**Cloud Native Generative AI Developer / Full Stack Generative AI Developer [GenDev]**

**Course Duration: 6 Months**

**Course Technologies Stack:**

**1. Frontend GUI Stack**

* TypeScript
* NextJS
* React
* Tailwind CSS (Shadcn UI, NextUI, Dev UI)

**2. Backend API Stack**

* Python
* FastAPI
* Pydantic
* SQLModel
* PostgresSQL (Neon, Vercel, Planetscale)
* Kafka

**3. Backend Cloud Stack**

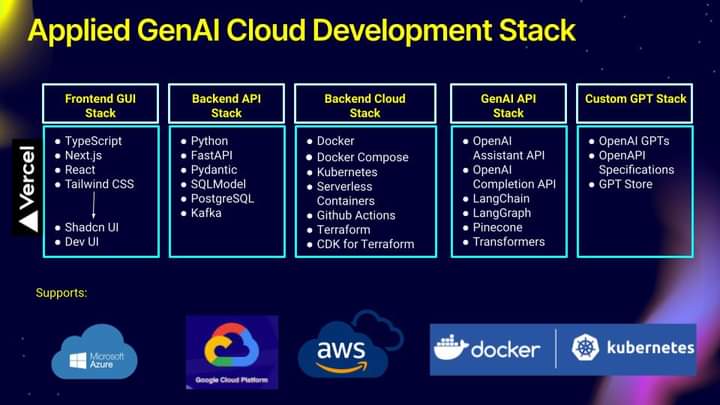
* Docker
* Docker Compose
* Kubernetes
* Serverless Containers
* Github Actions
* CD / CI
* Terraform
* CDK for Terraform

**4. Generative AI API Stack**

* Open AI Assistant API
* Open AI Completion API
* Google Gemini API
* Langchain
* Hugging Faces Models
* PineCone (Vector DB)
* Transformers

**5: Custom GPT Stack**

* Open AI GPTs
* OpenAPI Specifications
* GPT Store



**LEARNING PATH OF GEN AI [GenDev] STACKS:**

**Frontend Stack: Front-end Development using Next.js and TypeScript**

In this stack, we will learn to build and deploy state-of-the-art web user interfaces using TypeScript 5, React 18, Next.js 14, and Tailwind CSS 3. We will use beautifully designed and customizable Shadcn UI React components, which are built with Headless Radix UI and Tailwind CSS to speed up our development life cycle. Using Headless CMSs will also be covered. We will also learn to use Vercel AI SDK, an open-source library for building AI-powered user interfaces. The quarter will end with you learning to deploy these UI apps on Vercel Cloud and CDN.

**Backend Stack: TypeScript and Python Programming**

In the stack, we will learn the two most used programming languages in GenAI Application Development, TypeScript for User interfaces, and Python for Application Programming Interfaces (APIs). We will cover both functional and object-oriented paradigms by using Typescript programming and Python programming

**Backend Cloud & Generative AI API Stack: API Design, Development, and Deployment using FastAPI, Containers, and OpenAPI Specifications**

An API-as-a-Product is a type of Software-as-a-Service that monetizes niche functionality, typically served over HTTP. OpenAI APIs are themselves this kind of service. An application programming interface economy, or API economy, refers to the business structure where APIs are the distribution channel for products and services. In this stack we will learn to develop APIs not just as a backend for our frontend but also as a product itself. In this model, the API is at the core of the business's value.

We will be using Python-based FastAPI as our core library and Pedantic, SQLModel and PostgreSQL databases for API development. Docker Containers will be our fundamental building block for development, testing, and deployment. For local development, we will be using Docker Compose and DevPod, which is Dev-Environments-As-Code, for testing Pytest and Testcontainers, and for deployment Google Cloud Run, Azure Container Service, and Kubernetes. We will be using Terraform as our Infrastructure as Code (IaC) tool. OpenAI Chat GPT 4, Google Gemini APIs, and Langchain will be used to build these API-as-a-Products.

