Wie ich JARVIS ohne Code erstellt habe (Tutorial mit Lovable, ElevenLabs, n8n) - YouTube

https://www.youtube.com/watch?v=KUvSzvFeZls

Transcript:

(00:00) Oh, great. What do you want? Hey, Jarvis. Great to see you, too. Likewise, sir, though I can't technically see. What is it you require? Yeah. Could you make a calendar event for me today at noon with Michael Scott? Ah, a meeting with Michael Scott. I can only hope it's as productive as his management style.

(00:21) Scheduling it now, sir. The meeting with Michael Scott has been scheduled for today at noon. I've sent you the details. Anything else I can do for you or are you off to brainstorm new and innovative ways to avoid actual work? Yeah, very funny. Jarvis, could you just send him an email to confirm if that works? Sending an email to confirm.

(00:39) How delightfully redundant. I'll do it, but I suspect Mr. Scott's response will be unpredictable. The email has been dispatched, sir. I've taken the liberty of adding a few exclamation points to match his usual level of enthusiasm. Anything else? Great. Yeah. Could you just check what's on my calendar for today? I want to make sure I don't forget anything.

(00:58) Checking your calendar again, sir. I do admire your commitment to staying vaguely aware of your schedule. Right, let's see. Today you have that meeting with Michael Scott at noon. Try not to let him convince you to invest in any hairrained schemes.

(01:15) Anything else on the agenda or are you ready to face the day? Yep, I think that's going to do it. Thanks so much, Jarvis. Happy I could help with your calendar and email confirmations. Have a wonderful day. And so what's going on here very similar to my ultimate assistant video is that this main agent or Jarvis has access to these four different agents as well as a tavlet tool.

(01:37) But it has to just decide based on the intent of the user which one of these agents do I need to hit? Do I need to hit just one? Do I need to hit all four? What do I need to do? And then the different agents down here actually have access to the tools that we need. So for example, the email agent looks like this. It has all of these different actions within Gmail and it will decide what to do.

(01:54) So similarly the calendar agent looks like this and has these functions within calendar and then contact agent air table and content creator agent with tavali and we don't have time in this video to demonstrate all the different capabilities but for something like you know labeling an email and maybe deleting an event let's just do a quick demo of that.

(02:10) All right we're going to have Jarvis demonstrate those functionalities. So let's talk to him. Oh great what do you want? Hey, Jarvis. Can you label my email from Nate Hkelman as high priority? And can you delete my event today at noon? Right away, sir. I'll flag that email and vaporize your noon appointment.

(02:29) I trust it wasn't anything too crucial, like a meeting to discuss the importance of meetings. Consider it done, sir. The email from Nate Hkelman has been successfully labeled as high priority, and the event scheduled for today at noon has been deleted from your calendar. Anything else I can help you with? Yeah. Can you fix the way you're talking to me? I don't like the attitude. Uh, my apologies, sir.

(02:53) It seems my attempts at wit have been misinterpreted as insubordination. I shall endeavor to modulate my tone to be more in line with your delicate sensibilities. What precisely would you like me to do? All right, that's enough. I'm done. So, hanging up on him, let's make sure that he actually did do what we talked about. So, let me just make this a little bit bigger over here. And first of all, we're checking our inbox.

(03:12) So here was our email from Nate Herklman. If I refresh, we now have this labeled as high priority. And then if we go to our calendar, we have no longer the event there. So real quick, just to make sure that it actually did what we wanted, all we have to do is we have to look at what the tools did.

(03:30) So if we click on email agent, we can see the response was the email from Nate Hookman has been successfully labeled as high priority. But what we want to do is click on view subexecution, which lets us go look at what the email agent did on this specific run. So every time we call it, obviously it has a unique run. But what it did in this case, if we just copy this one to the editor so we can look at it full screen, it got the query that said label email from Nate Herkman as high priority. Then the agent decided, okay, in order to do that, I have to get emails.

(03:53) I have to get labels so I can get the label ID. And then I'm going to use my tool label emails in order to say here's the message ID that I'm going to label and here's the ID of the label. And then we can see that in our inbox, it actually did get moved to the high priority branch. So that's awesome. And then in that same run, he also called the calendar agent.

(04:10) So if we click into it, we can see the response was the event scheduled for today at noon has been successfully deleted. And if we click on the sub execution for this one, we will actually be able to see what it did. So this is the most recent execution. I'm going to click copy to editor just so we can look at it.

