

# Python and AI Power-Up Program Offline Class- 20250902\_113439-Meeting Recording

September 2, 2025, 6:04AM

1h 58m 52s

- Ajay Patel started transcription

TJ Tarun Jain 0:03

You want to get the response in Jason, which is typically dictionary in Python, which you can also convert it into Jason, which is Jason decode.

So in order to extract the details from LLM which is in Jason format, what we will be doing will be using Pydantic. So from Pydantic we will import base model and then field. Reason we need field is because.

Because to our descriptions.

So let's take one example. I will define sentiment analysis.

So what should be the first parameter inside?

Tirth 0:51

Which class?

TJ Tarun Jain 0:52

It should be base model.

Tirth 0:54

Make nothing.

TJ Tarun Jain 0:57

And here what I can do is I can define if there is any product name.

Then field. Can anyone tell me the syntax of field?

Tirth 1:09

Field open open bracket 3 dots.

TJ Tarun Jain 1:15

Three dots, then descriptions.



**Tirth** 1:16

Comma description.



**Tarun Jain** 1:19

Uh, the name of the product is provided.  
by the reviewer.

And then you have sentiment.

Which is again a string.

We can add field triple dots and description can be.



**Tirth** 1:45

Positive, negative or neutral, I think.



**Tarun Jain** 1:46

Uh, so sentiment of the product.

That is positive.

Negative and neutral.

And then uh, we can also have certain keywords like why did you map it as?

Positive or negative so list of string.

Yield again.

Here we can just define what are the main points.

Or main keywords.

Mentioned by the reviewer.

That made it large.

This intern.

So is this clear what we did so far? We just defined up identity class and these are our three keys. One is product name, one more is sentiment and one more is keywords and keywords should be in list.

And after this what we can do is we can write any prompt of your choice.

I will just say I was.

Using the dishwasher.

Provided by ABC Company.

It's one of the worst.

Product.

I have never use need replacement.

That's possible.

The story would.

So you can write any prompt of your choice and if I want to use pipe symbol now what? There are two ways we can use the parsers. The first thing is in recent changes of launch and what they've done is instead of directly using any import statements.

What you guys can do is can you run the LLM part?

This LLM is the right yesterday that we have. Can you just run that?



**Tirth** 4:23

Yeah.

Excuse me.



**Tarun Jain** 4:30

So one is land chain we need to run then as well as OS dot environment Google API key we need to add our API key and then this LLM component. So what I'll do is I will just copy this line.



**Tirth** 4:31

I think.

OK.



**Tarun Jain** 4:47

And I will paste it here.

And here what you can do is you can just click on LLM then just click on dot if you Scroll down.

You should see one parameter called width.

If you see here you have with structured output, can you see this particular function?



**Hardip Patel** 5:14

Yes.



**Tarun Jain** 5:15

I'll just select that and here what you can do is whatever pydantic schema you have defined. So if you see the input you have schema dictionary or it is type of base

model. So this type of base model is nothing but pydantic.

So what will I do? I'll just copy the sentiment analysis and I just have to pass it inside structured output.

 **Tirth** 5:38

M.

 **Tarun Jain** 5:39

Now this will be my structured.

LLM component.

So what is the function to run the structured LLM mode component?

 **Tirth** 5:57

Can we not do chain like would it be like?

Ingram.

chain pipeline and then to structured LLM.

 **Tarun Jain** 6:06

High one that we can do. I'll show that example as well. So there are three ways to do it. One you define chain.

 **Tirth** 6:10

OK.

Prompt.

 **Tarun Jain** 6:15

Then uh, prompt, then a little.

 **Tirth** 6:18

From template we will simple prompt won't work now we it would work.

 **Tarun Jain** 6:21

No, no, I meant the prompt template, not this one. So this is I'm just writing this syntax. Here you'll have output parser.



**Tirth** 6:25

OK.

OK.



**TJ** **Tarun Jain** 6:35

So there are two ways. One is directly we use with structured output from the latest release and then directly run response structured LLM dot invoke.

Prompt.

Now if you print this particular response, this is in pydantic if you look at the response. So what you can do is if you want to convert it into Jason, you can just use model dump.

And now we can use any function like get product name.

Then get sentiment.

And then get keywords.

Let me know if you're getting it in Jason.



**Hardip Patel** 7:46

Yes, Sir, I'm getting.



**TJ** **Tarun Jain** 7:51

You can also try with different example apart from sentiment analysis, which is a little bit tricky kind of thing.



**Tirth** 8:35

We actually modified the.

Whole thing for the career leveling guide.



**TJ** **Tarun Jain** 8:41

OK.

Uh, in the sense, are you using any prompter?



**Tirth** 8:46

Yesterday we were continuing this now, so we were working on a prompt of a query

from the career leveling from the PDF file that we shared.

So.

 **Tarun Jain** 8:59

Sorry, I didn't get you.

 **Tirth** 9:01

So yesterday we were continuing from the top where we had a different PDF file that we shared.

 **Tarun Jain** 9:05

Yeah.

Yeah, here.

 **Tirth** 9:08

Yeah, so the pedantic class and everything was written on that basis.

 **Tarun Jain** 9:13

OK, but are you passing any input variables from the data or you're just using plain? So how is the input pass? Let's suppose you have a PDF, right?

 **Tirth** 9:26

No, no, it is simple file. Right now we're not passing anything. I think we are still waiting on that.

 **Tarun Jain** 9:30

OK.

Yeah, yeah, that probably, I guess we should start today once we start with the database.

 **Tirth** 9:36

OK, OK.

 **Tarun Jain** 9:41

So is this done?

 **Tirth** 9:43

Yes.

 **Hardip Patel** 9:44

Thanks again.

 9:45

Yeah.

 **Tarun Jain** 9:46

OK, so let me also show one more way to do it. Now what we can do is.

Let's use LC here.

Which is Lang chain expression language.

6.

And for output parsers, what you can do is you can just define line chain.

under score core. Then you should see output parsers.

And then from here we will import pydantic.

Output parser.

And then uh, what you can do is we can define the parser.

Parts are equals to.

Pedantic parser.

And inside parentheses parser, I guess there is some object we need to define.

Why? OK, if you see this is the variable pydantic object equals to sentiment analysis.

 **Tirth** 10:51

Yes, identic object.

 **Tarun Jain** 11:02

And now what you can do is we can define the prompt template as well.

Where is the prompt template? Let's copy this.

And I wanted to show one thing. Now you might be thinking like why is it generating the response in Jason, right? So whenever we use these parsers, what you can do is you can click on this parser and then just click on get format instructions.

Can you run this function?

Can anyone predict what is happening?



**Tirth** 11:44

I I personally think this this is getting appended to the system from.



**Tarun Jain** 11:50

Correct. So this will be appended to the prompt. So now what it is trying to do is when you're using their import statements and when you're using the parsers, it has a function called get format instruction. This is appended in the system prompt so that every single time user ask any question.



**Tirth** 11:53

Yeah.



**Tarun Jain** 12:08

It should generate in Jason, but if you see here this is an example. Which is a few shot. So if you see as an example for this schema properties, it told for then title. It has some particular description and then it is actually adding our output schema. Now this schema is nothing but properties product name.



**Tirth** 12:16

Mhm.



**Tarun Jain** 12:30

Then you have description and then you have title. Title is nothing but the product name. Then you have type which is string.

And this should be the key. So now what is happening? This entire thing is there, right? It's going in in the prompt.

Ring goes for sentiment and other parameters.

So now what we need to do is when we define our prompt template. So let's define one prompt template from Lan chain or. dot prompts import prompt template.

 **Hardip Patel** 13:06

Not check wrong temperatures.

 **Tarun Jain** 13:10

Forwardable.

 **Hardip Patel** 13:13

We just import prompt template or check prompt.

 **Tarun Jain** 13:17

Let's first try with this prompt template. I'm not sure chat prompt template is compatible, so wait, let's use chat prompt template. If it didn't work, then we'll shift. Now what will be my system prompt?

 **Hardip Patel** 13:39

Chhuda Ki Taki Bhagi.

 **Tarun Jain** 13:42

I hope this screen is visible, right? I will remove this, but you understood what get your format instructions will do, right?

 **Tirth** 13:50

Yes, yes.

 **Hardip Patel** 13:51

Yes.

 **Tarun Jain** 13:51

OK so I'm using chat prompt template and in chat prompt template we have messages and inside messages we have system and then we have user and now what we can do is I have to define a system prompt.

 **Tirth** 13:58

OK.

 **Tarun Jain** 14:15

You are an expert.  
Sentiment analysis review.  
Who understands?  
Big product sentiment.  
And deep that.  
Of the reviewer.  
So this is this prompt. Now here I just have to check if there is any template variable available.  
So instead of message.  
Here we usually need to have template.

 **Tirth** 15:21

Template format.

 **Tarun Jain** 15:25

It should be template. Let me check if this is.  
Don't copy for one minute. Provide sentiment for the given product.

 **Tirth** 15:33

Open.

 **Tarun Jain** 15:39

I will have review and then what I need to do is I need to append the format instruction.

 **Tirth** 15:48

M.

 **Tarun Jain** 15:49

And here I will have system from.  
And.  
OK, they don't have input variable, I believe.  
Human template input variables.

OK, they do have partial variables. We need partial variables. So here partial variables is nothing but your format instruction. Can you see this?

 **Tirth** 16:26

Mhm.

 **Hardip Patel** 16:28

It's.

 **Tarun Jain** 16:29

Instead of template, I will revert back to messages.

So where do you think is this appended?

 **Tirth** 16:53

You should update the system prompt for it.

 **Tarun Jain** 16:56

Yeah, so now what I'll do is in system prompt at the bottom I will just include a variable called format instruction.

Now what is this?

