Distributed File Systems Made Easy with Python's fsspec

Guy Hardonag, February 2025

lakeFS

Guy Hardonag, lakeFS by Treeverse

✉ guy.hardonag@treeverse.io

in www.linkedin.com/in/guy-hardonag-2ab10264

🌐 https://lakeFS.io

STARTING LOCALLY

- Library support
- Python standard file interface
- Ease of use

STARTING LOCALLY (example)

Using pandas to read a CSV

Python

```python
import pandas as pd
df = pd.read_csv("/path/to/local/file.csv")
```

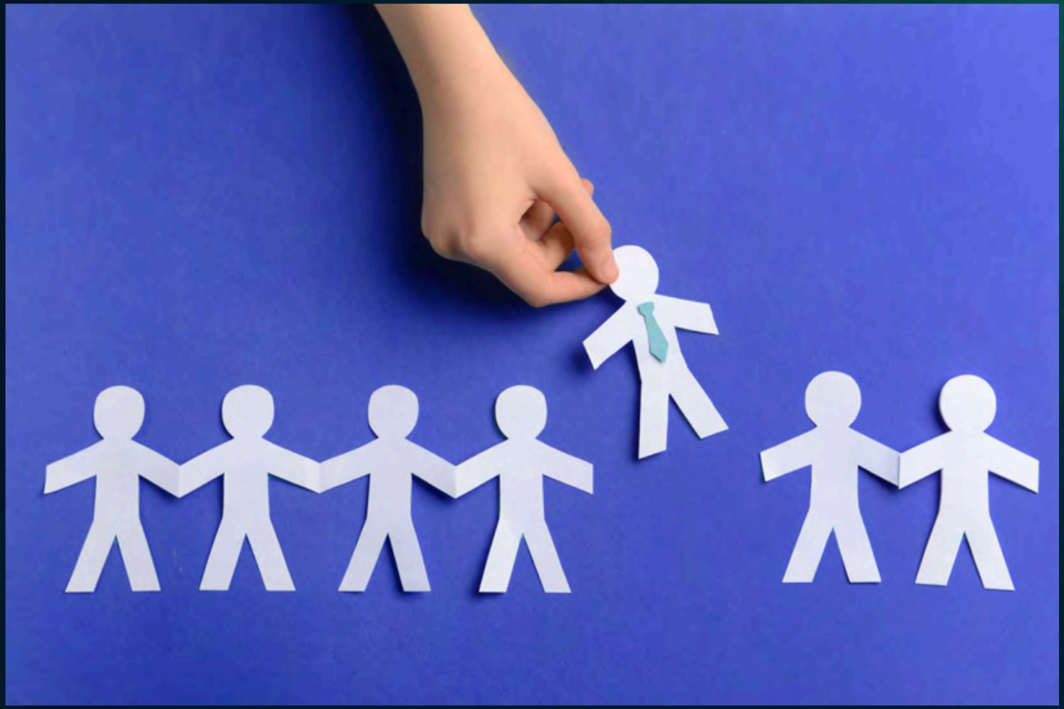
Loading a TensorFlow model

Python

```python
import tensorflow as tf
model = tf.keras.models.load_model("/path/to/local/model")
```

Why cloud?

✓ **REDUNDANCY**



Why cloud?

✓ **SCALABILITY**

Why cloud?

✓ **COLLABORATION**

Why cloud?

✓ **SECURITY**

Why cloud?

