**Implementation and Testing Plan**

**Phased Approach to Implementation**

**Phase 1: Core Backend Infrastructure**

**Description:**

* Build the modular structure for the project.
* Set up default storage backend as DuckDB.
* Configure environment for admin settings like:
  + API key/password
  + Logging level
  + Data source (default: PMC)
  + Storage backend

**Commands:**

poetry init

poetry add fastapi uvicorn duckdb httpx python-dotenv

mkdir pmc bern2 storage api

touch config.py main.py

**Phase 2: API Layer Development**

**Description:**

* Build RESTful API endpoints:
  + Upload PMCIDs
  + Query stored caption data
  + Download data (CSV or JSON)
* Secure endpoints using API key/password.

**Commands:**

poetry add python-multipart

poetry run uvicorn main:app --reload

To test endpoints:

curl -X POST http://localhost:8000/upload\_pmcs -H "x-api-key: <KEY>" -d '{"pmcids": ["PMC123"]}'

curl http://localhost:8000/query -H "x-api-key: <KEY>"

curl http://localhost:8000/download?format=csv -H "x-api-key: <KEY>" -o results.csv

**Phase 3: BERN2 Integration**

**Description:**

* Integrate locally hosted BERN2 instance.
* Extract entities from figure captions.

**Commands:**

docker pull dorajistyle/bern2

docker run -d -p 8888:8888 dorajistyle/bern2

To verify BERN2 is running:

curl -X POST http://localhost:8888/pubmed -d "text=Figure shows BRCA1 mutation"

**Phase 4: Data Export and Logging**

**Description:**

* Enable structured logging.
* Add endpoints to export caption and entity data.
* Support both CSV and JSON formats.

**Commands:**

curl http://localhost:8000/download?format=csv -H "x-api-key: <KEY>" -o output.csv

curl http://localhost:8000/download?format=json -H "x-api-key: <KEY>" -o output.json

**Phase 5: Extensibility and Plugin Support**

**Description:**

* Add abstract interfaces for:
  + Data sources (PMC, future APIs)
  + Storage backends (DuckDB, future: PostgreSQL)
* Allow easy registration of new components.

**Testing Plan**

**1. Functionality Testing**

**Description:**

* Ensure the system works end-to-end using both mocked and real data.
* Validate:
  + Uploading PMCIDs
  + Caption extraction
  + Entity extraction
  + Data retrieval and export

**Commands:**

# Run API

poetry run uvicorn main:app --reload

# Test upload with mock or real PMCIDs

curl -X POST http://localhost:8000/upload\_pmcs -H "x-api-key: <KEY>" -d '{"pmcids": ["PMC1234567"]}'

# Test query

curl http://localhost:8000/query -H "x-api-key: <KEY>"

# Test download

curl http://localhost:8000/download?format=csv -H "x-api-key: <KEY>" -o data.csv

**2. Security Testing (API Key)**

**Description:**

* Ensure all endpoints require a valid API key or password.
* Test for unauthorized access and proper error handling.

**Commands:**

# Test without API key

curl http://localhost:8000/query

# Expected: 401 Unauthorized

# Test with wrong key

curl http://localhost:8000/query -H "x-api-key: wrongkey"

# Expected: 401 Unauthorized

# Test with correct key

curl http://localhost:8000/query -H "x-api-key: <correct\_key>"

# Expected: 200 OK

**3. Performance Testing**

**Description:**

* Measure response times and batch handling.
* Identify performance bottlenecks when processing large PMCID batches.
* Evaluate memory and CPU usage during:
  + Upload
  + Extraction
  + Query and download

**Commands:**

# Test batch upload

curl -X POST http://localhost:8000/upload\_pmcs -H "x-api-key: <KEY>" -d '{"pmcids": ["PMC1", "PMC2", ..., "PMC100"]}'

# Monitor memory and CPU

top

htop

Optional benchmarking:

# Install and run simple benchmarks

poetry add --dev pytest pytest-asyncio

pytest tests/ --durations=10

For memory profiling:

poetry add --dev memory-profiler

mprof run main.py

mprof plot

**Mocked vs Real Data Testing**

**Mocked Data**

* Use static PMC XML/HTML files for unit tests.
* Simulate BERN2 responses to isolate caption pipeline.

**Real Data**

* Use PMCIDs from PubMed Central.
* Validate figure captions and entity output against expected results.
* Confirm correct parsing, storing, and exporting.

**Conclusion**

This phased plan outlines the step-by-step implementation of your PMC caption extraction and BERN2-enhanced system, paired with robust functionality, security, and performance testing. All relevant commands for setup, running, testing, and monitoring are included for smooth development and deployment.