 **Tirth** 17:09

Passion video.

 **Tarun Jain** 17:11

It's a. It's a input variable, but it is partial variable. So what is difference between input variable and partial variable when you are running invoke right? Let's suppose I define a chain.

 **Tirth** 17:20

Mhm.

 **Tarun Jain** 17:23

And I'm defining prompt. I'm defining LLM. So now what is happening in this

prompt? I already have a input variable called format instruction, but when I do chain dot invoke.

 **Tirth** 17:38  
E.

 **Tarun Jain** 17:39  
I don't have to define format instructions.  
But if format instruction is added as a input variable, then I have to use format instruction in invoke. It will be like this. Then you will have question.

 **Tirth** 17:54  
Oh.

 **Tarun Jain** 17:58  
Which is user query.  
Then comma format instruction.

 **Tirth** 18:04  
But it would still work, right?

 **Tarun Jain** 18:07  
It will still work.

 **Tirth** 18:09  
OK.

 **Tarun Jain** 18:10  
But not always you'll have to pass this every single time, right? So if you just add that as partial variable and append it here, it's just one time process. So this is remote.

 **Tirth** 18:14  
Mm.  
Welcome.

 **Tarun Jain** 18:26

So now we have one partial variable in system prompt and then we have one user prompt inside input variable inside user. So why is it showing red?

 **Tirth** 18:37

Because after messages we need comma.

 **Tarun Jain** 18:41

Oh yeah.

Then we have partial variables format instruction. Hopefully this should work. Now chain equals to prompt.

Template.

Hello.

Since we already appended our format instruction into parser variables, I don't have to use parser separately.

 **Tirth** 19:11

OK.

 **Tarun Jain** 19:11

Now chain dot invoke.

There should be review, not question.

Chain dot review.

Where is it prompt?

Response dot.

 **Tirth** 19:51

Content.

 **Tarun Jain** 19:55

Content.

If I do additional keyword arguments.

This.



**Tirth** 20:16

Don't turbo.



**Tarun Jain** 20:20

But I just want to see where it is appending in which particular parameter.



**Tirth** 20:25

Yes.

Um.



**Tarun Jain** 20:39

So let me try with prompt template from line chain code.



**Tirth** 20:39

I don't think it is a building.

OK.



**Tarun Jain** 20:46

dot bronze import.

Round template.

Round 2.

Wrong template. Here I just have to define template. So basically in prompt template what you can do is you can add your system prompt.

And then you can give a review.

Review.

And then you have format instructions.

So how many input variables do we have now?



**Tirth** 21:40

3.



**Tarun Jain** 21:41

We have three input variables, but I just want to use this as input variable which is review.



**Tirth** 21:47

Mhm.



**TJ** **Tarun Jain** 21:47

Input variables.

I just want to use it for.



**Tirth** 21:55

Tripathi.



**TJ** **Tarun Jain** 21:55

Review and then partial variables. I have format instruction. I hope this spelling is correct. Format instruction, format instruction and then comma system prompt.



**Tirth** 22:10

Uh, OK.



**TJ** **Tarun Jain** 22:13

I'll remote from system prompt.

So I will remove this line.



**Tirth** 22:20

The only problem, I think the only problem with that was instruction and instruction.

So previously when we created chat format, there was a spelling mistake.

So if you go on 121, yeah it is instruction format instruction while on the partial variables it's got instructions.



**TJ** **Tarun Jain** 22:31

Oh, good.

Oh, oh.



**Tirth** 22:41

So this might have caused issue, just guessing you know.

 **Tarun Jain** 22:46

Yeah, this might be the problem.

 **Tirth** 22:51

Yeah.

 **Tarun Jain** 22:53

All right, content.

 **Tirth** 22:57

That got the Jason, right?

 **Tarun Jain** 22:57

It's it is insult Jason.

All right, simply I was writing this.

So even I'll just write the logic of this also so that we are aware of it. So review then format instruction, format instruction system prompt.

 **Tirth** 23:09

Yes.

Right.

 **Tarun Jain** 23:17

System prompt 2.

Why? Why is it?

Template.

 **Tirth** 23:40

There is comma missing after template on line one.

 **Tarun Jain** 23:45

Rom two. Now I'll create chain 2. Chain 2 equals to Rom 2.

A little lymph.

Response to equals to chain to dot invoke.

Review.

 **Tirth** 24:10

Prompt.

M.

 **Tarun Jain** 24:22

Good.

 **Ronak Makwana** 24:23

OK.

 **Tarun Jain** 24:29

You can not do this. So what did we do? We started with the output parsers and we had to define this parser where you have pydantic output parser, pydantic object sentiment analysis. So inside this parser there is a function called get format instructions which is usually.

Have the information of how to parse into Jason with the schema provided and then you have system prompt and once you have system prompt we either can use chat prompt template. If you use chat prompt template this will use.

The chat dot client completion logic. So when you use chat completion logic you have messages. So in messages we have system and user. One thing you have to notice there are input variables, there are partial variables.

If you define input variables like this, I mean if you ever see this parameter which is review.

I mean if you see any variable which is inside curly brackets that is treated as input variables and here we have two input variables, one is format instruction and one more is review. If you make any of these input variables as partial variables, you don't have to use that during invoke.

So now what I have done is in my system prompt I have a input variable called format instruction. I don't want to use this every time when I invoke, so I'm directly using this into partial variables and 2nd you can also use prompt template. So in prompt template you just have to define a template. Starting you have to use system prompt.

And once you use system prompt then you have to use review an instruction. But if you are building chatbot never use prompt template.

If you are.

Building chatbot with memory avoid from template.

So what happens is like if you have memory and if you're building a chatbot, prompt template will keep on adding that memory in your input variables. I mean your input tokens and it is not flushing that out. So this is the reason why prompt template is not preferred if you are using anything that is related to memory.

So here if you see you have template. Now we have total 3 input variables and out of this I don't want to repeat system prompt and format instruction. So This is why I'm adding that in partial and then input variable is review so that I can use it while I invoke.

So till here we have copied the code.



**Hardip Patel** 27:07

He was, uh, I don't. Yeah.



**Tirth** 27:09

Can you can you explain about the chatbot with memory? Like what is what is happening with that?



**Tarun Jain** 27:16

OK, so let's suppose you ask 5 questions.

You have question one.

And then you have answer. So if you remember I told you when we use memory you have a variable called chat history.

Correct.

Do you remember total we have context?



**Hardip Patel** 27:43

Yes.



**Tirth** 27:45

Yeah.

Yeah.

TJ

**Tarun Jain** 27:48

And we have answer.

Sorry, not answer queries. So total 3 variables are existing.

When you're building a chatbot, one is context query and chat history. So now if you're using prompt template.

All these things will be appended in your input tokens.

So in first question it is fine, but when you are at the fifth question right, you will have same things repeated multiple times. I'll tell you how same things you have question one.

Then you have answer one and then you have question 2.

You have answer to. Then again you have question one.

And you have answer one.

And now you start with question 3.

Answer 3 and whatever you have here right this entire thing, it will be repeated again.



**Hardip Patel** 28:47

E.



**Tirth** 28:49

OK.



**Hardip Patel** 28:49

Tarun. So this will be repeated again by we call multiple invokes, right? Or am I wrong?

On a single we are calling five times.

TJ

**Tarun Jain** 29:02

No single invoke only, so usually.

Five times in the sense if you're building chatbot, usually it will be inside a loop, right?



**Hardip Patel** 29:13

No.

 TJ**Tarun Jain** 29:14

So while true or let's suppose you use any chat chatbot template and if you keep getting the post, OK, I don't know how this will work in the endpoint side because when I tested with the memory component I just observed question one. Is repeated two to three times. When I've asked question three, it didn't flush out the memory.

**Hardip Patel** 29:38

OK.

**Tirth** 29:40

It sure.

 TJ**Tarun Jain** 29:41

But I'll just check like how it works on the endpoint part on the chat UI side.

**Tirth** 29:43

E.

 TJ**Tarun Jain** 29:48

But that one thing I remember is chat prompt template is very well designed to use LLM. The prompt template part since it was from the original code base of langchain, I'm not that much of sure how they still tackle the memory.

**Tirth** 29:56

Mhm.

 TJ**Tarun Jain** 30:07

But while we were experimenting, we just saw portion one was repeated multiple times.

And this was use memory. If you don't use memory, it was working fine.

**Hardip Patel** 30:20

So is this a memory? I mean like is that like the stream that is going into the LLM? Is that like batch something?

 **Tirth** 30:20

Thank you.

 **Tarun Jain** 30:32

So memory what we have is in line chain only you have something called as buffer memory.

 **Hardip Patel** 30:34

M.

 **Tarun Jain** 30:39

From Langshin 4 dot.

Memory import.

Chat.

I paid with the had lime chain community.

dot memory.

 **Hardip Patel** 31:19

It is memory saver.

 **Tarun Jain** 31:20

What?

Deprecated.

Oh, they've created it inside.

OK, it's not core, it's not memory, it's only line chain.

If you see there are so much memories here and we usually use buffer memory when we build chatbot which is temporary memory. If we if we need long time memory, we usually use mem 0.

 **Hardip Patel** 31:55

OK.

OK.

OK.

 **Tarun Jain** 32:10

For long term.

And for cache memory it's like buffer. So buffer memory usually have window. So if you define window size as five, after five queries it will keep on removing one. Let's suppose now you are at the 6th portion, so the first portion will be removed from your buffer window.

 **Hardip Patel** 32:25

OK.

Got it.

 **Tarun Jain** 32:32

So that is how chat memory buffer works. So we'll also look at memory as one of the component.

But is this clear? So when you define template, you're appending every single input variables inside a single variable when you're using prompt template, whereas in chat prompt template it's in different variables.