✓ ACCESSIBILITY

BRIDGING THE GAP:

WORKING WITH CLOUD

BRIDGING THE GAP: MANUAL SYNC

Shell

```shell
aws s3 cp s3://bucket/file.csv ./file.csv
```

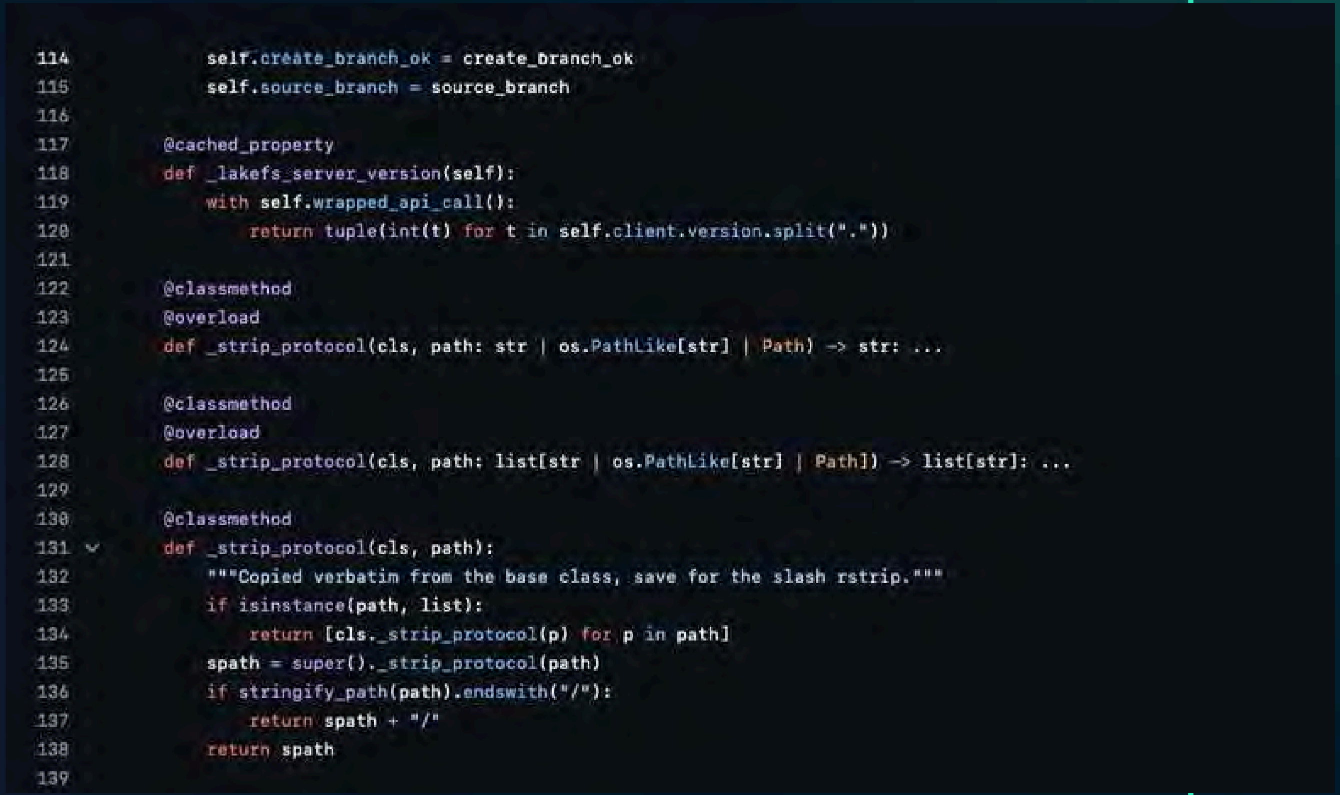
Python

```python
import boto3
s3 = boto3.client('s3')
s3.download_file('bucket', 'file.csv', 'file.csv')
```

BRIDGING THE GAP: LIBRARY-SPECIFIC CONNECTORS

Introducing

fsspec



```
        self.create_branch_ok = create_branch_ok
        self.source_branch = source_branch

    @cached_property
    def _lakefs_server_version(self):
        with self.wrapped_api_call():
            return tuple(int(t) for t in self.client.version.split("."))

    @classmethod
    @overload
    def _strip_protocol(cls, path: str | os.PathLike[str] | Path) -> str: ...

    @classmethod
    @overload
    def _strip_protocol(cls, path: list[str | os.PathLike[str] | Path]) -> list[str]: ...

    @classmethod
    def _strip_protocol(cls, path):
        """Copied verbatim from the base class, save for the slash rstrip."""
        if isinstance(path, list):
            return [cls._strip_protocol(p) for p in path]
        spath = super()._strip_protocol(path)
        if stringify_path(path).endswith("/"):
            return spath + "/"
        return spath
```

