GenAI Learning Roadmap: From Beginner to Job-Ready

Master the essential GenAI concepts, tools, and projects employers are looking for.

1. Introduction

This roadmap is designed to take a learner from basic awareness of Generative AI to the point where they can confidently build projects, explain technical concepts, and prepare for entry-level GenAI roles. It combines conceptual learning, tool practice, project work, and portfolio preparation so that learning is not limited to theory.

- **How to use this roadmap:** Follow the sections in order if you are new to AI. If you already know Python or software development, you may move faster through the developer stack and spend more time on RAG, agents, and portfolio projects.
- **Who it is for:** Students, career switchers, business analysts, developers, consultants, instructional designers, product managers, and professionals who want practical GenAI skills.
- **What you will achieve:** By the end, you should understand AI terminology, write effective prompts, build simple GenAI applications, create a GitHub portfolio, and prepare for interviews or internal AI project work.

1.1 Build Your AI Foundations

Before using advanced tools, build a clear understanding of the core AI landscape. Artificial Intelligence is the broad field of building systems that perform tasks normally associated with human intelligence, such as reasoning, perception, decision-making, and language understanding. Machine Learning is a subset of AI where systems learn patterns from data. Large Language Models are AI models trained on massive text and code

datasets to understand and generate language. Generative AI refers to systems that create new outputs such as text, code, images, summaries, audio, or structured data.

- **AI:** The broad discipline of making machines perform intelligent tasks.
- **Machine Learning:** A method where models learn from examples rather than being explicitly programmed for every rule.
- **Deep Learning:** A machine learning approach using neural networks with many layers.
- **LLM:** A large language model that predicts and generates text based on patterns learned from training data.
- **GenAI:** AI that creates new content, such as emails, reports, code, images, or answers.
- **Example:** A spam filter uses machine learning to classify emails, while a GenAI assistant can draft a reply, summarize an email thread, or create a knowledge-base article.

Key terms to learn early include tokens, context window, temperature, hallucination, grounding, embeddings, vector search, inference, fine-tuning, prompt engineering, RAG, model evaluation, and responsible AI. A beginner should not try to memorize everything at once. Instead, learn each term when you use it in a small exercise.

- **Recommended free resources:** Microsoft Generative AI for Beginners, Microsoft Learn AI Fluency modules, Google AI Essentials, DeepLearning.AI short courses,

OpenAI and Anthropic documentation, and beginner-friendly YouTube explainers.

- **Practice activity:** Pick one AI tool and ask it to explain the difference between AI, ML, LLMs, and GenAI in three ways: for a child, for a manager, and for a developer.

1.2 Learn Prompt Engineering

Prompt engineering is the skill of giving clear instructions to an AI model so it produces useful, accurate, and appropriately formatted responses. A prompt is not just a question. It can include role, context, task, constraints, examples, tone, output format, and evaluation criteria.

- **Prompting fundamentals:** Be specific, provide context, define the audience, specify the output format, and ask the model to make assumptions visible.
- **Prompting frameworks:** Use Role-Task-Context-Format, CLEAR, CREATE, or Chain-of-Thought style reasoning when appropriate. For business use, a simple structure is: role, goal, background, constraints, examples, and final format.
- **Example prompt:** "Act as an AI learning coach. Create a 4-week study plan for a beginner who can study 1 hour per day. Include concepts, practice tasks, and project milestones in a table."
- **Hands-on exercises:** Summarize a policy, classify customer tickets, rewrite a resume bullet, generate interview questions, extract action items from meeting notes, and create structured JSON from unstructured text.

- **Recommended tools:** ChatGPT, Claude, Gemini, Perplexity, Microsoft Copilot, NotebookLM, and API playgrounds from model providers.

A strong prompt engineer also learns how to test prompts. Do not accept the first output blindly. Compare outputs across tools, check factual accuracy, ask for sources when appropriate, and create prompt variants for different business scenarios. For example, a customer-support prompt should be tested against simple questions, angry customers, missing information, and policy-sensitive issues.

1.3 Master the GenAI Stack

The GenAI stack is the set of technologies used to build useful AI applications. A basic chatbot only sends a prompt to a model, but a real business application often connects the model to documents, databases, APIs, tools, permissions, workflows, and monitoring systems.

- **RAG explained:** Retrieval-Augmented Generation allows an AI system to retrieve relevant information from a knowledge source before generating an answer. This is useful when the answer depends on private, recent, or domain-specific information.

- **Embeddings:** Embeddings convert text into numerical vectors that capture meaning. Similar ideas are placed close together in vector space, enabling semantic search.

