al1nasir

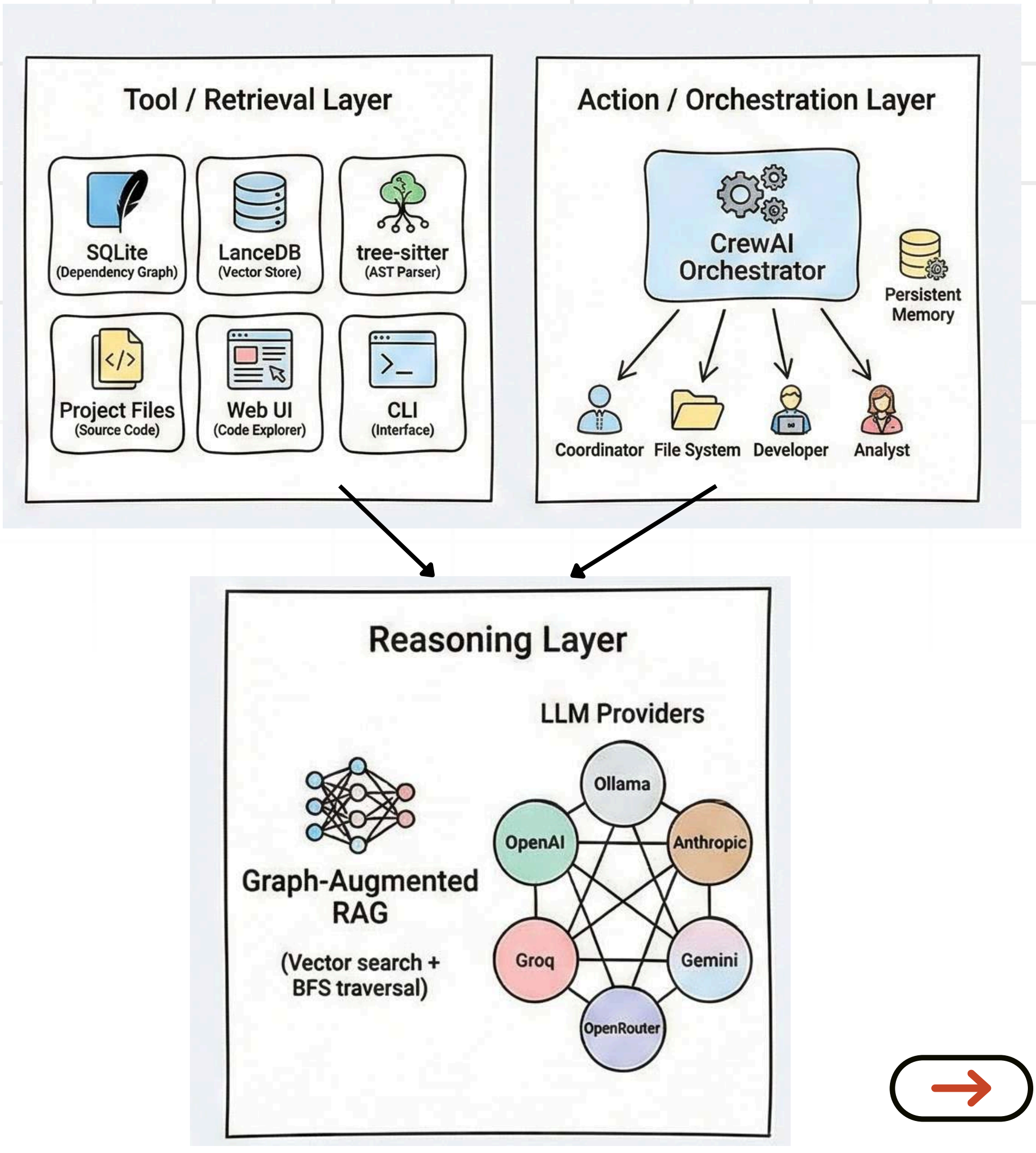
CodeGraph-CLI (local ai assistant)