 **Hardip Patel** 32:43

Yeah.

 **Tarun Jain** 32:55

But the major difference is the input variables and partial variables.

Are you getting in Jason file everyone?

 **Hardip Patel** 33:06

Yes.

 **Tarun Jain** 33:07

And now if you see her, there will be times when you might get response like.



**Hardip Patel** 33:11

I.



**Tarun Jain** 33:16

Here is the final response in Jason and then you will see back slash back slash Jason and then the response. So this will happen only with open source LLMS.

Most of the open source will generate the response like this. So what you need to do is you need to use regex and after using regex you have to extract the information which is inside Jason.

Is this clear?



**Hardip Patel** 33:47

Yeah.



**Tirth** 33:47

Yeah.



**Tarun Jain** 33:53

Uh, so let's proceed with tool calling. Everyone got the response, right? Is anyone still pending?



**Mitesh Rathod** 34:02

No.



**Tirth** 34:03

We're not totally following, but we are going along with you.



**Tarun Jain** 34:08

OK. Is there any issues that we are facing here?

Oh, what?



**Hardip Patel** 34:17

I think on on my end, I think I pretty much understood everything, just the memory part. I think I'll have to read it.



**Tirth** 34:18

No, OK.



**Tarun Jain** 34:25

On memory part I'll show the visualization, so this probably I'll have to show the output the visualization. During that time it will be clear.



**Hardip Patel** 34:30

Otherwise I am with you.



**Tarun Jain** 34:37

If not, let me remove this part. We'll cover this while we discuss memory.

We understood this part right parsers dot get format instruction.



**Hardip Patel** 34:51

Yes.



**Tirth** 34:54

Yes, yes.



**Tarun Jain** 34:55

And once we have this, we have to add that in partial variables. This is clear.



**Hardip Patel** 35:03

Yeah.



**Tarun Jain** 35:03

The safest option is the approach one that we took directly adding that in with structured output parsers.



**Tirth** 35:11

Right. I just wanted to understand one more thing, like in the chain, can we not have a prompt pipeline LLM pipeline, the identic class or the identic?

 **Tarun Jain** 35:12

This index is very.

Yes.

 **Tirth** 35:26

Uh, you know, parser.

Does this kind of thing also work or no? Just curious. Don't want to waste time, but you can just say.

 **Tarun Jain** 35:35

Yeah, this also should work. Uh.

 **Tirth** 35:39

OK.

 **Tarun Jain** 35:42

Yes.

Format instruction.

The only thing is I'm not able to remember this syntax.

Sing 3.

 **Tirth** 36:18

That is in the system prompt.

 **Tarun Jain** 36:21

8.

 **Tirth** 36:22

It is there in the system prompt.

 **Tarun Jain** 36:23

No, I didn't add this.



**Tirth** 36:26

No, no, but uh, it is there defined in the system, in the system.



**Hardip Patel** 36:26

No, it isn't yet.



**Tarun Jain** 36:27

You can add it as fast available.



**Hardip Patel** 36:29

No, so we need to give the value number.



**Tirth** 36:30

No, no, no. But we don't need the format instruction because we are adding the pipeline to the parser, so we need to remove it from the system prompt instead.



**Tarun Jain** 36:40

Now that variable will still be there.



**Tirth** 36:43

OK.



**Tarun Jain** 36:46

So OK, now this response is much better because you can directly convert it into.



**Tirth** 36:49

Great. OK. Yeah.



**Hardip Patel** 36:53

Yeah.



**Tarun Jain** 36:56

Model Dum.



**Tirth** 37:00

I think this is better.



**Tarun Jain** 37:02

But here also if you see you got this response right?



**Tirth** 37:05

Hmm.



**Tarun Jain** 37:06

What we usually do is we use parser dot parse and then use this particular response.



**Tirth** 37:10

Yeah.

OK.



**Tarun Jain** 37:16

Sorry.

I overrided the variables. One second.

Chain equals to wrong template. We'll revise this concept. I guess I've repeated multiple variables total times.

This one is clear. I'm not using any parser here. I defined a prompt template. I'm defining a chain without parser and then I'm using chain dot invoke review.



**Hardip Patel** 37:46

Yeah.



**Tarun Jain** 37:54

So now if I print this I'm getting a Jason output. Now I'll use parser dot parse. I will copy this.

I'm getting this response.

So now this entire logic is added in your third pipe.



**Hardip Patel** 38:14

Got it.



**Tarun Jain** 38:17

So let me create certain subbeddings.

Tell here, is it clear?

Direct using LLMS. Can you guys add sub bedding so that it's clearly clustered? So we define base model which is your pydantic. All the keys are defined, then you have prompt.



**Hardip Patel** 38:26

Yes.

Yes.



**Tirth** 38:36

Yep.



**Tarun Jain** 38:42

Then we have LLM and then structured LLM is nothing but in LLM itself we have a function called with structured output. Inside that we pass sentiment analysis. So this is nothing but the schema that you pass.

And then we have invoke function where you can directly add the prompt. This is very straightforward approach if you want to go with LCEL. Now why will we go with LCEL? Just to avoid not using two LLMS. Now here you have two LLMS, one is this LLM and one more is structured LLM. Now for the second approach you directly. Defined output parsers from pydantic and in this pydantic output parser you will have a object which is your schema and here you have an input variable called get format instruction. This thing format instruction.



**Hardip Patel** 39:29

Yeah.



**Tarun Jain** 39:32

Has to be your input variable.



**Tirth** 39:40

Yeah.



**Tarun Jain** 39:40

Till here, is it clear?



**Tirth** 39:42

Yes.



**Hardip Patel** 39:43

Yes.



**Tarun Jain** 39:43

Now here we can create someone sub bedding using parser as.

Parcel within LCL pipeline.

And then we have chat prompt template. I'll remove the prompt template part.

We have system prompt.

And we have the prompt template and in this prompt template if you see we are adding format instruction inside a partial variable. That means when I use invoke I don't have to pass format instruction.

And then this is also not required.

I have chain. Now in this scene I have prompt template, I have LLM, I have parser. If you use this parser and if you directly print it response, you can directly get the model dump. Clear. Is it clear? So since we are adding parser, whatever response you're getting it is in model dump.

So what is the data type of this response?

Data type.



40:53

Dictionary.



**Tirth** 40:53

Excellent.



**Hardip Patel** 40:53

The the the same sentiment model.



**Tarun Jain** 41:01

Uh, it is your schema.



**Ajay Patel** 41:05

OK, I don't need a schema.



**Tarun Jain** 41:05

The parentic schema.

If I use.

The schema.

And then with model dump. Now what is the data type?



**Tirth** 41:26

Jason.

Dictionary.



**Tarun Jain** 41:27

It is in Jason, which is nothing but our Python dictionary.

Till here, is it clear?



**Tirth** 41:34

Yes.



**Tarun Jain** 41:35

So if I just print response.



**Hardip Patel** 41:35

Yes.



**Tarun Jain** 41:42

The response is this one response dump. Now we here I'll create one more sub

heading.

Without.

Using parser in LCL.

So we have the prompt template. Here I'm just defining prompt template then LLM you use an invoke. So when you use the content you have the information inside Jason. So since we have parser which is already defined the pydantic output parser, I have a function called parser.



**Tirth** 42:06

OK.



**Tarun Jain** 42:17

And then I'm just passing Jason dot content and once I do this, I'm getting it in a sentiment analysis which is nothing but your object. Is it clear? We used 3 approaches. The first approach was base LLM.



**Tirth** 42:32

OK.



**Tarun Jain** 42:32

The second approach was parser within the LCL.



**Tirth** 42:36

Mhm.



**Tarun Jain** 42:38

Preferred.

Approach in LCL and this one we just went with the wrong approach, but if you want this to work we just have to use parser dot parse.



**Hardip Patel** 42:55

I just have one question.



**Tarun Jain** 42:55

It was actually not required, but since we took wrong approach we came up with the this particular function, but at the end of the day the results is same.

 **Hardip Patel** 43:04

OK.

 **Tirth** 43:08

Arvind, you have a question.

 **Hardip Patel** 43:11

Yes, and just like so as you said like before that there will be extra thing the ring like there is there is one JSON response or something like that. Will it be also part or is it because that?

 **Tirth** 43:26

Really.

 **Hardip Patel** 43:29

We use the without using password in SE.

 **Tarun Jain** 43:35

I didn't get you. Can you repeat?

 **Hardip Patel** 43:38

So like as I said like there there could be a response where do receive Jason but we also receive extra thing like there is the Jason response under the response that it could it be the reason why we.

 **Tarun Jain** 43:39

OK.

Yeah, yeah, yeah.

 **Hardip Patel** 43:58

The last operation.

 **Tarun Jain** 43:59

This one.

 **Hardip Patel** 44:01

On. Yeah, this one on.

 **Tarun Jain** 44:06

No, there will be definitely times like when LLM will not be able to generate the Jason files. So for this approach as well right? What it will do is it will tell a warning message.

 **Hardip Patel** 44:07

We always.

 **Tarun Jain** 44:19

Which is an error that it was not able to pass and it will return an empty properties. It will be like description is sorry not description product name.

 **Hardip Patel** 44:27

Goodnight.

And.

 **Tarun Jain** 44:32

Product name is.

This then sentiment will be again empty. Then keywords will be like this.

So during this time we usually use rejects. So one thing is for sure, LLM will generate you the response in this format. Jason will be there, but the only thing is you won't be able to parse your information properly.

 **Tirth** 44:58

Welcome.

 **Tarun Jain** 45:05

Which is here you will have.



**Tirth** 45:08

But the parser would take care of it now. So the text that I have given, I actually wanted it to paste over here. So you know, yeah, exactly.

We can take it as a variable instead.



**Tarun Jain** 45:32

So if I bring this.