- **Vector databases:** These store embeddings and support similarity search. Common options include FAISS, Chroma, Pinecone, Weaviate, Milvus, and Azure AI Search.
- **LangChain:** A framework for building LLM applications by connecting models, prompts, retrievers, tools, chains, and agents.
- **AI agents:** Systems that can reason through a task, choose tools, call APIs, search information, and take multi-step actions under constraints.
- **Model Context Protocol:** MCP is an open standard for connecting AI assistants to external tools and data sources through a consistent protocol, reducing the need for one-off integrations.

Example: A company policy assistant may use RAG to answer employee questions. The system loads HR policy documents, splits them into chunks, creates embeddings, stores them in a vector database, retrieves the most relevant chunks when a user asks a question, and sends those chunks to the LLM to generate a grounded response.

1.4 Learn Essential AI Tools

Learning GenAI is easier when you experiment with multiple tools. Each tool has strengths, limitations, and ideal use cases. The goal is not to become dependent on one interface, but to understand how to choose the right tool for the task.

- **ChatGPT:** Useful for brainstorming, writing, coding help, data analysis, and building custom workflows.

- **Claude:** Strong for long-form writing, document reasoning, analysis, and thoughtful explanations.
- **Gemini:** Useful for Google ecosystem workflows, multimodal tasks, and research-style exploration.
- **Perplexity:** Helpful for web research and source-aware answers.
- **GitHub Copilot:** Assists with code completion, refactoring, test generation, and developer productivity.
- **Cursor:** An AI-first code editor that helps developers work across files, debug code, and build applications faster.
- **NotebookLM:** Useful for learning from uploaded documents, summarizing sources, creating study guides, and generating Q&A from trusted materials.

Practice by giving the same task to different tools. For example, upload a short PDF and ask each tool to summarize it, extract action items, create flashcards, and identify risks. Compare accuracy, clarity, formatting, and source handling.

1.5 Build Your First Projects

Projects are the fastest way to move from learning to job readiness. Employers and clients want to see what you can build, explain, and improve. Start with small projects, then add better user experience, evaluation, documentation, and deployment.

- **AI Resume Reviewer:** Upload or paste a resume, compare it against a job description, and generate improvement suggestions. Example feature: score alignment with required skills.
- **RAG Chatbot:** Build a chatbot that answers questions from a set of PDFs or internal documents. Example feature: show source snippets used for the answer.
- **Document Q&A Assistant:** Let users ask questions about contracts, policies, manuals, or training material. Example: "What are the leave approval rules?"
- **Customer Support Bot:** Classify tickets, suggest replies, and escalate sensitive issues. Example categories: billing, login, refund, technical issue, complaint.
- **AI Research Assistant:** Summarize articles, compare viewpoints, create research briefs, and generate follow-up questions.
- **Knowledge Base Search:** Build semantic search over FAQs, SOPs, policy documents, or product documentation.

For every project, document the problem, users, data source, architecture, prompt strategy, evaluation method, limitations, and future improvements. A simple project with excellent documentation is often more impressive than a complex project that cannot be explained clearly.

1.6 Learn the Developer Stack

You do not need to become a full-stack engineer on day one, but job-ready GenAI learners should understand enough software development to build prototypes, work with APIs, manage code, and explain how systems are connected.

- **Python basics:** Variables, functions, lists, dictionaries, loops, file handling, virtual environments, and packages.
- **APIs:** Learn how applications send requests and receive responses. Practice calling an LLM API and handling JSON output.
- **Git and GitHub:** Track changes, publish projects, write README files, and collaborate.
- **VS Code:** Use extensions, terminals, notebooks, debugging, and environment management.
- **JSON:** Learn structured data formats because many AI tools return structured outputs.
- **Docker optional:** Useful for packaging applications consistently, especially when moving from local development to deployment.

Example learning task: Write a Python script that accepts a text file, sends the content to an AI model, asks for a summary and action items, and saves the result as a JSON file. This one exercise combines file handling, prompting, APIs, and structured output.

1.7 Portfolio Roadmap

Your portfolio should prove that you can solve real problems with GenAI. It should not be a random collection of tutorials. Each project should have a clear user, business value, architecture, and measurable outcome.

- **Projects employers love:** RAG document assistant, support ticket classifier, AI meeting summarizer, resume-job matcher, policy Q&A bot, research assistant, and code documentation generator.
- **GitHub checklist:** Clear README, problem statement, setup instructions, architecture diagram description, screenshots, sample prompts, limitations, evaluation examples, and future roadmap.
- **Demo video tips:** Keep it under five minutes, show the problem first, demonstrate the workflow, explain the architecture briefly, and close with what you would improve next.
- **Documentation template:** Project title, use case, target user, tech stack, data flow, prompt examples, retrieval strategy, evaluation method, risks, and deployment notes.