TALK TO YOUR CODEBASE From the Terminal

Open-source · Local first approach

HOW IT WORKS

Under the Hood: From Code → AI



2/7

CLI-overview

All commands at your fingertips

```
🧠 CodeGraph CLI — AI-powered code intelligence & multi-agent assistant.

╭─ Options ──────────────────────────────────────────────────────────────────────╮
│ --version  -v        Show version and exit.                                    │
│ --help               Show this message and exit.                               │
╰────────────────────────────────────────────────────────────────────────────────╯
╭─ Commands ─────────────────────────────────────────────────────────────────────╮
│ onboard  🚀 Auto-generate a project README from the code graph.                │
│ config   Configuration management                                             │
│ project  Project management                                                    │
│ analyze  Code analysis                                                         │
│ chat     Interactive chat with AI agents                                       │
│ explore  Visual code explorer in browser                                       │
│ export   Export project documentation                                          │
╰────────────────────────────────────────────────────────────────────────────────╯
```

```
(PyhelpCLI) ali-nasir@al1nasir:~/PycharmProjects/PyhelpCLI$ cg config --help

Usage: cg config [OPTIONS] COMMAND [ARGS]...

Configuration management

╭─ Options ──────────────────────────────────────────────────────────────────────╮
│ --help          Show this message and exit.                                    │
╰────────────────────────────────────────────────────────────────────────────────╯
╭─ Commands ─────────────────────────────────────────────────────────────────────╮
│ setup            Interactive setup wizard for LLM provider configuration.      │
│ set-llm          Quickly switch LLM provider without full setup wizard.        │
│ unset-llm        Reset LLM configuration to defaults (removes API keys and provider settings).
│ show-llm         Show current LLM provider configuration.                      │
│ set-embedding    Set the embedding model for semantic code search.             │
│ unset-embedding  Reset embedding model to default (hash — no download).        │
│ show-embedding   Show current embedding model configuration.                   │
╰────────────────────────────────────────────────────────────────────────────────╯
```

setup llm and embedding models

Semantic Search

Ever Joined a Codebase With 200K+ Lines?

Traditional grep	CodeGraph CLI
grep "auth" • 47 raw string matches • Searches for text only • No understanding of context • You read every file manually • You connect the dots yourself	cg analyze search "how does auth work" • 5 semantically relevant results • Understands code meaning • Returns full dependency chain • AI explains the relationships • Shows what calls what & why

```
~/PycharmProjects/PyhelpCLI$ cg analyze rag-context "kindly explain me the fft function"
[module] fft_processor
file: fft_processor.py:1
score: 0.623
```python
\"""FFT processing module for frequency domain filtering.

This module provides functions for generating filters, processing FFTs, and applying
frequency domain filtering using proper fftshift/ifftshift for zero-frequency centering.

Functions:
    generate_centering_mask: Generate manual centering mask (-1)^(x+y)
    create_gaussian_filter: Create Gaussian low-pass or high-pass filter
    create_butterworth_filter: Create Butterworth low-pass or high-pass filter
    create_ideal_filter: Create ideal (brick-wall) low-pass or high-pass filter
    process_fft: Complete workflow for FFT filtering
"""

# Type aliases for clarity
FilterType = Literal["Low Pass", "High Pass"]
FilterKind = Literal["gaussian", "butterworth", "ideal"]
FFTResult = Dict[str, np.ndarray]

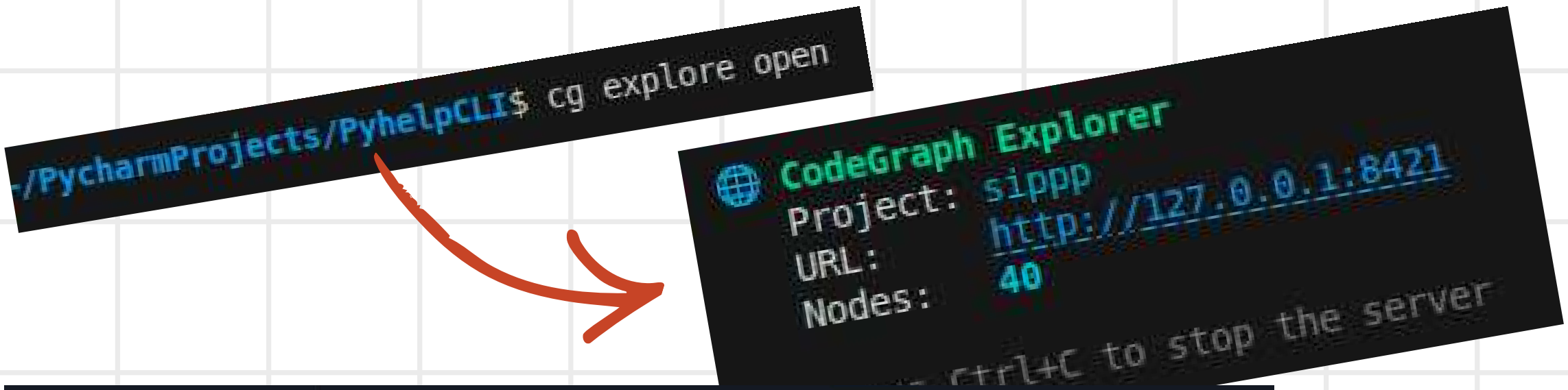
def generate_centering_mask(M: int, N: int) -> np.ndarray:
    """Generate a centering mask using (-1)^(x+y).

