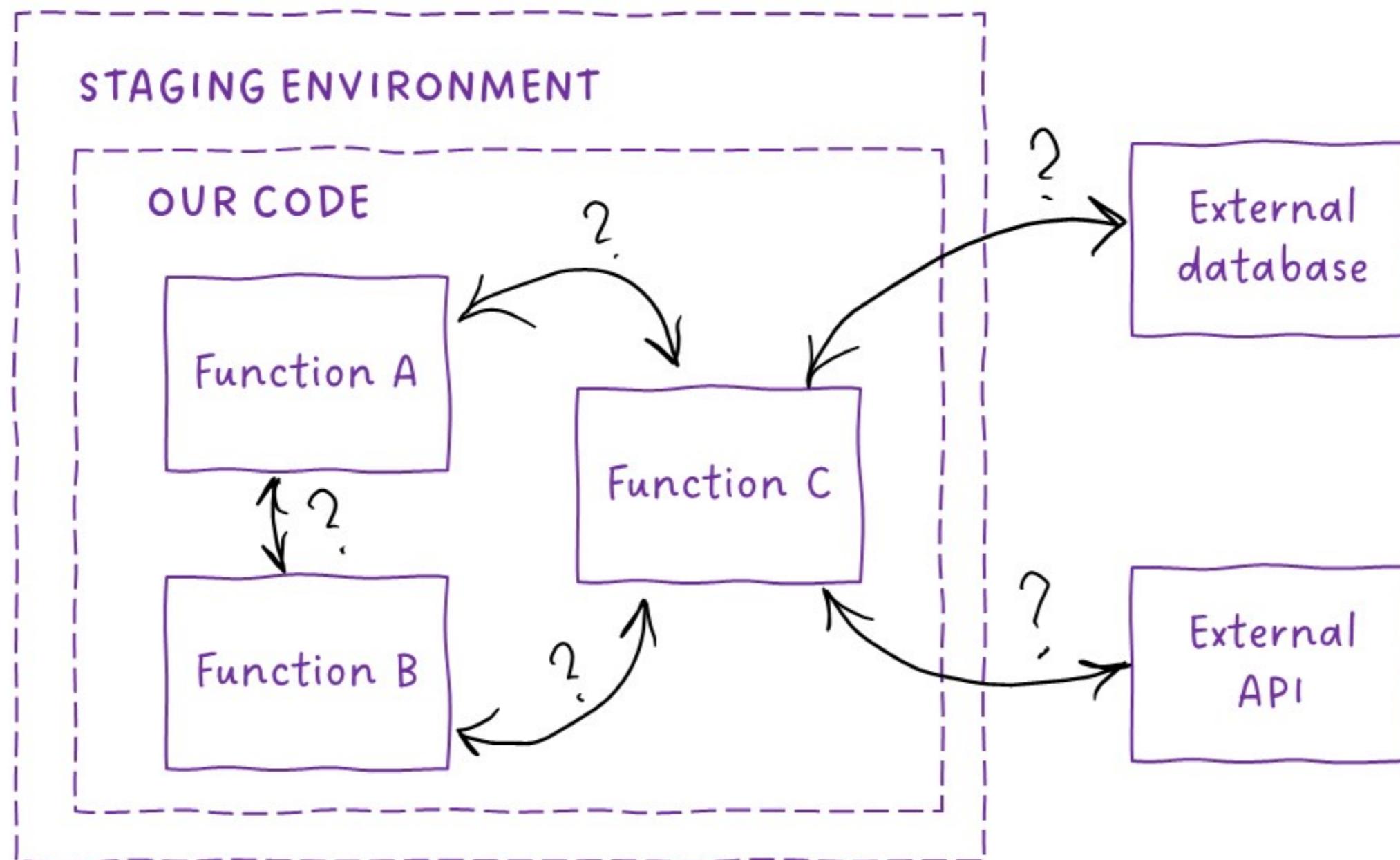


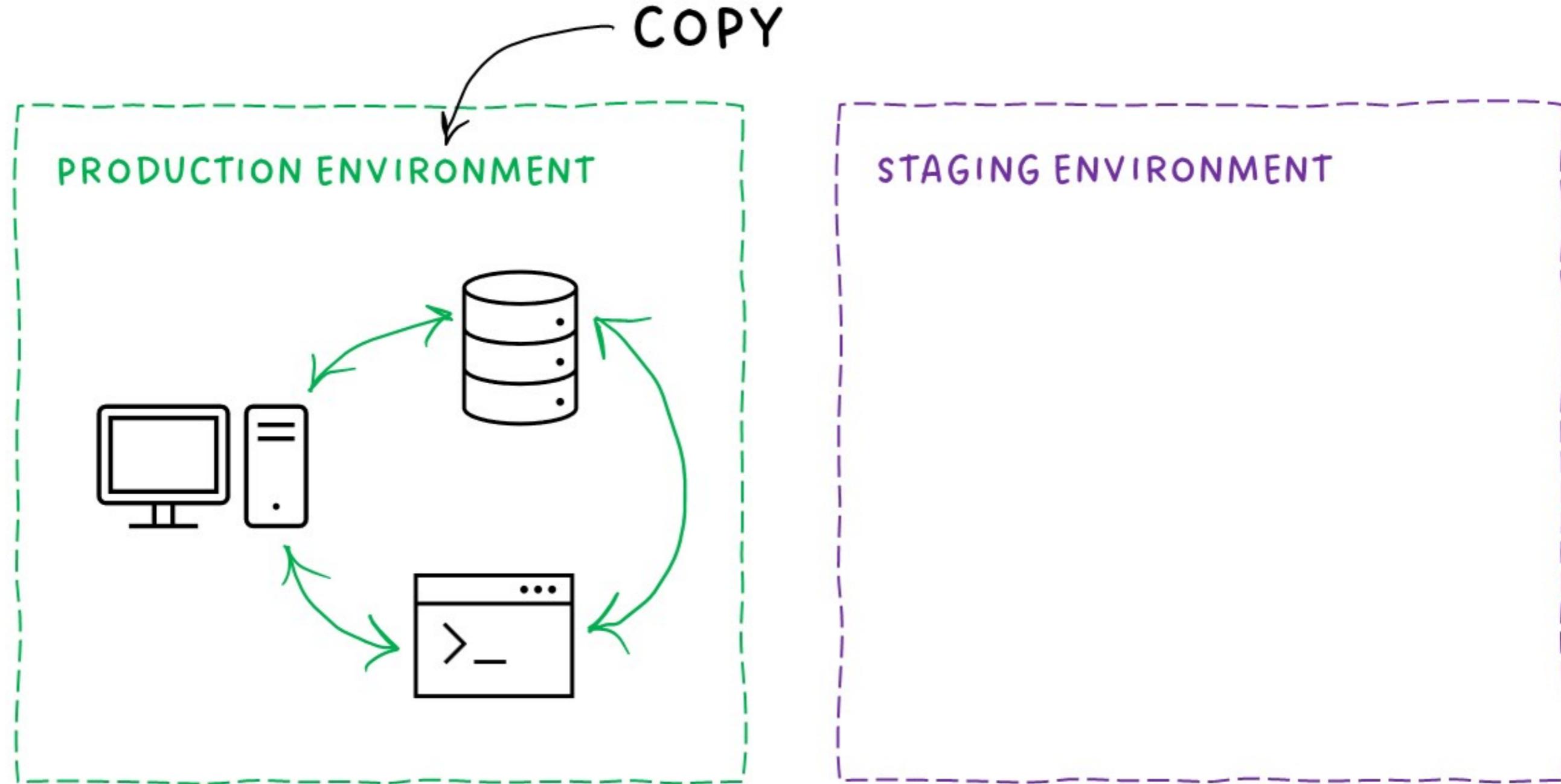


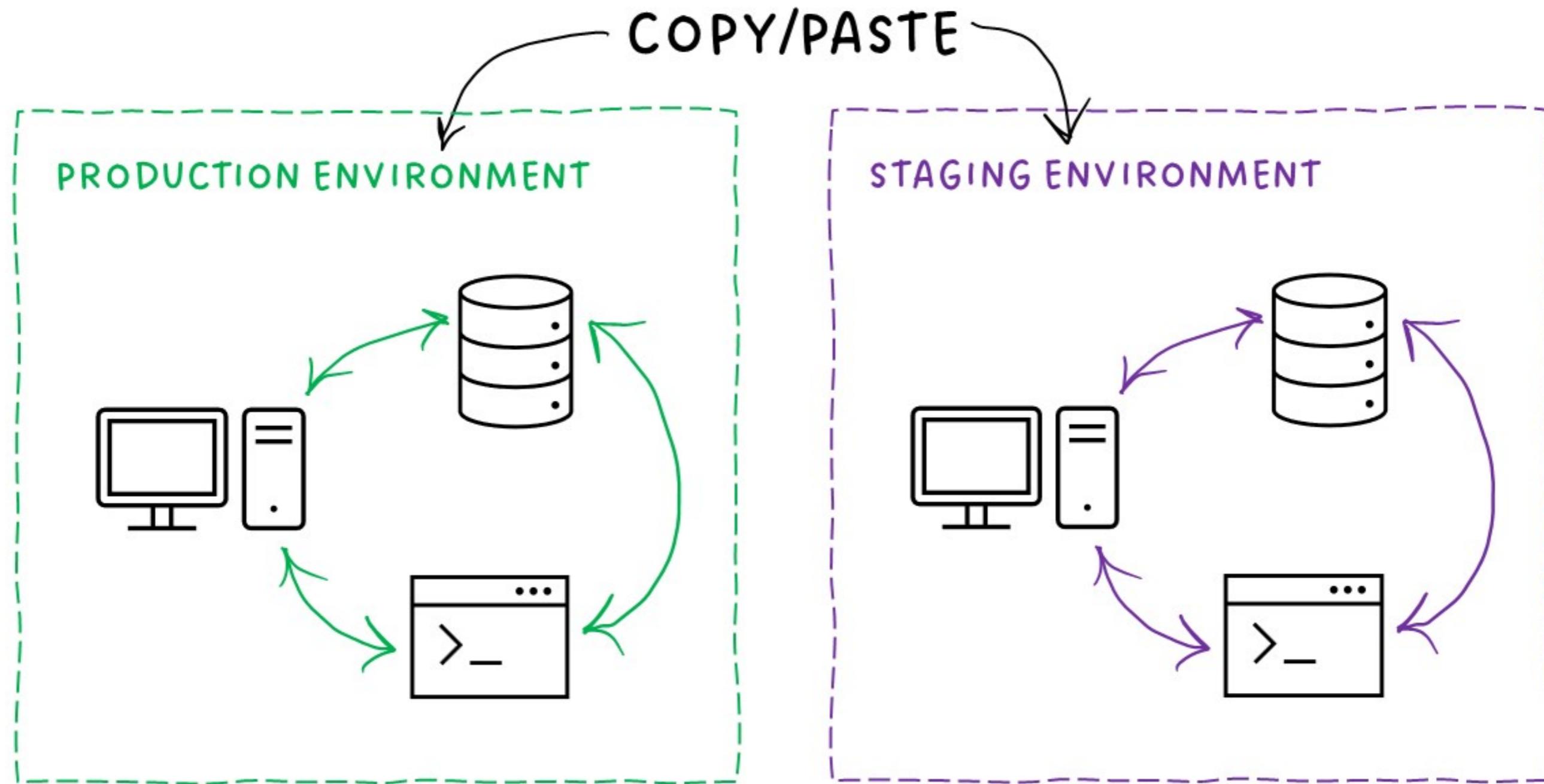
INTEGRATION TESTING

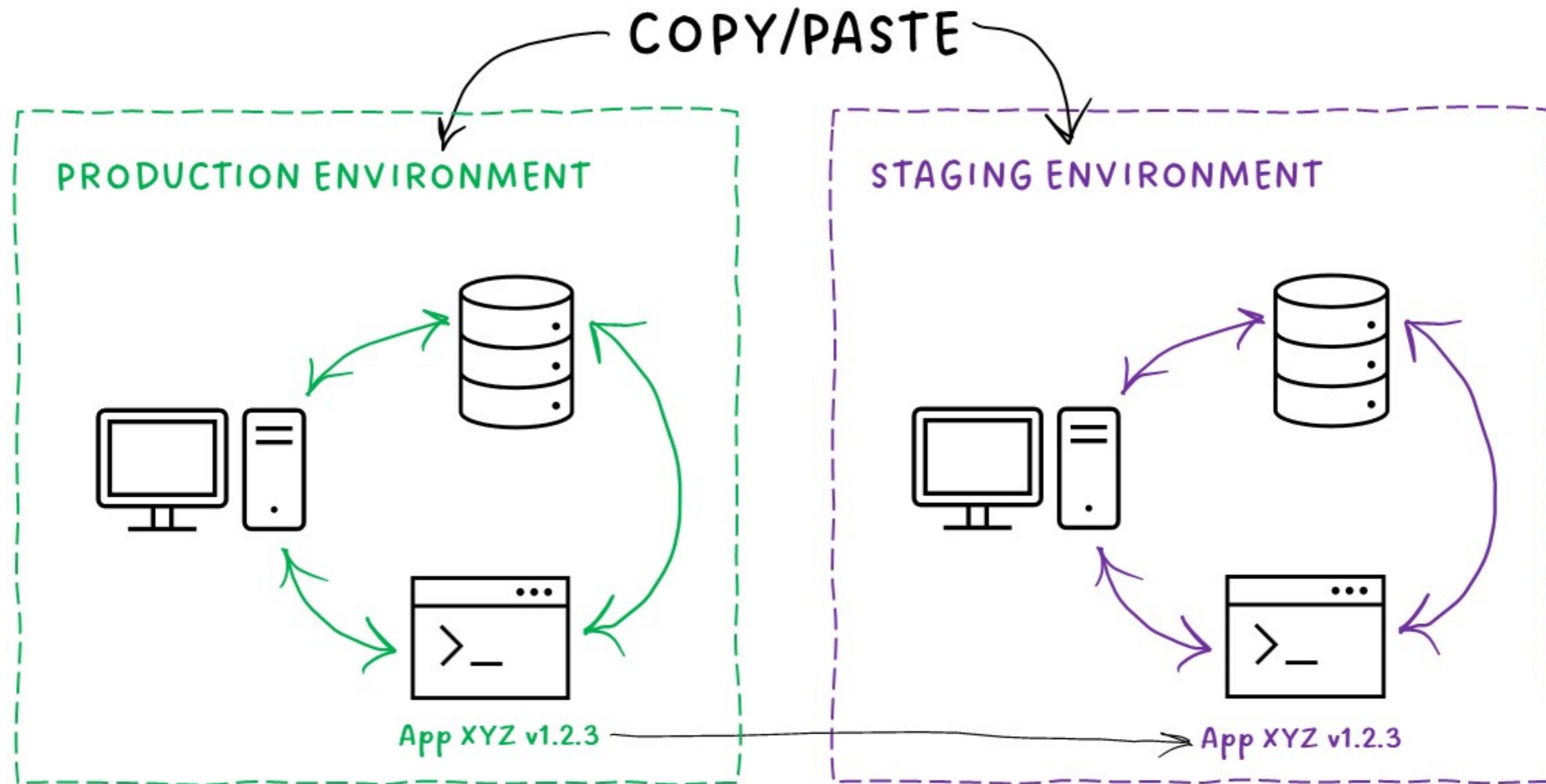


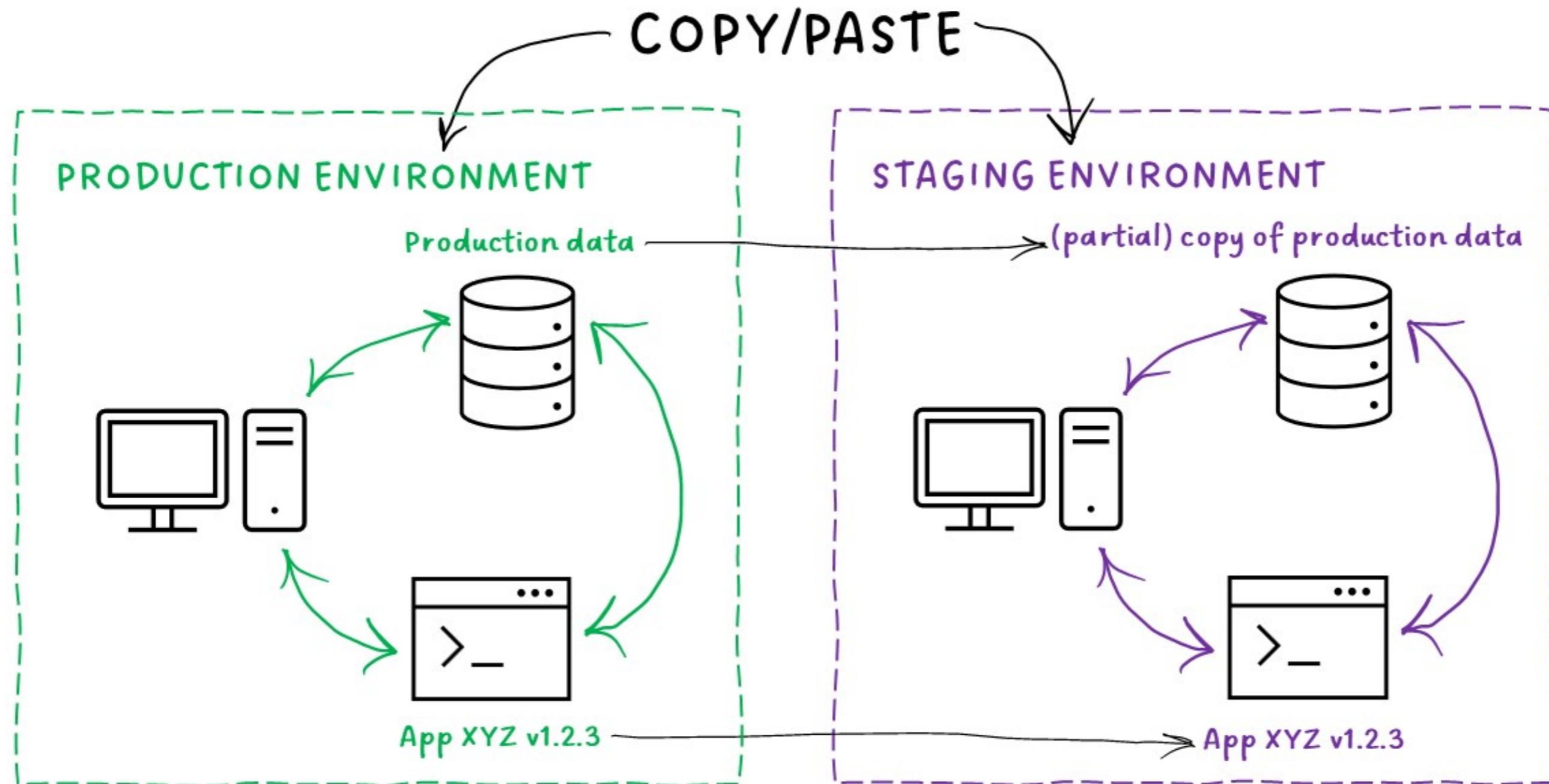
INTEGRATION TESTING