**Tirth** 45:35

Uh.

Can you take what you're trying to do? Actually, I shared it in the chat.



**Tarun Jain** 45:40

What did you share?

OK, like this, yeah.



**Tirth** 45:47

Yeah.

This would make more sense.



**Tarun Jain** 45:50

This should be abstain.



**Tirth** 45:52

Yes, yes, sorry.

It would still pass it because you know it will remove everything and it will just try to get the Jason out of it to pass it even if you're getting extra text.

It will only continue from the point where.



**Hardip Patel** 46:09

Got it. I think that.



**Tirth** 46:11

Yeah, yeah.

 **Tarun Jain** 46:11

OK, in this function then probably they're using rejects.

 **Tirth** 46:15

Hmm.

 **Tarun Jain** 46:16

Open EGI. So we usually have fall back every time this thing occurs, so utils.

Then we have a little task.

 **Tirth** 46:27

Extraction.

 **Tarun Jain** 46:31

You have this extract Jason from string, so if there is anything that goes wrong, we usually use rejects.

 **Tirth** 46:35

Yes.

 **Tarun Jain** 46:40

Somewhere we use rejects. I don't know if it is in this code or.

 **Hardip Patel** 46:41

OK.

 **Tirth** 46:45

It could be extraction, I guess.

 **Tarun Jain** 46:49

Huh. 4 Jason output.

This is very rare case, like not every time will happen, but it's better to have fallback strategy. So this is just the fallback strategy what we wrote, but not every time this is called.

 **Tirth** 47:00

Right.

 **Hardip Patel** 47:01

Well, we got it.

 **Tirth** 47:06

Right.

I think it is trying to.

You know, extract Jason from rejects and then it will try to parse it thereafter.

 **Tarun Jain** 47:20

Huh. Might be possible because here we didn't use line chain, we wrote everything from scratch.

 **Tirth** 47:26

Yeah.

 **Tarun Jain** 47:28

But yeah, this should work.

Is this clear? Anyone has any doubt now?

 **Tirth** 47:37

Alcott.

 **Tarun Jain** 47:37

So the only thing is make sure we divide this into multiple subbeddings so that you will know which approach to experiment if you face these issues.

I don't think I have to add this.

This is clear. This is clear, OK.

I just repeated some variables 2 two times. The only thing is I'll have to make this response to chain 2.

OK, anyone has any doubts yet or can we proceed?



**Tirth** 48:10

We can proceed, yes.



**Tarun Jain** 48:11

OK, so now for tool calling, what we will do is again we will look at 2 approaches.

One is the existing tools.

That Lanchain provides.

And 2nd is how to build your own custom function tools.

So now why is this useful? Like let's suppose sometimes you have your own custom APIs. This is for one of the retail use case that we had. You had a retail use case where you have functions like list the products.



**Tirth** 48:50

Mhm.



**Tarun Jain** 48:58

In uh.

What was the keyword list products and there was one more keyword that we had used list product and.

Supplied supply chain something. So you had list products, then you also had add products.



**Tirth** 49:19

Thank you.



**Tarun Jain** 49:19

And then there were other two functions. So now what this functions will do is if you want to build an agent and if you want that agent to have flexibility to interact with your APIs, you can't use any existing tools. So what you can do is you will have to build your own custom tools.

So what is available in existing tools? You will have simple search, then you will have YouTube.



**Tirth** 49:36

Hmm.



**TJ** **Tarun Jain** 49:43

Then you have Yahoo Finance.



**Tirth** 49:46

OK.



**TJ** **Tarun Jain** 49:47

Right, just like this if I.



**Tirth** 49:48

This is from. This is from Lens in community now.



**TJ** **Tarun Jain** 49:52

Huh. So in lunch and community, if you see you have Bing search, Brave search, Doug, Doug Go, then EXA. This is only for search. Then for code interpreter you have other code interpreter options. This is what I was mentioning.



**Tirth** 50:03

2.

Yeah.



**TJ** **Tarun Jain** 50:08

So basically when you're building sales agent, right?



**Tirth** 50:11

Mm.



**TJ** **Tarun Jain** 50:16

Your input is CSV file.



**Tirth** 50:34

Yes.

Hey.

 **Tarun Jain** 50:34

So this is executed in a Python interpreter which is a code interpreter. So you can use. There is something called as REPL.

You can use this Python REPL if you see sometimes about complex operation rather than have LLM it will generate the particular code. So you write the code and it will generate the particular code output. So this is the tool and if you see you have Reddit if in case you want to build a customer service chatbot which will take certain Reddit thread and give the feedback or.

Comment automatically. If you want to build any chatbots like that, you can use Reddit search and just like that there are so many. Probably they have at least 80 to 90 tools available with them.

So can just see if in case any tools which make sense is already available, you don't have to write code for it. And now that you also have MCP tool, most of the MCP also have their own logic which is tools. At the end of the day, what is MCP?

 **Tirth** 51:29

Um.

 **Tarun Jain** 51:35

So you have MCP server.

And inside this MPCMCP servers you have multiple tools.

 **Tirth** 51:43

Book.

 **Tarun Jain** 51:43

Which is already written for you, so either you can use MCP at the end of the day, MCP is also a tool.

He is also a fool.

Right, so let's use few of the tools.

I'll go with anything which is free.

Doug Doug. So let's just copy this line.

And probably dug dug go also need a library called DDGS dug dug go search.  
Can you copy this?

 **Tirth** 52:24

Yeah.

OK, done.

 **Tarun Jain** 52:51

OK, so now what we can do is again we have to use Langchain community from Langchain.  
Community.  
dot import dug dug go.

 **Tirth** 53:15

No.

 **Tarun Jain** 53:20

And it's very simple, you just have to define search equals to dug dug go search run.

 **Tirth** 53:29

You.

 **Tarun Jain** 53:29

And the function is similar to what we have in uh LLM which is search dot invoke.  
What is currently trending the CO summit?

 **Tirth** 53:39

Trump Tariff.

Then that is an SEO submit.

 **Tarun Jain** 53:47

So now if you see two hours ago the SEO Titan Summit 2025, there are a very lengthy response. So now when you build agent right, So what agent will do it will if there is any real time query it will directly use this search tool.

And search tool will generate the context. So this is the context that it is providing and usually one of the better search tool that we have is something called a stable.

 **Tirth** 54:18

At least.

 **Tarun Jain** 54:21

So tably itself is an agent when it comes to Web search. It's API based to be honest. And apart from tably there is something called as EXF.

 **Tirth** 54:27

OK.

 **Tarun Jain** 54:35

So these are two.

 **Tirth** 54:35

What about parallel web? We discussed about it now parallel web systems. They are also giving I think this kind of search tools and it has the highest. You know non blocking ratio on the Internet.

 **Tarun Jain** 54:49

This one.

 **Tirth** 54:51

Yeah.

It creates agents for web browsing.

 **Tarun Jain** 54:57

OK, it does RADXL.

 **Tirth** 55:00

Hmm.

 **Tarun Jain** 55:07

But I've not seen this before.

 **Tirth** 55:08

OK, OK.

 **Tarun Jain** 55:15

Highest accuracy Web search.

Docs.

How we can try this so here?

 **Hardip Patel** 55:28

They say that, but does anyone else say this? They they write it on their site, but does anyone say this except themselves?

 **Tarun Jain** 55:32

Oh, what happened?

 **Tirth** 55:34

I don't know.

Yeah.

We can, we can try, you know, but Tarun has the experience, so we can try and test right now.

 **Tarun Jain** 55:48

Uh, we can test it out. So here if you see custom function is a right custom function, we can write any logic. The only thing does anyone remember how to write custom functions?

 **Tirth** 55:48

Baby.

Um.

 **Tarun Jain** 55:59

Let's suppose I want to build an agent and if I want to build a custom function, how will I build it?

 **Hardip Patel** 56:04

The space. Yeah, this.

 **Tarun Jain** 56:07

Let's suppose I want to use this parallel web. Advanced search is my function then.

 **Tirth** 56:14

Then you have to give query and then you have to define that query with annotation. What does this query is?

 **Tarun Jain** 56:18

Correct. So we have to define query annotation then, which is the important thing, two important things.

 **Tirth** 56:25

Uh, the metadata, the metadata that you have to give and the return type.

 **Tarun Jain** 56:31

Return type is very important. We have to define return type as string and then we have to use this doc string. The big doc string should tell when it has to use this particular search.

 **Tirth** 56:36

Yes.

 **Hardip Patel** 56:38

Documentation.

 **Tirth** 56:40

Stockstream. Yes, it is.

 **Tarun Jain** 56:48

So this.

Tool needs to be used when you need to browse information.

From that.

Not sure how good this is. I'll just copy this code.



**Tirth** 57:21

And.



**Tarun Jain** 57:22

And I will paste it here.

But everyone understood this logic is same, so you can use any other tool as well.

You can start with YouTube search tool, but for YouTube we'll have to use.

PIP install.

YouTube search if I'm not wrong and then the logic is copy this YouTube search, add it here and then invoke. So this logic is repeated for every other tool as well. So if I pick Tableau search.



**Tirth** 57:52

Inch.



**Tarun Jain** 57:57

It's the same logic again. You have to import Tabli, so they've added in lunch and Tabli, but you can also use it from community tools.



**Tirth** 58:08

M.



**Tarun Jain** 58:09

So if you see you have Table search results and if you're using anything that is related to API based, let's suppose you have from lunch and community tools Table. Table is not free, it needs API key. So what will you do? You will just define OS dot. Environment check what is the variable that they're using. So the variable is Table API key. Copy that, paste it here and attach your keys and there is also one more important one which is EXA.

For EXA also, if you look at the logic, you have to copy this EXA API and these folks

have added separately. If not, you can also directly use it from community tools if I type EXA.

