Smart Search - Hướng dẫn sử dụng

Mục lục

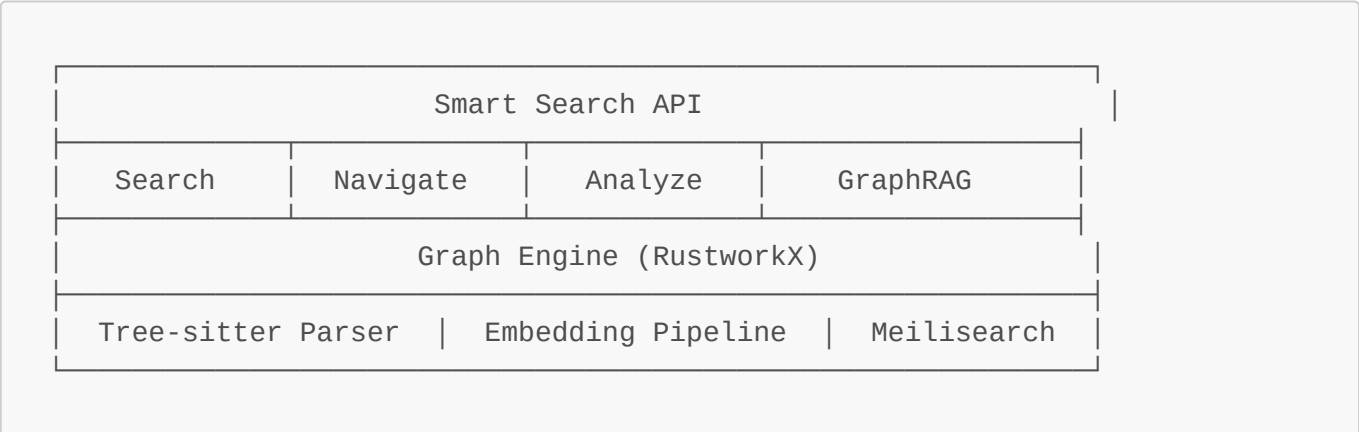
---

1. Giới thiệu

Smart Search là hệ thống tìm kiếm và điều hướng mã nguồn thông minh, kết hợp:

- **Hybrid Search**: Tìm kiếm kết hợp keyword và semantic
- **Code Graph**: Phân tích quan hệ giữa các thành phần code
- **GraphRAG**: Hỏi đáp thông minh về codebase bằng AI

Kiến trúc hệ thống



---

2. Cài đặt và Khởi động

2.1 Yêu cầu hệ thống

- Python 3.12+
- Meilisearch (optional, cho full-text search)
- Jina AI API key (optional, cho semantic search)

2.2 Khởi động server

```
cd /home/fong/Projects/smart_search
source .venv/bin/activate
PYTHONPATH=src python -m smart_search.main
```

Server sẽ chạy tại: `http://localhost:8000`

2.3 Kiểm tra trạng thái

```
curl http://localhost:8000/health
```

Response:

```
{
  "status": "healthy",
  "version": "0.1.0",
  "services": {
    "graph": "ok",
    "search": "ok"
  }
}
```

2.4 API Documentation

Mở browser tại: `http://localhost:8000/docs`

---

3. Index Codebase

Trước khi tìm kiếm, bạn cần index codebase của mình.

3.1 Index một thư mục

```
curl -X POST http://localhost:8000/api/v1/index \
  -H "Content-Type: application/json" \
  -d '{
    "paths": ["/path/to/your/project"],
    "recursive": true,
    "languages": ["python"]
  }'
```

Response:

```
{
  "job_id": "idx_abc123",
```

```
  "status": "started",
  "message": "Indexing started for 1 paths"
}
```

3.2 Kiểm tra tiến độ index

```
curl http://localhost:8000/api/v1/index/progress/idx_abc123
```

Response:

```
{
  "job_id": "idx_abc123",
  "status": "completed",
  "progress": 100,
  "files_processed": 150,
  "files_total": 150,
  "errors": []
}
```

