

ZERO TO HERO IN RETRIEVAL AUGMENTED GENERATION

Yogesh Haribhau Kulkarni



From Zero to Hero in RAG

Then from RAG to Riches

Learning Path: Python → DSA → System Design → NLP → ML → LLMs → RAG → Production

Full-Time Mode

- ▶ Duration: 15 working days (3 weeks)
- ▶ Daily: 8 hours (4 learning + 4 coding)
- ▶ Week 4: Independent capstone
- ▶ Saturdays: Overview + QnA (2 hrs)

Part-Time Mode

- ▶ Duration: 12 weeks (3 months)
- ▶ Weekly: Self-paced tutorials + coding
- ▶ Month 4: Independent capstone
- ▶ Weekends: Optional QnA sessions

What You'll Build:

- ▶ 12-15 progressive projects (choose 1 per day/week)
- ▶ 1 production-ready capstone system
- ▶ Complete end-to-end RAG portfolio

Target Audience: Software engineers with basic programming knowledge



Program Architecture

Phase 1: Python Foundations (Days 1-5; Weeks 1-4)

- ▶ Python Fundamentals
- ▶ File Handling & Decorators
- ▶ Async Programming
- ▶ DSA Essentials
- ▶ System Design (Optional)

Phase 2: NLP & ML (Days 6-10; Weeks 5-8)

- ▶ NLP with spaCy
- ▶ Machine Learning
- ▶ Word Embeddings
- ▶ LLMs & Prompting
- ▶ Transformers (Optional)

Phase 3: RAG (Days 11-15 Weeks 9-12)

- ▶ RAG from Scratch
- ▶ Docstring Parsing
- ▶ LangChain RAG
- ▶ Streamlit UI
- ▶ Cloud Deploy (Optional)

Daily/Weekly Structure:

- ▶ **Full-Time:** 09:00-13:00 Learning — 14:00-18:00 Coding (2 project options - pick 1)
- ▶ **Part-Time:** Self-paced tutorials + hands-on — Complete at your own pace

Milestones:

Day 1; Week 1: Python Fundamentals

Learning Topics (4 hrs)

- ▶ Variables & data types
- ▶ Control flow (if, loops)
- ▶ Functions & scope
- ▶ Object-oriented programming
- ▶ Classes & objects
- ▶ Inheritance & polymorphism

Project: Pick 1 (4 hrs)

- ▶ **Library Management:** Build system with books, members, lending using OOP classes and inheritance
- ▶ **Shopping Cart System:** Create product catalog, cart operations, checkout with OOP

Learning Resources:

- ▶ Python Basics (<https://www.freecodecamp.org/news/learn-python-basics/>)
- ▶ Python OOP (<https://www.freecodecamp.org/news/object-oriented-programming-python/>)
- ▶ Documentation: Python Official Docs (<https://docs.python.org/3/tutorial>)

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working code + README



Day 2; Week 2: Python File Handling

Learning Topics (4 hrs)

- ▶ File I/O operations
- ▶ Reading/writing text files
- ▶ JSON handling
- ▶ CSV processing
- ▶ Exception handling
- ▶ Decorators basics

Learning Resources:

- ▶ File Handling (<https://www.freecodecamp.org/news/file-handling-in-python/>)
- ▶ Decorators (<https://www.freecodecamp.org/news/python-decorators-explained-with-examples/>)

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working code + README

Project: Pick 1 (4 hrs)

- ▶ **CSV Data Processor:** Read CSV, filter/transform data, export with error handling and logging
- ▶ **Config File Manager:** Build JSON/YAML config reader/writer with validation and backup

Day 3; Week 3: Python Advanced

Learning Topics (4 hrs)

- ▶ List/dict/set comprehensions
- ▶ Generator expressions
- ▶ Async/await basics
- ▶ Asyncio fundamentals
- ▶ Concurrent operations
- ▶ Async file operations

Project: Pick 1 (4 hrs)

- ▶ **Async Web Scraper:** Fetch multiple URLs concurrently, parse HTML, save results using asyncio and aiohttp
- ▶ **Concurrent File Processor:** Process multiple large files in parallel with async I/O and progress tracking

Learning Resources:

- ▶ FreeCodeCamp: Concurrency vs. Parallelism (<https://www.freecodecamp.org/news/concurrency-vs-parallelism-whats-the-difference-and-why-should-you-care/>)
- ▶ Getting Started With Async Features in Python (<https://realpython.com/python-async-features/>)

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working code + README



Day 4; Week 4: Data Structs & Algorithms

Learning Topics (4 hrs)

- ▶ Arrays & lists
- ▶ Hash tables/dictionaries
- ▶ Stacks & queues
- ▶ Binary search
- ▶ Sorting algorithms
- ▶ Big O complexity

Project: Pick 1 (4 hrs)

- ▶ **LeetCode Problem Set:** Solve 5 medium problems on arrays, hashmaps, and sorting
- ▶ **Custom Data Structures:** Implement stack, queue, and hashmap from scratch with test cases

Learning Resources:

- ▶ Algorithms and Data Structures (<https://www.freecodecamp.org/news/algorithms-and-data-structures-free-treehouse-course/>)
- ▶ Practice: LeetCode (<https://leetcode.com/explore/learn/>)

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working code + README

Day 5: System Design Fundamentals

Learning Topics (4 hrs)

- ▶ Requirements analysis
- ▶ Scalability patterns
- ▶ Caching strategies
- ▶ Load balancing
- ▶ Database design
- ▶ API design

Project: Pick 1 (4 hrs)

- ▶ **URL Shortener:** Design TinyURL-like service with hashing, database, and caching
- ▶ **API Rate Limiter:** Build token bucket rate limiter with cache and API endpoints

Learning Resources:

- ▶ System Design (<https://www.freecodecamp.org/news/software-system-design-for-beginners/>)
- ▶ System Design Interview (<https://www.freecodecamp.org/news/systems-design-for-interviews/>)
- ▶ Gaurav Sen System Design Youtube Playlist

Note: Included in full-time Day 5 — Optional self-study for part-time learners

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Design doc + Code



Phase 1 Milestone Assessment

Learning Outcomes

- ▶ Python fundamentals mastery
- ▶ File & async operations
- ▶ DSA problem-solving
- ▶ System design thinking (optional)

Success Criteria

- ▶ 4-5 projects completed
- ▶ Code quality ≥ 80%
- ▶ Core concepts understood
- ▶ Ready for NLP phase

Assessment Activities:

- ▶ Code review of all projects
- ▶ Technical Q&A on concepts
- ▶ Readiness verification for Phase 2

Assessment Rubric (100 pts):

- ▶ Functionality (50 pts) - All features working correctly
- ▶ Code Quality (20 pts) - Clean, documented, readable
- ▶ Design Principles (20 pts) - Architecture and patterns
- ▶ Documentation (10 pts) - README and comments

Day 6; Week 5: NLP with spaCy

Learning Topics (4 hrs)

- ▶ spaCy pipeline basics
- ▶ Text preprocessing & tokenization
- ▶ Part-of-speech tagging
- ▶ Named Entity Recognition (NER)
- ▶ Dependency parsing
- ▶ Pipeline customization

Learning Resources:

- ▶ NLP with spaCy (<https://www.freecodecamp.org/news/natural-language-processing-with-spacy-python-full-course/>)
- ▶ Extract Insights from Text (<https://www.freecodecamp.org/news/extract-insights-from-text-using-named-entity-recognition/>)
- ▶ Documentation: spaCy Docs (<https://spacy.io/usage/spacy-101>)

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working code + README



Day 7; Week 6: Machine Learning

Learning Topics (4 hrs)

- ▶ Types of Machine Learning
- ▶ Classification algorithms
- ▶ Clustering techniques
- ▶ Scikit-learn essentials
- ▶ Model training & evaluation
- ▶ Cross-validation & metrics

Learning Resources:

- ▶ Machine Learning in Python (<https://www.freecodecamp.org/news/machine-learning-with-python-for-beginners/>)
- ▶ Scikit-Learn Course (<https://www.freecodecamp.org/news/scikit-learn-crash-course/>)
- ▶ Sentiment Analysis Tutorial (<https://www.freecodecamp.org/news/sentiment-analysis-with-text-mining/>)

Project: Pick 1 (4 hrs)