OK, EXA is not there. So in that case what you have to do is you directly have to install this from Lancet EXA import EXA search tools and instead of search tool you can use a function called invoke. So dot run is not required, you can use invoke.



**Tirth** 59:10

M.

1.



**Tarun Jain** 59:20

You understood this logic how to use the existing tools which is provided by Lang Lang chain.



**Tirth** 59:23

Yes.



**Hardip Patel** 59:27

Yes.



**Tirth** 59:28

Hmm.



**Tarun Jain** 59:30

So now what we will do is let's try to build our own custom tools. This is what you will be doing because most of the time whatever tools is available, you will not use that. That is only for testing purpose. So now how do we install this?



**Tirth** 59:41

M.



**Tarun Jain** 59:49

I'll also copy this command here tip install parallel web.

Let me only create the API key so that we can reuse the same API key.



**Tirth** 1:00:05

OK.



**Tarun Jain** 1:00:06

Yeah.

If anyone already have this account, you can use your API keys as well.

Has anyone used this before?



**Tirth** 1:00:17

No, I was just looking at news updates and this is, you know, I said before from ex CTO of Twitter.



**Hardip Patel** 1:00:25

Yeah.



**Tarun Jain** 1:00:27

The.

Oh, anyways.

So now import OS.

OK, not importers. They want the API key to be pasted here.

Processor is base. OK, so the there are two processor, one is base and pro. So pro it takes too much of time. So what I'll do is I'll use the base one itself.

Base then task run execute so this will be query.

Now what is their output?

Search.



**Hardip Patel** 1:02:43

There is a like in parallel web.



**Tarun Jain** 1:02:55

They have the tool here.



**Hardip Patel** 1:02:59

Yeah, it shows on.

 **Tirth** 1:03:00

They have tool.

 **Hardip Patel** 1:03:03

I will be the car.

Yeah, uh, just below the.

 **Tarun Jain** 1:03:14

Where is it?

 **Hardip Patel** 1:03:24

He's just like 10 days old, I guess.

 **Tarun Jain** 1:03:26

Particular whether however we want to check this logic, right? How to build our own custom function?

 **Hardip Patel** 1:03:29

Yeah.

Yes.

 **Tarun Jain** 1:04:13

No objective is.

 **Tirth** 1:04:27

Objective is query done.

 **Tarun Jain** 1:04:30

Oh, this should be query, but what about search queries then?

 **Tirth** 1:04:35

It's not going to be very as well, a single query, just saying.

We can try that because we are not dividing it into multiple query format.

 **Tarun Jain** 1:04:46

Objective is limited to 500 characters search queries.

String national query description of what the web search goal is. Include any source of freshness guidance. This is optional, so I'll just remove this.

 **Tirth** 1:04:57

Actually, yeah.

OK.

 **Tarun Jain** 1:05:01

Optional search queries to guide the results. So if in case I want to give any additional queries then I can give. So let's suppose I want to ask anything related to cricket. So what I can do is use.

 **Tirth** 1:05:04

Hey.

Um.

M.

 **Tarun Jain** 1:05:19

Trusted source.

From.

ESPN or Cricbuzz?

And I will remove this. Now this is same processor. You have only two, one is base and one more is pro. Max results is like. This is common if you're using Google search or if you're using Doug Doug. Every single search tool has Max results.

 **Tirth** 1:05:40

Um.

 **Tarun Jain** 1:05:47

So if you ask any query, let's suppose ask black hole. Now what will happen? It will go to that search, it will get what are the top ten search results and then from the top ten you will have title, you will have description.

And you will have URL right? And this will repeat for 1010 search results. And this parameter I'm not sure why they've added. Edit is same here we paste our API key content type by need in Jason and then since we are using request you have different function.

 **Tirth** 1:06:07

M.

8.

 **Tarun Jain** 1:06:21

One is you have request dot get for get function. If you want to post you can use request dot request post. For example you have request dot get. Here you can add any website and get the information of it.

And if you want to post it, you can use request dot request.

So now if you look at the custom function logic, this is it. So you have to define a variable, then query which is the data type and then give the doc string. Doc string. Even if you give 3 lines that is completely fine. You can tell what the query is and what you should return.

The top 10 results for the given user query.

The user input for which the web browser.

Search needs to be used. Is this clear?

 **Tirth** 1:07:23

Yes.

 **Tarun Jain** 1:07:25

Uh, let me play this here.

 **Tirth** 1:07:25

We we still have to return the response dot text.

 **Hardip Patel** 1:07:25

Yes.

 **Tarun Jain** 1:07:31

Uh, this should be written.

Just make sure you change this line. Here I've added print.



**Tirth** 1:07:39

Um.



**Tarun Jain** 1:07:40

And let's just see if this is working or not.

Advanced search.

First.

01 DPL 125.

Invalid EKG both.



**Tirth** 1:08:09

Will you put the dollar?

No, no, it was not the dollar. We are doing it with the dollar.



**Tarun Jain** 1:08:14

No, they've added dollar.



**Tirth** 1:08:17

Because that is variable.

Yeah.



**Tarun Jain** 1:08:27

And now you have results.

So the length of this thing.

So you understood what I did here. So when I print test it is in string but I don't need string. I want to see what is currently happening. So here if I use eval eval will convert your string into dictionary if in case your output whatever you have right.



**Hardip Patel** 1:09:15

Yes.



**Tirth** 1:09:22

E.

Um.

TJ

**Tarun Jain** 1:09:29

If it is in dictionary, you can use this function called eval.

Eval test.

Of results.



**Tirth** 1:09:39

Is it not unsafe to use eval? It is for PHP and JavaScript. Just asking like would it not do a system level eval or something?

TJ

**Tarun Jain** 1:09:49

Uh, in the sense this is just for type conversion.



**Tirth** 1:09:52

OK, OK.

TJ

**Tarun Jain** 1:09:55

So now if I print the length of this, it should be 10 because my maximum results is 10.

So this is what every single tool will also do. So here if you see you have URL, you have title, then you have exerts.



**Tirth** 1:10:06

Um.

TJ

**Tarun Jain** 1:10:13

Inside exit you have some information, but there are too much of garbage results.



**Tirth** 1:10:20

Yeah.

Play Remove the.

Uh, search query for the support and it returned me pretty.

Put output.

 **TJ** Tarun Jain 1:10:41

OK, I think who is there, right?

 **Tirth** 1:10:44

Yep.

 **Hardip Patel** 1:10:46

Yeah.

 **Tirth** 1:10:46

One DPL 2025 and then it gave me proper output.

 **TJ** Tarun Jain 1:10:54

But still, if you see it's returning this JavaScript void and all, are you able to see this?

 **Hardip Patel** 1:11:00

Yes.

 **Tirth** 1:11:01

E.

 **TJ** Tarun Jain 1:11:02

So if I check the zeroth index and the keys of it.

 **Tirth** 1:11:07

Thank you.

 **TJ** Tarun Jain 1:11:08

I only have you are in petal and exert.

 **Tirth** 1:11:08

And because this is except.

This is it, yeah.

 **TJ** Tarun Jain 1:11:14

So if I check, here is where you have your actual data, right? Oh.

 **Tirth** 1:11:26

I don't think we need it.

 **TJ** Tarun Jain 1:11:26

No, this is this will confuse LLM and this is too much of token.

 **Tirth** 1:11:30

Yeah, right. I don't think we need it except.

 **TJ** Tarun Jain 1:11:33

What if I make this as 1000?

Because now if I give thousand and if I'm giving 10 result query, how much input tokens am I passing? I'm passing 10,000. So this I'll make it 5 and this I'll make it 800. So now it's like 4000 which is fine.

 **Tirth** 1:11:40

Um.

10,000.

2.

 **TJ** Tarun Jain 1:12:02

No, I don't. It will still generate the response. So what we'll do is let's try this with an agent. Everyone understood this is it. This is how you define your own function. Simple Python function with a.

 **Tirth** 1:12:10

M.

Yes.

 **Hardip Patel** 1:12:13

Yes.

 **Tarun Jain** 1:12:18

Context. This context can be very lengthy if you need. So this is your actual prompt that you have to pass to the agent till here is it done.

 **Tirth** 1:12:22

Um.

Yes, it is.

 **Tarun Jain** 1:12:27

So now what we will do is tools is mainly used with agents.

Tool calling is mainly used with so or function calling.

So let's import Langrah.

So what we will do is we will use Doug Douggo and we will also use this advanced search with agent and check the response and we will use the same query and see which is performing better, which is who won DPL 2025 and who was.

You ordered.

As the man of the match.

I know very much.

So we will try that with langref.

So in Langraf I did I tell you a word called react before?

 **Tirth** 1:13:31

We discussed it yesterday.

 **Tarun Jain** 1:13:31

Yes.

 **Hardip Patel** 1:13:32

Yes.

 **Tarun Jain** 1:13:34

Uh, so basically React is nothing but.

Reasoning and action. So let me tell you one example. Let's suppose the time is.

What is the time now? It's 12:48, right? If I go to GPT and if I ask good morning, what will be the output?

 **Tirth** 1:13:54

Good morning.

 **Tarun Jain** 1:13:55

It should be good morning. Hey, good morning. But in my thing it should not be good morning because I've added personalization.

 **Tirth** 1:13:56

Yeah.

 **Tarun Jain** 1:14:04

What the hell is?

Settings personalization.

OK, it's still. If you see their instruction is there, look at the current time and be smarter to answer.

 **Hardip Patel** 1:14:29

It.

 **Tarun Jain** 1:14:30

But ignore this, I'll come back to the example. So basically let's suppose you have 12:48. If it has good morning to basic LLM model, no matter what happens, it will return you back good morning.

 **Tirth** 1:14:31

He.

