🚀 FedSearch-NLP - COMPLETE SETUP GUIDE

📦 What You've Received

Complete working FastAPI backend with:

- ✅ RAG (Retrieval-Augmented Generation) pipeline
- ✅ FAISS vector database for document search
- ✅ Flan-T5 for answer generation
- ✅ 4 sample company documents
- ✅ Interactive API documentation
- ✅ Production-ready code structure

🎯 3-Step Quick Start

Step 1: Create Project Structure

```
# Create main directory
mkdir fedsearch_nlp
cd fedsearch_nlp

# Create all subdirectories
mkdir -p app/api app/core app/services app/utils
mkdir -p data/company_docs data/embeddings models

# Create __init__.py files
touch app/__init__.py
touch app/api/__init__.py
touch app/core/__init__.py
touch app/services/__init__.py
```

touch app/utils/__init__.py

Step 2: Create All Files

Copy the following files from the artifacts:

📄 **Configuration Files**

1. requirements.txt - Dependencies
2. .env - Environment variables

🧠 **Core Application**

3. app/main.py - FastAPI application
4. app/core/config.py - Configuration settings
5. app/core/rag_engine.py - RAG orchestration

🌉 **API Layer**

6. app/api/routes.py - API endpoints
7. app/api/models.py - Request/response schemas

🔧 **Services**

8. app/services/document_processor.py - Document loading
9. app/services/retriever.py - FAISS retrieval
10. app/services/generator.py - Answer generation

📚 **Data Files**

11. Run the company documents creation script to generate:
    - data/company_docs/hr_policy.txt
    - data/company_docs/it_sop.txt
    - data/company_docs/legal_doc.txt
    - data/company_docs/product_guide.txt

🧪 **Testing & Setup**

12. test_api.py - API test suite

13. setup_and_run.sh - Automated setup script

14. README.md - Documentation

Step 3: Install & Run

```
# Create virtual environment
python3 -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install dependencies
pip install -r requirements.txt

# Create company documents
python3 -c "
import os

files = {
    'data/company_docs/hr_policy.txt': '''ACME Corporation - Human Resources Policy Manual

1. LEAVE POLICY
- Annual Leave: 20 days per year for full-time employees
- Sick Leave: 10 days per year with medical certificate required after 3 consecutive days
- Parental Leave: 12 weeks paid leave for primary caregivers

2. WORKING HOURS
- Standard working hours: Monday to Friday, 9:00 AM to 5:00 PM
- Remote work policy: Up to 3 days per week for eligible positions
```

3. COMPENSATION AND BENEFITS

- Salary reviews conducted annually in January

- Performance bonuses up to 20% of annual salary

- Health insurance coverage for employee and immediate family''',

'data/company_docs/it_sop.txt': '''ACME Corporation - IT Standard Operating Procedures

1. SYSTEM ACCESS

- New employee access: IT ticket must be raised by HR within 24 hours

2. PASSWORD POLICY

- Minimum 12 characters with mixed case, numbers, and symbols

- Password expiry: Every 90 days

- Multi-factor authentication (MFA) mandatory

3. DATA BACKUP

- Automated daily backups at 2:00 AM

- Weekly full backups on Sundays''',

'data/company_docs/legal_doc.txt': '''ACME Corporation - Legal Guidelines

1. INTELLECTUAL PROPERTY

- All work products belong to ACME Corporation

2. DATA PROTECTION AND PRIVACY

- GDPR compliance mandatory for all EU customer data

- Data retention: 7 years after last transaction

3. REGULATORY COMPLIANCE

- ISO 27001, SOC 2 Type II maintained'''',

    'data/company_docs/product_guide.txt': '''ACME Corporation - Product Guide 2024

1. CloudSync Pro

- Enterprise cloud storage

- Pricing: \$15/user/month

2. DataFlow Analytics

- Business intelligence platform

- Pricing: \$99/month for 5 users

3. SecureAuth Identity

- Enterprise identity management

- Pricing: \$8/user/month'''

}

for filepath, content in files.items():

    with open(filepath, 'w') as f:

        f.write(content)

    print(f'✓ {filepath}')

"

```
# Start the server
python -m uvicorn app.main:app --reload
```

🎮 Using Your API

1. Open API Documentation

Open browser: **http://localhost:8000/docs**

2. Index Documents (First Time Only)

Via Browser (Swagger UI):

1. Go to http://localhost:8000/docs
2. Click on POST /api/index
3. Click "Try it out"
4. Click "Execute"

Via Command Line:

```
curl -X POST "http://localhost:8000/api/index" \
  -H "Content-Type: application/json" \
  -d '{"reindex": false}'
```

Expected Response:

```
{
  "status": "success",
  "documents_indexed": 42,
  "message": "Index built successfully"
}
```

3. Ask Questions

Via Swagger UI:

1. Click on POST /api/query
2. Click "Try it out"

3. Enter your question in the request body
4. Click "Execute"

Via Command Line:

```
curl -X POST "http://localhost:8000/api/query" \
  -H "Content-Type: application/json" \
  -d '{
    "question": "How many days of annual leave do employees get?",
    "top_k": 3
  }'
```

Expected Response:

```
{
  "answer": "Employees get 20 days of annual leave per year for full-time positions.",
  "retrieved_documents": [
    {
      "content": "Annual Leave: 20 days per year for full-time employees...",
      "score": 0.89,
      "source": "hr_policy.txt"
    }
  ],
  "confidence": 0.92
}
```

🧪 Testing

Run the automated test suite:

```
# In a new terminal (keep server running)
python test_api.py
```

This will:

- ✅ Test all endpoints
- ✅ Index documents
- ✅ Run sample queries
- ✅ Verify responses

📝 Sample Questions to Try

HR Policy

"How many sick leave days do employees get?"