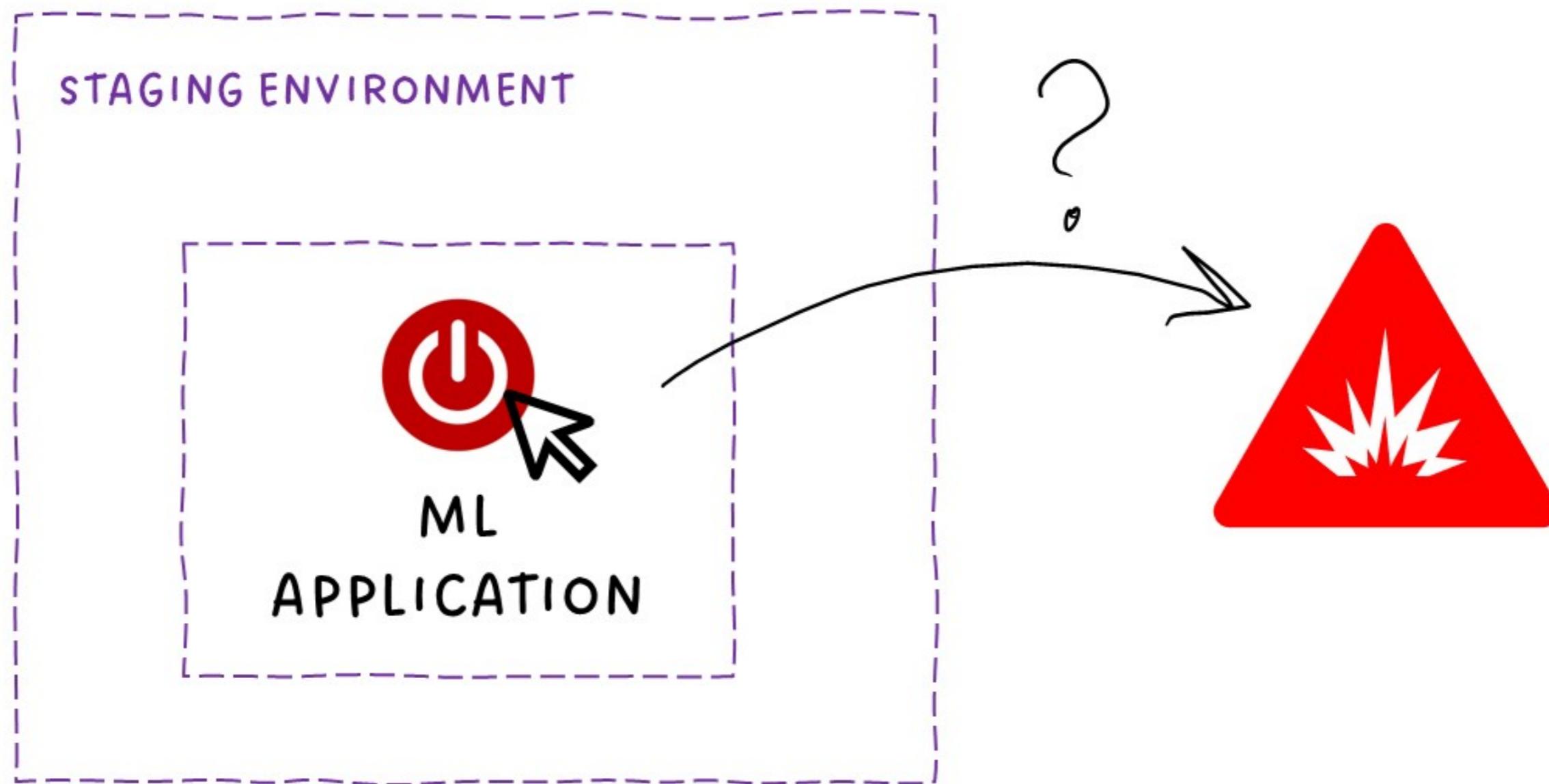




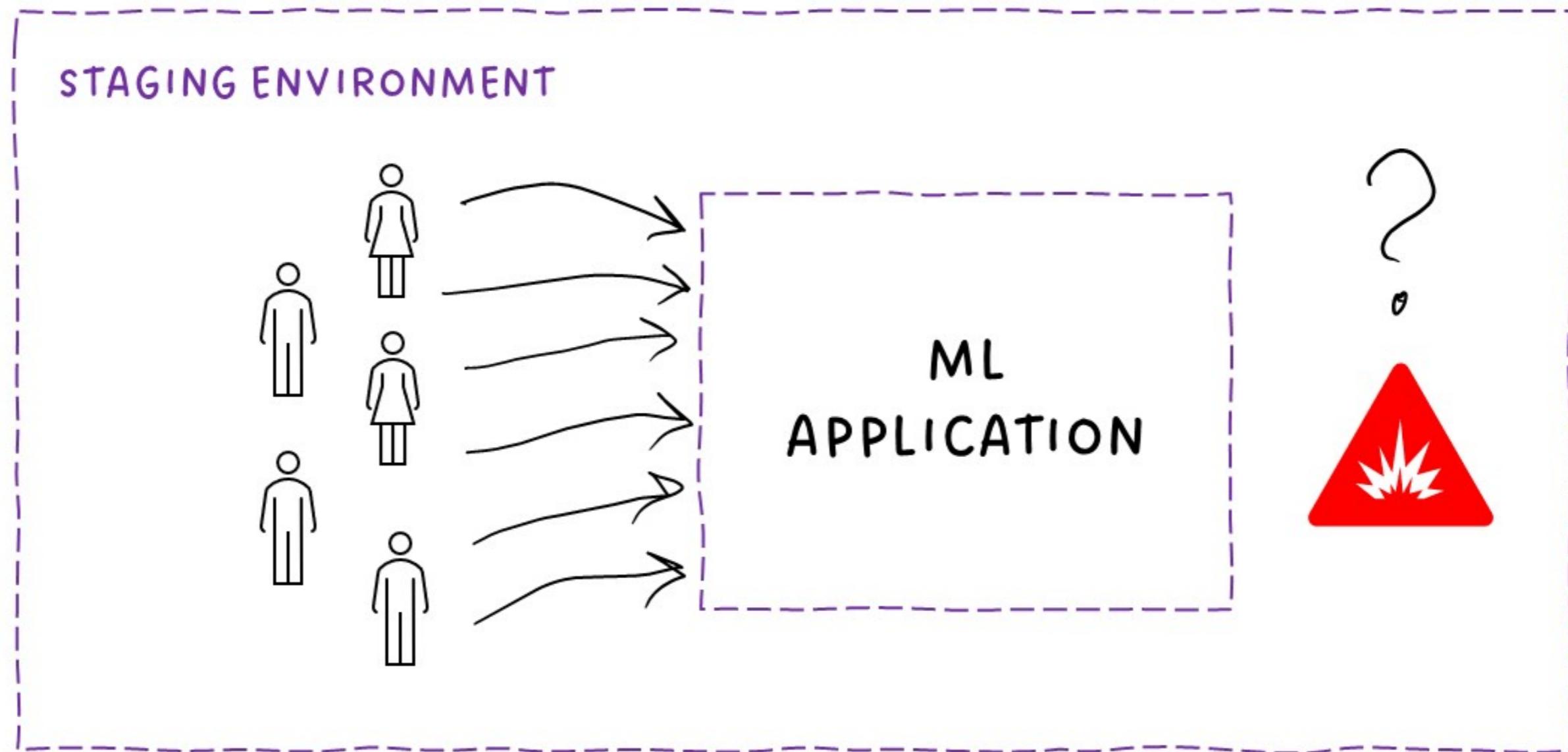




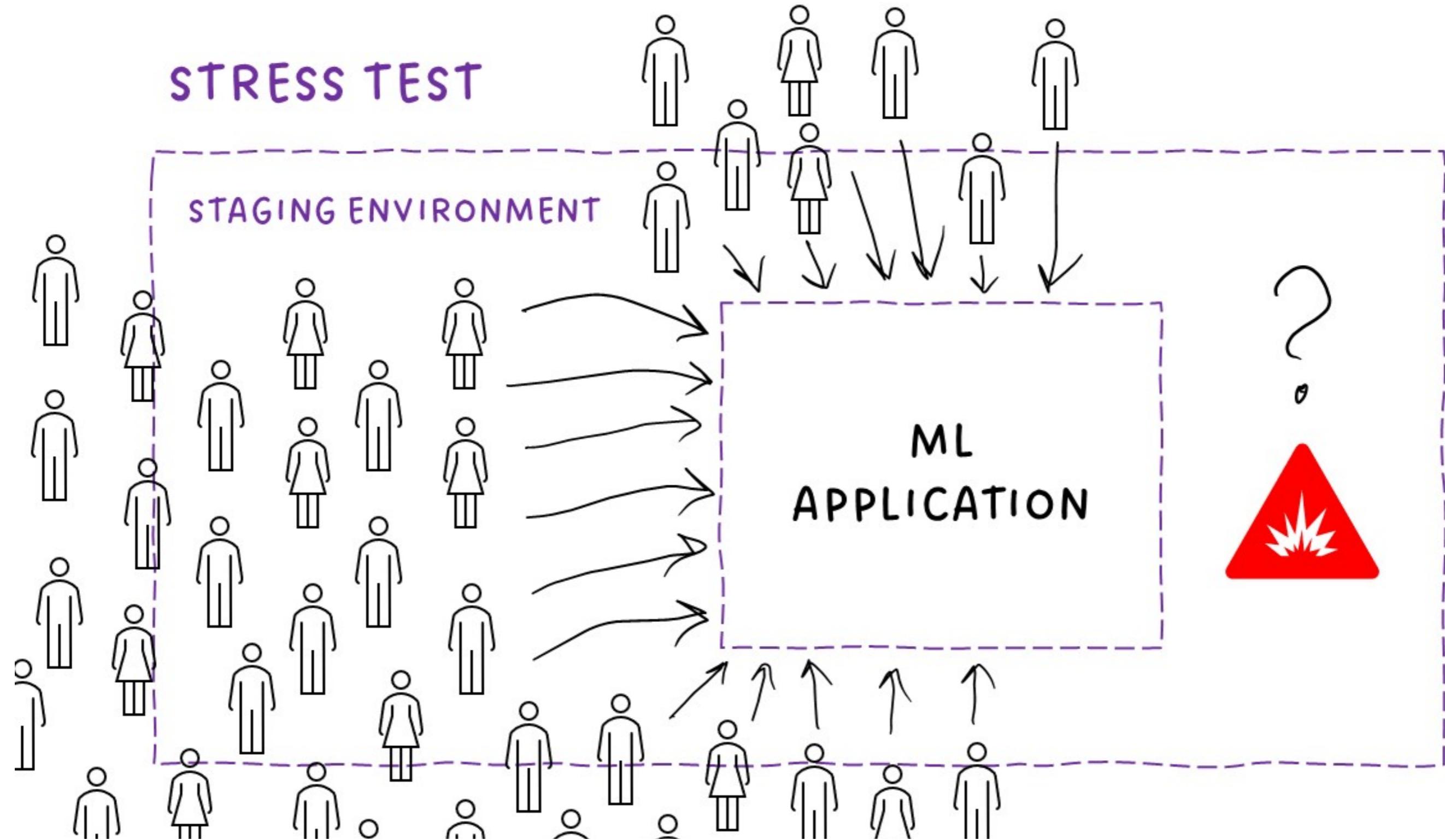
SMOKE TEST



LOAD TEST



STRESS TEST



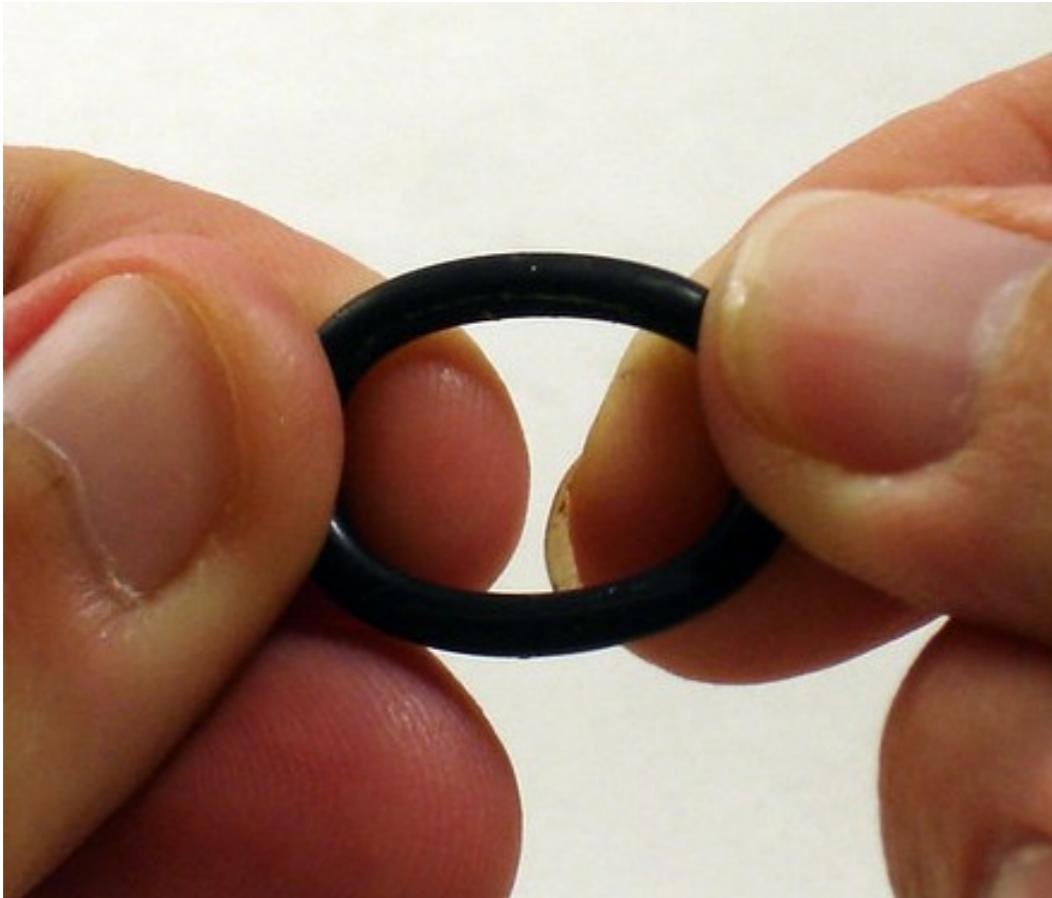
USER ACCEPTANCE TESTING (UAT)



Final “seal of approval” before deploying to production