3.3 Index với exclude patterns

```
curl -X POST http://localhost:8000/api/v1/index \
  -H "Content-Type: application/json" \
  -d '{
    "paths": ["/path/to/project"],
    "recursive": true,
    "languages": ["python", "javascript"],
    "exclude_patterns": [
      "**/node_modules/**",
      "**/__pycache__/**",
      "**/test/**",
      "**/.git/**"
    ]
  }'
```

3.4 Cập nhật incremental

Chỉ index các file đã thay đổi:

```
curl -X POST http://localhost:8000/api/v1/index/update \
  -H "Content-Type: application/json" \
  -d '{
    "paths": ["/path/to/project"],
```

```
    "mode": "incremental"
  }'
```

3.5 Xem thống kê index

```
curl http://localhost:8000/api/v1/index/stats
```

Response:

```
{
  "total_files": 150,
  "total_units": 1250,
  "languages": {
    "python": 120,
    "javascript": 30
  },
  "unit_types": {
    "function": 800,
    "class": 200,
    "method": 250
  },
  "last_indexed": "2025-12-14T10:30:00Z"
}
```

---

4. Tìm kiếm Code

4.1 Tìm kiếm cơ bản (Hybrid Search)

```
curl -X POST http://localhost:8000/api/v1/search \
  -H "Content-Type: application/json" \
  -d '{
    "query": "user authentication login",
    "limit": 10
  }'
```

Response:

```
{
  "query": "user authentication login",
  "hits": [
    {
      "id": "auth.py::login_user",
      "name": "login_user",
```

```
      "qualified_name": "auth.login_user",
      "code_type": "function",
      "file_path": "/project/auth.py",
      "line_start": 45,
      "line_end": 60,
      "score": 0.95,
      "highlights": ["def login_user(username, password):"]
    }
  ],
  "total": 15,
  "search_type": "hybrid"
}
```

4.2 Tìm kiếm theo loại

```
# Chỉ tìm keyword
curl -X POST http://localhost:8000/api/v1/search \
  -H "Content-Type: application/json" \
  -d '{
    "query": "parse json",
    "search_type": "keyword",
    "limit": 10
  }'

# Chỉ tìm semantic
curl -X POST http://localhost:8000/api/v1/search \
  -H "Content-Type: application/json" \
  -d '{
    "query": "function that validates user input",
    "search_type": "semantic",
    "limit": 10
  }'
```

4.3 Tìm kiếm với filters

```
curl -X POST http://localhost:8000/api/v1/search \
  -H "Content-Type: application/json" \
  -d '{
    "query": "database connection",
    "limit": 10,
    "filters": {
      "language": "python",
      "code_type": "function",
      "file_pattern": "**/models/**"
    }
  }'
```

4.4 Tìm code tương tự

Tìm các đoạn code tương tự với một đoạn code cho trước:

```
curl -X POST http://localhost:8000/api/v1/search/similar \
  -H "Content-Type: application/json" \
  -d '{
    "code": "def connect_database(host, port, user, password):\n    return
create_connection(host, port, user, password)",
    "limit": 5
  }'
```

4.5 Gợi ý tìm kiếm

```
curl -X POST http://localhost:8000/api/v1/search/suggest \
  -H "Content-Type: application/json" \
  -d '{
    "prefix": "user_",
    "limit": 10
  }'
```

Response:

```
{
  "suggestions": [
    "user_login",
    "user_logout",
    "user_register",
    "user_profile",
    "user_settings"
  ]
}
```

---

5. Code Navigation

5.1 Go to Definition

Tìm định nghĩa của một symbol tại vị trí cụ thể:

```
curl -X POST http://localhost:8000/api/v1/navigate/definition \
  -H "Content-Type: application/json" \
  -d '{
    "file_path": "/project/main.py",
    "line": 25,
    "column": 10
  }'
```

Response:


```
{
  "found": true,
  "definition": {
    "id": "utils.py::helper_function",
    "name": "helper_function",
    "file_path": "/project/utils.py",
    "line_start": 15,
    "line_end": 25,
    "code_type": "function"
  }
}
```