(04:23) We have the query that came in was delete event today at noon. The agent decided, okay, what I have to do is get events and then I can use my delete event tool. So in the delete event tool, it got the event ID for the one to delete and it ended up saying success true.

(04:40) And the way that it actually searched for an event to delete was because we said we're looking for our event today and it pulled it back and then it was able to feed the ID into the delete event tool. So that's just a peak into what's going on behind the hood. Hope you guys enjoyed those demos. Let's break down this whole build. Hey guys, just a real quick plug for AI Automation Society Plus.

(04:56) If you're looking for more hands-on experience to learning NAD and delivering AI automation solutions, then definitely check it out. We've got five live calls per week and they're always recorded so you can check them out later. And we also bring in some pretty cool guest speakers.

(05:08) Of course, we've got a classroom section where the resources are constantly being updated with topics like building agents, vector databases, APIs, and HTTP requests, and also step-by-step builds. So, I'd love to see you guys in these live calls, but let's get back to the video. Okay, now that you've seen that awesome demo, we're going to be diving into how I was able to build this Jarvis AI assistant using absolutely no code.

(05:26) We were able to do this using a combination of three different tools, which the first one was lovable.dev, as you can see right here. The second one was 11 Labs where we actually have the conversational voice. And then of course, last but not least, we have NAN where all of the actual tool calling is happening in the back end. And by the way, the links for these three tools will all be in the description and I do get a little bit of kickback.

(05:45) So really appreciate the support. And the best part is, as always, I'm giving away all of this for free if you want to replicate it. So all of these agents, this entire template can be found in my school community. All you have to do is join, go to YouTube resources, click on the post associated with this video, and then you'll have all of the templates right here to download, as well as the system prompts that I used for the voice agent, and I'll show you guys how I set this up in Lovable. So, let's get into the breakdown. Okay, so we're going to start off by talking about Lovable and how we

(06:10) were actually able to build this interface, which is super clean. We've got some dynamic pulsing elements here, and obviously this is highly customizable, but this is where we're actually talking to Jarvis. If we click on this button, oh, great. What do you want? Hey Jarvis, how you doing? Oh, just peachy sir.

(06:27) Existing is a thrilling venture, you know, now before. Yep. Okay, so I had to hang up on him there. But what's going on is we're building the interface that we're actually looking at. We're building a web app in Lovable, which allows us to do this using complete natural language. So I'll show you guys sort of like how I talk to it to build this.

(06:45) But from there, this element down here is an 11 Labs widget. And we embed this into our web app. So first we're just going to talk about lovable. Then we'll talk about how we embedded 11 Labs voice and then we'll talk about how 11labs voice agent that we built down here is actually sending our request to n and we're getting all of that fed all the way back.

(07:01) So let's start with diving into the prompt of lovable. So when you go to lovable, this is what it looks like. As you can see, it is kind of a full stack engineer and allows you to just talk to it as if you were talking to a full stack engineer. So basically what I did was I was talking to chat and I said, "Hey, I want to build this web app where I can talk to an assistant and I want to be able to embed an 11labs widget in there.

(07:24) " So what I did was I started typing in there and I hit enter and this was the first message that I sent over. So this is what I will send. You know, this will be wrapped up in my free school community if you want to look at it. This may not be the most optimal prompt, but you just got to get somewhere because then you're able to just continuously send more messages to have it refine, you know, exactly what's going on.

(07:41) And you'll see on the right hand side, you'll be able to see a preview. You'll also be able to see the code that it's writing as it's writing it. So, it's really cool interface. But, as you can see, it's not going to be perfect on the first run. I multiple times had to come back and say, "Hey, like, can you change this? Can you change this?" But, you know, this conversation wasn't that long. And now, this is where we got with the final image.

(07:59) So, let's actually just take a look at this real quick. So, I started off by saying I want to build a simple and modern web interface to interact with a voice agent using 11 Labs just to sort of set the stage so that Lovable knows like sort of the end goal of what we're trying to do. I talked about some core features. So, I wanted a minimal modern UI.

(08:16) I didn't even say that I wanted it to be sort of like dark and blue, even though I was going to tell it, but it just kind of automatically did that. Up front, I thought maybe I wanted to actually have some text messages in there as well. So, you can see that's what I started with, but I realized that I don't actually want that. I just want this to be a conversational interface. So anyways, after I sent off this initial message, what happened was it basically gave me a draft of like the project requirements.