Strong portfolio positioning example: "I built a RAG chatbot for HR policy documents that answers employee questions with source-grounded responses, uses chunking and embeddings for retrieval, and includes an evaluation set of 25 common employee questions."

1.8 AI Career Roadmap

GenAI careers are not limited to one job title. Different roles require different balances of technical depth, product thinking, communication, and domain expertise. Choose a path based on your strengths and experience.

- **AI Engineer:** Builds AI-powered applications, integrates models with software systems, and understands APIs, data pipelines, and deployment.
- **GenAI Engineer:** Specializes in LLM applications, RAG, agents, prompt design, evaluation, and production workflows.
- **Prompt Engineer:** Designs, tests, and optimizes prompts for business workflows, often working with content, operations, product, or support teams.
- **AI Product Manager:** Defines AI product strategy, user needs, success metrics, responsible AI requirements, and product roadmaps.
- **AI Consultant:** Helps organizations identify AI use cases, evaluate tools, design adoption plans, and manage change.
- **AI Solutions Architect:** Designs enterprise AI systems, including security, integration, scalability, governance, cloud architecture, and vendor selection.

If you are from a non-coding background, start with prompt engineering, AI product thinking, AI consulting, or domain-specific AI implementation. If you are comfortable with coding, move toward AI engineer, GenAI engineer, or solutions architect roles.

1.9 Certifications & Continuous Learning

Certifications can help structure learning and demonstrate commitment, but projects usually matter more than certificates alone. Choose certifications that match your career path and tools you expect to use.

- **Beginner certifications:** AI fundamentals, responsible AI, prompt engineering, and cloud AI basics.
- **Developer certifications:** Azure AI, AWS AI/ML, Google Cloud AI, Microsoft Generative AI Engineering, and GitHub Copilot learning paths.
- **Learning communities:** Hugging Face community, LangChain community, Microsoft Learn community, Kaggle, GitHub discussions, and local AI meetups.
- **Newsletters:** The Batch, Ben's Bites, Latent Space, TLDR AI, and vendor engineering blogs.
- **YouTube channels:** DeepLearning.AI, Microsoft Developer, Google Cloud Tech, IBM Technology, freeCodeCamp, and AI engineering creators.
- **Practice platforms:** Kaggle, GitHub, Hugging Face Spaces, Replit, Google Colab, Microsoft Learn, and coding challenge platforms for Python practice.

Continuous learning is essential because tools change quickly. Build a weekly habit: read one technical article, test one new tool feature, improve one portfolio project, and document one lesson learned.

2. 90-Day Learning Plan

This 90-day plan assumes you can study 5 to 8 hours per week. If you have more time, add extra project work. If you have less time, keep the same order but extend the timeline.

Month 1: Learn AI Fundamentals and Practice Prompting

- **Week 1:** Learn AI, ML, deep learning, LLMs, GenAI, tokens, context windows, hallucinations, and responsible AI basics.
- **Week 2:** Practice prompt writing for summarization, rewriting, classification, extraction, and ideation.
- **Week 3:** Learn prompt frameworks and compare outputs across ChatGPT, Claude, Gemini, and Perplexity.
- **Week 4:** Create a mini prompt portfolio with 15 reusable prompts for business, learning, coding, and research tasks.
- **Month 1 outcome:** You can explain GenAI concepts clearly and produce reliable prompts for common tasks.

Month 2: Build with LangChain, Learn RAG, and Work with Vector Databases

- **Week 5:** Learn Python basics required for GenAI projects, including functions, files, packages, and APIs.

- **Week 6:** Build a simple LLM-powered summarizer using an API and structured JSON output.
- **Week 7:** Learn RAG architecture: load, split, embed, store, retrieve, and generate.
- **Week 8:** Build a basic document Q&A system using LangChain and a vector database such as Chroma or FAISS.
- **Month 2 outcome:** You can build a working RAG prototype and explain how retrieval improves answer grounding.

Month 3: Complete Projects, Build Portfolio, and Prepare for Interviews

- **Week 9:** Choose two portfolio projects, such as an AI Resume Reviewer and a RAG Knowledge Base Search tool.
- **Week 10:** Improve the projects with better prompts, source grounding, error handling, and sample evaluations.
- **Week 11:** Create GitHub repositories with README files, screenshots, architecture explanations, and setup instructions.
- **Week 12:** Record short demos, prepare interview explanations, revise your resume, and practice explaining tradeoffs such as prompt engineering versus fine-tuning or keyword search versus vector search.