5.2 Find References

Tìm tất cả nơi sử dụng một function/class:

```
curl http://localhost:8000/api/v1/navigate/references/auth.py::login_user?
limit=50
```

Response:


```
{
  "code_id": "auth.py::login_user",
  "references": [
    {
      "file_path": "/project/main.py",
      "line": 45,
      "column": 12,
      "context": "result = login_user(username, password)"
    },
    {
      "file_path": "/project/api/routes.py",
      "line": 78,
      "column": 8,
      "context": "user = login_user(data.username, data.password)"
    }
  ],
  "total": 2
}
```


5.3 Find Callers

Tìm các function gọi đến một function:

```
curl
http://localhost:8000/api/v1/navigate/callers/utils.py::validate_input?
depth=2
```

Response:


```
{
  "code_id": "utils.py::validate_input",
  "callers": [
    {
      "id": "api.py::create_user",
      "name": "create_user",
      "file_path": "/project/api.py",
      "depth": 1
    },
    {
      "id": "main.py::process_form",
      "name": "process_form",
      "file_path": "/project/main.py",
      "depth": 2
    }
  ]
}
```


5.4 Find Callees

Tìm các function được gọi bởi một function:

```
curl http://localhost:8000/api/v1/navigate/callees/main.py::main?depth=2
```

5.5 Class Hierarchy

Xem cây kế thừa của một class:

```
curl http://localhost:8000/api/v1/navigate/hierarchy/models.py::User?
max_depth=3
```

Response:


```
{
  "code_id": "models.py::User",
  "parents": [
    {
      "id": "models.py::BaseModel",
```

```
      "name": "BaseModel"
    }
  ],
  "children": [
    {
      "id": "models.py::AdminUser",
      "name": "AdminUser"
    },
    {
      "id": "models.py::GuestUser",
      "name": "GuestUser"
    }
  ]
}
```

5.6 File Outline

Xem cấu trúc của một file:

```
curl "http://localhost:8000/api/v1/navigate/outline?
file_path=/project/auth.py"
```

Response:

```
{
  "file_path": "/project/auth.py",
  "outline": [
    {
      "name": "User",
      "type": "class",
      "line_start": 10,
      "line_end": 45,
      "children": [
        {
          "name": "__init__",
          "type": "method",
          "line_start": 12,
          "line_end": 18
        },
        {
          "name": "validate",
          "type": "method",
          "line_start": 20,
          "line_end": 30
        }
      ]
    },
    {
      "name": "login_user",
```

```
      "type": "function",
      "line_start": 50,
      "line_end": 65
    }
  ]
}
```

5.7 Symbol Search

Tìm kiếm symbols theo tên:

```
curl "http://localhost:8000/api/v1/navigate/symbols?
query=User&type=class&limit=10"
```

---

6. Phân tích Code

6.1 Explain Code

Giải thích một đoạn code bằng AI:

```
curl -X POST http://localhost:8000/api/v1/analyze/explain \
  -H "Content-Type: application/json" \
  -d '{
    "code_id": "auth.py::login_user",
    "include_context": true
  }'
```

Response:

```
{
  "code_id": "auth.py::login_user",
  "explanation": "Hàm login_user thực hiện xác thực người dùng...",
  "summary": "Xác thực người dùng với username và password",
  "complexity": "medium",
  "related_functions": ["hash_password", "verify_token"]
}
```

6.2 Impact Analysis

Phân tích ảnh hưởng khi thay đổi một function:

```
curl -X POST http://localhost:8000/api/v1/analyze/impact \
  -H "Content-Type: application/json" \
  -d '{
```

```
    "code_id": "utils.py::format_date",
    "max_depth": 3,
    "include_indirect": true
  }'
```

Response:

```
{
  "code_id": "utils.py::format_date",
  "impact": {
    "direct_dependents": 5,
    "indirect_dependents": 12,
    "affected_files": [
      "/project/api/routes.py",
      "/project/views/dashboard.py",
      "/project/reports/generator.py"
    ],
    "risk_level": "medium",
    "recommendations": [
      "Cần update 5 functions gọi trực tiếp",
      "Kiểm tra test coverage cho các file bị ảnh hưởng"
    ]
  }
}
```