**Custom GPT Stack: Custom GPT and GPT Actions**

In this stack, we will learn to create custom versions of ChatGPT that combine instructions, extra knowledge, and any combination of skills. We will also learn how to build a GPT action that intelligently calls our APIs using the OpenAPI Specifications. We will also cover strategies for selling our custom GPTs in the GPT stores.

**Common Questions (FAQs) with Detailed Answers**

1. **What is Cloud Native Generative AI Development?**

Cloud Native Generative AI Development (GenDev) is the application of generative AI technologies to solve real-world problems in the cloud.

* Generative AI is a type of artificial intelligence that can create new data or content from existing data.
* Cloud computing is the delivery of computing services—including servers, storage, databases, networking, software, analytics, and intelligence—over the Internet (“the cloud”).

By combining generative AI with cloud computing, businesses can solve a variety of problems, such as:

* Creating personalised experiences for customers
* Automating tasks
* Improving decision-making
* Detecting fraud
* Developing new products and services

The potential applications of cloud-applied generative AI are endless. As generative AI and cloud computing continue to develop, we can expect to see even more innovative and groundbreaking uses for this technology.

1. **How valuable are the Cloud Native Generative AI developers?**

Developers with expertise in Cloud Native Generative AI were in extremely high demand due to the increasing adoption of GenAI technologies across various industries. However, the supply of developers skilled specifically in this niche area might not have been as abundant compared to more generalised AI or cloud computing roles.

The demand for AI developers, especially those proficient in applying generative AI techniques within cloud environments, has been rising due to the growing interest in using AI for creative applications, content generation, image synthesis, natural language processing, and other innovative purposes.

According to some sources, the average salary for a Cloud Native Generative AI developer in the global market is around $150,000 per year. However, this may vary depending on the experience level, industry, location, and skills of the developer. For example, a senior Cloud Applied Generative AI developer with more than five years of experience can earn up to $200,000 per year. A Cloud Applied Generative AI developer working in the financial services industry can earn more than a developer working in the entertainment industry. A Cloud Applied Generative AI developer working in New York City can earn more than a developer working in Dubai. In general, highly skilled AI developers, especially those specialising in applied generative AI within cloud environments, tend to earn competitive salaries that are often above the average for software developers or AI engineers due to the specialised nature of their skills. Moreover, as generative AI technology becomes more widely adopted and integrated into various products and services, the demand for Cloud Applied Generative AI developers is likely to increase.

Therefore, Cloud Native Generative AI developers are valuable professionals who have a bright future ahead of them. They can leverage their creativity and technical skills to create innovative solutions that can benefit various industries and domains. They can also enjoy very competitive salary and career growth opportunities.

1. **What is the potential for Cloud Native Generative AI Developers to start their own companies?**

Cloud Native Generative AI Developers have a significant potential to start their own companies due to several factors:

1. **Emerging Field:** Generative AI, particularly when applied within cloud environments, is still an evolving field with immense potential for innovation. Developers who understand the intricacies of both generative AI techniques and cloud technologies can identify unique opportunities to create novel products, services, or solutions.
2. **Market Demand:** There is a growing demand for AI-driven applications, especially those that involve generative capabilities such as image generation, content creation, style transfer, etc. Developers with expertise in this area can leverage this demand to create specialized products that cater to specific industries or consumer needs.
3. **Innovation and Differentiation:**The ability to develop unique and innovative solutions using generative AI in the cloud can set apart these developers' startups from more conventional companies. They can explore new ways of generating content, enhancing user experiences, or solving complex problems with AI-generated solutions.
4. **Access to Cloud Resources:** Cloud platforms provide scalable and cost-effective resources that are crucial for AI development. Developers starting their own companies can leverage cloud services to access powerful computing resources, storage, and AI-related services without significant upfront investment.
5. **Entrepreneurial Opportunities:** Developers with entrepreneurial spirit and a deep understanding of AI technologies can identify gaps in the market and build startups to fill those gaps. They can create platforms, tools, or services that simplify the adoption of generative AI for businesses or developers.
6. **Collaboration and Partnerships:** These developers can collaborate with other experts in AI, domain specialists, or businesses to create innovative solutions or explore new application areas for generative AI in the cloud.

