Jewel

File Syncing with AWS

Kevin Riehm

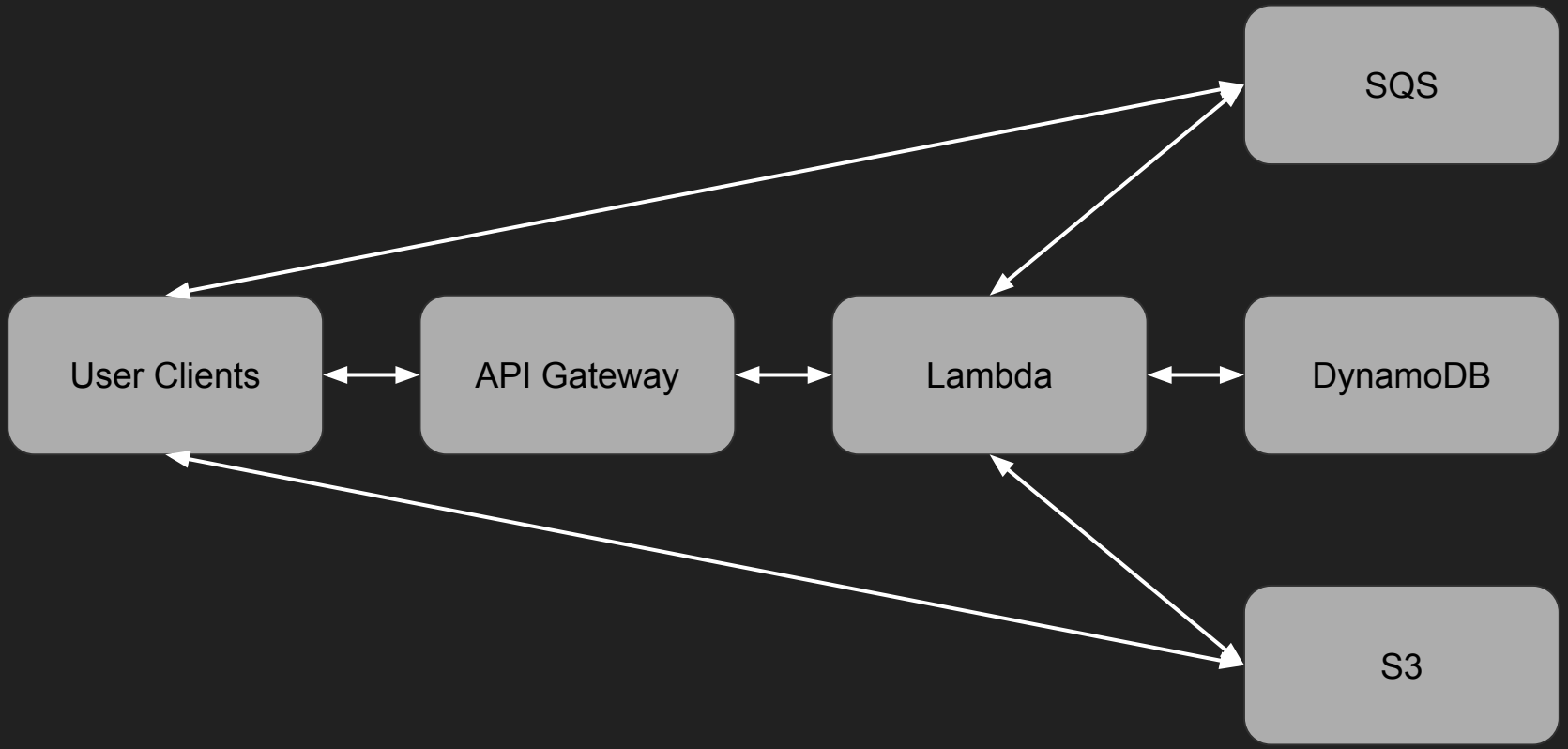
Project Goal

- Synchronize files across many machines
- Act transparently, with minimal user interaction
- Handle many users
- Handle many files per user