- ▶ **Text Classifier:** Build sentiment analysis or spam detection classifier with scikit-learn
- ▶ **Document Clusterer:** Create text clustering system with K-means and visualization

Difficulty: Medium | **Time:** 4 hours | **Deliverable:** Working code + README



Day 8; Week 7: Word Embeddings

Learning Topics (4 hrs)

- ▶ Bag of Words & TF-IDF
- ▶ Word embeddings concepts
- ▶ Word2Vec (CBOW, Skip-Gram)
- ▶ GloVe embeddings
- ▶ Similarity measures
- ▶ Vector operations

Project: Pick 1 (4 hrs)

- ▶ **Semantic Search Engine:** Build document search using embeddings with similarity ranking
- ▶ **Word Analogy Solver:** Create tool for word relationships using pre-trained embeddings

Learning Resources:

- ▶ Understanding Word Embeddings (<https://www.freecodecamp.org/news/understanding-word-embeddings-the-building-blocks-of-nlp-and-gpts/>)
- ▶ Word2Vec Tutorial (<https://www.freecodecamp.org/news/how-to-get-started-with-word2vec-and-then-how-to-make-it-work-d0a2fca9dad3/>)
- ▶ TF-IDF in Python (<https://www.freecodecamp.org/news/how-to-process-textual-data-using-tf-idf-in-python-cd2bbc0a94a3/>)

Difficulty: Medium | **Time:** 4 hours | **Deliverable:** Working code + README

Day 9; Week 8: Large Language Models & Prompt Engineering

Learning Topics (4 hrs)

- ▶ Introduction to LLMs
- ▶ OpenAI API & models
- ▶ Prompt engineering basics
- ▶ Few-shot learning
- ▶ Prompt templates
- ▶ Best practices & safety

Project: Pick 1 (4 hrs)

- ▶ **AI Writing Assistant:** Build text generation tool with prompt templates and OpenAI API
- ▶ **Intelligent Q&A Bot:** Create chatbot with context-aware prompts and conversation memory

Learning Resources:

- ▶ ChatGPT Prompt Engineering (<https://www.freecodecamp.org/news/prompt-engineering-chatgpt-course/>)
- ▶ OpenAI API Tutorial (<https://www.freecodecamp.org/news/how-to-use-the-openai-api-to-build-apps/>)
- ▶ LLM Handbook (<https://www.freecodecamp.org/news/large-language-models-explained/>)

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working code + README



Day 10: Transformers

Learning Topics (4 hrs)

- ▶ Transformer architecture
- ▶ Attention mechanism
- ▶ BERT, GPT models
- ▶ Hugging Face ecosystem
- ▶ Transfer learning
- ▶ Fine-tuning basics

Project: Pick 1 (4 hrs)

- ▶ **Transformer Classifier:** Fine-tune pre-trained BERT for text classification task
- ▶ **Custom Text Generator:** Fine-tune GPT-2 on custom dataset for domain-specific generation

Learning Resources:

- ▶ Transformers Explained (<https://www.freecodecamp.org/news/transformer-neural-network-explained/>)
- ▶ Hugging Face Course (<https://www.freecodecamp.org/news/get-started-with-hugging-face/>)
- ▶ Fine-Tuning Tutorial (<https://www.freecodecamp.org/news/how-to-fine-tune-llms/>)

Note: Included in full-time Day 10 — Optional self-study for part-time learners

Difficulty: Advanced | **Time:** 4 hours | **Deliverable:** Working code + README



Phase 2 Milestone Assessment

Learning Outcomes

- ▶ NLP with spaCy mastery
- ▶ ML classification skills
- ▶ Word embeddings understanding
- ▶ LLM & prompting expertise

Success Criteria

- ▶ 4-5 NLP projects completed
- ▶ ML model ≥ 75% accuracy
- ▶ Vector operations working
- ▶ Ready for RAG phase

Assessment Activities:

- ▶ NLP pipeline demonstration
- ▶ Sentiment model evaluation
- ▶ Component integration check

Assessment Rubric (100 pts):

- ▶ Functionality (50 pts) - All NLP components working
- ▶ Model Quality (20 pts) - Accuracy and performance
- ▶ Code Quality (20 pts) - Clean and documented
- ▶ Integration (10 pts) - Components work together