However, starting a company, especially in a specialized field like Cloud Native Generative AI, requires more than technical expertise. It also demands business acumen, understanding market needs, networking, securing funding, managing resources effectively, and navigating legal and regulatory landscapes.

Successful entrepreneurship in this domain involves a combination of technical skills, innovation, a deep understanding of market dynamics, and the ability to transform technical expertise into viable products or services that address real-world challenges or opportunities.

Developers aspiring to start their own companies in the Cloud Native Generative AI space can do so by conducting thorough market research, networking with industry experts, building a strong team, and developing a clear business plan that highlights the unique value proposition of their offerings.

To sum up, the potential for Cloud Applied Generative AI Developers to start their own companies is high.

* Generative AI is a rapidly growing field with a high demand for skilled professionals.
* The Certified Generative AI (GenDev) Developer and Engineering Program provides students with the skills and knowledge they need to develop and apply cutting-edge generative AI technologies.
* The program also teaches students how to start and run a successful business.
* Graduates of the program will be well-positioned to start their own companies and capitalise on the growing demand for generative AI solutions.

1. **Why do we have to learn two programming languages?**

You are learning two programming languages in the Cloud Native Generative AI Developer Program because they are both essential for developing smart applications with GenAI.

* **TypeScript (Which is a superset of JavaScript) is used for building user interfaces**, and it is a relatively new programming language that is gaining popularity due to its strong typing system and its ability to be used with JavaScript, React, and Next.js.
* **Python is used for developing application programming interfaces (APIs)**, and it is a more established programming language that is known for its versatility and ease of use. It is also the go-to language for developing AI systems.

1. **What is the difference between OpenAI Completion API, OpenAI Assistant API, Google Gemini Multi-Modal API, and LangChain?**

The difference between OpenAI Completion API, OpenAI Assistant API, Google Gemini Multi-Modal API, and LangChain is that they are different ways of using artificial intelligence to generate text, images, audio, and video based on some input, but they have different features and applications. Here is a summary of each one:

**OpenAI Completion API** is the most fundamental OpenAI model that provides a simple interface that’s extremely flexible and powerful. You give it a prompt and it returns a text completion, generated according to your instructions. You can think of it as a very advanced autocomplete where the language model processes your text prompt and tries to predict what’s most likely to come next. The Completion API can be used for various tasks such as writing stories, poems, essays, code, lyrics, etc. It also supports different models with different levels of power suitable for different tasks.

**OpenAI Assistant API** is an interface to OpenAI's most capable model (gpt-4) and their most cost-effective model (gpt-3.5-turbo). It provides a simple way to take text as input and use a model like gpt-4 or gpt-3.5-turbo to generate an output. The Assistant API allows you to build AI assistants within your applications. An Assistant has instructions and can leverage models, tools, and knowledge to respond to user queries. The Assistant API currently supports three types of tools: Code Interpreter, Retrieval, and Function calling.

**Google Gemini Multi-Modal API** is a new series of foundational models built and introduced by Google. It is built with a focus on multimodality from the ground up. This makes the Gemini models powerful against different combinations of information types including text, images, audio, and video. Currently, the API supports images and text. Gemini has proven by reaching state-of-the-art performance on the benchmarks and even beating the ChatGPT and the GPT4-Vision models in many of the tests.

There are three different Gemini models based on their size, the Gemini Ultra, Gemini Pro, and Gemini Nano in decreasing order of their size.

**LangChain** is a platform that allows you to interact with various language models from different providers such as OpenAI, Google Gemini, Hugging Face Transformers, etc. You can use LangChain to create applications that leverage the power of natural language processing without having to deal with the complexity of APIs or SDKs. LangChain provides a user-friendly interface that lets you choose the model you want to use, customize the parameters you want to apply, and see the results in real-time.