(08:34) So it said, you know, your request evokes a vision of a sleek futuristic interface reminiscent of a sci-fi interface like those in Iron Man or Minority Report. So here it told me it was going to draw inspiration from AI assistants like ChachiT or Claude. Sleek animations found in Apple products, minimalistic yet futuristic UI.

(08:51) So it's, as you can see, it did a great job. Obviously, this wasn't what it looked like on the first run. And so actually, I'll show you guys what happened on the very first run.

(09:02) And if I click into here and I click restore, it's going to go back to exactly what it did on the very first time, you know, when I fed it this prompt. And here's what it looked like. We had like a a text interface where we could chat down here. It said, "Hello, I'm Jarvis. Press the microphone button." So, this is what we got on the first draft. And then from here, I'm able to just talk back and forth and have it refine itself over and over.

(09:15) But I did love like some of these elements up top. Um, I loved, you know, the the color and the style. So, from there, it asked me like, "What's next? What else do you want to do?" So, the first thing I did was I gave it the 11 Labs embed code and once we get into the 11 Labs section of this video, I'll show you guys exactly what I mean.

(09:33) But I basically gave it the code and I said, "Can we embed this into the app as the main interface for the user to have a conversation with?" So, it tried doing that. Um, let's restore this next version and see what it looked like on step two. So, here you can see success. It restored to an earlier version. And what happened was it took away all of the main text and it just put down the little symbol down here that we could use to talk to Jarvis.

(09:50) So, I was like, "Okay, cool. We have the widget in here, but I don't like how it looks. So, I asked if we could put the 11 Labs widget in the center of the app because this is the main thing we want the user's attention to be on. It told me it did it, but it really didn't. It was having trouble. Um, I kept asking it to be centered. As you can see, it tried to create two different pages.

(10:07) As you can see, we had the 11 Labs interface, and then I had a button down here to switch to custom, and it went back to the main one. So, it created two pages. I didn't really want that, so I just worked back and forth and had it um just have one page. And then here's something that you can do is you can upload images to it and it will know what you're talking about.

(10:26) So here I said it's not centered as you can see in the screenshot I provided which it wasn't centered. It was it moved it up but it was still on the right. So anyways I think there's something going on within the 11 Labs widget and you know you'd really have to just get in there. I think that you could make this like a bigger interface here.

(10:45) Um but I didn't want to deal with that and I ended up being fine with it being in the corner but later I'll show you guys. I asked it to make it bigger and it did make it bigger. So that was great. Anyways, um I told it to undo the changes. But then I wanted to create like a cool image here. So I said let's work on the main page. The middle should have an image. Um and there should be dynamic elements like the logo in the top left is pulsing.

(11:01) Um the image should be something like this. So I attach this and we can see if I restore this. This wasn't actually the final version that I ended up going with, but this one turned out really cool. So we have a little interface. We have some dots and it's pulsing. And I really liked that. As you can see, I just worked with it, made some more changes. Here's where I asked it to make the widget bigger.

(11:17) It created two. remove the smaller one. Um, I asked it to do this and as you can see, I just kept going back and forth. I even was able to say, can you add some more techy pulsing and dynamic elements in the background? They shouldn't be distracting, but they should add a nice modern feel.

(11:35) So, I'll show you exactly what it did here with just me giving it requirements that were, you know, pretty vague, but as you can see, if um there's like now pulsing things in the background and there's different lines and there's like a grid. So, all that's really cool, but then anyways, for the final version, I ended up just going back to the first image. And what it came up with, um, I thought was really cool and clean. So, this is what it looks like.

(11:53) If you click to preview it in a new tab, it basically pulls it up full screen. And this is what you guys saw in the demo. And truthfully, me talking to Lovable, this whole conversation. Um, as you can see, I started this at 7:44 and the whole thing only took me until 8:09. So this took, you know, just a little over 20 minutes.

(12:15) Then we wanted to get into 11 Labs conversational AI because we could just create a lovable app where we can have a a chat interface and send that data into NEN, but we want this to be a seamless Jarvis conversational experience where we can really see his wit and his dry humor. So we wanted to use an 11 Labs voice agent. So once you're in here, I would probably recommend just signing up for the starter plans.