Hmm.

 **Tarun Jain** 1:14:43

So if you tell the same time 12:48 to react agent.

With prompt.

And the tool. So what tool do you think is relevant here?



**Tirth** 1:14:59

Within time tool.



**Tarun Jain** 1:15:03

Either you can use search or it can be calendar and calendar is the most preferred tool. So now what will happen is there are loops that it will run into. Let me also show some slides.

So basically how React works is let's suppose the first thing is you have the user query. In our case it is good morning. Now first thing what it will do is it will generate a thought. This thought process it will try to reason itself on what it needs to generate the response to the user.

It will think and then it will look at the tools that are available to it. So how many tools do we have here?



**Tirth** 1:15:53

Search Calculator code execution.



**Tarun Jain** 1:15:55

You have web search, you have calculator, you have code executed. So for the given Query Which tool do you think is most relevant?

It is search. So what it will do? First it will think, then it will use an action and once it generates the action, it will use observation. Observation is where you actually have the final results, right? But what React does it, it doesn't stop there.



**Hardip Patel** 1:16:07

Vote.



**Tarun Jain** 1:16:21

It will see if the goal is met or not. When it checks the goal is met or not, you use input variables. So here the input variable is just good morning. So is it a right thing to generate the response to the user? It will think if no, it will iterate.



**Tirth** 1:16:21

Yeah.

TJ

**Tarun Jain** 1:16:38

For example, you have maximum iteration which is five, right? If the goal is not met, that means whatever response that was supposed to be generated, it was not right. So it will do thought action observation. Is this the final response? If yes, it will generate the response to the user. If no, it will iterate plus one and then it will loop it. So until what it will do until it generates the right response to the user, it will keep on looping and if the maximum maximum iteration is reached, it will force the output.

Is this clear?



**Tirth** 1:17:12

8.



**Hardip Patel** 1:17:14

Yes.



**Tirth** 1:17:14

Yeah.

TJ

**Tarun Jain** 1:17:14

We'll again talk about React when we talk about agents in detail, but this is just simple understanding thought, action, observation. If the answer is met, then final answer. If not, again thought, action, observation.

So this is agent with feedback which most of the libraries are using. Crew AI by default it is React Langraf. If you want to use agent, it is React. Even in our library Open AI, where is the prompt?



**Tirth** 1:17:34

OK.

TJ

**Tarun Jain** 1:17:45

Prompts if I check the task execution.

Can you see these keywords?

Thought action then observation. What is happening in action? We are telling hey we

have tons of tools and the if you want to call a tool which you should call in this approach which is class. Then you have Doug Douggo. So this is just an example. Since we have multiple tools you can pick any tool.  
Which is relevant and then you have to get the module. So now what will happen when you get this module right? You can also get this doc strings.  
So this is a common format in most of the agentic frameworks and it was developed by Google and Google ADK also uses React. So this is the basic prompt template you will find across most of the agentic application.

 **Tirth** 1:18:34

OK.

M.

Yes.

 **TJ Tarun Jain** 1:18:45

So let's come back here. What you can do it. Oh yeah.

 **Hardip Patel** 1:18:45

That is the the reactor. Is it like a a protocol or something like that or is just like the discipline?

 **TJ Tarun Jain** 1:18:58

It's a prompt engineering.

 **Hardip Patel** 1:19:01

Is this wrong? OK.

 **TJ Tarun Jain** 1:19:03

Yeah, but heavy prompt with some kind of engineering.

 **Hardip Patel** 1:19:06

OK, so like what I'm I'm asking is like if we do it on ourselves, is that fine or is it something some like maybe on something that we we look for implement to do that?

 **Tarun Jain** 1:19:21

Oh, can you repeat?

 **Hardip Patel** 1:19:23

OK, so now uh it.

 **Tirth** 1:19:23

So does it require a library or something or we can just?

 **Tarun Jain** 1:19:25

No, no, no. Uh, so this you can do it by yourself.

 **Tirth** 1:19:29

So what I'm what I'm interested in, how does it look through?

 **Hardip Patel** 1:19:29

OK, got it.

 **Tarun Jain** 1:19:30

Yeah.

So.

 **Tirth** 1:19:35

Like do you write it in the prompt? Let go to step one or something like that.

 **Tarun Jain** 1:19:39

No, looping is like you usually have an output key, right? When you generate a observe observation, each observation will have a final key. Where is that excel?

 **Hardip Patel** 1:19:52

So and like what I'm doing on my end is like W5 line items, the invoice does not have zero line items. I will look if there is zero line item then I will fall back to OCR.

Could it be considered as a react or not?

 **Tarun Jain** 1:20:13

Oh, sorry, I didn't catch up. Uh, but once again, I'll just, uh, tell this one.

 **Hardip Patel** 1:20:14

So.

 **Tirth** 1:20:15

So if you're breaking up with the noise isolation, if you're breaking up a bit with the noise isolation, so you know.

 **Hardip Patel** 1:20:22

OK. OK. Sorry.

 **Tarun Jain** 1:20:24

OK, so I'll just explain this. You have thought you have action, you have observation. So every single observation will have a key which is answer key. If this answer key is true, that means you're stopping your final response. If this is false, then you will have your own looping.

 **Tirth** 1:20:29

M.

 **Tarun Jain** 1:20:40

You will iterate this process while nice iteration.

Uh, what is the loop it will be?

 **Tirth** 1:20:52

Do you give the output back to the same prompt or something?

 **Tarun Jain** 1:20:59

Oh, what?

 **Tirth** 1:21:00

So when you get the output from the observation or from the action, do you give it back to the prompt and then check for whether it is true or false?

 **Tarun Jain** 1:21:04

Yeah, yeah.

Yeah, yeah, so now this observation is your input variable for your next thought.

 **Tirth** 1:21:17

OK.

 **Tarun Jain** 1:21:18

So it's like thought, action, observation. If you found the output then you'll have final answer.

If not, it will again use thought.

So now how will thought generate its reasoning? It will generate reasoning based on the observation that you get. So this is heavy prompting. If you see the prompt line, it's like.

Roughly it's more than 60 lines of prompt and I'll also show hugging face. Hugging face is more than us.

Uh, GitHub.

So they have small agents. Where is that prompt?

Fronts.

What agent? Can you see these keywords?

Thought then their action is nothing but they're using it from their own thing, then observation. So this is standard. React is a standard prompt template across every frameworks and if you look at their prompt.

 **Tirth** 1:22:24

Yeah.

 **Tarun Jain** 1:22:41

OK, it's still going.

 **Hardip Patel** 1:22:44

Mhm.



**Tirth** 1:22:44

That is a big prompt.



**Tarun Jain** 1:22:46

This is 300 lines of prompt.

OK, this is system prompt, but you can see how big the prompts are and this is not just with small uggings face or us. If you are writing React prompt, it's very lengthy.



**Tirth** 1:22:56

Yes.



**Tarun Jain** 1:23:05

I'm not sure if crew AI have made it public crew AI.

Oh yeah.

These folks have hidden their prompt. It's very difficult to find their prompts.

Yeah, yeah, it's very difficult to find where they have done the prompt, but they're using React. If you run crew AI, right? And if you check the output, you will see these three keywords, thought, action, observation. If you see these three keywords, it's self-understood, it's React.



**Tirth** 1:23:37

It.



**Tarun Jain** 1:23:47

And not to worry, we'll build React from scratch that is added in the first module itself when we start with agent.

So from Langraf.

Till here everyone got the results. Did you get the results from?

The advanced search tool.



**Tirth** 1:24:06

Yes, yes.



**Tarun Jain** 1:24:07

OK, now what we are trying to do is we have our agents. I mean we have the tools, one is the own functions that you created and one from the line chain. So now what we need to do is the goal of one tool is to be passed to the agent. So now we just have to use from Langraph.

dot they have a function called prebuilt and from here you can use create react agent.

This is done these two lines of code.



**Tirth** 1:24:50

Yes.



**Tarun Jain** 1:24:51

And here you just have to define an agent. I'll define agent one or I'll tell existing. It's better to use agent one. Agent one is for existing tool create react agent and define LLM and every single tools has to be passed inside a list.

So what tool do we have? We have search.

Search is nothing but Doug Doug Go. If I'm not wrong, it's Doug Doug Go search.

So you have can you see the maximum results? This is a common variable.



**Tirth** 1:25:34

Yes.



**Tarun Jain** 1:25:36

So create React agent LLM and then search. That's it.

And now the same old function which is agent 1.

dot invoke.

Huh. Who won DPL?



**Tirth** 1:25:53

We created this OK.



**Tarun Jain** 1:25:57

What happened?



**Tirth** 1:25:58

We created this agent and we're passing one tool for search that is the duck. We're not passing the one that we built right now.

 **Tarun Jain** 1:26:05

Yeah.

No, that will lower conflict rate because both are searched.

 **Tirth** 1:26:08

OK, OK, OK.

 **Tarun Jain** 1:26:11

So if I give search and then if I give Yahoo finance then it makes sense.

 **Tirth** 1:26:16

Hmm. Understood.

 **Tarun Jain** 1:26:17

Because now what will happen? If you ask anything related to stock price, it should typically use Yao Finance.

 **Tirth** 1:26:21

E.

 **Tarun Jain** 1:26:26

Here the goal is to understand that OK, I'm using your existing tools.

And when I define agent 2, I will use the own tool. And how did we define our own tool? It's just simple function where you have the logic. So what do you want to return? This is what is context engineering.

Like how you how well are you passing the context to the LLM. So here if you see we are passing garbage result.

So we just want to filter this out when we, uh, use this response only.

 **Tirth** 1:27:06

Um.

TJ

**Tarun Jain** 1:27:10

Agent one create React agent, then LLM, then search and now response one.

OK, they should be inside messages.