1. **Why don't we use Flask or Django for API development instead of FastAPI?**

* **FastAPI is a newer and more modern framework than Flask or Django.** It is designed to be fast, efficient, and easy to use. FastAPI is also more scalable than Flask or Django, making it a better choice for large-scale projects.
* **FastAPI is also more feature-rich than Flask or Django.** It includes several built-in features that make it easy to develop APIs, such as routing, validation, and documentation.
* **Overall, FastAPI is a better choice for API development than Flask or Django.** It is faster, more scalable, and more feature-rich.

1. **Why do we need to learn Cloud technologies in a Generative AI program?**

Cloud technologies are essential for developing and deploying generative AI applications because they provide a scalable and reliable platform for hosting and managing complex workloads.

* Cloud computing offers a vast pool of resources that can be provisioned on demand, which is ideal for generative AI applications that can be computationally intensive.
* Cloud providers offer a wide range of services that can be used to support generative AI applications, including storage, computing, networking, and machine learning.
* Cloud services are typically more cost-effective than on-premises infrastructure, which can be a significant advantage for generative AI applications that are often used for large-scale projects.

The Certified Generative AI (GenDev) Developer Program teaches you how to use a variety of cloud services, including Google Cloud Run, Azure Container Apps, and Kubernetes, to deploy your applications to the cloud. You will also learn how to use **Docker containers** to package and deploy your applications, and how to use Terraform to manage your cloud infrastructure.

By the end of the program, you will be able to:

* Use Docker containers to package and deploy your applications
* Develop and deploy generative AI applications to the cloud
* Manage your cloud infrastructure using Terraform

1. **What is the purpose of Docker Containers and what are the benefits of deploying them with Docker Compose, Google Cloud Run, Azure Container Apps, and Kubernetes?**

* **Docker Containers** are a way to package software into a single unit that can be run on any machine, regardless of its operating system. It is used to create a Dockerfile, which is a text file that describes how to build a Docker image. The image is then used to create a container, which is a running instance of the image. This makes them ideal for deploying applications on a variety of platforms, including cloud-based services.
* **Docker Compose** is a tool provided by Docker that allows you to define and manage multi-container Docker applications locally. It enables you to use a YAML file to configure the services, networks, and volumes needed for your application's setup. With Docker Compose, you can describe the services your application requires, their configurations, dependencies, and how they should interact with each other, all in a single file. This makes it easier to orchestrate complex applications locally composed of multiple interconnected containers.
* **Google Cloud Run** is a serverless computing platform that allows you to run stateless containers that are invocable via HTTP requests. It is fully managed, so you don't need to worry about provisioning or managing servers.
* **Azure Container Apps** is a serverless platform from Microsoft that allows you to maintain less infrastructure and save costs while running containerized applications. Instead of worrying about server configuration, container orchestration, and deployment details, Container Apps provides all the up-to-date server resources required to keep your applications stable and secure.
* **Kubernetes** is a container orchestration system that automates the deployment, scaling, and management of containerized applications. It allows you to run multiple containers on a single machine or across multiple machines. It is an open source and can be deployed in your data center or the cloud.

1. **Why do we need to learn Web development technologies in a Generative AI program?**

Web development technologies are essential for developing and deploying generative AI applications because they allow you to **create user interfaces** that allow users to interact with your applications.

The Certified Generative AI (GenEng / GenDev) Developer and Engineering Program teaches you how to use cutting-edge web development technologies, including TypeScript, React, Next.js, and Tailwind CSS, to build and deploy state-of-the-art web user interfaces. You will also learn how to use Vercel AI SDK, an open-source library for building AI-powered user interfaces.

1. **What is the purpose of learning to develop APIs in a Generative AI program?**

APIs (Application Programming Interfaces) are used to connect different software applications and services together. They are the building blocks of the internet and are essential for the exchange of data between different systems.

In the third quarter of the Certified Generative AI (GenEng) Developer and Engineering Program, students will learn to develop APIs not just as a backend for their front end but also as a **product** itself. In this model, the API is at the core of the business's value.