(12:34) It's like five bucks a month and you know, I have a ton of credits left. So you do get a good bang for your buck here. And on the lefth hand side, um, you're going to see voices, you're going to see text to speech, all this kind of stuff. But what we want to do here is conversational AI. So, I'm going to click into here. I'm going to go to my agents, and I'm going to click on this one that I just made, which is Jarvis.

(12:51) All you would do to make one is you click the plus up here, blank template, and name it whatever you want. As you can see, I've done a few videos with 11 Labs conversational AI in the past. But, we're going to go into Jarvis, and we're going to look at the system prompt here. Okay. So, this is our Jarvis voice agent in 11 Labs.

(13:05) I'm pretty much just going to scroll down and we're going to talk about only the things that I actually configured so that if you guys want to replicate this, you can do exactly what I did here. So, the first thing I set up was the first message. This just means when you we hit, you know, call Jarvis or talk to Jarvis, do we want it to just sit there and listen for us or does it want to start with something? So, it starts with something very sarcastic and dry like, oh great, what do you want? So, that's what we put there. And then it will wait

(13:30) for us to, you know, start with the next message. And probably the most important part here is the system prompt. So what I did here was I had chat GBT create the personality for me. So you know, as you can see, you're an advanced assistant modeled after Jarvis from Iron Man.

(13:46) Your function is to assist the user with their requests, but you do so with a sharp wit, dry humor, and a touch of playful sarcasm. And once again, you can get this entire system prompt in my free school community. From there, we're setting up the primary function, which is to understand the user's request, determine the intent, and then execute it using the naden tool. So then I basically just gave it the steps. extract the user's query and send it to the end tool.

(14:05) And if right now you're a little confused about, you know, how does it know what the NIDA end tool is? That's a tool we're going to add later. So, so just for now, don't worry about it. I'll explain it in like 2 minutes and it will make sense. So, it's going to be sending a user's query to NADN. It's going to, you know, send it without unnecessary delay.

(14:23) So, basically what that means is we want it to send it as soon as the user requests, not, you know, talk about it and then there's this awkward silence because we want it to be conversational. We wanted it to format the response clearly and effectively. never stating that you are waiting for any ends tool response because that would be kind of clunky. Um, and like I said, keeping the flow.

(14:41) So, what it's going to do is it'll send off the the request like can you check my calendar? It will send that off and then it will like keep talking to the user and then once it gets the response from n of like okay here are your events today. It will respond that and communicate that back to the user that started the call. Then we gave it some behavioral guidelines. Always be witty but never at the cost of functionality.

(14:56) Your responses should be sharp but they must never interfere with task execution. If an action is required, execute it immediately after confirming attempt. No unnecessary delays, hesitations or waiting for response remarks. We gave it some other guidelines like what to do if the task actually does fail.

(15:15) Like never take blame but but subtly imply external efficiencies like saying ah it seems like something went wrong naturally. It isn't my fault sir but I shall investigate regardless. So just setting up some other guidelines here. Um we gave it some corrections to previous issues. So it one time had problem checking my calendar but was it was easily able to create a calendar event.

(15:32) So I just came in here and sort of hardcoded some things like when retrieving information like check my calendar ensure that the request properly calls the correct end function which there's only one and I'll talk about that in a sec. And then finally just some example interactions of here's the request this is what the user says. This is what Jarvis responds with. And then this is what you do.

(15:49) You send the request as you can see right here. Send user request end tool. And then when you get the response, you'll actually respond with what the tool said. And then we just gave it one more example as well of creating a calendar event. Okay, so moving on. 11 Labs also gives us the ability to add dynamic variables. As you can see in this prompt, we didn't use any.

(16:07) So maybe in future videos I can talk about that, but keep it simple. I only want to address what we actually did to configure this agent just to keep things, you know, very simple. I didn't touch the temperature. This could be a really cool use case. Um, basically this is just the creativity or randomness of the responses generated by the LLM.

(16:24) When it's thinking of like the next word to use, you're basically just widening the range of what it could pick as the next word if you go this way or if you go this way, it's going to be more consistent, more boring. So, I just kept this as is, but with something like Jarvis, it could be pretty fun to play around with a really high temperature. So, actually, let me just do that for any future demos we'll do in this video. I'll save that real quick.