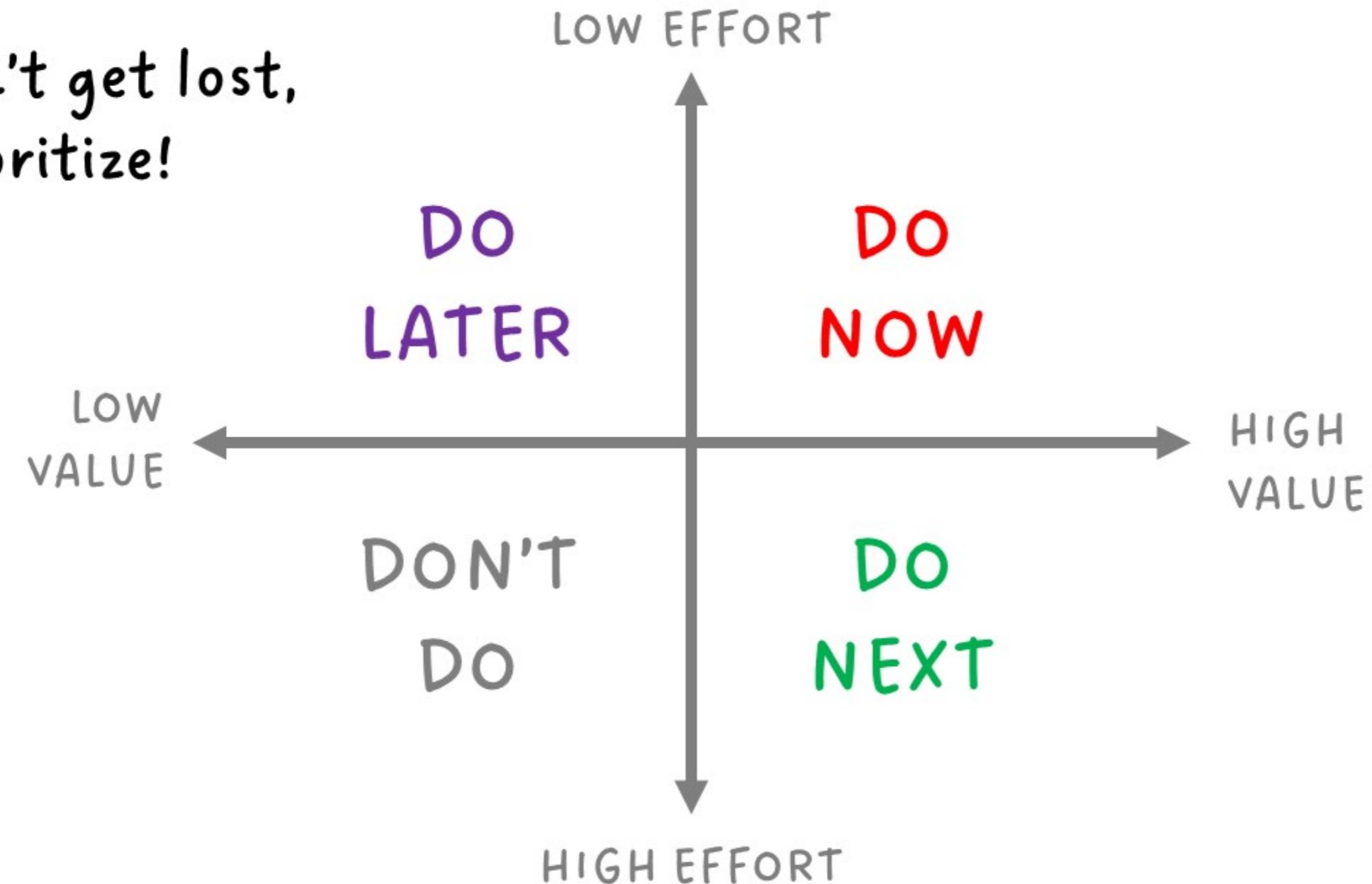
A single weak link breaks the system

We must be careful what we introduce in production.



The Challenger shuttle disaster in 1986 was caused by a failed rubber sealing ring

**Don't get lost,
prioritize!**



Let's practice!

