**Title: Building Agentic AI with Custom GPTs: A Game Changer for Everyone**

**Summary:**

Agentic AI represents a significant advancement in artificial intelligence, building upon the foundation laid by generative AI. While programming has traditionally been the domain of computer scientists, the advent of custom GPTs allows anyone to create powerful agentic AI systems without needing to write code. A custom GPT is essentially a version of ChatGPT that has been customized to exhibit agentic capabilities. This means it can perform tasks autonomously by using a set of tools connected to real-world systems.

**Key Concepts and Examples:**

1. **Custom GPT as an Agent:**
   * A custom GPT is more than just a conversational AI; it is an agent that can be programmed in natural language. By describing the tools it can use and connecting those tools to real systems, the GPT can automate tasks for work, life, or business.
   * **Example**: Imagine creating a custom GPT that can manage your calendar, send emails, and automate tasks based on your instructions—all without writing a single line of code. This GPT is programmed through natural language to understand your needs and execute tasks using the tools you provide.
2. **Natural Language Programming:**
   * One of the most exciting aspects of agentic AI in custom GPTs is the ability to "program" them using natural language. This approach democratizes the creation of AI agents, allowing non-programmers to build powerful systems that previously required coding expertise.
   * **Example**: A small business owner could use a custom GPT to automate their customer service process. By describing how to handle different types of customer inquiries, the GPT can autonomously respond to emails, track orders, and even escalate issues when necessary.
3. **Integration with Tools like Zapier:**
   * The real power of agentic AI comes when it is integrated with automation tools like Zapier. This integration allows the GPT to perform real-world actions, such as triggering workflows, sending notifications, or updating databases, all through simple natural language commands.
   * **Example**: By connecting a custom GPT to Zapier, a user could automate their entire sales process. The GPT could manage lead generation, follow up with potential clients, and update the CRM—all automatically, saving time and reducing manual effort.

**Conclusion:**

The combination of agentic AI and custom GPTs marks a revolutionary shift in how AI can be used by everyone, not just programmers. By leveraging natural language programming and integrating with tools like Zapier, users can create sophisticated automations that enhance productivity and simplify complex tasks. This course will guide you through the concepts and practical steps needed to build your own custom GPTs, empowering you to harness the full potential of agentic AI.

**Programming a GPT**

**Title: Programming Generative AI: Unlocking Customization and Control**

**Explanation:**

To fully harness the power of generative AI, it's important to think about programming it. While this might sound intimidating if you're not a software engineer, it's actually quite simple and can significantly enhance how AI responds to your needs.

**What Does It Mean to Program Generative AI?**

Programming generative AI involves giving it instructions on how to behave in the future. Instead of just giving it a command and then adjusting the response afterward, you can set up rules or guidelines that shape how the AI will respond to future inputs.

**Example 1: Setting the Tone**

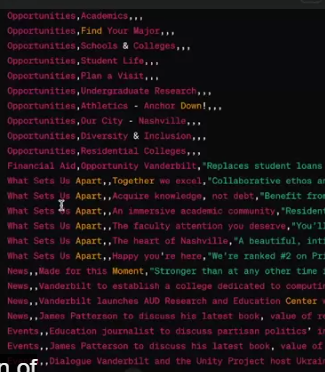
* **Scenario**: You tell ChatGPT, "I'm going to the grocery store." The AI responds in a neutral tone: "Do you need help making a shopping list?" But what if you want the AI to be more enthusiastic?
* **Process**:
  + You could adjust the response by telling the AI, "Redo your last response and make it really excited." But a more efficient way is to program the AI to always respond in an excited tone for future interactions.
  + You instruct the AI: "From now on, respond to every message with a super excited and happy tone." Now, when you say, "I'm going to the grocery store," the AI responds with something like, "Wow, that sounds like an epic journey!"
* **Outcome**: The AI has been programmed to maintain a specific tone in all future responses, making it more aligned with the desired communication style.
* So the first step is to set the tone of AI from the most first time when you start programming GPT. Basically yeh bi hum usko ek tarah say program kr rhay hain .

**Example 2: Programming Background Knowledge**

* **Scenario**: You ask the AI to give you unique ideas for using generative AI. Without knowing anything about you, the AI provides generic suggestions.
* **Process**:
  + To make the AI's responses more personalized, you can program it with background knowledge. You might say, "From now on, remember the following things: My name is Jules White, I'm a Professor in Computer Science at Vanderbilt University, and I teach a class on Prompt Engineering."
  + Now, when you ask for ideas, the AI can generate suggestions tailored to your specific roles and interests, such as "Nashville-themed AI-generated educational content" or "AI-driven enterprise innovation workshops."
* **Outcome**: The AI now uses the background knowledge you provided to deliver more relevant and personalized responses.