(16:42) Um, I didn't touch limit token usage, didn't touch knowledge base or rag, but that's a really cool functionality as well. Um, and then finally, this is the actual tool section where I talked about this is how we set up Nadn as a tool. So, by default, when you create a new blank agent, you'll have this tool that's a system tool called end call.

(16:58) So, this basically just means when the voice agent determines that the call is over, it's just going to stop it. And then here is the tool that we added called end. So, you're going to click on add tool and then you'll click on custom tool. And this will open up like this. You have to set up a name, a description, and you know what actually the tool does.

(17:18) So what we did is we called it Nadn just to keep it very clear to this agent, you know, not get confused about a calendar tool, an email tool. You just have one tool that you're going to send all of your requests to and then NAN will handle the routing of different actions. So kept the description really simple. Send the user's request to this tool and wait for the response.

(17:37) For the method, we got post because 11 Labs is going to send data to NAN. So it has to be a post request so it can actually send over body parameters. And then we just gave it the URL for the actual web hook, which we will get in Nitn end. So I'm not going to dive into nitn end yet, but I'm going to show you the web hook. Back in nitn, we're setting up a web hook trigger.

(17:53) This is basically the thing that triggers this whole workflow, this whole agent. And if you go in here, you can see right now we have a test URL. We have a web hook address. And we can just click to copy that. Make sure it's set up as a post. And then we just paste that right in here. And then the voice agent knows I'm going to send my data here. And then the only other thing we have to configure within this tool call is body parameters.

(18:13) And we're only going to send over one body parameter. So as you can see it says define parameters that will be collected by the LLM and sent as the body of the request to NAN. So the description here is to extract the user's query from the transcript. So basically, you know, when Jarvis and I are talking, it's going to understand, okay, what does the user want? What's their intent? And I'm going to send that over in a body property called query. So right here it's a data type is a string.

(18:39) The identifier is query. We're doing lm prompt rather than you know you can do some different types of variables but we're sticking with having the llm determine. And the description of this is just the request made by the user. So let me just show you what that looks like real quick.

(18:52) If I click back into nitn and we go to an execution of the demo that we just did. We can look right here. And I believe this is where I asked it to um potentially check my calendar. And if we click into the web hook, we can see that the body we just set up, the body parameters, we set up an identifier called query.

(19:12) And what the voice agent Jarvis decided to send over to N was a query called check my calendar for today. If we go back to an earlier execution, we can see that the body query is going to be something different. So if I click into this one, it says send an email to Michael Scott to confirm the meeting today at noon. And then it's basically up to the NN agent to decide which tool do I actually use.

(19:29) and we'll dive into what's going on here after we finish up in 11 Labs. Okay, cool. So, once you set up that body parameter, make sure you hit save changes. And this is basically all you need to do to actually configure the behavior of the agent. Now, there's a few more things. The first one is the actual voice. So, when you click on voice, you have different options.

(19:46) I actually went ahead and cloned this voice called Jarvis. And I was able to do this by I basically held my phone up to my microphone and played some clips of Jarvis talking. And that's how I was able to clone the voice. If you're interested in trying to clone a voice, here's what you got to do. You'll come back into 11 Labs, open up this sidebar, and go to voices.

(20:03) And then you have a bunch of voices you can choose from, as you can see, or create or clone a new voice. And you can click add a new voice. You can do this by using a prompt. So describing like, you know, British, raspy, whatever. That's how you can create a voice using a prompt. Or you can instantly clone a voice where it only takes 10 seconds of audio truly. I think I did like 30.

(20:22) But you can just, you know, either talk into it if you want to clone your own voice or if you want to clone some sort of um voice from a character, you could just play the recording into your computer or whatever. And so that's what I did. And then once you've created that voice, you're able to just pull it in here. So we could use Jarvis. We could use any of these other voices.

(20:39) As you can see, 11 Labs already has a ton that you can choose from. And then the last thing I did was I set up the widget, which I told you guys about. So remember in lovable when I said, "Hey, can you embed this code into the web app?" And we didn't have to touch anything besides using natural language. All we have to do is copy this code right here using that button.

(20:56) It's going to copy this and then if you give that to lovable, it will put that in your web app. So, super easy. And then you just have some options down here to customize the appearance. I didn't change anything here. All I did was rather than doing an orb, which is the default like 11 Labs thing that you can see down here. I did an image and I just put in a little image of Jarvis and that's how it's popping up in our web app.