- **Month 3 outcome:** You have at least two demonstrable GenAI projects and can discuss your learning, architecture choices, limitations, and next steps with confidence.

Final Advice

To become job-ready, focus on depth, not just tool familiarity. Learn the concepts, build small applications, test outputs carefully, document your work, and explain tradeoffs in plain language. The most employable GenAI professionals are not only people who can use AI tools; they are people who can identify a real problem, design a responsible solution, build a working prototype, evaluate its quality, and communicate its value clearly.

3. GenAI Resource Library

A strong GenAI learner should build a personal resource library. The goal is not to collect hundreds of links, but to maintain a focused set of books, documentation, courses, code repositories, communities, and podcasts that support continuous learning. Use this section as a practical reference list and update it every month as tools and best practices change.

Books

- **For beginners:** Choose books that explain AI, machine learning, and LLMs in simple language before going deep into mathematics. Look for practical examples, diagrams, and business use cases.
- **For developers:** Read books that cover transformers, tokenization, attention, embeddings, fine-tuning, RAG, and LLM application development.
- **For product and business professionals:** Focus on books about AI strategy, responsible AI, automation, AI product management, and human-AI collaboration.
- **How to use books effectively:** Do not read passively. After each chapter, write a one-page summary, create three prompts related to the topic, and identify one small project idea.

Documentation

- **Model provider documentation:** Study OpenAI, Anthropic, Google Gemini, Meta, and Microsoft documentation to understand model capabilities, API usage, safety guidance, and pricing considerations.
- **Framework documentation:** Use LangChain, LlamaIndex, Semantic Kernel, and Hugging Face documentation when building applications.
- **Vector database documentation:** Review Chroma, FAISS, Pinecone, Weaviate, Milvus, and Azure AI Search documentation to understand indexing, similarity search, metadata filtering, and retrieval patterns.
- **Cloud documentation:** If you plan to work in enterprise environments, explore Azure AI, AWS Bedrock, Google Vertex AI, and related security or deployment documentation.
- **Practical habit:** Whenever you use a library, read the official quickstart first, then build one small example before watching tutorials.

Courses

- **Beginner courses:** Start with structured introductions to Generative AI, LLMs, prompt engineering, and responsible AI.
- **Hands-on developer courses:** Choose courses that include APIs, LangChain or LlamaIndex, RAG pipelines, vector databases, agents, evaluation, and deployment.

- **Cloud learning paths:** Use Microsoft Learn, Google Cloud Skills Boost, AWS Skill Builder, and other cloud learning platforms if your target role involves enterprise implementation.
- **Recommended learning approach:** Complete one beginner course fully before starting multiple advanced courses. Take notes, complete labs, and turn course exercises into portfolio projects.
- **Example:** A beginner can complete a Generative AI foundations course in Week 1, a prompt engineering course in Week 2, and a RAG mini-project course in Weeks 3 and 4.

GitHub Repositories

- **Microsoft Generative AI for Beginners:** A structured beginner-friendly course with lessons on GenAI concepts, prompt engineering, RAG, agents, application lifecycle, and responsible development.
- **Awesome Generative AI Guide:** A broad collection of GenAI research updates, interview resources, notebooks, tools, and free learning materials.
- **LLMs from Scratch:** Useful for learners who want to understand tokenization, attention, transformer architecture, and how LLMs work internally.
- **LLM Zoomcamp:** A practical learning path focused on building LLM applications, RAG systems, vector search, evaluation, and deployment.

- **Awesome LLM Apps:** Helpful for exploring practical applications such as RAG assistants, agents, voice tools, and multimodal workflows.
- **How to learn from GitHub:** Do not only star repositories. Clone one project, run it locally, change a feature, document what you learned, and publish your own adapted version with proper attribution.

Communities

- **Hugging Face community:** Useful for open-source models, datasets, Spaces, demos, and model experimentation.
- **LangChain and LlamaIndex communities:** Useful for RAG, agents, application architecture, and troubleshooting implementation issues.
- **Kaggle:** Good for data science practice, notebooks, competitions, and learning from public examples.
- **GitHub discussions:** Useful for following project updates, issues, implementation patterns, and open-source contribution opportunities.
- **LinkedIn and local meetups:** Helpful for networking, sharing portfolio work, learning from practitioners, and finding project ideas.
- **Community habit:** Post one learning update each week. For example, share a screenshot of a project, a short explanation of a RAG concept, or a lesson learned from debugging an AI application.