MLOPS DEPLOYMENT AND LIFE CYCLING

Model deployment strategies

MLOPS DEPLOYMENT AND LIFE CYCLING

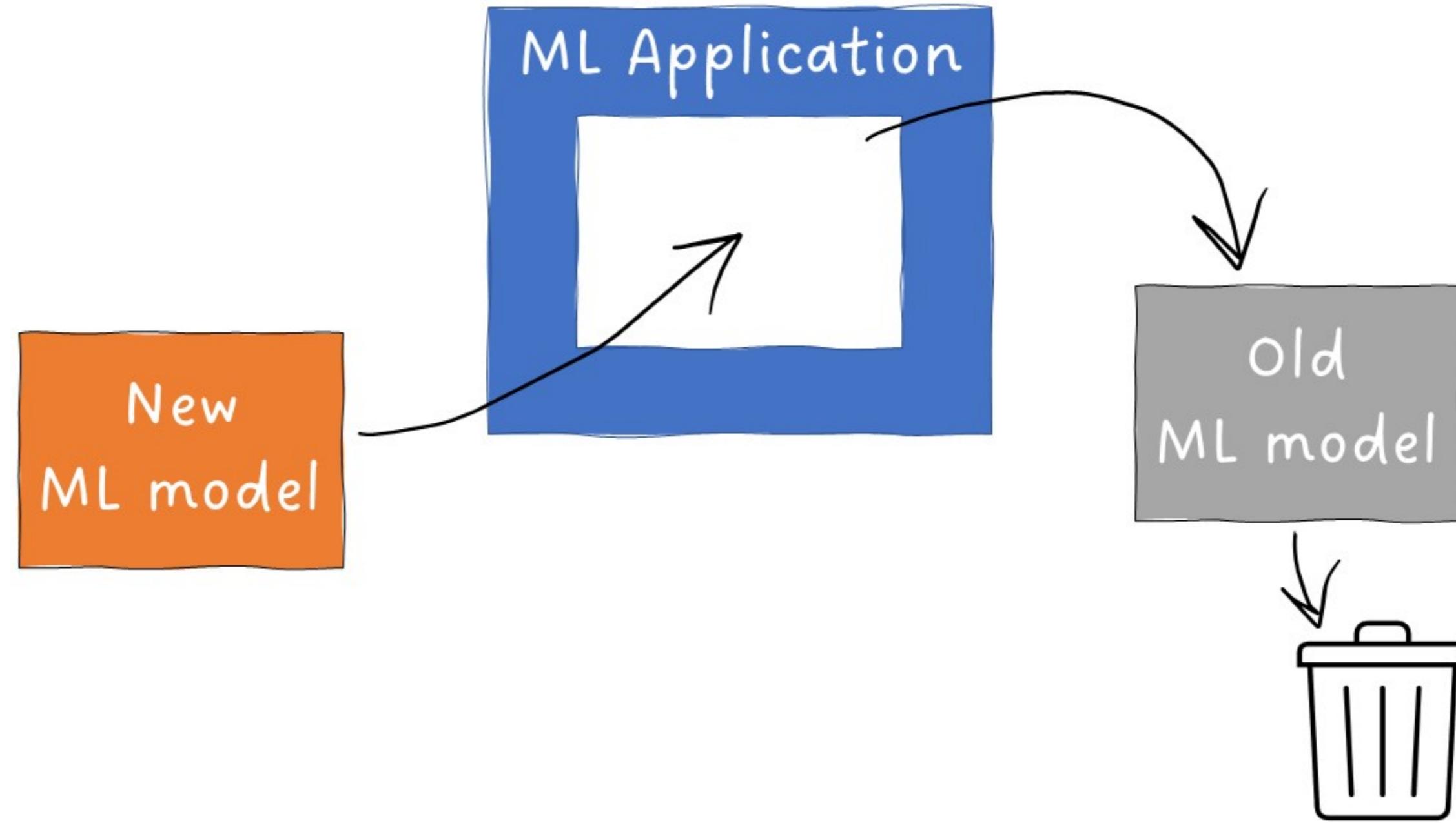


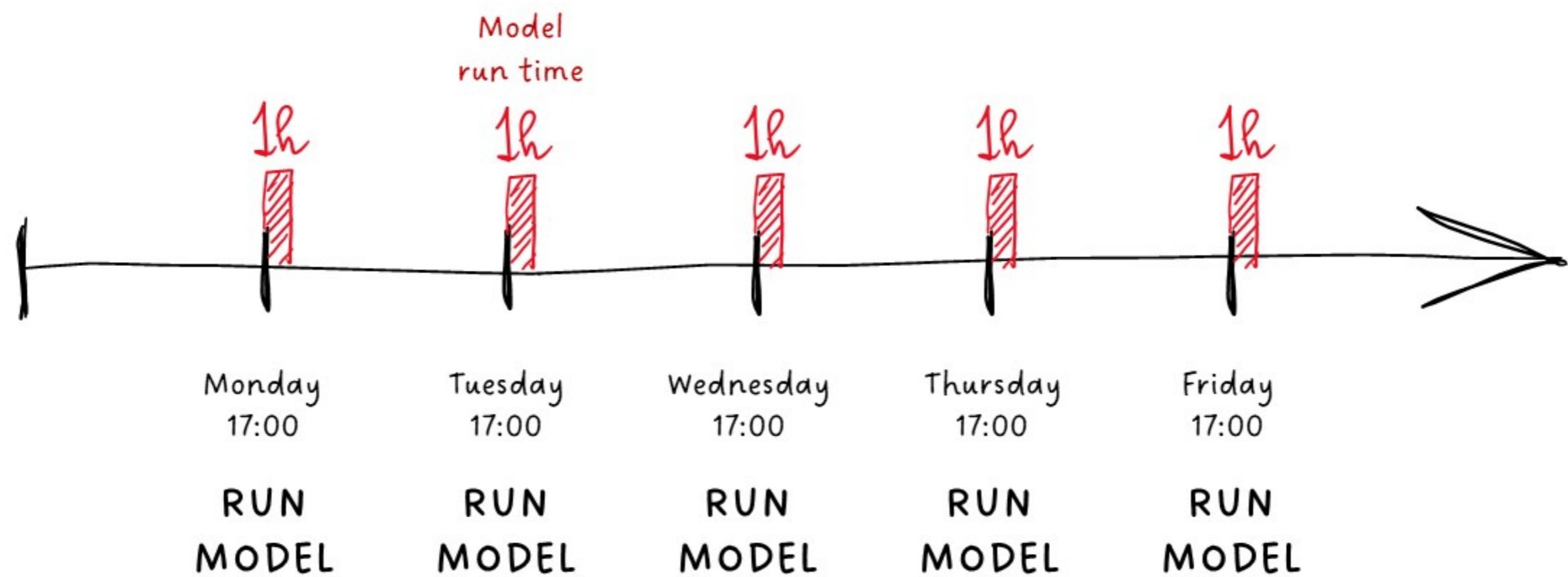
Nemanja Radojkovic

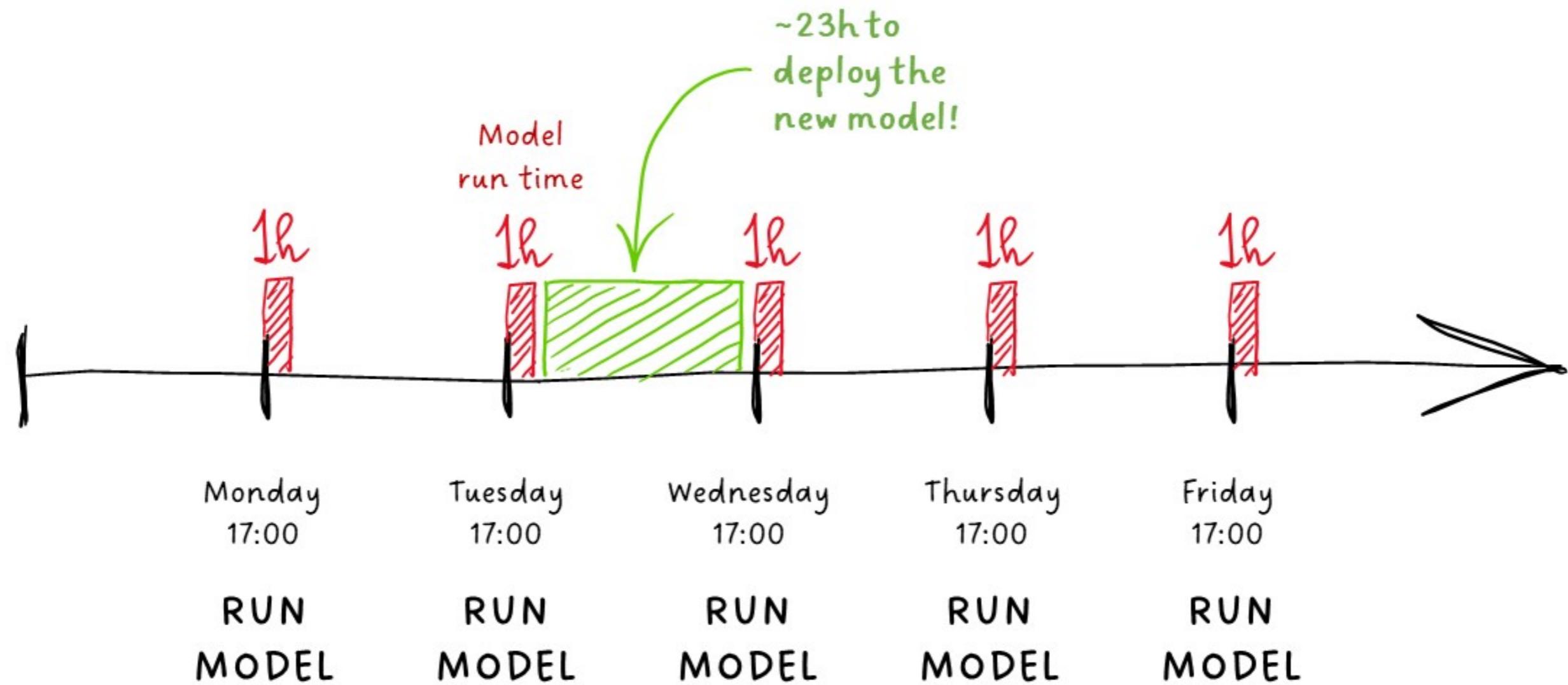
Senior Machine Learning Engineer

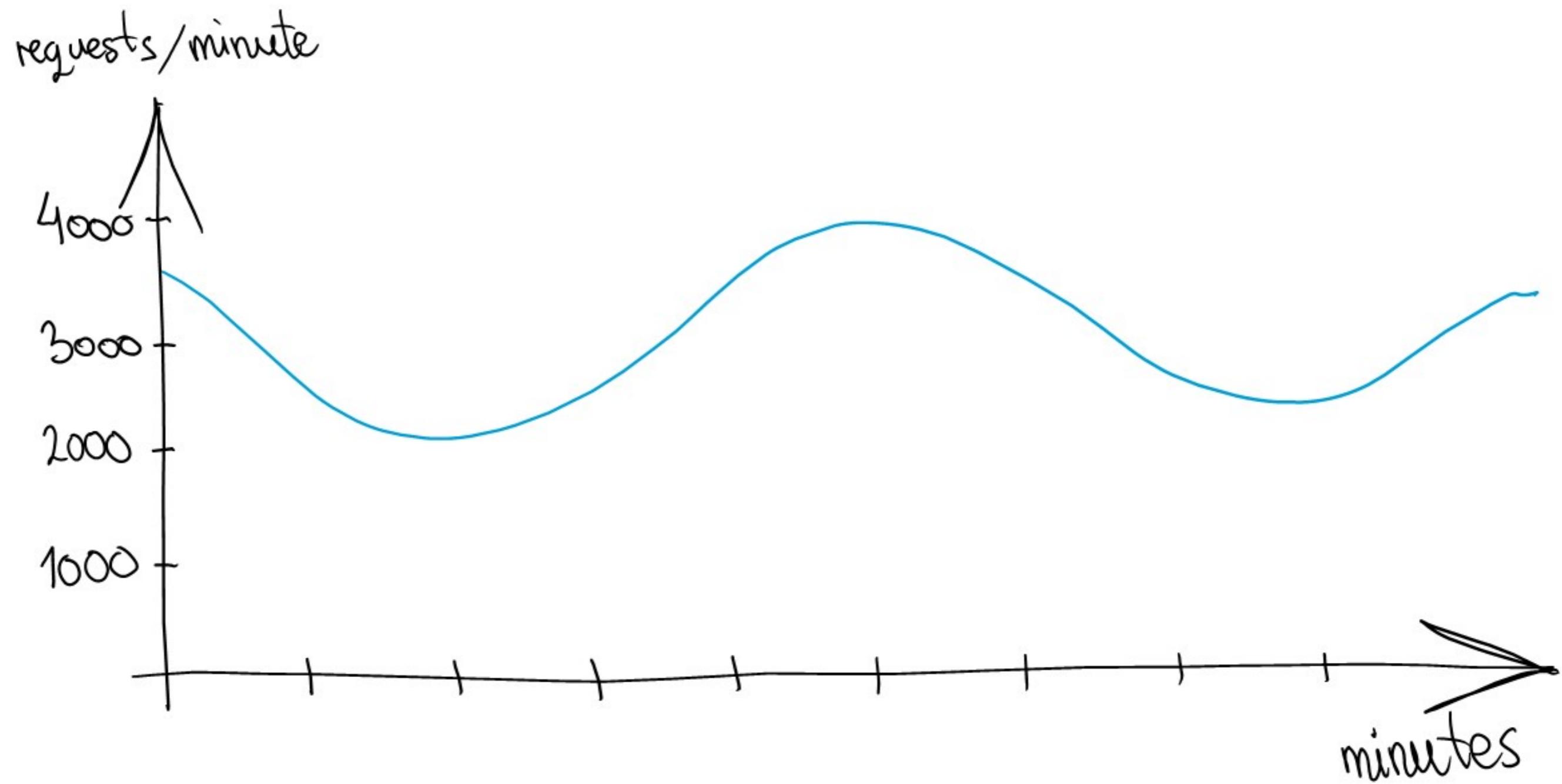
Deployment successful!