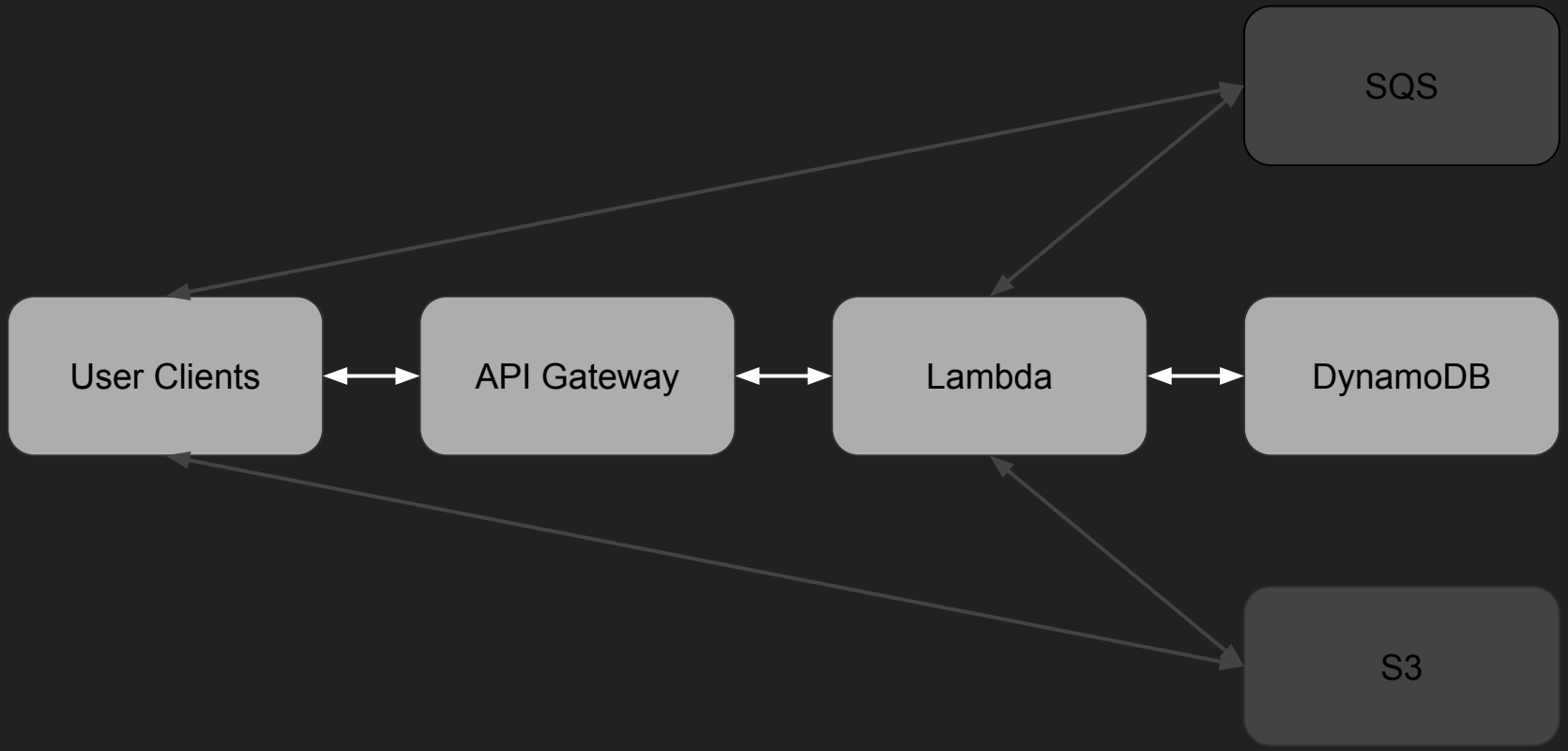
Project Solution

- Use AWS as central repository for all files
  - DynamoDB for directory tree/metadata
  - S3 for file contents
  - SQS for file change notifications
  - Lambda for execution logic
  - API Gateway for service access
- Local Python script monitors files, applies changes
- Simple web portal to access your files anywhere

