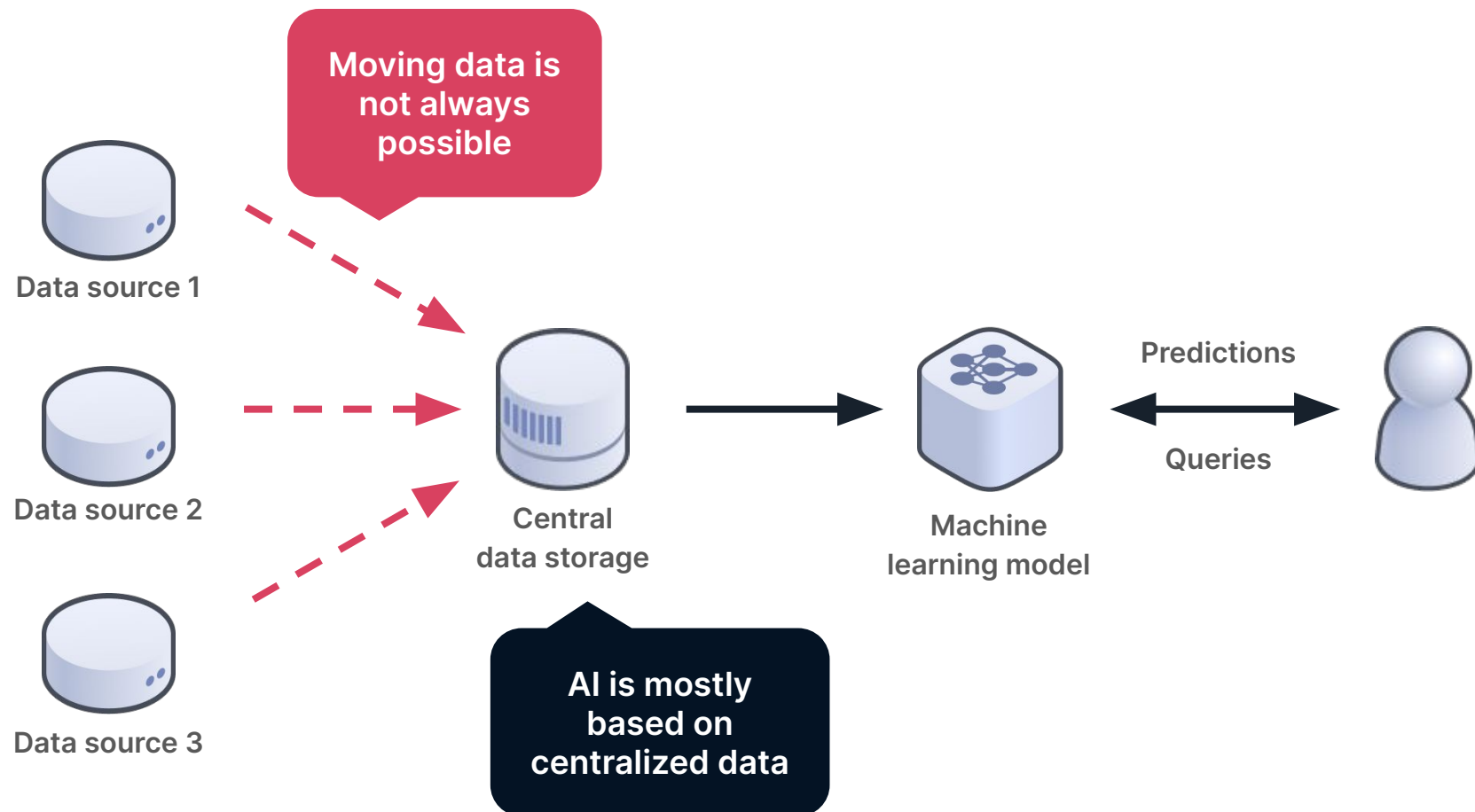


# Federated learning

AI training without moving data

# The AI problem

Data is out-of-reach in machine learning projects



# No data access

Common cases where data is out-of-reach in AI

“Cannot move data”

Data that cannot be shared due to ownership rights, confidentiality agreements, or practical issues in transferring edge-generated data.

- High volumes of data
- Network latency
- Connectivity issues

“Not allowed to move data”

Information that is restricted from being shared due to legal, regulatory, or compliance reasons to ensure data privacy and security.