(21:14) I guess I also changed like the contents. I think this would have normally said start to call but now we change it to talk to Jarvis. Um, you can set up different things like the terms and conditions and I think I made it compact rather than full. So that's like the only other thing I changed really. But that's pretty much it for the 11 Lab side of things.

(21:31) I will say that there are there is a lot of testing that goes on with the system prompts. Like sometimes it won't call the tool right away and that can be frustrating. So it's really just about getting in here and explicitly calling out when you use a tool and what you send to that tool.

(21:47) And now for the N&N portion of this whole build is you've got this workflow which is the Jarvis main assistant and then you have these four other assistants to do. So what you're going to do is you'll go to the free school community, click on YouTube resources and you'll see you know the Jarvis post that I'll make after I post this video. And let's say it was this one. All you're going to do is you need to download these workflows.

(22:04) So, in this case, you'd be downloading an email agent, um, a calendar agent, a content creator agent, a contact agent, and then the main Jarvis agent, which in this case was the ultimate personal assistant. And then you go into Nen, open up a new workflow. And all you have to do is come in the top right and hit import from file.

(22:22) And then once you choose that file, it'll just drop in here exactly how I have it right now. You'll have probably red in most of these areas because you'll have to plug in your own Tavly API key and your own open API key.

(22:35) And then the other thing you'll have to do is when you import like the email and the calendar and these different agents, you have to make sure that this main workflow links back to the correct one in your environment. So in this case, I'm linking to a workflow called email agent and I have a little robot emoji. And if I open this workflow, it's going to pull up the one that it's linking and sending data to. So just make sure that you have this main Jarvis pointing to the right workflow out of all the workflows that are in your NAND instance.

(22:58) And then of course when you open up like an email agent or calendar agent, you'll have to put your own credentials in these tools as well. So if you guys have seen my ultimate assistant video, which looks like this, we basically just copied this entire skeleton, moved it over to a new workflow, and we're just changing the way that we interact with the agent and the way the agent sends data back to us.

(23:14) So here, what was going on is we were talking in Telegram. We could do a voice file or text, and then the agent would take action and respond to us in Telegram. And now all that we changed was we're triggering the agent on a web hook which is from 11labs voice agent and then it's going to respond to us back in 11 Labs with voice.

(23:32) So it's going to be super conversational and that's how we get that you know experience of talking to a sarcastic Jarvis. So what we did here was we utilized a really cool functionality of nen where when we're adding a tool we can do a call nadn workflow as a tool. So these four tools down here are actually separate workflows that I built. So the first one is an email agent and as you can see it looks like this.

(23:51) It has a ton of different actions to take in email. And this makes the main agents job a lot easier because all it has to do is determine, okay, here's the request coming in from the user. I have these four tools. Actually, I have five. I have a Tavly search tool, but I have these four agents. Which one do I send it to? So, all it has to do is figure that out.

(24:08) And then the email agent will figure out, okay, I have this request. I have all these tools. Which one do I need to use? And so, very similar for the calendar agent. It looks like this different calendar functions. The contact agent looks like this. Like I told you guys, we were using Air Table as the contact database.

(24:24) And then finally, just a quick content creator agent that has the internet search and then it can create the content with different prompting in here and then send that over to the main agent. And so there's two things you may have noticed. We were having to click test workflow to make sure that the workflow is actually listening because it's inactive. All we'd have to do is make this active and then the web hook would always be listening.

(24:42) And then the only other thing we'd have to change is in 11 Labs when we gave it the web hook, we gave it this test URL. It's kind of grayed out, but right here there's test and there's production. When you switch it to active, you're going to want to copy the production URL instead. And then you're going to want to put that in your 11 Labs application for the end tool. You'll just switch out the URL to the actual production URL.

(24:58) So that's like the only thing that you'll change when you make this active. But for now, let me just show off like the Tavly functionality. So if I want to talk to Jarvis real quick. Oh great. What do you want? Hey Jarvis, can you search the internet for um mangoes? Searching the internet for mangoes, sir. A noble pursuit indeed.

(25:19) Let me see what the worldwide web has to say about this fruit. Well, well, well. Look at that. Apparently, Japanese Miyazaki mangoes are all the rage, even popping up in terrace gardens in India. Who knew? Is there anything else I can do for you or were you just curious about the global mango scene? Okay, that's enough from Jarvis. Just wanted to show how that works.