✓ UNIFIED INTERFACE

✓ MULTIPLE BACKEND SUPPORT

✓ EASE OF INTEGRATION

✓ ENHANCED CAPABILITIES

fsspec

AUDIENCE

- End users
- Library implementers
- Backend implementations

USING MY CODE WITH S3 VIA fsspec

END USERS

```
import fsspec
# Open a file from S3
fs = fsspec.filesystem('s3', key='ACCESS_KEY', secret='SECRET_KEY')
with fs.open('s3://bucket/file.csv', 'r') as f:
    data = f.read()
```

```
with fsspec.open('s3://minio-bucket/ ... ') as file:
    content=file.read()
```

USING PANDAS WITH S3 VIA fsspec

LIBRARY DEVELOPERS

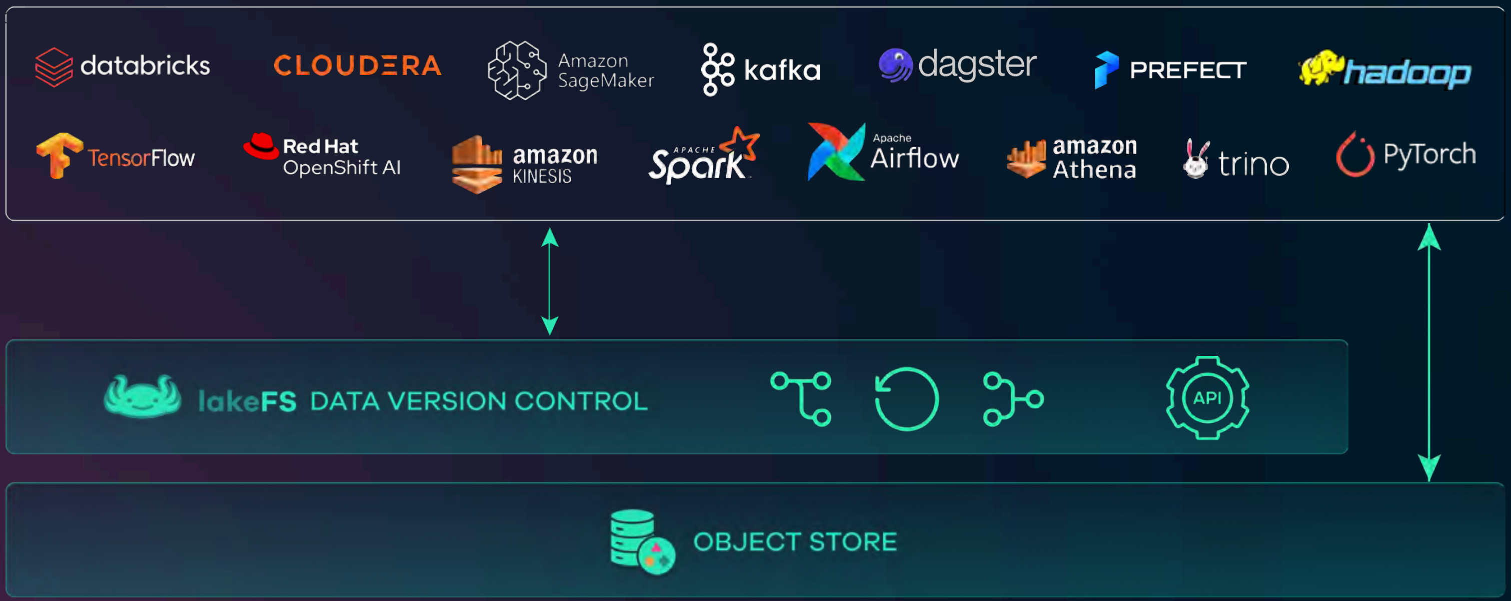
```
# Load a CSV from S3
df = pd.read_csv('s3://bucket/file.csv', storage_options={
'key': 'ACCESS_KEY',
'secret': 'SECRET_KEY'
})
```