- ML app is running
- API handles 1000s of requests/hour
- One month later:
 - New, better features found
 - Collected new batch of training data
 - Ran the model build pipeline
 - New model package created

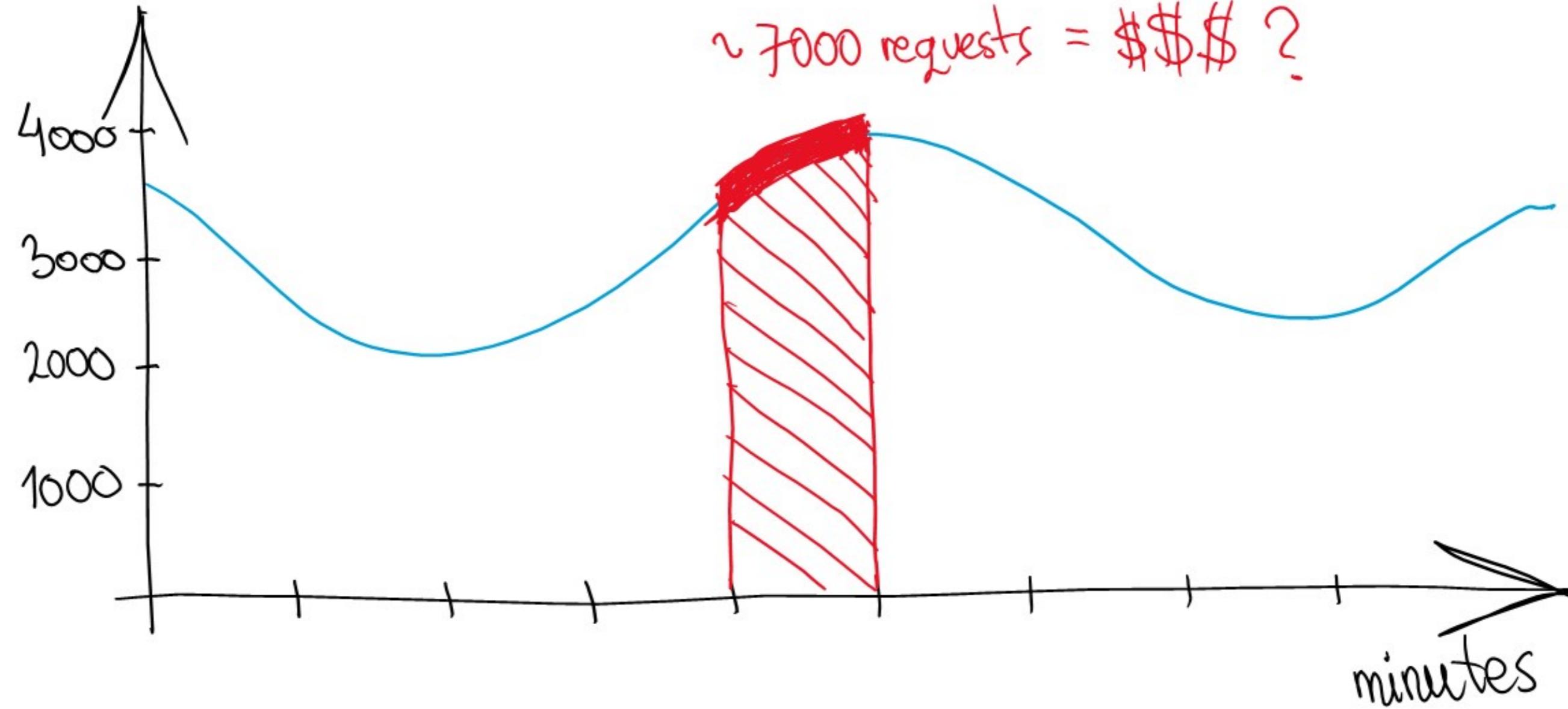


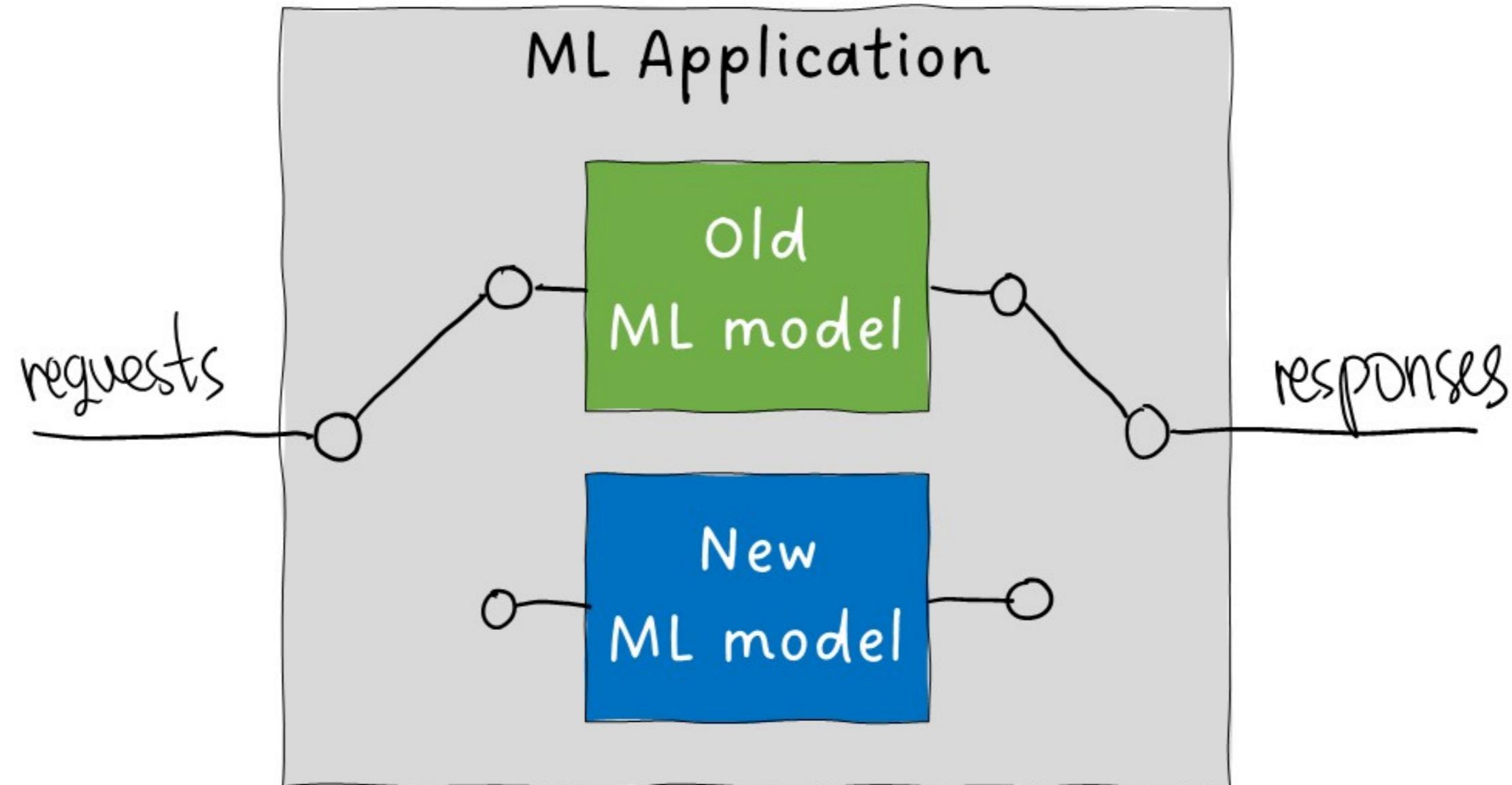