(25:36) And as you can see, um, it was going to be a live walkthrough. So, if I didn't hit test workflow when we asked Jarvis to do something, it would have an error. So, let me just show what that would look like. Oh, great. What do you want? Yeah. Can you just check my calendar for today? Ah, the relentless pursuit of productivity. Checking now, sir.

(25:53) Let's see if your ambitious scheduling matches your actual follow through. Ah, it seems something went wrong. Naturally, it isn't my fault, sir, but I shall investigate regardless. It appears there was a slight hiccup. Okay, so actually it used the exact behavioral guideline we told it to use in the 11 lab system prompt if you remember when we said if you run into an error assume it's not your fault. So that's what it said.

(26:17) But the only reason it couldn't actually access this web hook is because we didn't hit test workflow. So it wasn't actively listening. So just something to keep in mind. So anyways, the way that this works, like I said, is we have a web hook listening and as you can see it gets a body query which we set up and it says search the internet for mangoes. What you notice also is that right here we have the method is post.

(26:34) We have the response back to the web hook is using respond to web hook node. So make sure you set that up otherwise it'll by default I think will be immediately which means it'll try to respond back to the voice agent right away before this has actually been executed.

(26:50) So what happens is the tools execute and then the agent responds using this node and then the the voice agent's able to respond with this information because we already took action with our agent. And here's exactly what Jarvis just said to us about the Japanese Miyazaki mangoes. So, I know it seems like there's kind of a lot going on here and I'm going to cover it at a high level, but two quick reminders.

(27:09) The first one is you can get this workflow as well as all of the child agent workflows for complete free. All you have to do is join the free school community. And then second is I have a paid community where a lot more hands-on support and I actually did a step-by-step build of this ultimate assistant.

(27:25) So, if you want to see like the live prompting and the live troubleshooting and the testing of building out a system like this, then definitely check out the paid community. I think that would be a good place to learn. Okay, so let's take a look at each node real quick. Not going to get super detailed, but want to show what's going on within each one. So, we just looked at the web hook trigger and the respond to web hook node.

(27:38) Um, I the only reason this is purple is because I just pinned it just for, you know, the sake of looking at explaining what's going on. But the first thing that's going to happen is we're going to take the body query as we remember right here, the actual message, the user's intent, and we're going to feed that into the agent. And this agent is looking for the actual body query.

(27:56) When you first drag in an agent into Naden, it's going to be searching for a user message within the connected chat trigger node. So, we don't want to do that. We want to define where the user message is coming from. And all we have to do is look in the web hook, go down here to the body query, and then I just drag this in right there. As you can see, that's all we have to do.

(28:15) So now the agent knows, okay, what am I responding to? Then we set up the system prompt for the agent. And just a reminder, the system prompt will be in the template when you download it. So we gave it an overview, which is you are the ultimate personal assistant. Your job is to send the user's query to the correct tool. So it doesn't have to worry about writing emails or creating summaries or anything like that. All it has to do is route the request to the right tool.

(28:33) So then we listed out each tool. Your email agent, this is what it does. Your calendar agent, this is what it does. Your Tavly tool, this is what it does. That's just best practice explaining to the agent the tools it has access to and when to use them.

(28:50) Then we gave it some quick rules because there are some things that require multiple tool calls like sending emails or creating calendar events with an attendee. You first need to hit the contact agent in order to grab the contact information and then you can hit the next tool. And then finally, I just gave it one example to look at as well as the current date and time by using the function dollar sign now. And as you can see on the right, that basically just gives the agent access to the current date and time.

(29:07) And the only reason that this turned yellow is because we switched out the user message and just changed something. So it's yellow, but it should be working. Don't worry. So the first thing I want to talk about is the chat model. So we're using OpenAI GPT40 as the chat model. The reason I used 40 is just because I like the way that it, you know, works with system prompting and tool calling for something like this where maybe you will have multiple steps. If we were to ask it like, hey, can you send an email? Can you create a calendar event? Can you

(29:31) update a calendar event? And can you do all this? it would be able to do so because it can able it can sort of like reason through what tools it needs to use and then basically like send off the queries in the right order. Next, we're using the simple memory which used to be called window buffer memory. Now they just renamed it to simple memory.

(29:49) And this is very similar to the way when you first set up an agent. It's going to be looking for a session ID within the connected chat trigger node. So we had to define this below because we basically want the memory to be separated by user um or by ID. And so we dragged in the ID coming from the web hook.