BUILDING YOUR OWN fsspec BACKEND

Using a real world example with

lakeFS

MANAGE DATA LIKE CODE WITH lakeFS

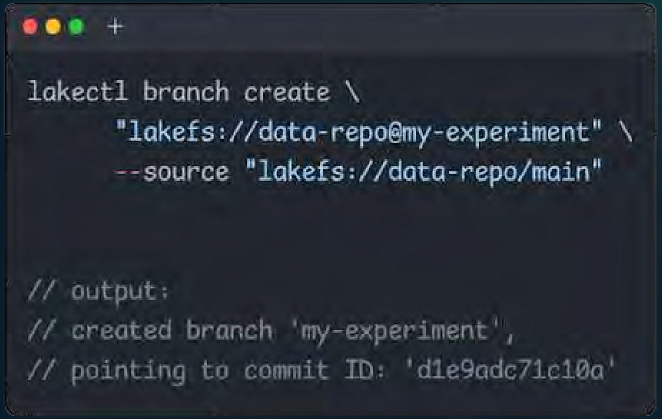


lakeFS DATA VERSION CONTROL

OBJECT STORE

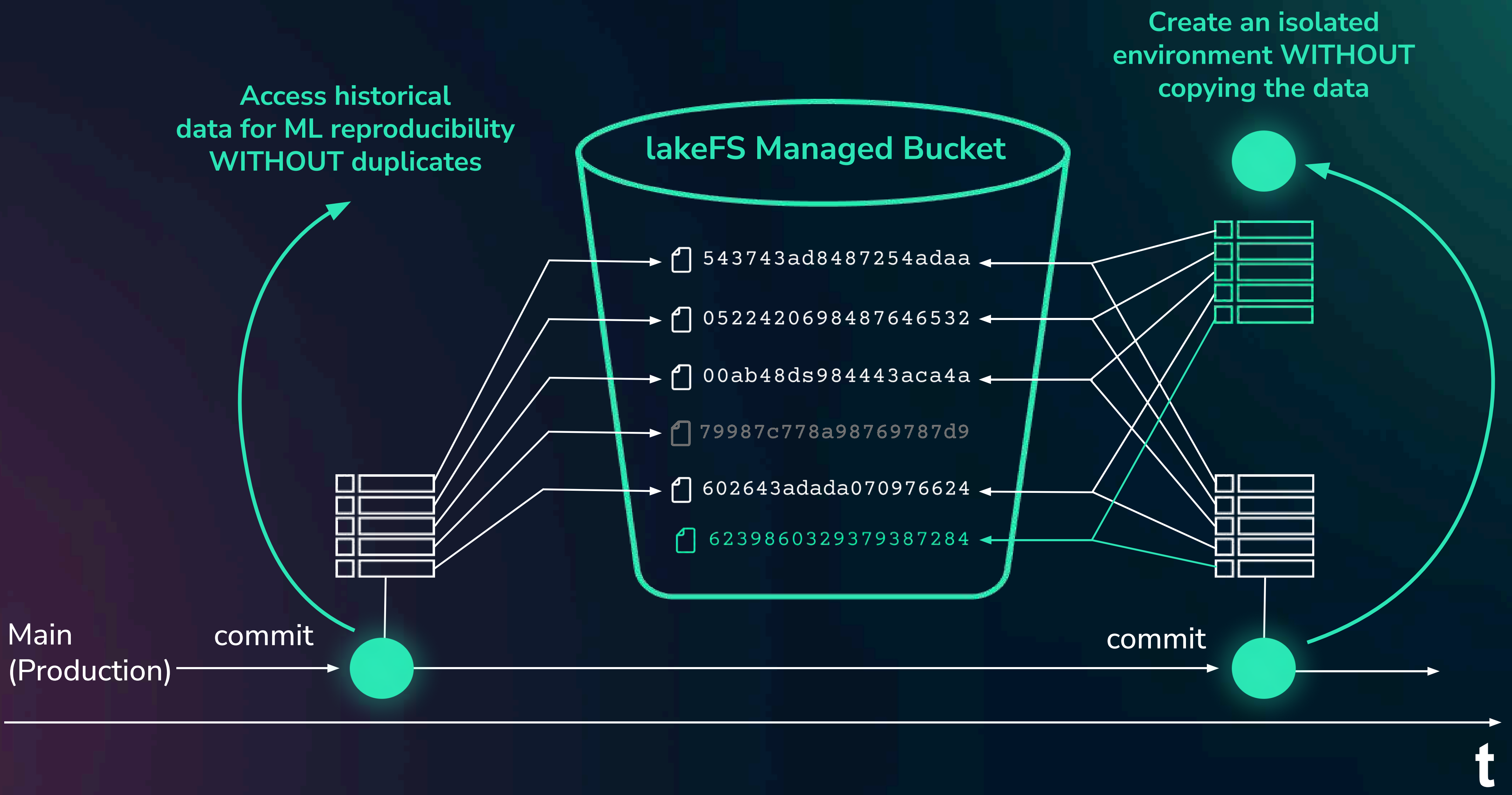
```
s3://data-repo/collections/foo
```

```
s3://data-repo/main/collections/foo
```



```
lakectl branch create \
      "lakefs://data-repo@my-experiment" \
      --source "lakefs://data-repo/main"


// output:
// created branch 'my-experiment',
// pointing to commit ID: 'd1e9adc71c10a'
```