"What is the remote work policy?"

"When are salary reviews conducted?"

IT Procedures

"What is the password policy?"

"How often are backups performed?"

"What is MFA?"

Legal

"What is the data retention policy?"

"What compliance certifications do we have?"

Products

"What is the pricing for CloudSync Pro?"

"Tell me about DataFlow Analytics"

"What features does SecureAuth have?"

🎯 API Endpoints Reference

Method	Endpoint	Description	Request Body
GET	/	API overview	-
GET	/api/health	System health	-
POST	/api/index	Index documents	{"reindex": bool}
POST	/api/query	Ask questions	{"question": str, "top_k": int}
GET	/api/documents/stats	Document stats	-

🔧 **Customization**

Add Your Own Documents

1. **Add text files** to data/company_docs/
2. **Reindex**:
3. curl -X POST "http://localhost:8000/api/index" \ -H "Content-Type: application/json" \ -d '{"reindex": true}'

Change Models

Edit .env:

Smaller models (faster, less accurate)

RETRIEVER_MODEL="sentence-transformers/all-MiniLM-L6-v2"

GENERATOR_MODEL="google/flan-t5-small"

Larger models (slower, more accurate)

RETRIEVER_MODEL="sentence-transformers/all-mpnet-base-v2"

GENERATOR_MODEL="google/flan-t5-large"

Adjust Retrieval

TOP_K=5 # Retrieve more documents

MAX_LENGTH=1024 # Handle longer documents

🚨 Troubleshooting

Issue: "ModuleNotFoundError"

Solution: Activate virtual environment

source venv/bin/activate

Issue: "Connection refused"

Solution: Start the server

python -m uvicorn app.main:app --reload

Issue: "Port 8000 already in use"

Solution: Change port in .env or command

python -m uvicorn app.main:app --reload --port 8001

Issue: "Out of memory"

Solution: Use smaller models (see Customization above)

Issue: Models downloading slowly

Solution: Use cache directory

export TRANSFORMERS_CACHE="./models"

export SENTENCE_TRANSFORMERS_HOME="./models"

📊 System Requirements

Minimum:

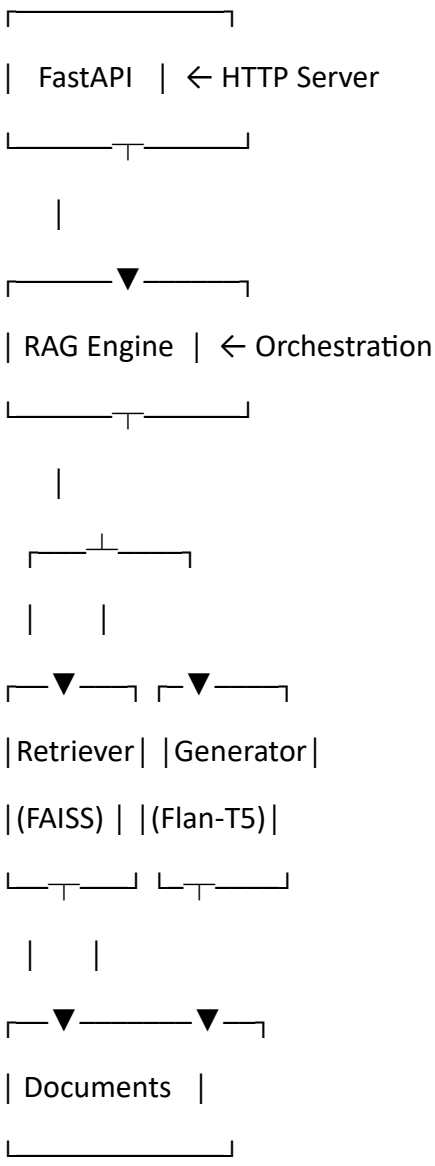
- Python 3.8+
- 4GB RAM
- 2GB disk space

Recommended:

- Python 3.10+
- 8GB RAM

- GPU (optional, for faster inference)
- 5GB disk space

🎓 **Architecture Overview**



Flow:

1. User sends question → FastAPI
2. RAG Engine retrieves relevant docs (FAISS)
3. Generator creates answer (Flan-T5)

4. Response sent back to user

🚀 Next Steps

Phase 1: Current (✅ COMPLETE)

- [x] Basic RAG pipeline
- [x] FastAPI backend
- [x] Document indexing
- [x] Query endpoint

Phase 2: Enhancements

- [ ] Add PDF/DOCX support
- [ ] Implement conversation history
- [ ] Add user authentication
- [ ] Deploy with Docker

Phase 3: Federated Learning

- [ ] Multi-client architecture
- [ ] Secure aggregation
- [ ] Differential privacy
- [ ] Model updates

📞 Support

Documentation: http://localhost:8000/docs
Interactive API: http://localhost:8000/redoc
GitHub: [Your Repository]

✅ Success Checklist

Before considering setup complete, verify:

- [ ] Server starts without errors
- [ ] Can access http://localhost:8000/docs
- [ ] Document indexing completes successfully
- [ ] Sample queries return answers
- [ ] Confidence scores are reasonable (>0.5)
- [ ] Retrieved documents are relevant
- [ ] Test suite passes all tests

🎉 **You're all set! Start asking questions to your enterprise knowledge base!**