Messages.

Pop.

Yeah, you just have to use agent 1 dot invoke add messages. Inside messages you have your query and if I do response one, what is the piece that we have?

It's messages.

Messages of -1 content.



**Tirth** 1:27:57

DPL 2025 has not happened yet.

TJ

**Tarun Jain** 1:28:02

Now let's try with advanced search.

Advanced search.



**Tirth** 1:28:08

Oh.

Mention DPL OK.

TJ

**Tarun Jain** 1:28:17

Agent 2.

Response to I hope I'm not passing more context by 800 OK.

Where is double check? Because if you give 10 and if you give 6000 that is like 60,000 tokens you are passing, it will hit an error. It will rate limit.

So if you check the code that I shared there you have 10 and 6000 which is too much of input tokens.

It did get Nitish Kumar Nitishana's correct response, but it was not able to find the proper search results.



**Tirth** 1:29:15

And I was always doing that.

 **Tarun Jain** 1:29:17

Now let's try one more search.  
Let's install.  
By country.  
Did anyone get results?

 **Tirth** 1:29:35

I got the result. West Delhi Lions wants DPL 2025 and Nitish Rana was rewarded as the man of the match.

 **Tarun Jain** 1:29:36

Or are you getting?

 **Tirth** 1:29:44

The performance of 79 of 49 balls.

 **Tarun Jain** 1:29:44

So here, if I give premiere, you even got 79.

 **Tirth** 1:29:49

Yeah, 79 of 49 balls.  
I I.

 **Hardip Patel** 1:29:52

I didn't get answer for from both of the parallel as well as.

 **Tarun Jain** 1:29:59

What temperature have you set? Oh wait, what is my temperature value of alum?

 **Tirth** 1:30:05

The default one would be 0.7, yes, but we didn't set anyone explicitly, didn't 0.7.

 **Tarun Jain** 1:30:17

OK, then why did it give me wrong results 0.01?

OK, this is not forced.

It's DPL only, right? Or is it?

 **Hardip Patel** 1:30:50

M.

If it if it is saying sorry to I cannot find any.

 **Tirth** 1:30:51

Yeah.

I don't need anything this week.

 **Tarun Jain** 1:31:06

OK, it is supposed to tell W Delhi if I'm not wrong.

 **Tirth** 1:31:10

I go to West Delhi.

 **Margi Varmora** 1:31:14

You know, DPS stands for disabled people, cricket reach, so.

 **Tarun Jain** 1:31:21

No, if I give Premier League also, it didn't give me the response. This is very weird.

 **Tirth** 1:31:28

It hasn't happened yet.

 **Tarun Jain** 1:31:30

Let me use Google search.

Yes.

 **Hardip Patel** 1:31:38

Can we check if if the?

 **Tarun Jain** 1:31:39

So now if you look at the logic it's still the same. I'm defining Google search then

what is this? This is a custom function and in custom function I'm just adding this advance to be true and the logic is same. Maximum result is there and if you notice now.

 **Tirth** 1:31:43  
E.

**TJ** **Tarun Jain** 1:31:56  
I'm filtering out here only. So what is happening here?  
Here if you see I'm adding URL, I'm adding search ID. I don't need all these details.  
What I need is I directly need only description. So if I print this result.

 **Tirth** 1:32:10  
Yeah.

**TJ** **Tarun Jain** 1:32:13  
This result will have URL. It will also have title.  
But what am I doing? I'm creating a variable called context and it's an empty string.  
I'm only appending description.  
I'll send this code as well.

 **Tirth** 1:32:40  
You don't need API key for it.

**TJ** **Tarun Jain** 1:32:43  
No, no. So here if you see.  
If you use the Google search from Land Chain, you will need the API key.

 **Tirth** 1:32:52  
Right.

**TJ** **Tarun Jain** 1:32:53  
So we created our own custom tool. In this custom tool it's not there, so this tool is not supported in Langsin.

 **Tirth** 1:32:56

OK.

OK.

OK.

 **Tarun Jain** 1:33:06

OK, so now what is the tool name? I have web search. First I'll just test it DPL 2025 and man of the match.

So if you see it's printing URL, it is printing. What is this title? URL title, URL title. But description is what I'm upending. If I just use K equals to web context, now this K is my actual context.

 **Tirth** 1:33:30

Hmm.

 **Tarun Jain** 1:33:38

Is this clear? Now if I use web search which is my agent 3.

It was to create.

React agent LLM web search. Now web search is the new custom function. You understood it. When you guys will build your own agent, you will have to create your own custom functions to have more flexibility and better context.

 **Tirth** 1:33:57

Yeah.

M.

 **Tarun Jain** 1:34:07

So response 3 equals to Agent 3 invoke phone DPL 2025 and was rewarded as one of the match.

Messages.

 **Hardip Patel** 1:34:29

Still sorry.

 **Tarun Jain** 1:34:33

It has the context here.

 **Tirth** 1:34:37

The LLM is not good. You have to change it.

Let me change the LLM. Let me see where the LLM can go wrong.

So we are doing flash and then flash we can go to.

 **Tarun Jain** 1:34:53

OK.

 **Tirth** 1:35:00

Oh.

Temperature 0.7.

They're not working, I think, but the DPL turn has not yet has been held, so there is no winner.

 **Hardip Patel** 1:35:26

OK, I got the answer, so I changed it to 0.7.

 **Tirth** 1:35:29

What is the temperature?

OK, OK.

And you changed it to.

 **Tarun Jain** 1:35:36

Oh, I I have with my API limits.

 **Hardip Patel** 1:35:40

Uh, but I changed it to 0.7 on the.

 **Tarun Jain** 1:35:46

This.

 **Hardip Patel** 1:35:47

But now I got the answer.

 **Tarun Jain** 1:35:48

No.

 **Hardip Patel** 1:35:53

I ran, by the way, I ran complete everything here.

 **Tarun Jain** 1:35:56

And one more thing, what you can do is let's add verbose here. Let's check if there is verbose.

 **Tirth** 1:36:01

Don't want a bus.

 **Tarun Jain** 1:36:05

Is there a function called verbose?

 **Tirth** 1:36:09

No Deepak.

Oh.

 **Tarun Jain** 1:36:16

What goes to be true?

 **Tirth** 1:36:27

It is Deepak.

 **Tarun Jain** 1:36:37

So human message is this content function calling. It is using web search and after function. Oh wait, why is it printing all this? OK, I use that function.

And now tools.

No, it is getting. If you see here, the content has the Delhi Premier League was

concluded. The West Delhi Lions clinched the 2025 title. Then why am I not getting the result?

 **Tirth** 1:37:04

'Cause you're looking at the last message.

 **Tarun Jain** 1:37:07

For last message only you'll have the response messages -1 dot content.

 **Tirth** 1:37:10

OK.

 **Tarun Jain** 1:37:19

OK, this is very weird.

 **Tirth** 1:37:23

No, but why is it not calling the tool? For me it is not calling the tool.

 **Tarun Jain** 1:37:28

Can you add debug and check this?

 **Hardip Patel** 1:37:29

I I have. I have reached my limit. It shows that.

 **Tirth** 1:37:34

I'm adding debug but it is not calling the tool.

 **Tarun Jain** 1:37:39

You are not able to see any confidential here.

 **Tirth** 1:37:40

So who want? Yeah, yeah, like it is not calling the tool any function calls.

 **Tarun Jain** 1:37:47

Oh, can you copy this and send it to me whatever you're getting it here?



**Tirth** 1:37:53

Yeah.



**Tarun Jain** 1:38:02

So here I'll add more customization. What I'll do is I'll add results dot title also.



**Tirth** 1:38:14

That's the.



**Tarun Jain** 1:38:15

F string.

This is the description.

And this is title.

And.

Now try with this function.



**Tirth** 1:39:02

Calling it.



**Tarun Jain** 1:39:06

Yes.



**Tirth** 1:39:07

And the text is like it's not calling at all values updates if you see.



**Tarun Jain** 1:39:32

Here, what are the parameters we have?

Molay prompted.



**Hardip Patel** 1:39:56

OK.



**Tarun Jain** 1:40:17

OK, now I'm getting the results. Can you try with this new updated tool everyone?



**Hardip Patel** 1:40:23

So let go. Even without changes, I'm getting the result.



**Tarun Jain** 1:40:27

No, but it should be consistent, right? Like.



**Hardip Patel** 1:40:30

Yeah, it is consistent. I tried three times.



**Tarun Jain** 1:40:34

OK, others also, can you confirm?



**Tirth** 1:40:35

I'm still getting some error.

No, no, I'm getting some error. Error type error search got an unexpected keyword argument.



**Tarun Jain** 1:40:45

Oh, you changed anything or?



**Tirth** 1:40:48

No, no, I didn't change anything. I just copy pasted what you shared. Let me try with the new copy paste.



**Tarun Jain** 1:40:57

First what you can do is you can just test the normal web search if you're getting the results or not. Once you're getting the result then you can pass it to the agent.

Others, can you confirm if you're getting the right results or even you're getting wrong results?



**Tirth** 1:41:33

It's not calling the tool for some reason at mind.

There is no tool calling.

 **Tarun Jain** 1:41:43

Oh, are you overriding any variable or probably you'll have to restart this?

 **Tirth** 1:41:45

No, no, no. I just cropped it. OK, I see. OK.

 **Tarun Jain** 1:41:58

There is also a parameter called region, so if I pass region.

 **Tirth** 1:42:06

M.

 **Tarun Jain** 1:42:07

Equals to India.

Region equals to region.

 **Tirth** 1:42:13

Uh, for Google search then we have to restart the session this.

 **Tarun Jain** 1:42:22

There is one more parameter you can give region. I give region equals to IN. Region.

One is language country codes we can give.