HOW DOES lakeFS WORK?



Access historical
data for ML reproducibility
WITHOUT duplicates

Create an isolated
environment WITHOUT
copying the data

lakeFS Managed Bucket

543743ad8487254adaa

0522420698487646532

00ab48ds984443aca4a

79987c778a98769787d9

602643adada070976624

6239860329379387284

Main
(Production)

commit

commit

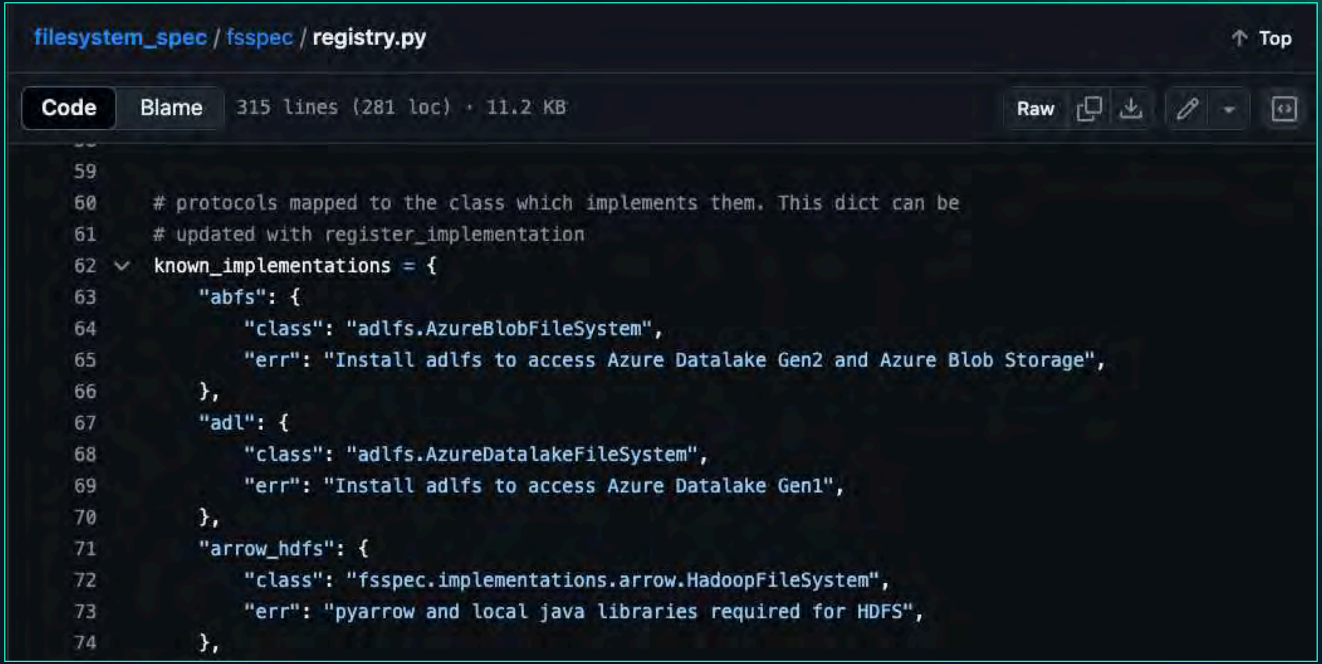
t

Implementing the fsspec package for lakeFS

AbstractFile

IMPLEMENTING THE fsspec PACKAGE FOR lakeFS

```
entry_points={
    'fsspec.spec': [
        'myfs = myfs:MyFS',
    ]
}
```


filesystem_spec / fsspec / registry.py
Top
Code
Blame
315 lines (281 loc) · 11.2 KB