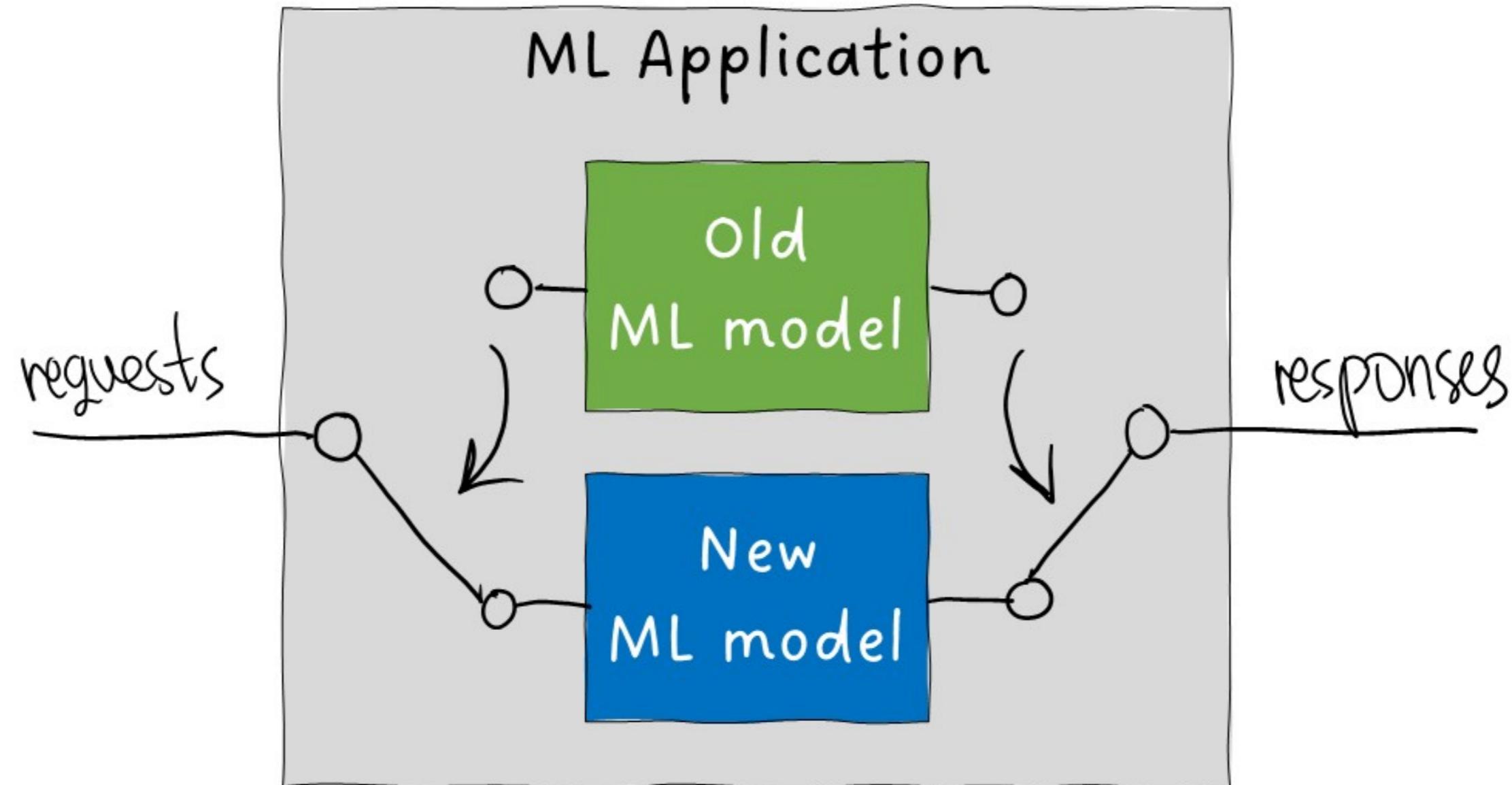




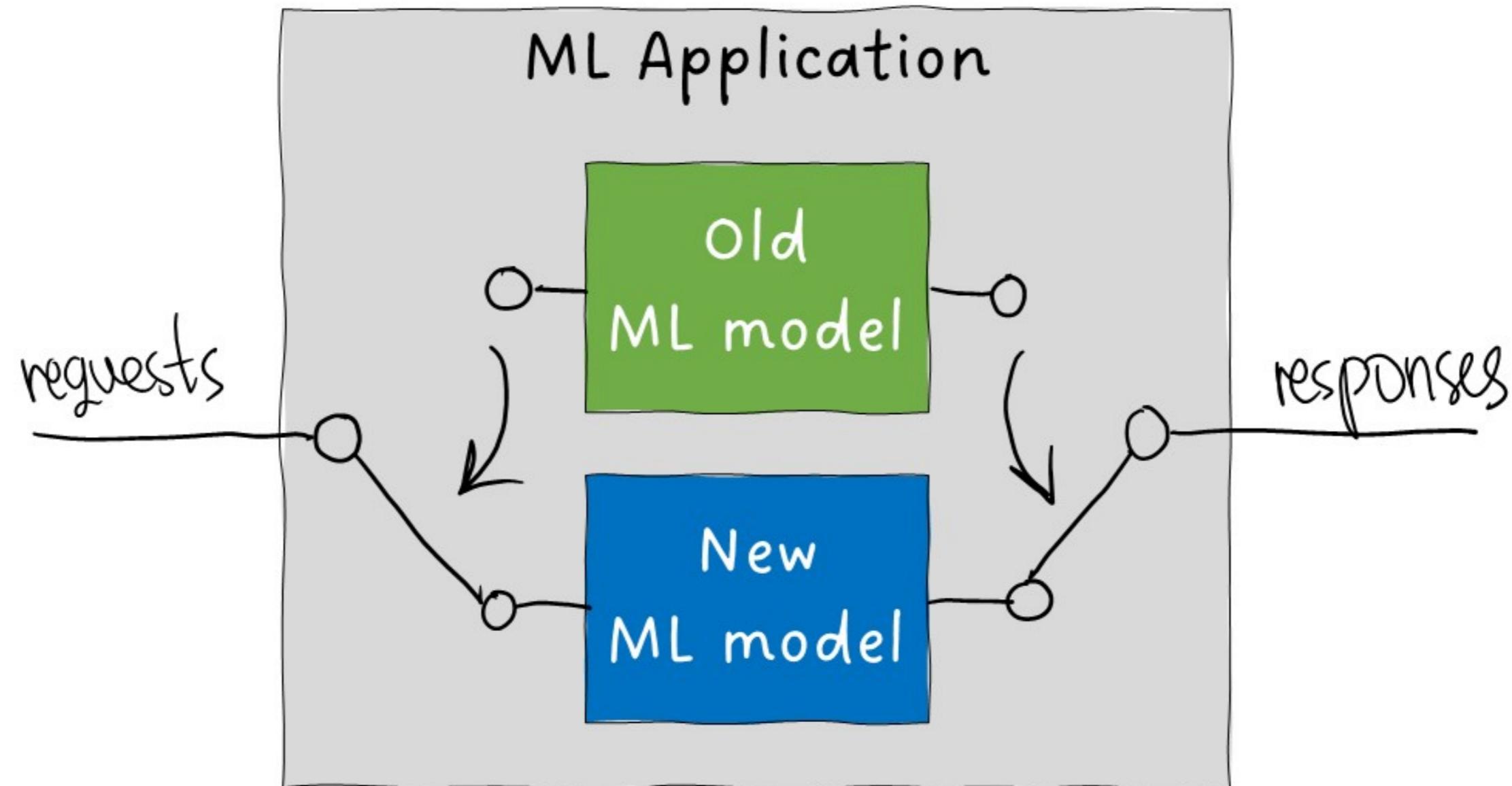
requests/minute



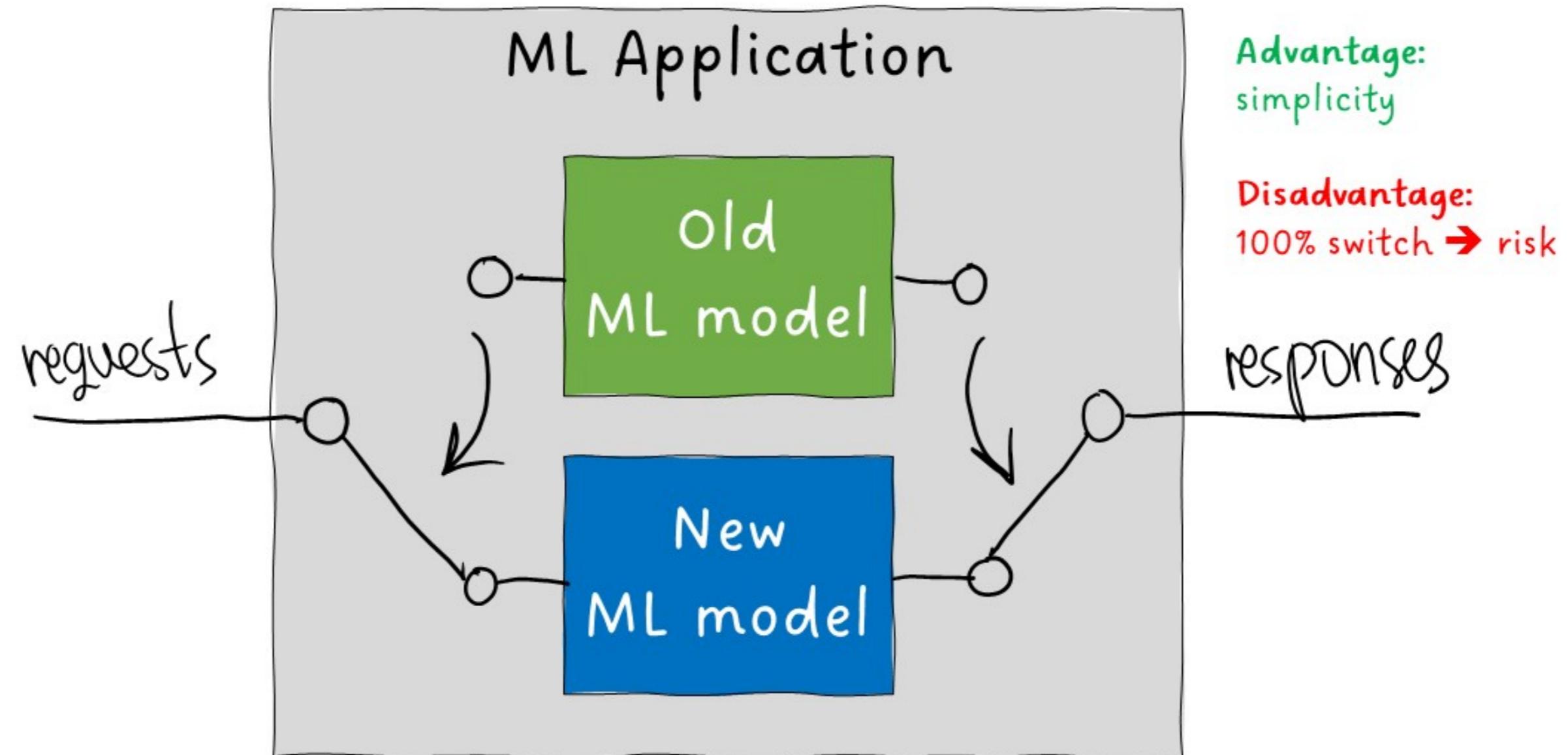




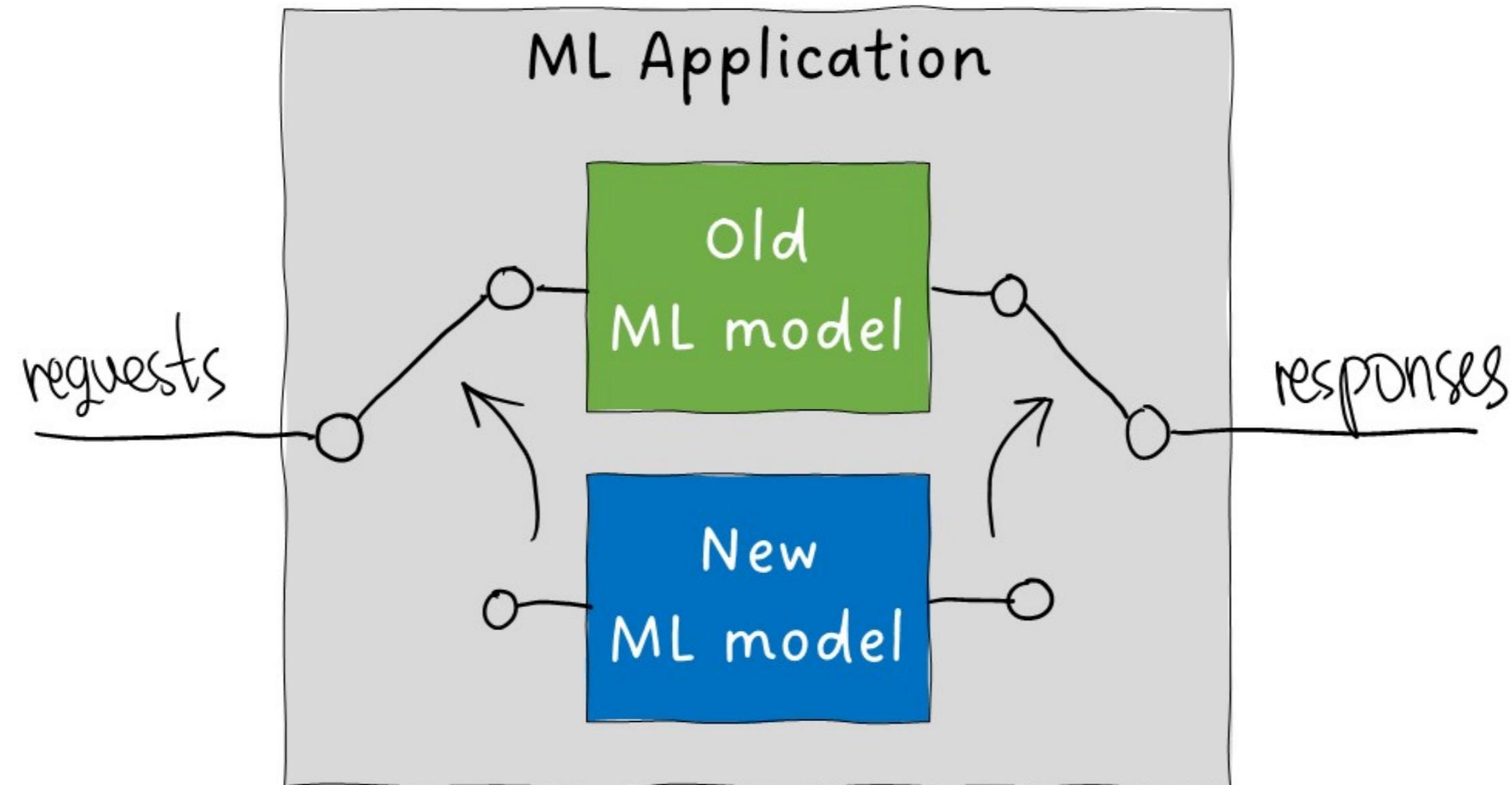
Blue/green deployment



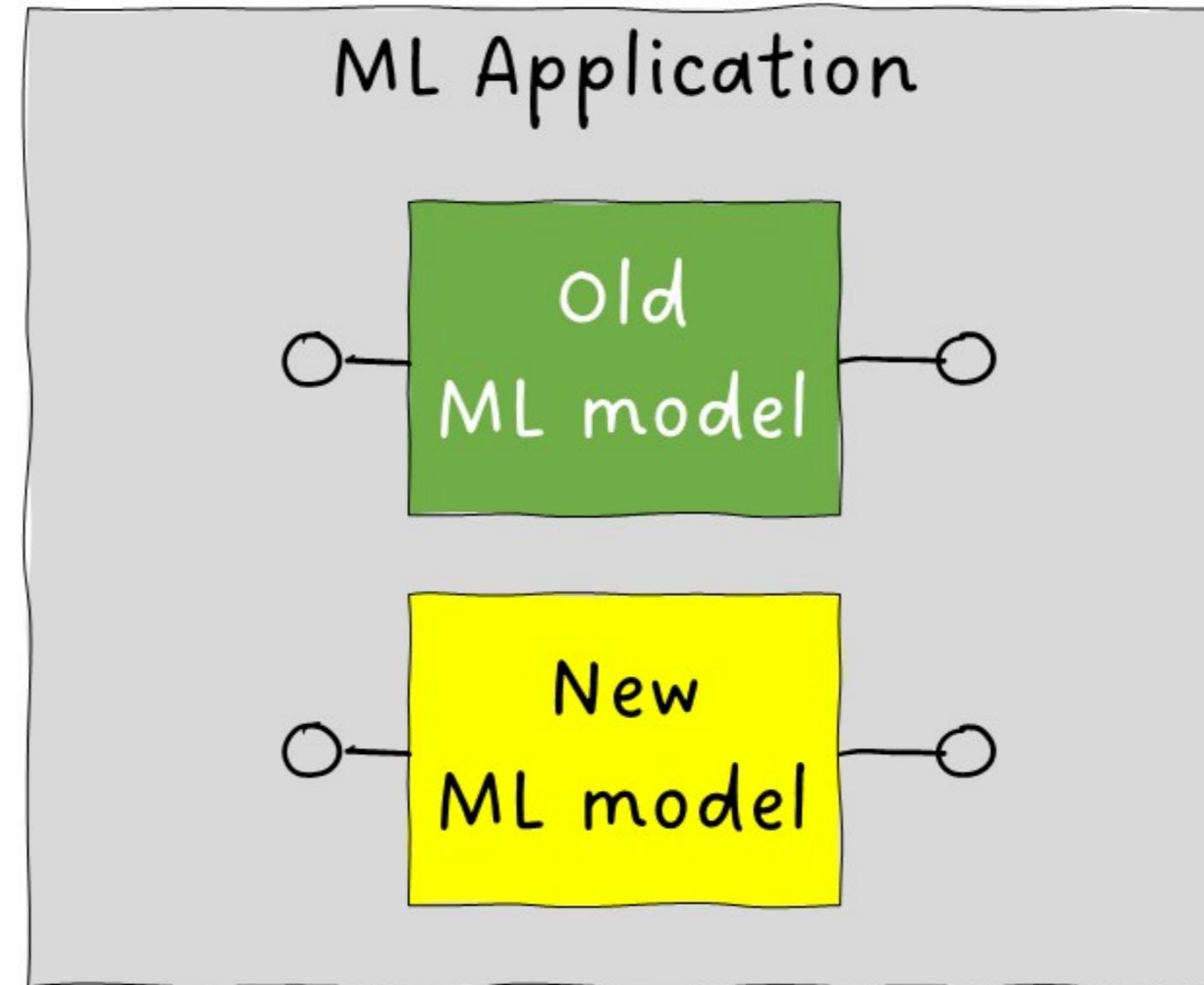