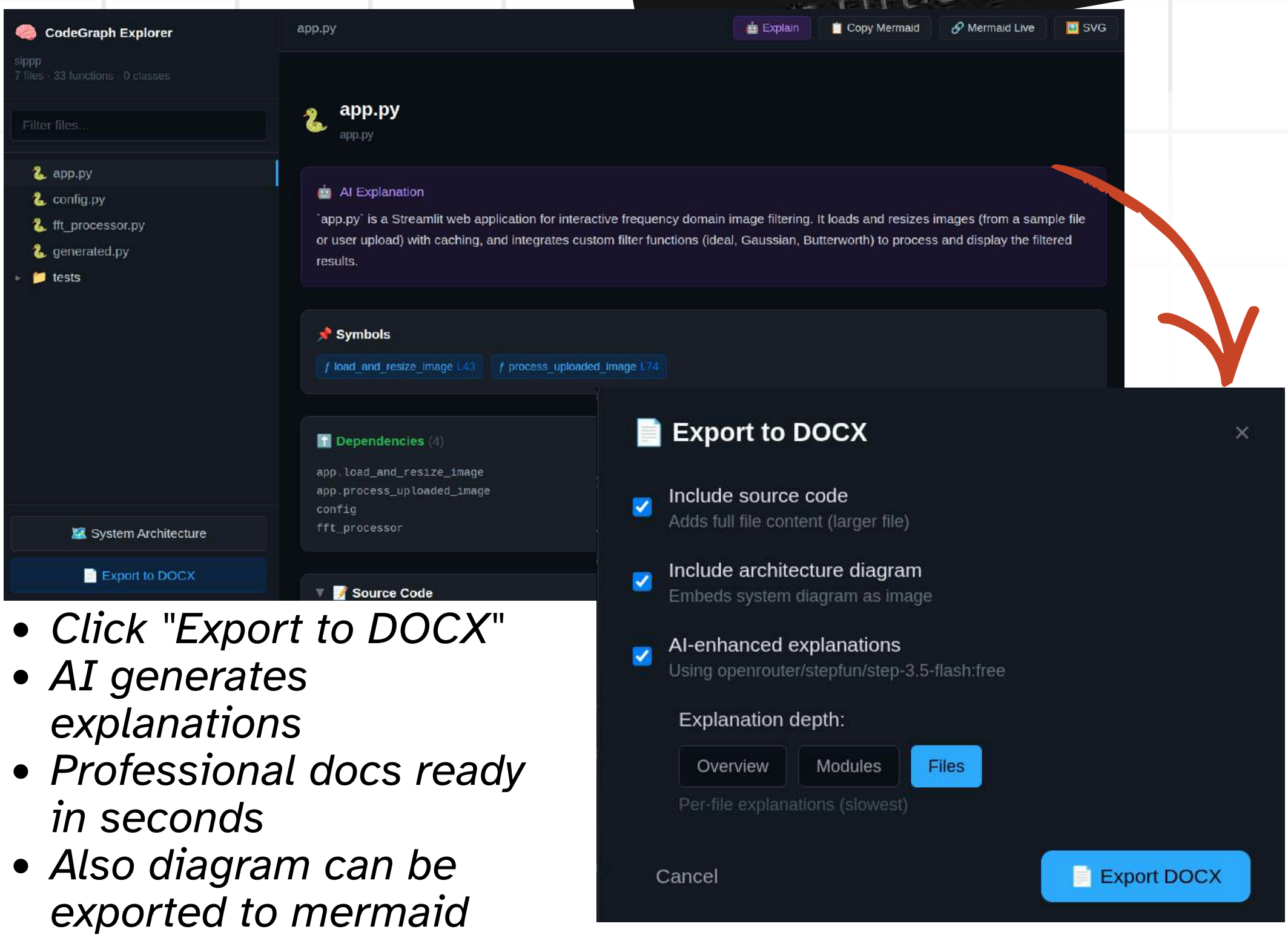
    This function creates the manual centering mask that can be used to shift
    the zero-frequency component to the
# ... (truncated)
```
```