Day 11; Week 9: RAG from Scratch

Learning Topics (4 hrs)

- ▶ RAG architecture overview
- ▶ Document chunking strategies
- ▶ Vector stores (FAISS/ChromaDB)
- ▶ Retrieval mechanisms
- ▶ Re-ranking techniques
- ▶ Basic RAG pipeline

Project: Pick 1 (4 hrs)

- ▶ **Basic RAG System:** Build end-to-end RAG with loading, chunking, embedding, retrieval, generation
- ▶ **Multi-Document RAG:** Create RAG handling multiple documents with metadata filtering

Learning Resources:

- ▶ RAG Handbook (<https://www.freecodecamp.org/news/retrieval-augmented-generation-rag-handbook/>)
- ▶ LangChain RAG (<https://www.freecodecamp.org/news/mastering-rag-from-scratch/>)
- ▶ YouTube: “Complete RAG Crash Course” - Krish Naik

Key Pipeline: Load → Chunk → Embed → Store → Retrieve → Generate

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working RAG + README



Day 12; Week 10: Document Parsing with Docling

Learning Topics (4 hrs)

- ▶ Docling architecture
- ▶ PDF/DOCX/PPTX parsing
- ▶ Layout analysis
- ▶ Table extraction
- ▶ Image extraction
- ▶ Metadata handling

Project: Pick 1 (4 hrs)

- ▶ **Multi-Format Parser:** Build parser supporting PDF/DOCX/PPTX with text, tables, images
- ▶ **Structured Data Extractor:** Create tool for hierarchical content extraction

Learning Resources:

- ▶ Docling: Step-by-Step Guide (<https://www.datacamp.com/tutorial/docling>)
- ▶ Documentation: Docling Official Docs (https://docling-project.github.io/docling/getting_started/)

Difficulty: Medium | **Time:** 4 hours | **Deliverable:** Working parser + README



Day 13; Week 11: LangChain RAG Implementation

Learning Topics (4 hrs)

- ▶ LangChain Essentials course
- ▶ Document loaders
- ▶ Text splitters
- ▶ Embeddings integration
- ▶ Vector stores
- ▶ Retrieval chains
- ▶ LCEL expressions

Learning Resources:

- ▶ Course: LangChain Academy Essentials (<https://academy.langchain.com/courses/langchain-essentials-python>)
- ▶ What is RAG? (<https://www.datacamp.com/blog/what-is-retrieval-augmented-generation-rag>)

Difficulty: Medium |**Time:** 4 hours |**Deliverable:** Working RAG + README

Project: Pick 1 (4 hrs)

- ▶ **PDF Q&A System:** Build LangChain-based RAG for PDF documents with chat history
- ▶ **Conversational RAG:** Create multi-turn RAG chatbot with memory and context awareness

Day 14; Week 12: Streamlit UI Development

Learning Topics (4 hrs)

- ▶ Streamlit basics
- ▶ Layout components
- ▶ Interactive widgets
- ▶ File uploads
- ▶ Session state management
- ▶ Chat interfaces

Learning Resources:

- ▶ Build 12 Data Science Apps with Streamlit
- ▶ Documentation: Streamlit Official Docs (<https://docs.streamlit.io/>)

Difficulty: Medium | **Time:** 4 hours | **Deliverable:** Working UI + README

Project: Pick 1 (4 hrs)

- ▶ **RAG Chatbot Interface:** Create interactive chat UI with file upload and message history
- ▶ **Document Q&A Dashboard:** Build multi-page app with document upload and query interface

Day 15: Cloud Deployment

Learning Topics (4 hrs)

- ▶ Cloud RAG architecture patterns
- ▶ AWS Bedrock: Foundation models & Knowledge Bases
- ▶ Azure AI Search & OpenAI integration
- ▶ Serverless deployment strategies
- ▶ Managed vector stores
- ▶ Cost optimization & scaling

Learning Resources:

- ▶ AWS Bedrock Tutorial (<https://www.freecodecamp.org/news/how-to-use-amazon-bedrock-to-build-generative-ai-applications/>)
- ▶ Build AI Apps with Azure (<https://www.freecodecamp.org/news/build-ai-apps-with-azure-openai-and-semantic-kernel-sdk/>)
- ▶ AWS Docs: Bedrock Knowledge Bases (<https://docs.aws.amazon.com/bedrock/>)