6.3 Code Metrics

Xem metrics của một code unit:

```
curl http://localhost:8000/api/v1/analyze/metrics/auth.py::User
```

Response:

```
{
  "code_id": "auth.py::User",
  "metrics": {
    "lines_of_code": 35,
    "cyclomatic_complexity": 8,
    "cognitive_complexity": 12,
    "method_count": 5,
    "dependencies": 3,
    "dependents": 8
  }
}
```

6.4 Dependency Analysis

Phân tích dependencies của một module:

```
curl http://localhost:8000/api/v1/analyze/dependencies/auth.py?depth=2
```

Response:

```
{
  "file_path": "auth.py",
  "imports": [
    {
      "module": "hashlib",
      "type": "stdlib"
    },
    {
      "module": "utils",
      "type": "internal",
      "functions_used": ["validate_input", "format_error"]
    }
  ],
  "imported_by": [
    "main.py",
    "api/routes.py"
  ]
}
```

6.5 Find Duplicates

Tìm code trùng lặp:

```
curl
http://localhost:8000/api/v1/analyze/duplicates/utils.py::process_data?
threshold=0.8
```

Response:

```
{
  "code_id": "utils.py::process_data",
  "duplicates": [
    {
      "id": "helpers.py::handle_data",
      "similarity": 0.92,
      "file_path": "/project/helpers.py",
      "line_start": 45
    }
  ]
}
```

6.6 Summarize Module

Tóm tắt một module/file:

```
curl -X POST http://localhost:8000/api/v1/analyze/summarize \
  -H "Content-Type: application/json" \
  -d '{
    "file_path": "/project/auth.py"
  }'
```

---

7. GraphRAG - Hỏi đáp thông minh

7.1 Hỏi về codebase

```
curl -X POST http://localhost:8000/api/v1/search/rag \
  -H "Content-Type: application/json" \
  -d '{
    "query": "How does user authentication work in this project?",
    "mode": "local"
  }'
```

Response:


```
{
  "query": "How does user authentication work in this project?",
  "answer": "User authentication trong project này hoạt động như sau:\n\n1.
**Login Flow**:\n   - User gửi credentials đến `login_user()` trong
`auth.py`\n   - Password được verify bằng `hash_password()`\n   - Nếu thành
công, JWT token được tạo bởi `create_token()`\n\n2. **Token Validation**:\n
- Mỗi request được validate bởi middleware `verify_token()`\n   - Token
expired sẽ trả về 401 Unauthorized\n\n3. **Related Functions**:\n   -
`auth.py::login_user` - Entry point\n   - `auth.py::verify_password` -
Password verification\n   - `utils.py::create_jwt_token` - Token
generation",
  "sources": [
    {
      "id": "auth.py::login_user",
      "relevance": 0.95
    },
    {
      "id": "auth.py::verify_password",
      "relevance": 0.88
    }
  ],
```

```
  "mode": "local"
}
```

7.2 Các mode của GraphRAG

```
# Local mode - Focus vào chi tiết cụ thể
curl -X POST http://localhost:8000/api/v1/search/rag \
  -H "Content-Type: application/json" \
  -d '{
    "query": "What parameters does login_user accept?",
    "mode": "local"
  }'

# Global mode - Nhìn tổng quan architecture
curl -X POST http://localhost:8000/api/v1/search/rag \
  -H "Content-Type: application/json" \
  -d '{
    "query": "What is the overall architecture of this project?",
    "mode": "global"
  }'

# Hybrid mode - Kết hợp cả hai
curl -X POST http://localhost:8000/api/v1/search/rag \
  -H "Content-Type: application/json" \
  -d '{
    "query": "How do the authentication and authorization modules
interact?",
    "mode": "hybrid"
  }'

# Drift mode - Follow relationships
curl -X POST http://localhost:8000/api/v1/search/rag \
  -H "Content-Type: application/json" \
  -d '{
    "query": "Trace the flow from user login to dashboard access",
    "mode": "drift"
  }'
```

