* AI agents for business users you may have heard generative agents are the next big thing but what exactly are they during this session, lets demystify generative AI agents and share real examples by the end of this series you will have a clear understanding of the definition of generative AI agents how to design effective agents how to equip your agents with tools and most importantly how to get started building AI agents without knowing how to code so let's start by talking about what are
* generative AI agents you can think of them as a combination of two concepts as the name suggests generative Ai and AI agents now ai agents are not A New Concept they've been around for a long time so let's start by defining what an AI agent is you can think of an agent as an autonomous entity meaning it operates without human intervention that interacts with its environment gets feedback from those interactions in order to make decisions and Achieve goals the key question distinguishing an agent versus a non-agent is this can the
* AI autonomously get feedback from interactions with its environment in order to make better decisions if yes it's an agent if not it's not an agent now generative AI agents are agents powered by generative AI such as large language models llama 3 or chat GPT compared to non-generative agents generative agents are able to solve way more complex and diverse problems they're more adaptable and flexible they respond in believable language thanks to the Power of llms Now typically when you interact with Chat GPT you run a prompt
* review the output manually and notice a few things to improve here on the left hand side I'm running a prompt create a marketing copy for our company awesome culture a platform for planning virtual team bonding events and you can see the output that chat GPT gives me it's very long though maybe it's too long for a social media post so I interact with Chat GPT I want to iterate on the answer and so I give it feedback I say make it shorter for a social media post and now the answer looks much more usable it's
* much closer to what I want now the key Point here is that this interaction between me and chat GPT is what leads to better answers aligned with what I'm looking for and what I need but chat GPT didn't arrive at this better answer autonomously I still had to tell chat gpt what to change so in contrast generative AI agents are autonomous that means they work without human intervention they use large language models to ask and answer their own questions which serves as feedback to further improve its thinking and output
* generative AI agents can think about their own answers judge the quality of their answers and try to create better responses until finally they're satisfied with the quality of their own answers this means for us we don't have to sit there constantly interacting with chat TBT to iterate towards better more reliable answers and this is really powerful because generative AI agents can work independently from us learning over time from interactions and feedback and getting better for the rest of this video I'm going to use the term agents
* to refer to generative AI agents when it comes to designing effective agents consider these six aspects number one identity number two memory number three Planning number four narrow scope number five use of external tools and number six collaboration with other agents we'll dive into each one of these starting with identity the identity you give an agent directly impacts the quality of its answers for example let's compare these two prompts and answers the first prompt on the left hand side is simply what is llama 3 no additional
* Identity or context is given to chat TPT and you can see the answer it leans more towards the technical side it contains quite a bit of technical detail that would be hard to understand if you're for example a sixth grade science teacher teaching sixth graders so on the prompt in the right we are now giving chat an identity we're telling chat you are a sixth grade science teacher and then we ask what is Lama 3 and you could read the total change in its response with a very brief brief one sentence identity assignment Chachi BT assumed
* the role of a sixth grade science teacher and it completely changed the content the writing style it started off its answer with a joke the comprehension level of its answer uh the amounts of technical detail notice that there are no references or sources or technical details included after we gave chat upt the identity of a sixth grade science teacher so be mindful of the identity in context that you give each agent I recommend you experiment a lot to see what works best for your use case in Industry because identity can make a
* really big difference in the types of answers you get the second design aspect to consider is memory now memory plays a huge role in many real world agent systems inspired by human memory constructs agent memory enables agents to remember past actions and decisions learn from them through a process called reflection and apply these learnings to future decisions so memory is incredibly powerful for agents because it's what allows them to get better and better over time learning from past decisions over time now we have at least two types
* of memory just in Broad categories we have short-term memory that starts fresh each time an agent runs each time an agent is assigned a new task and then there's long-term memory typically stored in a database and this is what allows agents to learn learn from previous runs each time an agent finishes a run the agent reflects on its work on its answers and tries to figure out how it could have done better what could it have done to improve its answers to do it better and faster for next time so when faced with a new task
* agents query these reflection insights in order to make better and more reliable decisions in the future so memory is really really powerful especially for AG to be able to work reliably autonomously over time planning is another key aspect of agent design many tasks are complicated they can't be accomplished in a single step and for complex tasks you may not be able to specify all the necessary subtasks ahead of time with planning you let your agent decide dynamically based on the information it has at hand so you let
* the agent think through the steps it needs to take to accomplish the goal for example suppos you have a QA agent reviewing customer service replies drafted by another agent the QA agent finds that part of the reply can't be verified by information in the knowledge base so the QA agent then decides to search through the company's technical documentation hosted outside separate from the knowledge base to find supporting information and so what it did there it didn't find what it was looking for in the initial Source the