Raw

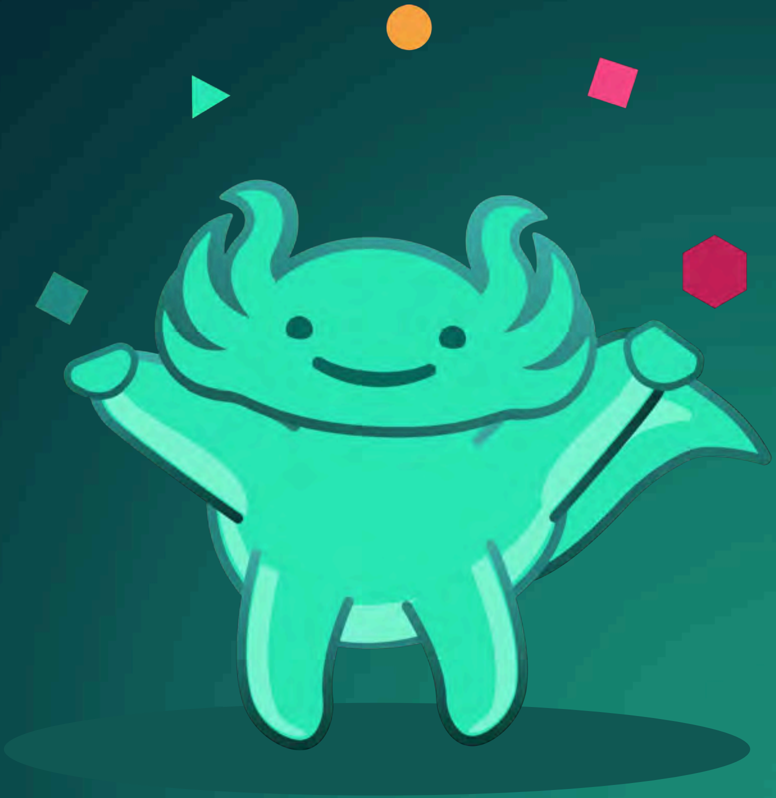
```

# protocols mapped to the class which implements them. This dict can be
# updated with register_implementation
known_implementations = {
    "abfs": {
        "class": "adlfs.AzureBlobFileSystem",
        "err": "Install adlfs to access Azure Datalake Gen2 and Azure Blob Storage",
    },
    "adl": {
        "class": "adlfs.AzureDatalakeFileSystem",
        "err": "Install adlfs to access Azure Datalake Gen1",
    },
    "arrow_hdfs": {
        "class": "fsspec.implementations.arrow.HadoopFileSystem",
        "err": "pyarrow and local java libraries required for HDFS",
    },
```