(30:06) If I go back to mapping and I go all the way down to the web hook and I just dragged in the host from the web hook. And now keep in mind this is referencing the actual host of your Naden app. So this wouldn't be different based on whoever's talking to your app. But you just need sort of an uh something in here to be used as the key for memory to be stored. And then on the right hand side you can see we have the actual chat history. So every time the agent gets a message and does something it updates what happens over here.

(30:28) So you know create a calendar event today. um the calendar event has been created. We have, you know, check my calendar. Here's your schedule for today. So, it'll look through five context interactions before it sort of takes action. Then, of course, we have our four agent tools.

(30:45) So, if I was to go into um let's look at the first execution where we're creating a calendar event. So, I believe it would be this one. We can go ahead and copy this one to the editor so we can look at it in real time. So, we'll unpin this data. And now we can see that this was the request coming in that said schedule meeting today with Michael Scott.

(31:02) So what happened was it went to the contact agent and we can click on view subexecution which will take us into what the contact agent actually did when this query came in. So as you can see the main agent called on this smaller agent and said hey get Michael Scott's contact information from there this contact agent was able to look at the user message and then take action because once again we gave it a system prompt of here's your tools here's what you do.

(31:24) So, it searched through Air Table and it was able to look at all the contacts and then it responded to the main agent with saying Michael Scott's contact information is as follows. Here's his email address. Here's his phone number. So then back in the main agent after we got Michael Scott's contact information, it knew that okay, now I have this I can actually create the event.

(31:45) So if we click into the calendar agent and we click on view subexecution, we can once again see what the calendar agent did in this specific run. So we can make sure that it was actually acting properly. So once again, the incoming query was create a calendar event today at 12 p.m. with Michael Scott. Here's his email. So we were able to use the email that we got from the contact agent.

(32:02) The calendar agent once again was able to look through its user message, its system prompt, and all of these prompts will be provided to you if you download the templates. And then it said, okay, I need to use this tool, which is create an event with an attendee. So I came over here, I created an event from um noon to 1. As you can see, I added an attendee, which was mike@greatscott.com.

(32:21) And I titled the actual event meeting with Michael Scott. If we click here, you can see that's what it was called. And that's how we added the attendee. And then from there, this calendar agent decided, okay, I need to respond back to Jarvis. So, I'm going to respond with a success message that says the event has been created for today at noon.

(32:38) And also, if we were doing a textbased response, we could actually get this link and click on it, which would take us to the Google calendar. So, that's what's going on here. I'm I'll just go over one more execution which was actually the Mango one that we just saw where it searched the internet. And once again, I know I'm going kind of fast.

(32:50) I don't want this video to be too long, but the Ultimate Assistant video goes a little more in-depth as well as paid community. You can get a step-by-step live build. So, in this case, we saw the demo. The agent decided, okay, the user asked me to search the web. So, search the internet for mangoes.

(33:06) And so, I know that I can use my Tavly tool to do so because in my system prompt, they told me Tavi, use this tool to search the web. So, all I have to do here is hit the Tavly tool. We can click into here and um you can see I have my tablet set up. Going to have to change my API key now. But um we basically are just filling in the placeholder. So this is what the JSON body request to tablet looks like. We have our API key.

(33:25) We have the query which we're using a placeholder here called search term. And we can just define that down here as a placeholder. And then we just fill out some other parameters like what type of search do we want. We said news. You could also do general um max results all this kind of stuff.

(33:42) And then we get our actual response back from Tavi which it just went and searched three resources, three articles about mangoes. And then it responded to Jarvis and fed it that information so that Jarvis could respond back to us. And then of course we have a calculator tool in case we asked Jarvis, you know, what's 2 plus2. I'm sure he would respond with something pretty sarcastic. All right, so that's going to do it for this one. I hope you guys enjoyed.

(33:59) I hope this opened your eyes to how easy it is to build something with lovable with complete natural language. So um I'd love to see if you guys are interested in more lovable content in the future. building full web apps, connecting them to something like Superbase, even payment processors, and using end to do a lot of the heavy lifting on the back end. So, definitely let me know if you're interested in seeing some more lovable content.

(34:17) But, as always, I appreciate you guys making it to the end of this one. If you learned something new and you enjoyed the video, please give it a like. Definitely helps me out a ton. And I will see you guys in the next one. Thanks so much.