FAISS Semantic Search Setup - Operational Procedure

Purpose

This procedure enables semantic search across markdown files in a GitHub repository using FAISS indexing and sentence transformers. Use this to set up RAG (Retrieval Augmented Generation) capabilities on your forked repository in GitHub Codespaces.

Prerequisites

□ GitHub account with access to the target repository
□ Repository forked to your account
□ GitHub Codespaces enabled for your account

Section 1: Environment Setup

1.1 Launch GitHub Codespace

□ Navigate to your forked repository on GitHub
□ Click **Code → Codespaces → Create codespace on main**
□ Wait for the codespace to fully initialize

1.2 Create Python Virtual Environment

```bash
python3 -m venv .venv
source .venv/bin/activate
```

Verification:

□ Command prompt shows `(.venv)` prefix
□ Run `which python3` - should show path inside `.venv/bin/`

Section 2: Dependency Installation

2.1 Upgrade Core Package Tools

```bash
pip install --upgrade pip setuptools wheel
```

Expected output: Successfully installed/upgraded messages for pip, setuptools, wheel

2.2 Install Project Dependencies

```bash
pip install -r 6_Symbols/requirements.txt
```

What gets installed:

- `sentence-transformers` - embedding model framework
- `torch` - PyTorch deep learning library
- `faiss-cpu` - Facebook AI Similarity Search
- `markdown-it-py` - Markdown parser

Notes:

□ Installation downloads model `all-MiniLM-L6-v2` (cached by Hugging Face)
□ Process may take 2-5 minutes depending on connection speed
□ Verify no error messages during installation

Section 3: Index Creation

3.1 Run Initial Indexing

```bash
python3 6_Symbols/index.py --folder .
```

What this does:

1. Scans repository root recursively for `.md` files
2. Extracts text content from each markdown file
3. Generates embeddings using `all-MiniLM-L6-v2` model
4. Builds FAISS index (`IndexFlatL2` with `IndexIDMap`)
5. Writes index and filepath mapping to disk

Generated artifacts:

□ `faiss_index.bin` - binary FAISS index file created
□ `filepaths.txt` - index ID to filepath mapping created

Verification:

```bash
```

```
ls -lh faiss_index.bin filepaths.txt
wc -l filepaths.txt  # Should match number of .md files indexed
```

Section 4: Search Testing

4.1 Run Sample Queries

Execute each query and verify results:

```bash
# Query 1: Technical concept
python3 6_Symbols/search.py --query "retrieval augmented generation"
```

- □ Returns relevant technical documentation paths
- □ Shows distance scores (lower = better match)

```bash
# Query 2: General topic
python3 6_Symbols/search.py --query "people"
```

- □ Returns person-related documents
- □ Verify presence of docs like `jane.md`, `john.md`, `mehmet.md`

```bash
# Query 3: Phrase query
python3 6_Symbols/search.py --query "who is"
```

- □ Returns biographical or identity-related content
- □ Results ranked by semantic relevance

4.2 Document Search Results

Record top results for audit trail:

- □ Document filepaths returned for each query
- □ Note any unexpected results for index tuning

Section 5: Repository Maintenance

5.1 Configure .gitignore

Create or update `.gitignore` with these entries:

gitignore

```
# Python virtual environment
.venv/
venv/
env/

# Generated index files
faiss_index.bin
filepaths.txt

# Python artifacts
__pycache__/
*.py[cod]
*.pyo
*.pyd
.Python

# Model cache (optional - uncomment to exclude)
# .cache/
# huggingface/
```

□ `.gitignore` file updated

□ Verified patterns match your setup

5.2 Clean Committed Virtual Environment (if applicable)

If `.venv` was previously committed:

bash

```
git rm -r --cached .venv
git add .gitignore
git commit -m "Remove .venv from repo and update .gitignore"
git push
```

□ Virtual environment removed from git tracking

□ Changes committed and pushed

Section 6: Re-indexing Workflow

6.1 When to Re-index

Re-index when:

- New markdown files are added

- Existing markdown content is updated
- Documents are deleted or moved

6.2 Re-index Procedure

```bash
# Activate virtual environment if not active
source .venv/bin/activate

# Re-run indexer
python3 6_Symbols/index.py --folder .
```

□ Previous `faiss_index.bin` and `filepaths.txt` overwritten
□ Verify updated file timestamps
□ Test search with updated content

Section 7: Integration Guidelines

7.1 Programmatic Search Integration

For RAG applications, implement a search module:

Key functions to implement:

1. Load FAISS index and model on startup (once)
2. Create `semantic_search(query, k=5)` function that:
   - Encodes query with SentenceTransformer
   - Runs `index.search(query_embedding, k)`
   - Maps IDs to filepaths using `filepaths.txt`
   - Returns list of (filepath, distance) tuples

7.2 RAG Application Pattern

```python
# Pseudo-code pattern
results = semantic_search(user_query, k=3)
context = [read_file(path) for path, _ in results]
llm_prompt = f"Context: {context}\n\nQuery: {user_query}"
# Send to LLM...
```

Section 8: Next Steps Checklist

8.1 Enhancement Options

□ Create `chat_app.py` integrating search with LLM

□ Add snippet extraction to show preview text

□ Implement result caching for common queries

□ Add metadata filtering (by directory, date, etc.)

□ Consider upgrade to Qdrant/Milvus for production scale

8.2 Documentation

□ Create `6_Symbols/README.md` with quick-start commands

□ Document custom search parameters and tuning

□ Add example queries specific to your repository content

Troubleshooting

Common Issues

Issue: `ModuleNotFoundError` during search

- **Solution:** Verify virtual environment is activated (`source .venv/bin/activate`)

Issue: FAISS index not found

- **Solution:** Run indexing step (Section 3.1) before searching

Issue: No results returned for queries

- **Solution:** Check `filepaths.txt` is populated; verify `.md` files exist in scanned folder

Issue: Out of memory during indexing

- **Solution:** Index subdirectories separately: `python3 6_Symbols/index.py --folder ./specific_dir`

Completion Checklist

□ Virtual environment created and activated

□ All dependencies installed successfully

□ FAISS index generated (`faiss_index.bin` exists)

□ Filepath mapping created (`filepaths.txt` exists)

□ Sample searches executed and verified

□ `.gitignore` configured appropriately

□ Re-indexing procedure tested

□ Documentation updated (if applicable)

Procedure completed by: _______________

Date: _______________

Notes: _______________