Project: Pick 1 (4 hrs)

- ▶ **AWS Bedrock RAG:** Deploy RAG using Bedrock Knowledge Bases, S3, Lambda
- ▶ **Azure AI RAG:** Build RAG with Azure AI Search, OpenAI, Blob Storage

Note: Included in full-time Day 15 — Optional self-study for part-time learners

Phase 3 Milestone & Capstone Kickoff

Learning Outcomes

- ▶ Complete RAG systems
- ▶ Document processing mastery
- ▶ LLM orchestration
- ▶ Cloud deployment (optional)

Success Criteria

- ▶ All components working
- ▶ Code well-documented
- ▶ Ready for capstone

Production Readiness Checklist:

- ▶ Local RAG implementation complete
- ▶ Error handling & validation implemented
- ▶ Performance optimized
- ▶ Comprehensive documentation

Capstone Preview:

- ▶ Week/Month 4: Independent capstone development
- ▶ Build Document Intelligence System OR Enterprise Knowledge Base
- ▶ Integrate: Docling + LangChain + Streamlit + (optional) Cloud
- ▶ Prepare for final presentation

Week/Month 4: Capstone Project

Choose 1 Capstone Project:

- ▶ **Document Intelligence System:** Multi-format parser (Docling) + RAG + Streamlit UI + (optional) Cloud deployment
- ▶ **Enterprise Knowledge Base:** Multi-document RAG with advanced search, filters, analytics dashboard

Integration Requirements:

- ▶ Docling: Multi-format document parsing (PDF/DOCX/PPTX)
- ▶ LangChain: Complete RAG orchestration with chains
- ▶ Streamlit: Professional interactive web interface
- ▶ Cloud: AWS Bedrock OR Azure AI (optional but recommended)
- ▶ All components seamlessly integrated

Development Schedule:

- ▶ **Full-Time:** Week 4 independent development with async mentor support
- ▶ **Part-Time:** Month 4 self-paced development with optional check-ins
- ▶ Daily/weekly progress tracking available
- ▶ Code review sessions on request

Capstone Evaluation (100 pts): Functionality (40) | Code Quality (30)

Implementation (20) | Documentation (10)

Capstone Presentation Day

Session Structure (4 hours):

- ▶ 10:00-12:30: Individual capstone presentations (20 mins each)
- ▶ 12:30-13:00: Break
- ▶ 13:00-14:00: Open Q&A & technical discussions

Presentation Format (20 mins):

- ▶ Project demo (8 mins) - Live demonstration
- ▶ Architecture overview (5 mins) - System design & components
- ▶ Code walkthrough (5 mins) - Key implementation details
- ▶ Q&A (2 mins) - Questions from instructors

Evaluation Criteria:

- ▶ **Functionality (40 pts):** All features working, error-free
- ▶ **Code Quality (30 pts):** Clean, modular, documented
- ▶ **Architecture (20 pts):** Sound design, scalable structure
- ▶ **Documentation (10 pts):** Clear README, setup instructions

Q&A Topics:

- ▶ Technical challenges & solutions
- ▶ RAG optimization techniques

Congratulations! From RAG to Riches

Technical Skills Acquired

- ▶ Advanced Python programming
- ▶ NLP & text processing
- ▶ RAG architecture & implementation
- ▶ Full-stack AI development
- ▶ Production deployment (optional)

Career Pathways

- ▶ AI/ML Engineer
- ▶ RAG Solutions Architect
- ▶ LLM Application Developer
- ▶ AI Product Engineer
- ▶ NLP Engineer

Recommended Certifications:

- ▶ LangChain Academy Certification (academy.langchain.com)
- ▶ Azure AI Fundamentals (AI-900)
- ▶ AWS Machine Learning Specialty

Continuous Learning Resources:

- ▶ LangChain Academy (academy.langchain.com)
- ▶ DeepLearning.AI Short Courses (deeplearning.ai)
- ▶ Hugging Face NLP Course (huggingface.co/learn/nlp-course)
- ▶ LangChain Documentation (python.langchain.com)
- ▶ Explore open-source RAG projects on GitHub