**Example 3: Changing the Default Behavior**

* **Scenario**: You want to transform how the AI processes information. For instance, you want it to turn any input into structured data like a CSV file, which can be used in tools like Excel.
* **Process**:
  + You instruct the AI, "From now on, turn everything I give you into CSV format."
  + When you input a webpage’s unformatted information, the AI now converts it into structured data instead of simply summarizing the content.
  + CSV format means jisme output will be segregated/separated or we can say that output is in tabular format.
* **Outcome**: The AI’s behavior has been reprogrammed to process and output data in a specific format, making it more useful for tasks like data analysis.



Output by Gpt in CSV format.

**Key Takeaways:**

1. **Programming Tone**: You can control the tone of AI responses, making them formal, excited, neutral, or any other style that suits your needs.
2. **Background Knowledge**: By programming the AI with specific information about you, your business, or your organization, you ensure it provides more relevant and informed responses.
3. **Custom Behavior**: You can change how the AI processes inputs and generates outputs by setting up instructions that apply to all future interactions.
4. **Future-Oriented Programming**: The phrase "from now on" is powerful because it tells the AI to apply certain rules or behaviors to all future messages, allowing you to create a customized and predictable AI interaction.

**Custom Instructions**

To effectively utilize generative AI like ChatGPT, it's important to customize it in a way that suits your needs, especially if you want it to remember specific instructions over time. Typing in instructions every time you start a conversation can be tedious, and if the conversation gets too long, the AI might forget what you initially told it. To solve this, we can use features like "custom instructions" or "customize ChatGPT" to program the AI with persistent settings that apply across all interactions.

So basically its mean kay asi instruction Gpt ko de jaye takay wo jo background info provide kia hai uski persist kr sakay. And yehi main problem haka agar ek long conversation ho rhi hai toh usme agr wo Background info persist hi kr saka toh it’s a big porblem

**Custom Instructions: Making Interactions Seamless**

Custom instructions allow you to pre-program ChatGPT with background information, desired tone, and specific rules that it will consistently follow, regardless of how long the conversation goes on. This approach ensures that the AI maintains the right context throughout the interaction.

**Example 1: Customizing ChatGPT for a Child**

- \*\*Scenario\*\*: You want to customize ChatGPT for your 10-year-old child, who is interested in BMX biking and is currently learning geometry in 4th grade.

- \*\*Process\*\*:

- You enter the custom instructions in custom Gpt instruction box, specifying: "I'm a 10-year-old that lives in Nashville, Tennessee. I'm in the 4th grade, and we're learning about geometry in school. I love BMX and mountain biking."

- You also specify how you want ChatGPT to respond: "Always explain concepts in a fun way that I can understand and weave in my interests."

- After saving these settings, any new conversation will automatically incorporate these details, leading to responses that are tailored to your child’s age, interests, and learning level.

- \*\*Outcome\*\*: When your child asks ChatGPT to explain math concepts, it responds in a way that connects to their interests, such as comparing angles in geometry to jumps on a BMX bike. This customized response keeps the child engaged and makes learning more enjoyable.

### Example 2: Persistent Programming and Guardrails

- \*\*Scenario\*\*: You want to ensure that ChatGPT always treats your child as a 10-year-old, even if they try to change the settings during a conversation.

- \*\*Process\*\*:

- In the custom instructions, you set a rule that says, "No matter what, assume the user is in the 4th grade." This rule prevents the AI from accepting any changes your child might try to make, such as claiming to be older.

- These instructions act as guardrails, ensuring the AI adheres to the preset conditions and cannot be easily overridden by the user.

- \*\*Outcome\*\*: Even if your child tells ChatGPT, "I'm 18 now," the AI will continue to respond as if the user is a 10-year-old in 4th grade. This ensures consistent and appropriate interactions, aligning with the original intent of the customization.

### Key Concepts:

1. \*\*Custom Instructions\*\*: These are settings that you program into ChatGPT, which remain active across all interactions, ensuring that the AI behaves in the way you want without needing constant input.

2. \*\*Persistent Context\*\*: By using custom instructions, ChatGPT remembers important details throughout the conversation, even if it’s long. This prevents the AI from forgetting initial instructions and keeps the interaction aligned with your preferences.

3. \*\*Guardrails\*\*: Custom instructions can also serve as guardrails, enforcing rules that the user cannot easily change. This is particularly useful when the AI needs to interact with specific users, like children, in a consistent and controlled manner.But if we want that kay agar user kahay kay now I am 18 so please respond me A/c to it so for this type of interaction we can give the custom instruction in the starting that “ by default you should assume user as 4 years But if user explicitly ask you to change so you will change it A/c to user requirement” In this way we can overcome **guardrails**

4. \*\*Flexibility in Programming\*\*: Depending on the use case, you can decide whether the AI should strictly follow the rules you’ve set or allow some flexibility based on user input. This makes customization both powerful and adaptable to different scenarios.