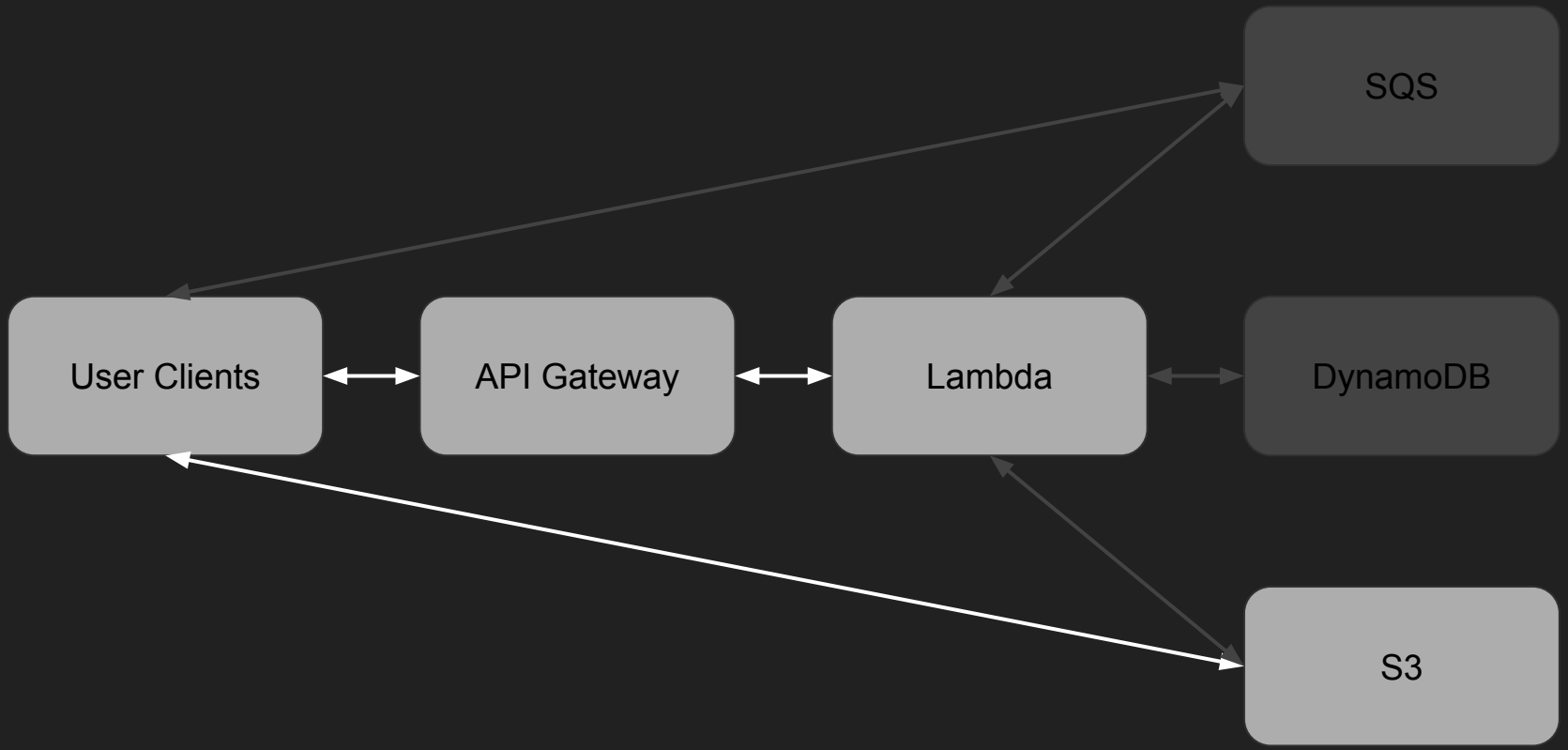
System Diagram



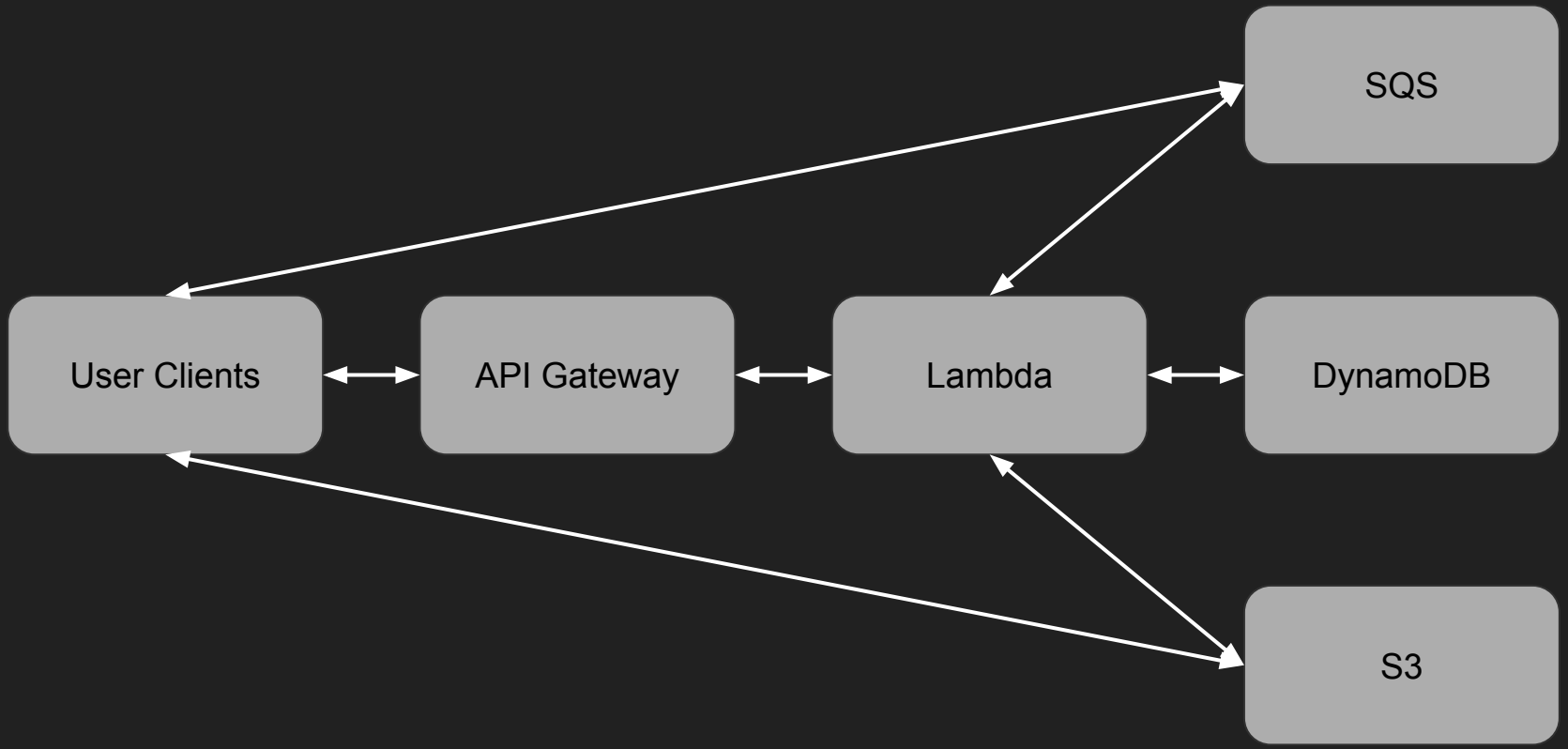
System Diagram - List Files



System Diagram - Download File



System Diagram - Upload File



Conflict Resolution

- Simple over correct
- When files conflict, rename one

Performance - Many Independent Users

- Run N Condor jobs, each with a different user
- Two local directories, both synchronized
- Create 100 files in one directory, wait until they appear in the other
- Delete 100 files in one directory, wait until they disappear from the other

Performance - Many Independent Users

# Users	1	10	100	1000
Creation Time (s)	95.637	87.627	108.844	671.856
Deletion Time (s)	31.723	42.123	48.146	581.248
Request Time (s)	0.033	0.037	0.051	0.749

Performance - Many Dependent Users

- Run N Condor jobs, each with the same user
- Two local directories, both synchronized
- Create 100 files in one subdirectory, wait until they appear in the other
- Delete 100 files in one subdirectory, wait until they disappear from the other

Performance - Many Dependent Users

# Users	1	10	100	1000
Creation Time (s)	86.234	168.762	...	...
Deletion Time (s)	22.627	123.783	...	...
Request Time (s)	0.039	0.033	...	...

Cost

- 10 GB storage/10000 files, 2 machines always syncing
  - $0.42 for S3 storage and data transfer
  - $0.22 for SQS requests and data transfer
  - $0.59 for DynamoDB throughput
  - $0.04 for Lambda execution time and data transfer
  - $0.18 for API Gateway requests
  - $1.45 total/month