4/7

cg-explore

Visual Code Explorer Navigate, Understand, Document - All in Your Browser





- *Click "Export to DOCX"*
- *AI generates explanations*
- *Professional docs ready in seconds*
- *Also diagram can be exported to mermaid*

5/7

cg chat start

Talk to Your Codebase Ask Questions in Plain English , Get Intelligent Answers

```
(PyhelpCLI) ali-nasir@alinasir:~/PycharmProjects/PyhelpCLI$ cg chat start
  📂  Resumed session (8 messages)

  ⚡  CodeGraph Chat

  Project   sippp
  LLM       openrouter/stepfun/step-3.5-flash:free
  Indexed   7 files • 33 functions • 0 classes
  Recent    ONBOARD.md, README.md, app.py
  Type /help for commands, /exit to quit

  ● You › explain me this project

  ● Assistant
  This is a **Frequency Domain Image Filtering** project built with Streamlit
ere's what it does:

  ## **Core Concept**
  The project applies filters to images in the **frequency domain** rather th
chnique where:
  1. An image is transformed from spatial domain to frequency domain using FF
  2. A filter mask is applied to modify specific frequencies
  3. The result is transformed back to spatial domain using inverse FFT

  ## **Project Structure**
```

Works With Your Favorite LLM

Ollama · OpenAI · Anthropic . Groq · Gemini · OpenRouter

5 Embedding Models: *hash (0 bytes) → qodo-1.5b (6.2 GB)*

Zero env vars. *One TOML file.*

6/7

cg impact (Impact Analysis)

Know What Breaks BEFORE You Change It

- *Multi-hop BFS on the dependency graph*
- *LLM explains what would break and why*

```
ice' to find similar symbols
~/PycharmProjects/PyhelpCLI$ cg analyze impact app.process_uploaded_image  --hops 3
e
Root: app.process_uploaded_image
Impacted symbols: none found

ASCII graph:
app.process_uploaded_image
  |- calls -> st.error
  |- calls -> cv2.imdecode
  |- calls -> np.frombuffer
  |- calls -> st.error
  |- calls -> cv2.resize
  |- calls -> st.error
  |- calls -> str

Explanation:
Based on the provided dependency information for `app.process_uploaded_image`, here's the downstream impact analysis:

## 1) Main Risks

**Signature/Return Type Changes:**
- If you modify what the function returns (e.g., change from returning an image array to returning a tuple, or change data type/s
 caller expecting the original format will break immediately**.
- Since no direct callers were detected, this risk is lower but still exists if the function is called from outside the analyzed
 from Streamlit UI code, notebooks, or other modules not in the dependency graph).

**Behavioral Changes:**
- **Error handling modifications**: The function calls `st.error()` multiple times. Removing/changing these could:
  - Hide critical failure modes from users
  - Cause unhandled exceptions if error conditions aren't properly managed
  - Break UI flow expectations (Streamlit apps often rely on error messages for user guidance)
main* ⊗0 ⚠4
```

7/7

cg onboard

create auto-readme with rag-context and llm support

```
Usage: cg onboard [OPTIONS]

🚀 Auto-generate a project README from the code graph.

Analyzes the indexed project's structure, dependencies, entry points,
and docstrings to produce a comprehensive README.md — either via LLM
or from a pure template.

Examples:
  cg onboard                    # print to stdout
  cg onboard --save             # save as ONBOARD.md in project dir
  cg onboard -o README.md       # save to specific file
  cg onboard --no-llm           # template only, no LLM call
```

> ***and many more!***

7/7

Its not ends here

Star It. Fork It.
Break It. Improve It.

GITHUB

github.com/al1-nasir/codegraph-cli

PyPI

pip install codegraph-cli

(see readme for different versions info)

LICENSE

MIT — use it however you want

What I'd love feedback on:

Is graph-augmented RAG useful for code search?

LanceDB vs FAISS/Chroma — thoughts?

What languages should come next?
(Go, Rust, Java, TypeScript?)

Multi-agent vs single agent — which do you prefer?