Blue/green deployment



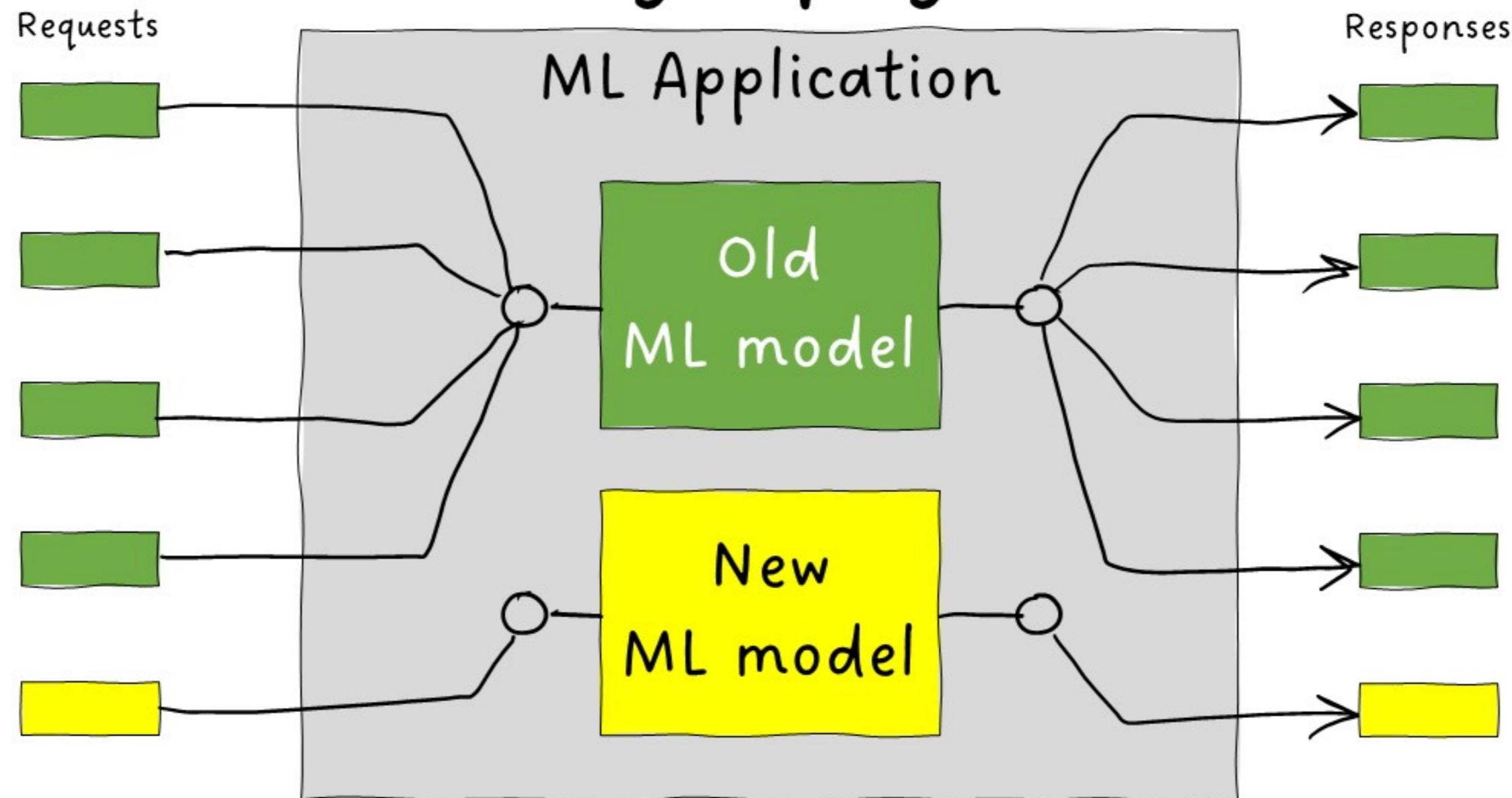
Rollback



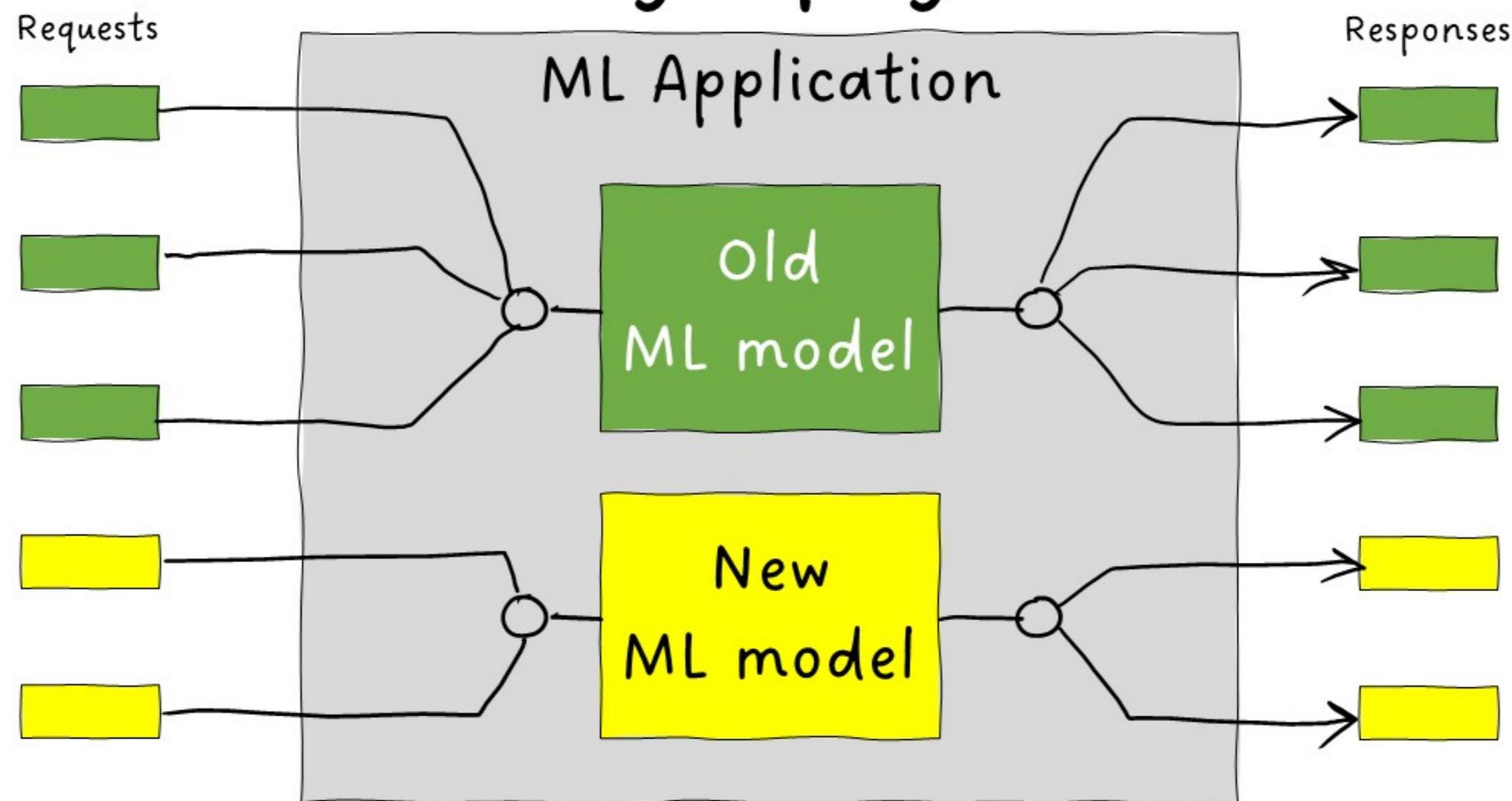
Canary deployment



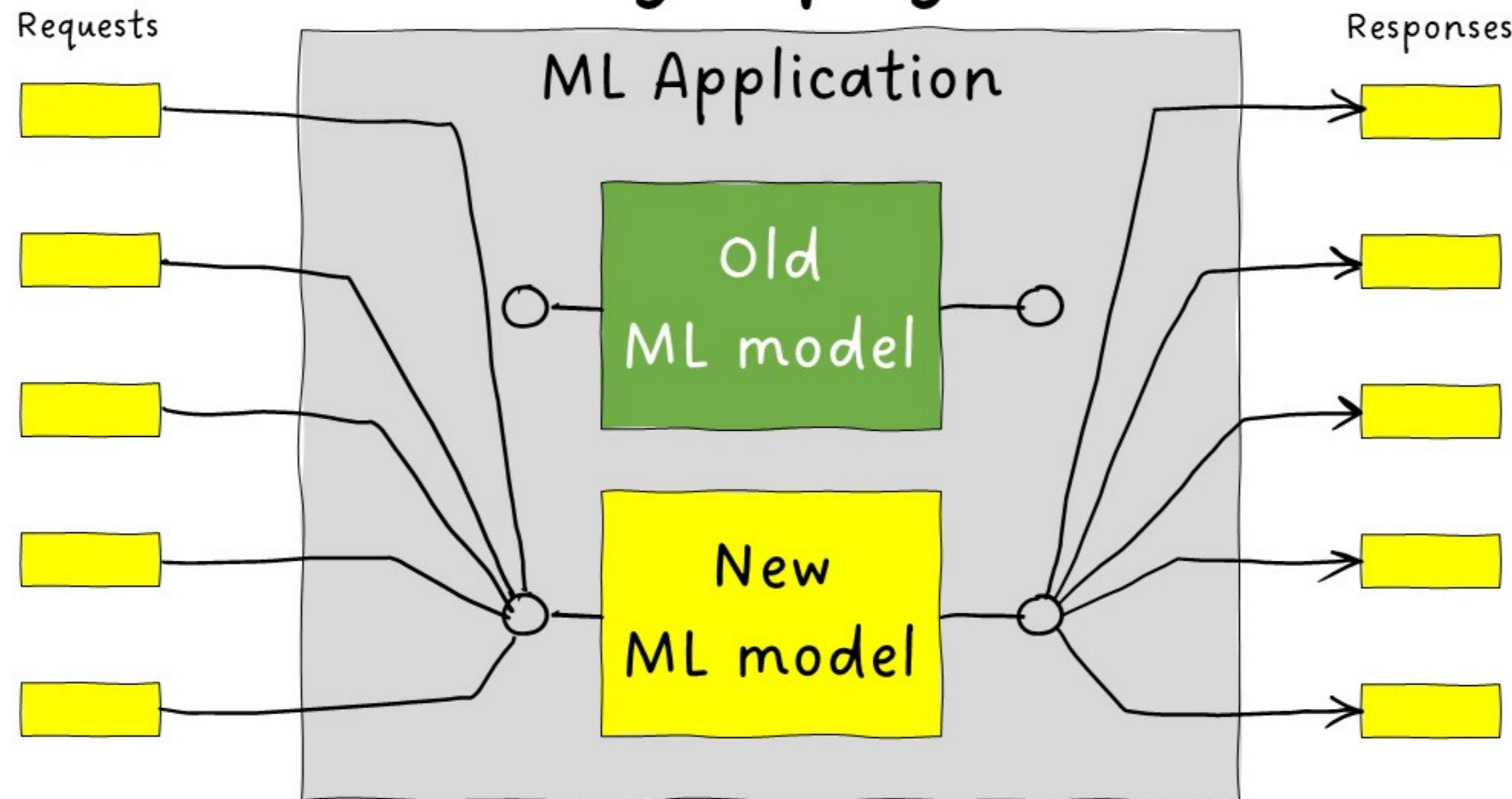
Canary deployment



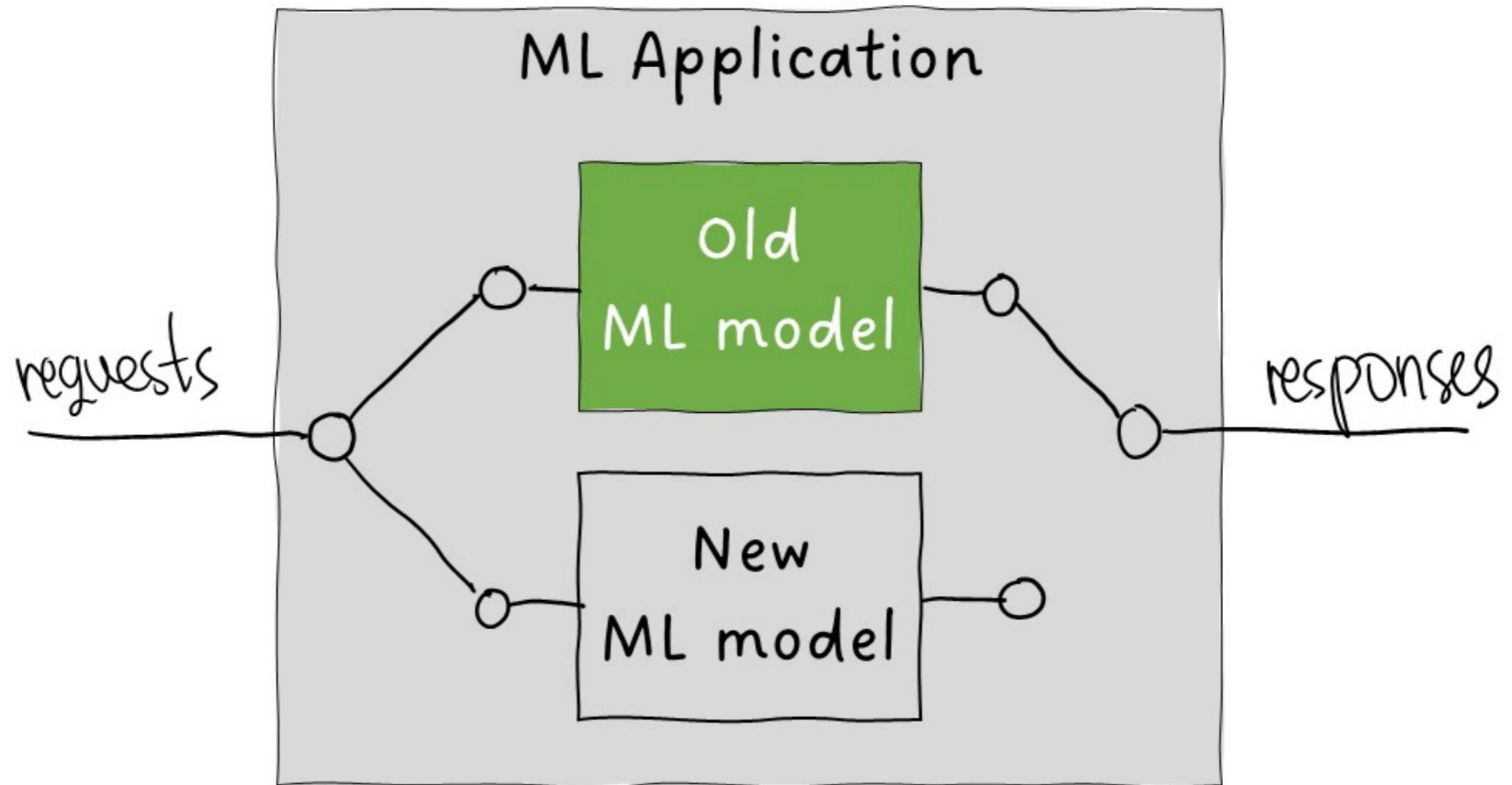
Canary deployment