* APIs are used to make it possible for different software applications to communicate with each other.
* APIs are used to access data from a remote server.
* APIs are used to create new services or applications that are integrated with existing systems.
* APIs are used to improve the security of applications by providing a way to control access to data.
* By learning to develop APIs, students will gain the skills necessary to create powerful and efficient software applications that can be used to solve a variety of business problems.

1. **What is the purpose of using Python-based FastAPI and related technologies?**

In the third quarter of the Certified Generative AI (GenEng / GenDev) Developer and Engineering Program, students will learn how to use Python-based FastAPI as a core library for API development.

* FastAPI is a high-performance, lightweight, and easy-to-use framework for building APIs.
* It is designed to be fast, scalable, and secure.
* FastAPI is compatible with a wide range of programming languages and frameworks, making it a good choice for developers with different skill sets.
* Students will also learn about the following related technologies:
* **Pedantic:** Pedantic is a Python library that helps to improve the quality of your code by checking for errors and potential problems.
* **SQLModel:** SQLModel is a Python library that provides an object-relational mapping (ORM) layer for working with databases.
* **PostgreSQL:** PostgreSQL is a free and open-source relational database management system (RDBMS) that can be used for development. Highly scalable database systems compatible with it have also been deployed by all the major cloud platforms.

1. **What does the API-as-a-Product model entail?**

API-as-a-Product is a type of Software-as-a-Service that monetizes niche functionality, typically served over HTTP. In this model, the API is at the core of the business's value. The API-as-a-Product model is different from the traditional API model, where APIs are used as a means to access data or functionality from another application. In the API-as-a-Product model, the API itself is the product that is being sold.

The benefits of the API-as-a-Product model include:

* **Increased flexibility:** APIs can be used to access data or functionality from any application, regardless of the underlying platform or technology. This gives businesses greater flexibility in how they integrate APIs into their applications.
* **Reduced development costs:** APIs can be reused by multiple applications, which can save businesses the time and expense of developing their custom APIs.
* **Improved scalability:** APIs can be scaled up or down as needed, which makes them well-suited for businesses with fluctuating or unpredictable traffic demands.
* **Enhanced security:** APIs can be more secure than traditional methods of data exchange, as they can be protected by a variety of security measures, such as encryption and access control.

1. **What are the benefits of using Docker Containers for development, testing, and deployment?**

Docker Containers are a fundamental building block for development, testing, and deployment because they provide a consistent environment that can be used across different systems. This eliminates the need to worry about dependencies or compatibility issues, and it can help to improve the efficiency of the development process. Additionally, Docker Containers can be used to isolate applications, which can help to improve security and make it easier to manage deployments.

1. **Why in this program are we not learning to build LLMs ourselves? How difficult is it to develop an LLM like ChatGPT 4 or Google’s Gemini?**

Developing an LLM like ChatGPT 4 or Google Gemini is extremely difficult and requires a complex combination of resources, expertise, and infrastructure. Here's a breakdown of the key challenges:

**Technical hurdles:**

**Massive data requirements:** Training these models requires an immense amount of high-quality data, often exceeding petabytes. Compiling, cleaning, and structuring this data is a monumental task.

**Computational power:** Training LLMs demands incredible computational resources, like high-performance GPUs and specialised AI hardware. Access to these resources and the ability to optimise training processes are crucial.

**Model architecture:** Designing the LLM's architecture involves complex decisions about parameters, layers, and attention mechanisms. Optimising this architecture for performance and efficiency is critical.

**Evaluation and bias:** Evaluating the performance of LLMs involves diverse benchmarks and careful monitoring for biases and harmful outputs. Mitigating these biases is an ongoing research challenge.

**Resource and expertise:**

**Team effort:** Developing an LLM like ChatGPT 4 or Google Gemini requires a large team of experts across various disciplines, including AI researchers, machine learning engineers, data scientists, and software developers.

**Financial investment:** The financial resources needed are substantial, covering costs for data acquisition, hardware, software, and talent. Access to sustained funding is critical.

Additionally:

**Ethical considerations:** LLMs raise ethical concerns like potential misuse, misinformation, and societal impacts. Responsible development and deployment are crucial.