Podcasts

- **Technical podcasts:** Choose shows that discuss LLM engineering, model evaluation, AI infrastructure, open-source models, agents, and production deployment.
- **Business podcasts:** Listen to episodes about AI adoption, responsible AI, AI strategy, productivity, product management, and organizational transformation.
- **Research-focused podcasts:** Use these to understand emerging topics such as multimodal AI, reasoning models, synthetic data, evaluation methods, and agentic workflows.
- **How to use podcasts:** Listen during commute or exercise, but capture notes afterward. Write down one new term, one tool, and one project idea from each episode.
- **Example podcast activity:** After listening to an episode on AI agents, create a small workflow diagram showing how an agent chooses tools, calls APIs, checks results, and returns an answer.

4. Beginner Checklist

Use this checklist as a one-page readiness tracker. Mark an item only when you can explain it in your own words and show practical evidence such as notes, code, a demo, or a portfolio link.

- □ **Understand LLMs:** I can explain what large language models are, how tokens and context windows work, why hallucinations happen, and why human review is important.
- □ **Learn prompting:** I can write prompts with role, context, task, constraints, examples, and output format. I can also test multiple prompt versions and compare results.
- □ **Build a RAG application:** I can build a simple application that loads documents, chunks text, creates embeddings, retrieves relevant content, and generates source-grounded answers.
- □ **Use LangChain:** I can create a basic chain, connect a model to a retriever, use prompt templates, and explain the role of chains, tools, and agents.
- □ **Work with a vector database:** I can store embeddings, run similarity search, use metadata filters, and explain when vector search is better than keyword search.

- □ **Complete 3–5 portfolio projects:** I have built projects such as a resume reviewer, document Q&A assistant, RAG chatbot, support bot, or research assistant.

- □ **Publish projects on GitHub:** Each project has a clear README, setup steps, screenshots, architecture explanation, sample prompts, limitations, and future improvements.

- □ **Earn a GenAI certification:** I have completed a relevant GenAI certification or learning path that supports my target role and validates my knowledge.

Checklist example: If you built a RAG application but cannot explain chunking, embeddings, retrieval, and grounding, keep the item open. The purpose of the checklist is not just completion; it is confidence, clarity, and proof of skill.

Next Steps

After completing the beginner roadmap, the next goal is to move from guided learning to independent execution. This means building more realistic projects, contributing to communities, exploring advanced topics, and choosing a structured certification path if you want formal validation.

- **Continue building projects:** Improve your existing projects instead of always starting new ones. Add better data handling, user interface improvements, evaluation test cases, logging, deployment, and documentation.
- **Join the GenAI community:** Participate in discussions, ask questions, answer beginner questions, attend meetups, share demos, and follow builders who publish practical implementation examples.
- **Explore advanced topics:** Learn model evaluation, LLMOps, fine-tuning, function calling, AI agents, MCP, multimodal AI, security, privacy, responsible AI, guardrails, and enterprise deployment patterns.
- **Learn with GSDC's Certified Generative AI Professional program:** Use the program as a structured path to validate your understanding of GenAI concepts, practical applications, prompt engineering, responsible AI, and business use cases. It can be especially useful for professionals who want a formal credential alongside portfolio projects.

A practical next-step plan is to select one advanced project and one certification goal. For example, build a RAG knowledge-base assistant for a real business domain, publish it on

GitHub with a demo video, then complete a GenAI certification to strengthen your resume. This combination shows both hands-on capability and structured learning.

Final recommendation: Treat GenAI as an ongoing professional skill, not a one-time course. Keep building, keep testing tools, keep documenting your work, and keep connecting AI capabilities to real business problems. The learners who become job-ready fastest are those who consistently turn concepts into working prototypes and explain their decisions clearly.

GSDC
Global Skill Development Council

CERTIFIED GENERATIVE AI PROFESSIONAL

GET GLOBAL RECOGNITION AND STAND OUT AS A LEADER IN THE FIELD OF GENERATIVE AI .

ABOUT GSDC CERTIFICATION

EBOOK

Extensive and exclusive Ebook created by world's experts to help you with understanding core concepts.

CREATED BY EXPERTS

GSDC certifications are created and authored by world's leading experts in the field.

LEARNING MATERIALS

Get access to learning materials such as videos, ebooks, templates, and practice exams, which will help you clear the certification exam.

LEARNING OBJECTIVE

- Contribute to the dynamic field of artificial intelligence.
- Validate practical application skills in Gen AI.
- Propel advancements in generative AI technology.
- Exhibit practical expertise in generative AI.

Enroll now with the code **LEARN20** To avail **20%** discount

Enroll Now

www.gsdcouncil.org