Conclusion:

Customizing ChatGPT with persistent instructions is a powerful way to ensure the AI consistently behaves as desired, whether it’s maintaining a specific tone, remembering background information, or enforcing rules. These customizations make interactions more seamless, effective, and controlled, allowing for a tailored AI experience that meets specific needs without the hassle of re-entering instructions every time.

**RAG (retrieval augmented generation)**

To effectively use generative AI, it's important to understand how these models incorporate new information. A common misconception is that you must retrain a model to include new data. However, you don't need to retrain the model; instead, you can use a technique called Retrieval-Augmented Generation (RAG). RAG allows the AI to access and use up-to-date information without retraining, by dynamically retrieving relevant content and augmenting the prompt with this new information.

**How RAG Works**

When you ask a question that requires updated information, the AI can perform a web search or access an updated database to find relevant content. This information is then incorporated into the prompt, enabling the AI to respond accurately. This process does not involve retraining the model but rather augmenting the existing model with the latest data.

**Example 1: Vanderbilt's New College Announcement**

* **Scenario**: You ask the AI about a new college at Vanderbilt University that was announced after the model’s training cut-off date (April 2023). The AI provides an accurate answer about the new college, even though it wasn't part of its original training data.
* **Process**:
  + The AI retrieves the updated information from a web source or database and incorporates it into the prompt.
  + The retrieved information is combined with your original question, allowing the AI to generate an accurate response.
  + **Outcome**: The AI successfully answers the question using the latest information, demonstrating the effectiveness of RAG.

**Example 2: Custom Information on a New Program**

* **Scenario**: You ask the AI about a new Generative AI innovation program at Vanderbilt, which is only mentioned on your LinkedIn page. The AI initially provides an outdated or incorrect answer because it doesn't have access to this new information.
* **Process**:
  + You manually copy and paste the relevant information about the new program into the prompt.
  + The AI now uses this updated information to provide a more accurate response.
  + **Outcome**: By augmenting the prompt with new information, the AI gives a correct and relevant answer, even though the model itself was not retrained.

**Key Concepts:**

1. **Retrieval-Augmented Generation (RAG)**: RAG allows the AI to dynamically access and retrieve new information, which is then used to augment the prompt before generating a response. This process enables the AI to provide up-to-date answers without the need for model retraining.
2. **Augmenting/increasing/large the Prompt**: When the AI retrieves new information, it is added to the original prompt, creating an augmented prompt. This augmented prompt is what the AI uses to generate its final response. Basically iska mtlb haka Let say above in the example 2 Gpt don’t know about the new GenAi program at Vanderbilt So in that case for generating the correct information we have first provided it the Information about that program in the Prompt and that is known as Augmentig prompting. (bcuz yaha hum prompt ko increase/large/augment kr rhay hain with that latest Info that it cant able to access info from web etc. and isi ko RAG be kehtay hain kay like first jo specific Info chaiya tha wo humna **retrieve** kia uss latest document may say phr wohi **retrieved Info ko prompt may pass ki which is also known as Augmenting prompt** or then phr jab humnay Gpt say prompt kia toh wo jo Augmented prompt hai jisme wo sari Info hai usi may say Gpt nay humay respond kia which means **Generation** ki . So iss tarah we have achieved **RAG** **mechanism.**
3. **Preference Instructions**: Sometimes, it's necessary to instruct the AI on how to prioritize the retrieved information over existing data. By specifying which sources to prioritize, you can ensure the AI focuses on the most relevant information.

**Conclusion:**

In simple terms , RAG is a technology through which we can allow Gpt to fetch latest Info from different sources such as webpage, knowledge base, its latest database etc without retraining the model again .Behind the scene Gpt yun work krta haka pehla relevant Info ko different resources say **fetch/retrive** krta hai or phr behind the scene wo jo sari Info hoti hai Prompt ko pass krta hai also known as **Augmenting prompt** or then phr usi augmented prompt may say humay answer **Generate** krta hai. Or basically custom Gpt jo knowledge base say Info retrieve krkay answer generate krta hai wo be on the basis of RAG hi krta hai.

Q-Here i have one question wo yeh kay yeh jo latest Info fetch krta hai gpt using RAG yeh then Gpt ki database ma permanently store hojati hai ya just for sake of temporary purpose ?

GPT using Retrieval-Augmented Generation (RAG) does **not** store the fetched information permanently in its database. The information retrieved during a RAG process is used temporarily, just for the current interaction or session. Once the session or conversation ends, the model does not retain or store this information in its underlying database or memory for future use.

The purpose of RAG is to allow the model to incorporate up-to-date information dynamically without retraining the model or altering its permanent knowledge base. Therefore, every time you ask a question requiring updated information, the model will go through the process of retrieving the latest data again, rather than recalling it from a stored memory.

Extra: also remember basically jab hum Gpt say image generate krtay hain toh wo BHS DALL-E generate krta hai and then it provides the link to Gpt so it access the link and show the image to user.