- Data privacy and transfer laws
- Intellectual property rights
- Non-disclosure agreements

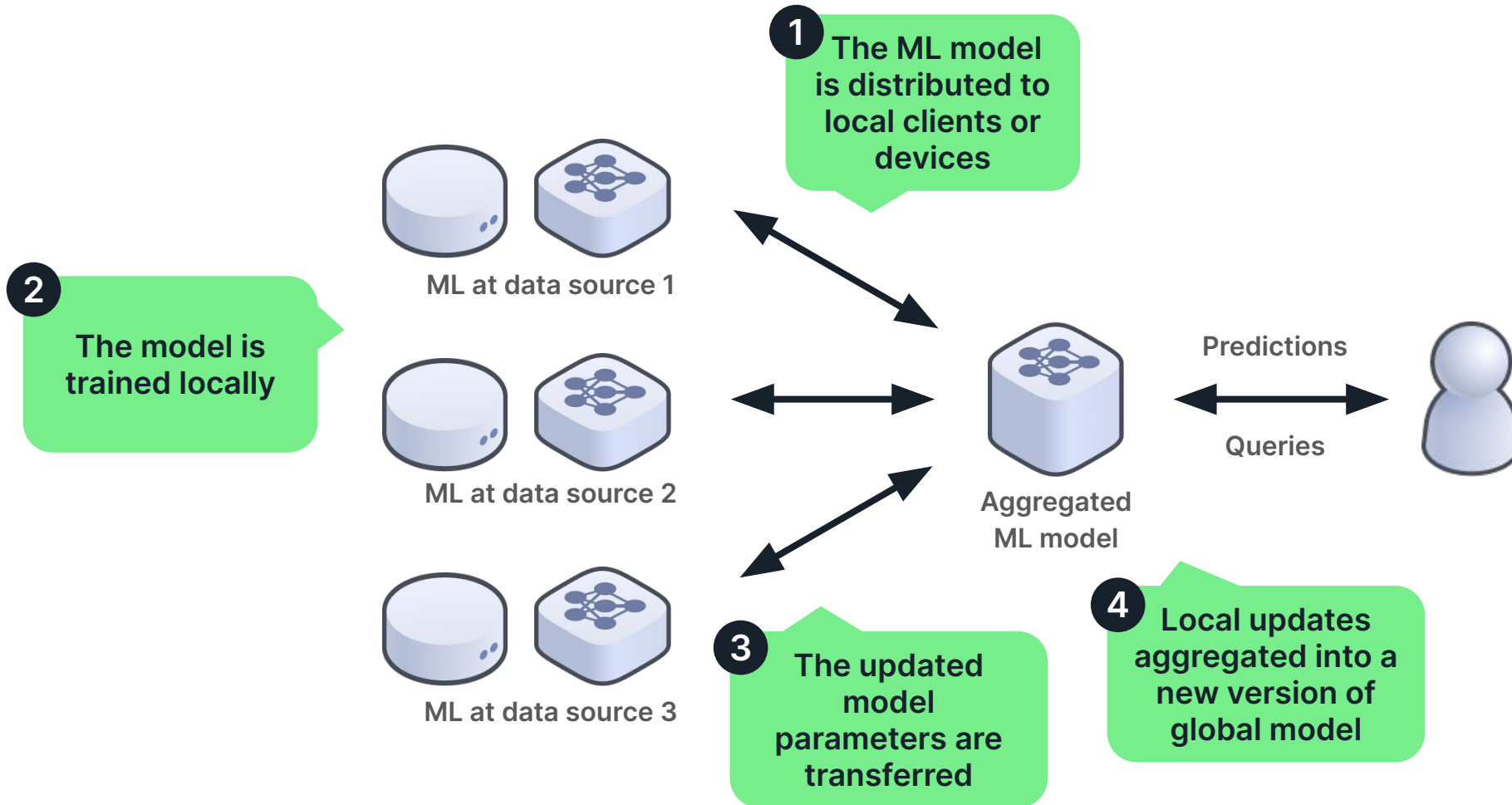
“Don’t want to move data”

Sensitive data kept private to protect privacy, ensure confidentiality, prevent misuse, and guard against cybersecurity threats.

- Business sensitive
- Private data
- Proprietary data

# Federated learning

Bring the ML to the data instead of data to the ML



# The FEDn framework

Federated learning by Scaleout:

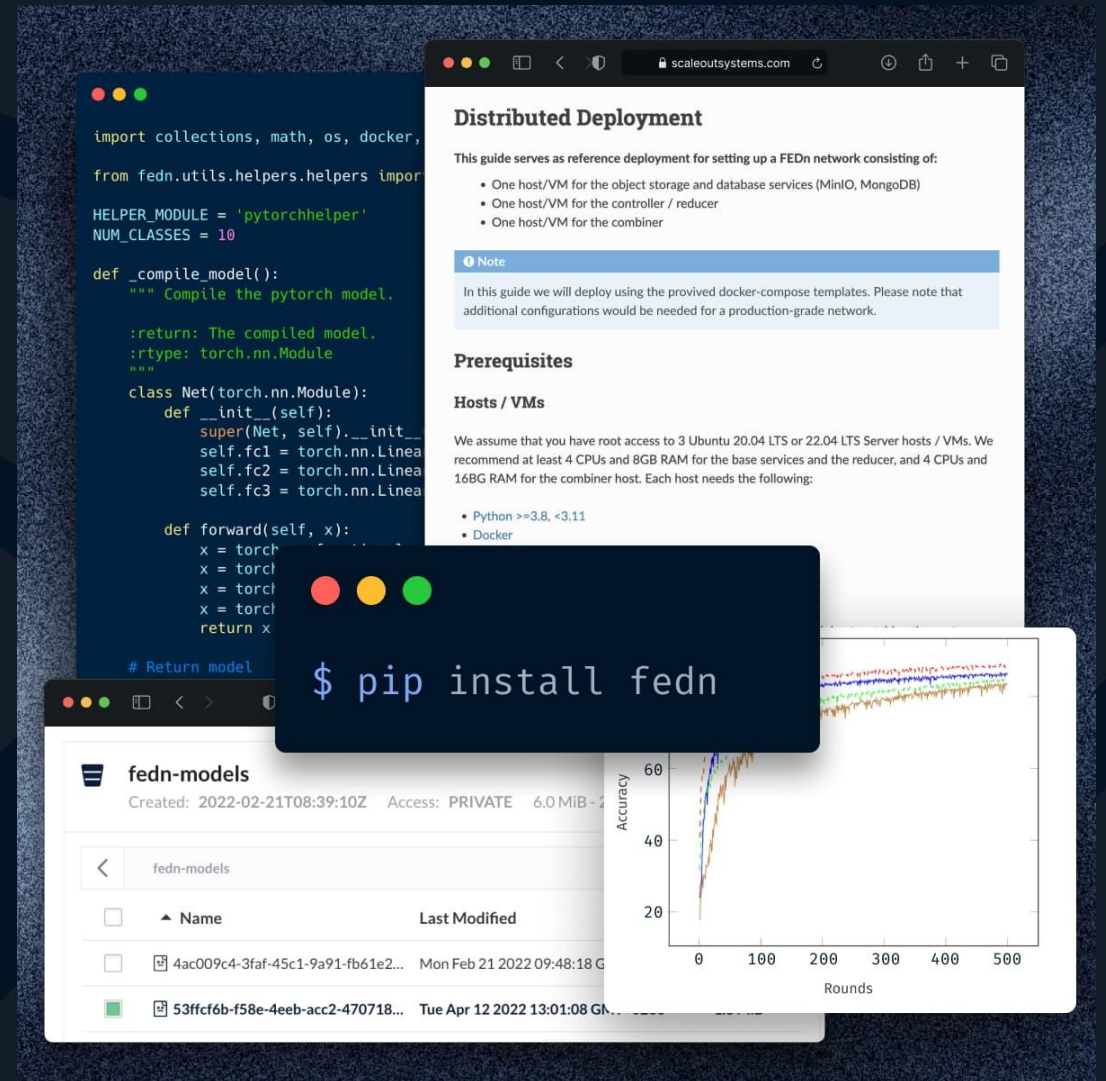
- with truly distributed (not simulated) federated learning
- with instant-start so you can focus on development
- built for scale

Take your project from R&D to a highly secure production.

# FEDn SDK

FEDn, our open-source SDK, is designed for scalable and secure federated learning.

It ensures a smooth transition from development to large-scale deployment, emphasizing data security and efficient model aggregation.



```
import collections, math, os, docker,
from fedn.utils.helpers.helpers import

HELPER_MODULE = 'pytorchhelper'
NUM_CLASSES = 10

def _compile_model():
    """ Compile the pytorch model.

    :return: The compiled model.
    :rtype: torch.nn.Module
    """
    class Net(torch.nn.Module):
        def __init__(self):
            super(Net, self).__init__()
            self.fc1 = torch.nn.Linear
            self.fc2 = torch.nn.Linear
            self.fc3 = torch.nn.Linear

        def forward(self, x):
            x = torch
            x = torch
            x = torch
            x = torch
            return x

    # Return model
```

**Distributed Deployment**

This guide serves as reference deployment for setting up a FEDn network consisting of:

- One host/VM for the object storage and database services (MinIO, MongoDB)
- One host/VM for the controller / reducer
- One host/VM for the combiner

**Note**

In this guide we will deploy using the provided docker-compose templates. Please note that additional configurations would be needed for a production-grade network.