7.3 Các câu hỏi mẫu

```
# Hỏi về function cụ thể
"What does the process_payment function do?"

# Hỏi về design patterns
"What design patterns are used in this codebase?"

# Hỏi về dependencies
"What external libraries does this project depend on?"
```

```
# Hỏi về error handling
"How are errors handled in the API layer?"

# Hỏi về testing
"How is the authentication module tested?"

# Hỏi về data flow
"How does data flow from user input to database?"
```

---

8. Graph Operations

8.1 Graph Statistics

```
curl http://localhost:8000/api/v1/graph/stats
```

Response:

```
{
  "node_count": 1250,
  "edge_count": 3500,
  "node_types": {
    "function": 800,
    "class": 200,
    "method": 250
  },
  "edge_types": {
    "calls": 2000,
    "imports": 500,
    "inherits": 100,
    "contains": 900
  }
}
```

8.2 Get Node Details

```
curl http://localhost:8000/api/v1/graph/node/auth.py::User
```

8.3 Get Subgraph

Lấy subgraph xung quanh một node:

```
curl "http://localhost:8000/api/v1/graph/subgraph/auth.py::login_user?
depth=2&max_nodes=50"
```

8.4 Find Path

Tìm đường đi giữa hai nodes:

```
curl "http://localhost:8000/api/v1/graph/path?
source=main.py::main&target=db.py::connect&max_depth=5"
```

8.5 Community Detection

Phát hiện các module/community trong codebase:

```
curl http://localhost:8000/api/v1/graph/communities
```

8.6 Files in Graph

Xem các files đã được index:

```
curl "http://localhost:8000/api/v1/graph/files?pattern=**/*.py&limit=50"
```

---

9. Cấu hình nâng cao

9.1 Environment Variables

```
# Server
export SMART_SEARCH_HOST=0.0.0.0
export SMART_SEARCH_PORT=8000
export SMART_SEARCH_DEBUG=true

# Meilisearch (optional)
export MEILISEARCH_URL=http://localhost:7700
export MEILISEARCH_API_KEY=your_api_key

# Jina AI (optional, for semantic search)
export JINA_API_KEY=your_jina_api_key

# LLM for GraphRAG (optional)
export OPENAI_API_KEY=your_openai_key
# hoặc
export ANTHROPIC_API_KEY=your_anthropic_key
```

9.2 Config File

Tạo file `config.yaml`:

```yaml
server:
  host: 0.0.0.0
  port: 8000
  debug: true

indexing:
  batch_size: 100
  languages:
    - python
    - javascript
    - typescript
  exclude_patterns:
    - "**/node_modules/**"
    - "**/__pycache__/**"
    - "**/venv/**"

search:
  default_limit: 20
  max_limit: 100
  hybrid_weight: 0.5  # 0 = only keyword, 1 = only semantic

graph:
  max_depth: 10
  community_detection: true

rag:
  model: "gpt-4"  # hoặc "claude-3-opus"
  temperature: 0.1
  max_tokens: 2000
```

9.3 Chạy với config

```
PYTHONPATH=src python -m smart_search.main --config config.yaml
```

Troubleshooting

Server không khởi động được

```
# Kiểm tra port đã được sử dụng chưa
lsof -i :8000

# Kill process nếu cần
kill -9 <PID>
```

Index chậm

- Thêm exclude patterns cho các thư mục không cần thiết
- Sử dụng incremental update thay vì full index

Search không trả về kết quả

- Kiểm tra đã index codebase chưa: `GET /api/v1/index/stats`
- Kiểm tra file có trong danh sách languages được hỗ trợ không

GraphRAG không hoạt động

- Kiểm tra đã cấu hình LLM API key chưa
- Kiểm tra logs: `tail -f logs/smart_search.log`

Liên hệ & Hỗ trợ

- GitHub Issues: Report bugs
- Documentation: http://localhost:8000/docs