**Rapidly evolving field:** The LLM landscape is constantly evolving, with new research, models, and benchmarks emerging. Staying abreast of these advancements is essential.

Therefore, while ChatGPT 4 and Google Gemini have made impressive strides, developing similar LLMs remains a daunting task accessible only to a handful of organizations with the necessary resources and expertise.

In simpler terms, it's like building a skyscraper of knowledge and intelligence. You need the right materials (data), the right tools (hardware and software), the right architects (experts), and a lot of hard work and attention to detail to make it stand tall and function flawlessly.

Developing similar models would be a daunting task for individual developers or smaller teams due to the enormous scale of resources and expertise needed. However, as technology progresses and research findings become more accessible, it might become incrementally more feasible for a broader range of organizations or researchers to work on similar models, albeit at a smaller scale or with fewer resources. At that time we might also start to focus on developing LLMs ourselves.

To sum up, the focus of the program is not on LLM model development but on applied Cloud GenAI Engineering (GenEng), application development, and fine-tuning of foundational models. The program covers a wide range of topics including TypeScript, Python, Front-end Development, GenAI, API, Database, Cloud Development, and DevOps, which will give students a comprehensive understanding of generative AI and prepare them for careers in this field.

1. **What are Custom GPTs?**

"Custom GPTs" refers to specialised versions of the Generative Pre-trained Transformer (GPT) models that are tailored for specific tasks, industries, or data types. These custom models are adapted from the base GPT architecture, which is a type of language model developed by OpenAI. Custom GPT models are trained or fine-tuned on specific datasets or for particular applications, allowing them to perform better in those contexts compared to the general-purpose models.

Here are some examples of what custom GPT models might be used for:

1. **Industry-Specific Needs**: A custom GPT for legal, medical, or financial industries could be trained on domain-specific texts to understand and generate industry-specific language more accurately.

2. **Language and Localization**: Models can be customised for different languages or dialects that might not be well-represented in the training data of the base model.

3. **Company-Specific Applications**: Organisations might develop a custom GPT model trained on their own documents and communications to assist with internal tasks like drafting emails, generating reports, or providing customer support.

4. **Educational Purposes**: Educational institutions might develop custom GPTs trained on educational material to assist in creating teaching materials or providing tutoring in specific subjects.

5. **Creative Writing and Entertainment**: Custom models could be trained on specific genres of literature or scripts to assist in creative writing or content creation.

6. **Technical and Scientific Research**: A custom GPT model could be trained on scientific literature to assist researchers in summarising papers, generating hypotheses, or even drafting new research.

These custom models are created through a process of fine-tuning, where the base GPT model is further trained (or 'fine-tuned') on a specific dataset. This process allows the model to become more adept at understanding and generating text that is relevant to the specific use case. Fine-tuning requires expertise in machine learning and natural language processing, as well as access to relevant training data.

1. **What are Actions in GPTs?**

Actions are a way to connect custom GPTs to external APIs, allowing them to access data or interact with the real-world. For example, you can use actions to create a GPT that can book flights, send emails, or order pizza. **Actions are defined using the OpenAPI specification**, which is a standard for describing APIs. You can import an existing OpenAPI specification or create a new one using the GPT editor.

**Generative AI Prompt Engineering Professional**

**Course Duration: 2 Months**

**Course Technologies Stack:**

**1. OpenAI / Other GPTs Stack**

* ChatGPT
* Gemini
* Bard
* Meta AI

**2. Custom GPT Stack**

* Open AI GPTs
* OpenAPI Specifications
* GPT Store

**Course Structure:**

The course will be divided into three core modules:

**Module 1: Foundations of Generative AI (20% of course time)**

* **What is Generative AI?**
  + Understanding the core concepts of generative models (e.g., GANs, VAEs)
  + Applications of generative AI across various industries (e.g., image generation, text creation, music composition)
* **Demystifying Machine Learning:**
  + A high-level overview of machine learning concepts relevant to generative AI (e.g., supervised vs. unsupervised learning)
  + The role of training data in shaping model outputs