**Prerequisites**

**Hosts / VMs**

We assume that you have root access to 3 Ubuntu 20.04 LTS or 22.04 LTS Server hosts / VMs. We recommend at least 4 CPUs and 8GB RAM for the base services and the reducer, and 4 CPUs and 16GB RAM for the combiner host. Each host needs the following:

- Python >=3.8, <3.11
- Docker

```
$ pip install fedn
```

**fedn-models**

Created: 2022-02-21T08:39:10Z Access: PRIVATE 6.0 MiB - 2

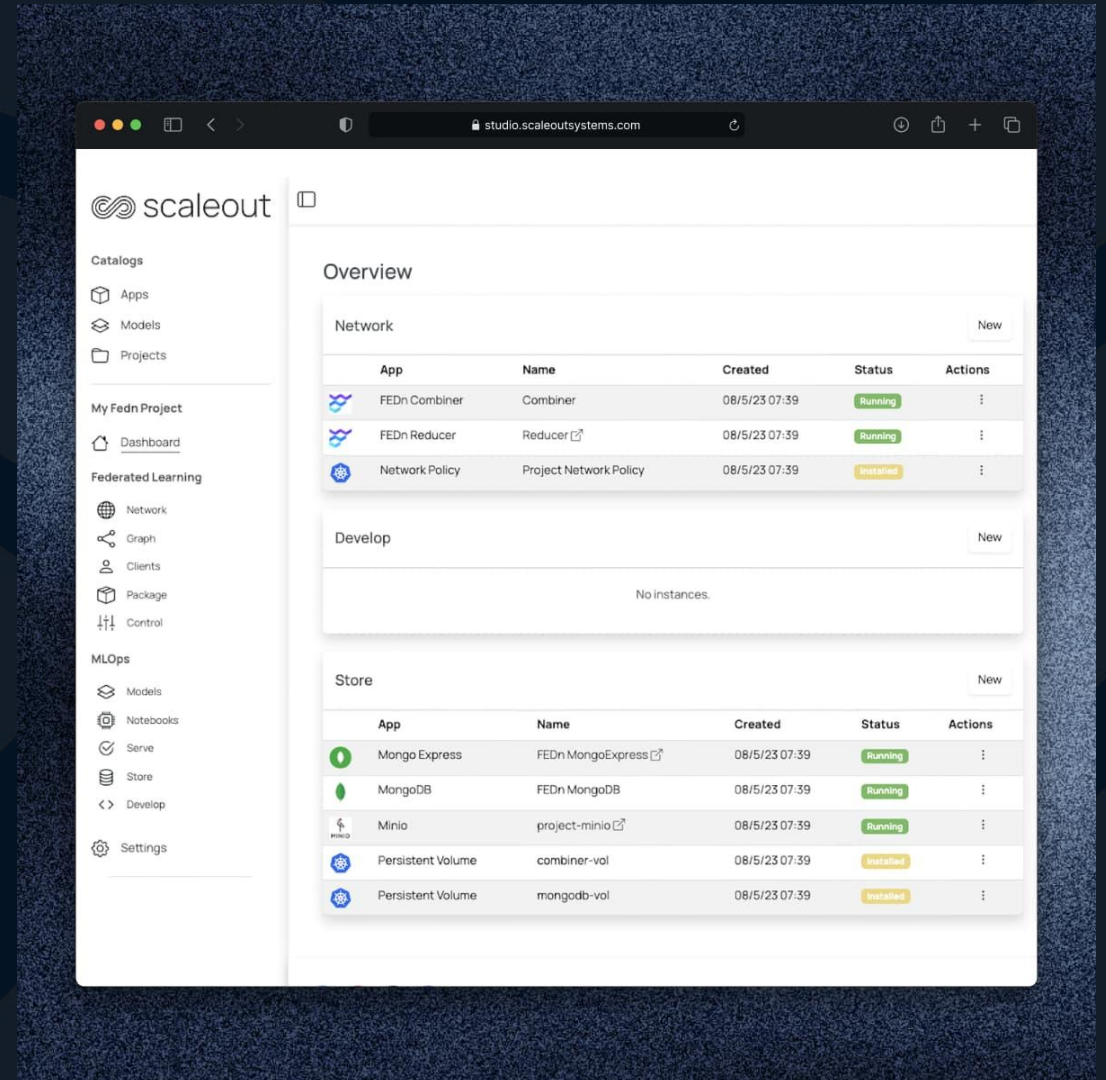
Name	Last Modified
4ac009c4-3faf-45c1-9a91-fb61e2...	Mon Feb 21 2022 09:48:18 G
53fcc6b-f58e-4eeb-acc2-470718...	Tue Apr 12 2022 13:01:08 G

Accuracy vs Rounds graph showing training progress over 500 rounds.

# FEDn Studio

FEDn Studio, our cloud-native platform, delivers seamless and secure federated machine learning, designed to integrate effortlessly with your existing MLOps pipeline.

It scales to any demand, ideal for both early-stage experimentation and fast iterations, as well as for on-premise deployment in production stages.





[scaleoutsystems.com](https://scaleoutsystems.com)