* general knowledge base so the agent thought about what to do next and decided to search another more technical Source the technical documentation and that's a really powerful capability when you think about agents that need to operate autonomously they need to you know look at a complex situation or task be able to break it down into subtasks and create plans for the best way to proceed next this is why planning is a powerful design pattern to increase the overall flexibility and adaptability of your agent systems you want to empower
* agents to analyze available information their goals and what tools they have at their disposal so that the agent can formulate a reasonable plan consisting of specific discrete steps it can take now let's talk about scope research has shown that providing llms with excessive information or context actually decreases output quality and increases hallucinations so it's crucial to keep the scope narrow give each agent a single identity and sufficient context to succeed but not so much context that the agent loses sight of what's
* important so when designing your agents ensure that each agent has one specific goal instead of a single agent trying to do everything assemble a team of Agents where each agent has one goal focusing on a narrow specialty for example if you're using agents for coding tasks then start by assembling a team of technical agents with distinct responsibilities for example One agent writes an architecture plan by analyzing requirements from the product manager another agent could be the one writing the code and a third agent could be in
* the role of QA testing the code another Point here about scope don't overwhelm your agent with tools either equip each agent with the critical tools it needs to achieve its goal and do its work access to too many tools is counterproductive because your agent may get confused about which tool to use I've mentioned tools a couple times so now let's talk about tools agents can leverage tools which massively extend their capabilities such as calling external apis collecting data and searching the internet this makes agents
* especially powerful and extensible I'm going to dive into tools a lot in the next section to wrap up the list of design criteria as you think about creating your agent systems the last on the list is collaboration now remember how interacting with chat TBT leads to better answers the back and forth conversation helps chat GPT learn what you're looking for and the same is true for agents can talk to each other collaborate with each other incorporate feedback from one another and together iterate towards better answers I'll
* discuss multi-agent collaboration a couple videos from now when we talk about designing multi-agent systems to summarize what we've talked about so far take these six aspects into serious consideration in order to design effective agents and agent teams we talked aled about identity giving each agent a clear role we talked about memory which is what enables agents to learn from past decisions and make better informed future decisions number three we talked about planning letting agents decide dynamically what are the
* best next steps to take we talked about narrow scope keeping each agent's scope focused and that means the role is focused their goal is focused they aren't given too many tools each agent has a very clear and distinct responsibility we briefly touched on tools I'm going to talk about that a lot in the next section and last collaboration with other agents is really fascinating we're going to dive into that a lot more later during this session series I want to expand a bit On Tools because tools play a big role in agent
* systems tools are what allow agents to communicate with the external world including APIs your company's internal systems and databases and the internet so for example here on this slide on on the right hand side this is a list of built-in tools supported by Lang chain which is a framework for building AI agents and you can see all different types of tools supported here many different Google tools file system search tools SEO tools uh python based tools many different tools being able to plug into these tools is what makes
* agents incredibly powerful low code agent Builder platforms offer many out of the box Integrations with tools for common use cases including operations marketing research and sales so for example imagine an AI agent tasked with outbound sales prospecting to be maximally effective the agent should have tools to enrich lead information um look up the lead's company and website read and scrape the company's website and also Google search for further research an agent equipped with these tools will be far more productive and
* reliable than an agent without but here's the challenge agent out puts are text based and probabilistic but most external tools require a structured input such as Json so you need to convert your agent's output into a structured format required by most tools now what makes this particularly complicated is the nature of llms is that they're probabilistic the same input does not guarantee the same output what that means is outputs can vary and break the connection to tools resulting in errors or exceptions I experienced
* this myself when I recently built for fun an AI powered test prep app even though I explicitly asked chat GPT for a Json output there's a nonzero probability that occasionally it will return an invalid Json and I found out on the 20th run that chat GPT did exactly that it mixed latch and Json without escaping resulting in an invalid Json if I didn't have robust error handling in place this would have crashed my app one thing you want to think about when designing AG agent systems is think about what should happen when the agent runs into a
* failure or exception using a tool should the agent stop should the agent try again what I recommend is looking for tools that have built in error handling built-in retry mechanisms and that return helpful error messages to your agent who may use its planning capabilities to decide on a new plan if it encounters a failure or issue connecting to one tool perhaps it'll decide to connect to another tool to try to achieve goal tools are incredibly important and they give your agents superpowers by being able to connect with the external world
* and all the services out there that we already have including APIs databases and so forth in the next video I'm going to show you how to build an outbound sales prospecting agent that's equipped with a variety of tools to help it do research efficiently and draft outbound emails