lakeFS-fsspec implementation

TRANSACTIONS

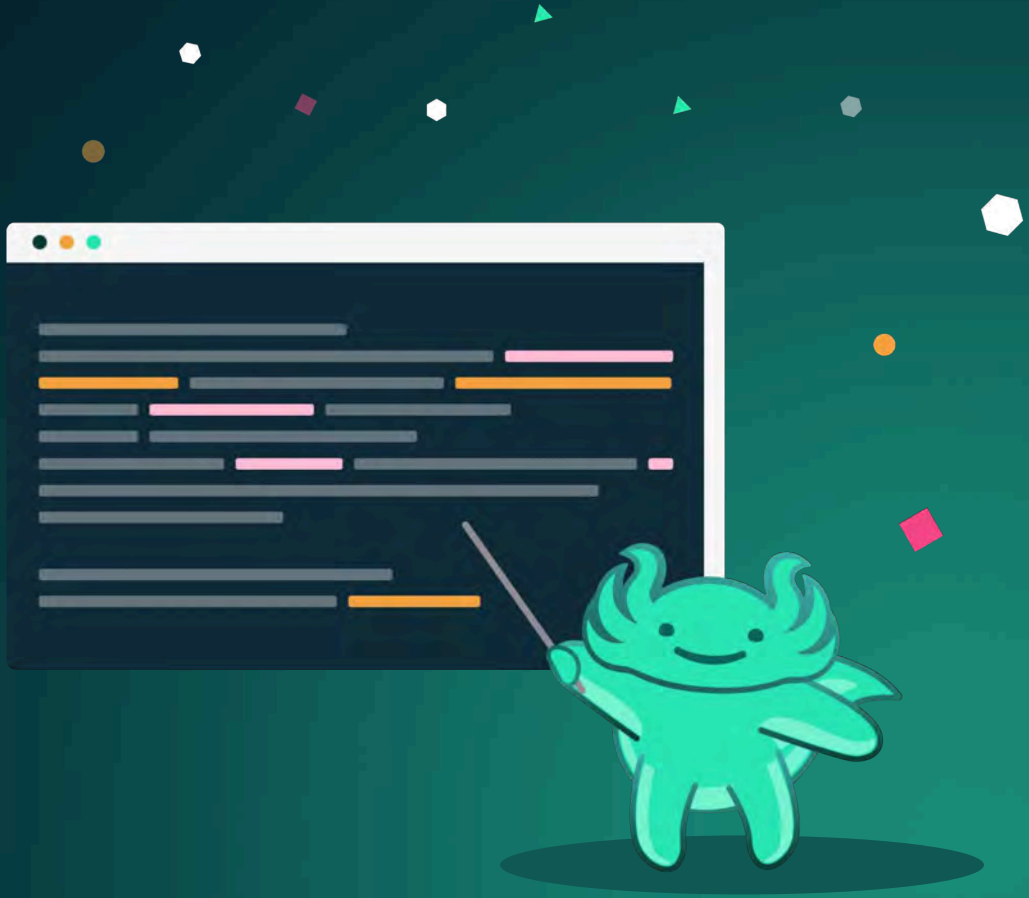
Backend implementation

lakeFS-fsspec



DEMO TIME

Using lakeFS-fsspec



Join the lakeFS Community

https://lakefs.io

https://lakefs.io/slack

Learn more about fsspec

https://github.com/aai-institute/lakefs-spec

https://lakefs.io/blog/lakefs-spec/

https://lakefs-spec.org/latest/quickstart/

Thank You!

Guy Hardonag, lakeFS by Treeverse

✉ guy.hardonag@treeverse.io

in www.linkedin.com/in/guy-hardonag-2ab10264

https://lakeFS.io