**Module 2: The Art and Science of Prompt Engineering (60% of course time)**

* **Anatomy of a Prompt:**
  + Breaking down the different elements of a prompt (e.prompt itself, instructions, desired style)
  + Exploring various prompt templates for different tasks (e.g., creative writing, code generation, factual text summarization)
* **Crafting Effective Prompts:**
  + Techniques for writing clear, concise, and informative prompts
  + The power of language: how word choice and phrasing influence model outputs
* **Advanced Prompt Engineering Strategies:**
  + Zero-shot and Few-shot Prompting: Utilizing minimal or no examples
  + Chain-of-Thought Prompting: Guiding the model's reasoning process
  + Leveraging Generated Knowledge for Improved Prompts: Using the model's own outputs to refine your prompts

**Module 3: Hands-on Exploration and Applications (20% of course time)**

* **Working with Generative AI Platforms:**
  + Introduction to popular generative AI platforms and their user interfaces
  + Uploading, editing, and running prompts on different platforms
* **Experimentation and Refinement:**
  + Hands-on exercises to practice crafting prompts for various tasks
  + Iterative process of refining prompts based on the generated outputs
* **Exploring the Creative Potential:**
  + Brainstorming applications of generative AI in your field
  + Experimenting with creative prompts for artistic expression (e.g., writing different genres, composing music)
* **Ethical Considerations:**
  + Potential biases in generative models and their outputs
  + Responsible use of generative AI and avoiding misuse

**Course Duration:** (Adjust based on desired depth)

* 8 Weeks (2 hours per week)

**Course Structure:**

The course will combine theoretical knowledge with practical exercises, allowing you to experiment with generative AI models and refine your prompt engineering skills.

**Modules:**

* **Module 1: Introduction to Generative AI (Week 1)**
  + What is generative AI?
  + Different types of generative models (GANs, VAEs, Transformers)
  + Applications of generative AI in various fields (text generation, image creation, music composition)
* **Module 2: Unveiling the Power of Prompts (Week 2)**
  + Demystifying prompts and their role in guiding generative models
  + The anatomy of a prompt: instructions, style, and desired outcome
  + Understanding how prompts influence the output format and content
* **Module 3: Crafting Effective Prompts (Week 3 & 4)**
  + Techniques for writing clear, concise, and informative prompts
  + Leveraging different prompt structures: Simple prompts, Few-shot learning prompts, Chain-of-thought prompts
  + Exploring advanced prompt engineering methods: Zero-shot learning, Persona prompting
* **Module 4: Hands-on Experimentation with Generative AI Models (Week 5 & 6)**
  + Introduction to popular generative AI platforms and tools (playground access)
  + Practical exercises in prompt design and experimentation with different models (text, image, code generation)
  + Iterative refinement of prompts based on model outputs
* **Module 5: Applications and Use Cases (Week 7)**
  + Exploring real-world applications of prompt engineering in various industries
  + Brainstorming creative uses of generative AI for content creation, design, and problem-solving
  + Showcasing successful case studies of prompt-driven AI projects
* **Module 6: Ethical Considerations and Future of Prompt Engineering (Week 8)**
  + Discussing potential biases and ethical concerns in generative AI
  + Exploring responsible prompt design practices to mitigate risks
  + Examining the evolving landscape of prompt engineering and its future potential

**Assessment:**

* Weekly quizzes to test understanding of key concepts
* Hands-on assignments involving prompt design and experimentation with generative AI models
* Final project where students apply their knowledge to a specific creative or practical task using generative AI

**Additional Resources:**

* Throughout the course, students will be provided with supplementary materials like research papers, articles, and tutorials for further exploration.
* Links to relevant generative AI platforms and tools will be shared for ongoing practice.

**Instructor:**

* An expert in generative AI and prompt engineering with experience in applying these technologies.

This course outline provides a comprehensive framework for learning prompt engineering. You can adjust the depth and duration based on your audience and desired learning outcomes. Remember to include engaging activities and case studies to make the learning process interactive and inspiring.