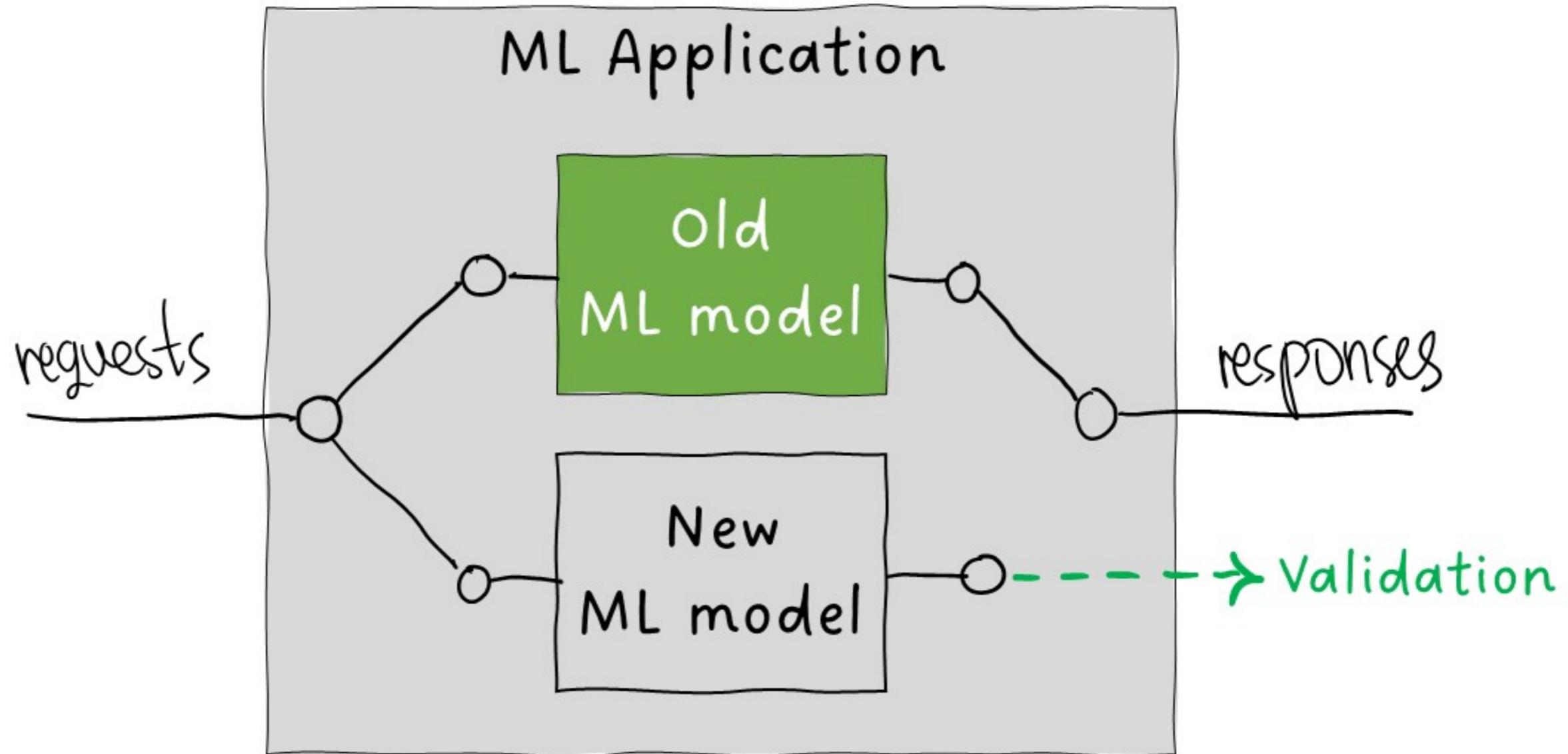
Canary deployment



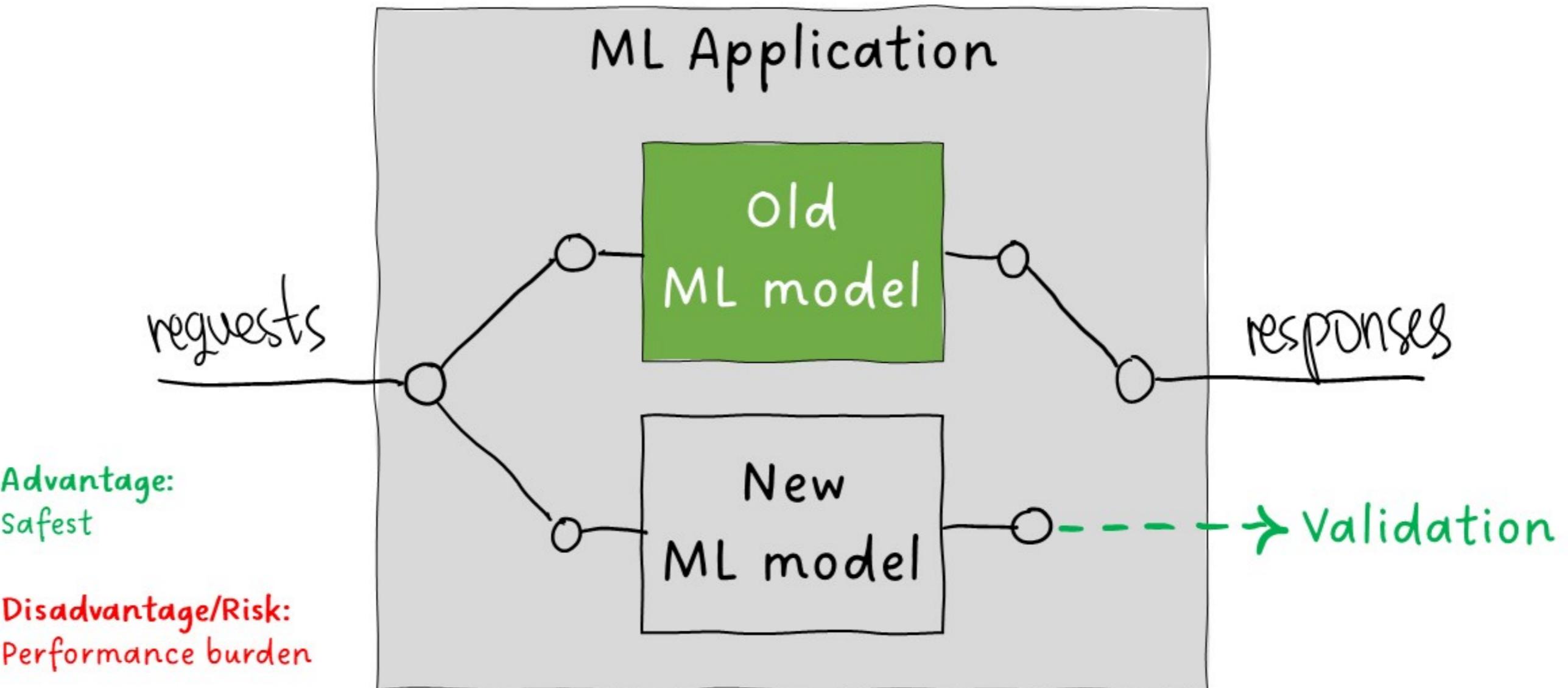
Shadow deployment



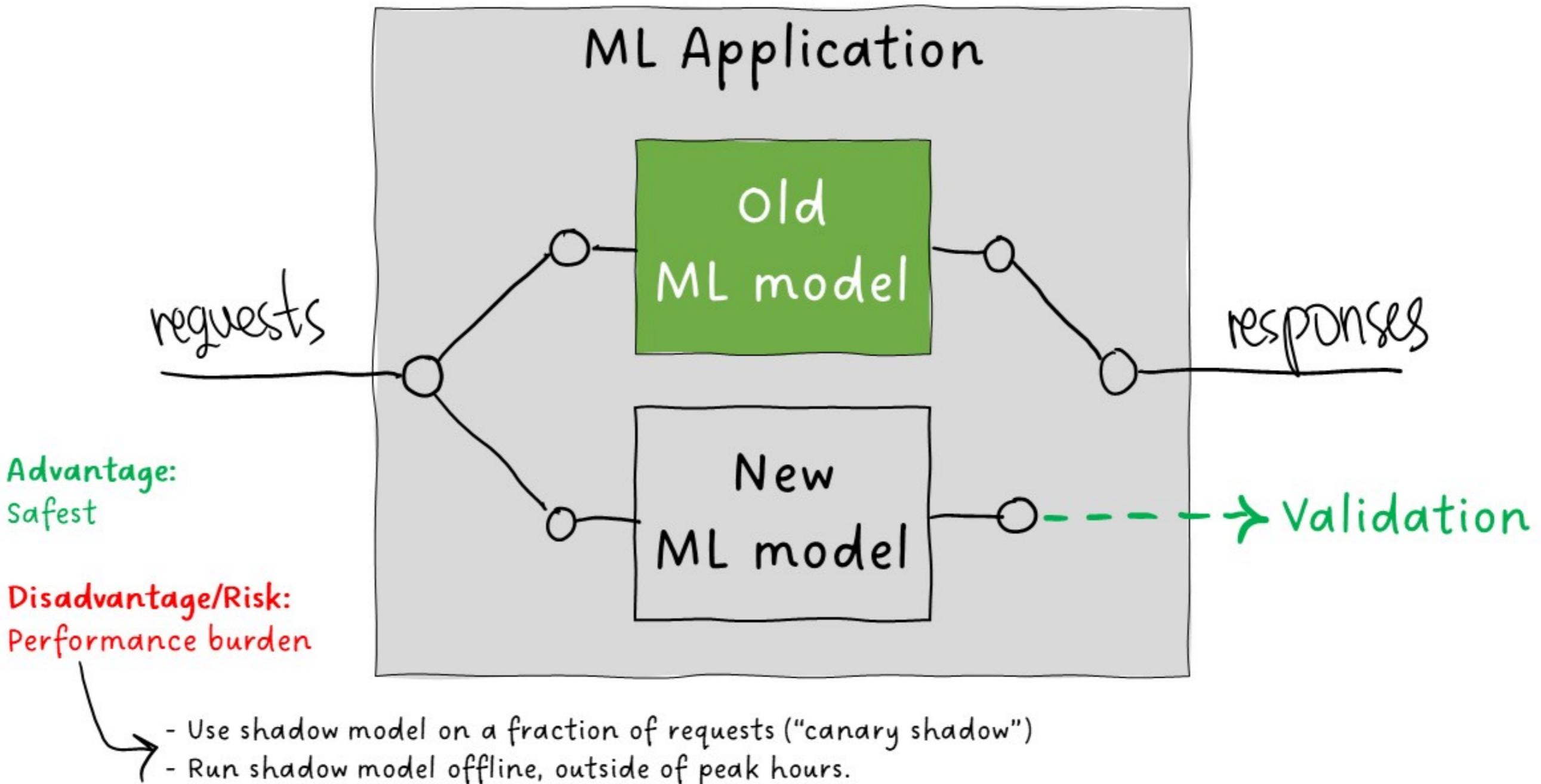
Shadow deployment



Shadow deployment



Shadow deployment



Let's practice!

MLOPS DEPLOYMENT AND LIFE CYCLING