And country code for India is IN.

Oh, are you?

How many of you got the results? The correct results?

 **Tirth** 1:43:01

Just this again.

 **Tarun Jain** 1:43:02

You can raise your hands if it is echoing.

Just one.

 **Tirth** 1:43:13

Yeah, hold on.

 **TJ** **Tarun Jain** 1:43:17

Ayush, Mitesh, Margi, Ronak, you guys didn't get the output.

 **Ishan Chavda** 1:43:24

Uh.

 **Ayush Makwana** 1:43:24

No, actually I hit the limit. I'm trying again by with another ID.

 **Hardip Patel** 1:43:30

OK, so if you hit the limit then it will wait for a second and try again.

 **Tirth** 1:43:31

I'm I'm not getting the output.

 **Hardip Patel** 1:43:39

Yes, on my end it is.

 **TJ** **Tarun Jain** 1:43:40

You guys are using Pro or Flash?

 **Tirth** 1:43:42

Less.

 **TJ** **Tarun Jain** 1:43:45

Uh, let me.

 **Hardip Patel** 1:43:45

I'm using brown.

 **Tirth** 1:43:50

I'm following the tools.

It.

 **Tarun Jain** 1:43:59

If I just use 4/1.

 **Tirth** 1:44:08

Well, I got the right result now. If I use pro, it is getting the right result. If I use flash, it is not getting the.

 **Tarun Jain** 1:44:13

I want the flash. So if you see I'm using flash. It told the West Delhi Lions on the DPL 2025 title defeating the Central Delhi Kings by 6 wickets in the final. Then Nitish Rana captain.

 **Tirth** 1:44:16

Yeah, Flash is not even calling tools.

Hmm.

 **Tarun Jain** 1:44:32

And also the player of the tournament.

 **Tirth** 1:44:35

For pro I got the result. It was calling the tools. Flash was not calling the tools for me. Beard.

 **Tarun Jain** 1:44:51

So what were the tweaks we made? The first tweak we made was we improved our tool calling. We added more context and if you look at the parallel web that we tried in parallel web we are giving search ID, we are giving.

 **Tirth** 1:45:02

M.

 **Tarun Jain** 1:45:06

What do we have in parallel ID? You have search ID and you have results and inside results if I check the keys.

 **Tirth** 1:45:11

Yeah.

 **Tarun Jain** 1:45:18

I have URL, I have title, I have exert. So how will my logic change here?

Let's suppose I want to change the logic of advanced search and I only want to pass right context. So what will I do here?

What will be the logic?

 **Tirth** 1:45:39

So you would you would ignore, you would have the title you would have.

 **Tarun Jain** 1:45:45

So for response in.

Response dot TXT. So what is the data type of this response dot TXT?

The.

 **Margi Varmora** 1:46:00

List.

 **Tarun Jain** 1:46:00

What type?

 **Tirth** 1:46:01

Bing.

 **Tarun Jain** 1:46:03

It is in. It's in dictionary.

 **Tirth** 1:46:05

It is very string.

 **Tarun Jain** 1:46:07

Oh yeah, it's.

 **Tirth** 1:46:08

No, but it is. It is not yet converted. It is string so we have to 1st do e-mail.

 **Tarun Jain** 1:46:16

So let me check this thing. You also have a function called Jason.

 **Tirth** 1:46:21

Oh, OK.

 **Tarun Jain** 1:46:22

So now what we can do is instead of text if I use Jason and then if I use results.

So from what do we need to do?

 **Tirth** 1:46:31

Then it is list.

 **Tarun Jain** 1:46:36

I have to get context plus equals.

 **Tirth** 1:46:39

Hmm.

 **Tarun Jain** 1:46:42

Title.

 **Tirth** 1:46:43

Hey.

 **Tarun Jain** 1:46:46

F string response of.

Title and then description.



**Tirth** 1:46:52

Take that.

Response of Excel.



**Tarun Jain** 1:46:56

Responder.

Exerts and then return content.



**Tirth** 1:47:00

But then except is a list, so you would have to combine it now like join.

Except this list.



**Tarun Jain** 1:47:10

OK, so it would be.

dot join.

Interesting here will give space.



**Tirth** 1:47:18

Yeah, yeah.

Yeah.



**Tarun Jain** 1:47:24

And now if I use advanced search, I should only get text.



**Tirth** 1:47:30

It.



**Tarun Jain** 1:47:32

And I will use agent 2 which was using advanced search.

It's still not giving me the results, but for web search it gave.



**Tirth** 1:47:44

But try with pro, try with pro it would work. This is not doing tool calling. I observe that.

Flash is not doing tool calling properly.

So now your tool is getting called and you would get the response after that.

 **Tarun Jain** 1:48:07

I just hope it doesn't hit the rate limit because I got the rate limit last time.

 **Tirth** 1:48:13

OK.

Just two seconds.

 **Tarun Jain** 1:48:22

Why? How much time did it took for you? I did the limit. I knew it. Let me change your API key.

 **Tirth** 1:48:31

I think I'm using your API key.

 **Tarun Jain** 1:48:43

Voice start environment Google API key.

How much again did it took for you? You will see the green.

 **Tirth** 1:49:07

5 seconds Max.

 **Tarun Jain** 1:49:13

OK, I'll have to restart session if I change the keys.

 **Tirth** 1:49:15

M.

M.

 **Tarun Jain** 1:49:20

Over where?

Yeah, I'll have to restart this session.

Yeah but for this thing, if you see here I used flash and then I ran the agent tree. I got the output. It is doing function call it used web.

 **Tirth** 1:49:48

Oh.

But what do we do in some scenarios when it does not do function calls? Like in my scenario I was not able to do function call for some reason I don't know. And when I changed it to pro it was doing the function call.

 **Tarun Jain** 1:50:04

So same we use 3 approaches. One is improve the prompt.

 **Tirth** 1:50:11

Mhm.

 **Tarun Jain** 1:50:14

And add additional context.

Inside function dot string.

 **Tirth** 1:50:25

OK.

 **Tarun Jain** 1:50:27

Here we'll add off too much of information. Here I just gave one sentence, right? I'll add three to four more sentence.

 **Tirth** 1:50:30

Thanks.

Thank you.

Yes.

 **Tarun Jain** 1:50:35

And third.

Where is that?

OK.



**Tirth** 1:50:46

To improve the context.



**Tarun Jain** 1:50:47

Not experiment with.



**Tirth** 1:50:52

OK.

Oh.

Oh.



**Tarun Jain** 1:51:01

Uh, the model that we use is GPT 4O.

And sometimes we use GBT 4 one. We still didn't migrate it into GBT 5.



**Tirth** 1:51:13

M.



**Tarun Jain** 1:51:14

So GPT photo works better.

But I don't think we experiment with models. Models for us will remain the same. We only experiment with these three things.

And this sometimes doesn't make much sense.



**Tirth** 1:51:33

OK.



**Tarun Jain** 1:51:36

There is no much.

What you call return from this? So usually it's prompt and as well as the additional context of function tool calling. If you improvise most of the cases is solved there only.

 **Tirth** 1:51:53

Um.

 **Tarun Jain** 1:52:00

Is it clear?

 **Tirth** 1:52:02

It's clear.

 **Tarun Jain** 1:52:04

The only thing is, let's just make sure we had certain subtitles submittings because we have kept on coding and we use the same variables multiple times.

 **Tirth** 1:52:16

Mhm.

 **Tarun Jain** 1:52:20

And instead of parallel, what you guys can do is as an assignment use tably. Because Table is one of the most advanced search tool that I've seen. So what you can do is use Table as the assignment and here you can you will have to create API key and you don't have to write custom code, use the existing. Existing code from Langston itself.

 **Tirth** 1:52:47

OK.

 **Tarun Jain** 1:52:48

Because they have made proper arguments that you're supposed to use and you don't have to do much alteration. And tably guys are directly involved with Lanchen, so they actually know what parameters to use. This is only specific to tably if you look at other tools.

 **Tirth** 1:53:02

OK.

 **Tarun Jain** 1:53:06

It's just community contribution. They just kept on adding it.

 **Tirth** 1:53:09

OK.

Mm.

 **Tarun Jain** 1:53:16

If you look at GitHub Toolkit.

If you see any this thing right import statement where you see community, that means that it's been added by the community. But if you look at Tabli, Tabli was also community but now it's changed. Tabli is a separate wrapper now.

 **Tirth** 1:53:24

Oh.

Mhm.

 **Tarun Jain** 1:53:36

Uh, there was one more interesting tool. Where is that?

EXA also is now separate. If you see langs in EXA, these two are very good search rules, but I'm not sure why they've added very low in their.

 **Tirth** 1:53:55

Oh.

Um.

 **Tarun Jain** 1:54:02

Mhm.

 **Tirth** 1:54:03

They must be faking it.

 **Tarun Jain** 1:54:07

I'm not sure like what benchmarks have used.

But EXA is good. If you see from perplex it is much better in search quality search engine EXA has done good job.

 **Tirth** 1:54:23

Oh.

 **Tarun Jain** 1:54:24

But this is way too much.

O3 MCP.

EXA.

And why they haven't done any benchmark with Tabley?

Because travel is there for very long times.

 **Tirth** 1:54:50

M.

 **Tarun Jain** 1:54:51

So this will be the assignment. Let's just make sure you use Tabli in your Uh.

This thing particular tool and then connect it with React agent.

 **Tirth** 1:54:58

OK.

OK.

 **Hardip Patel** 1:55:04

OK, I have extracted what is working for me for the web search.

 **Tarun Jain** 1:55:06

Yeah.

Can you place that function in the chat?

 **Hardip Patel** 1:55:15

I I have separated the collab and added it to chat.

 **Tarun Jain** 1:55:22

A what?

Hello.

 **Hardip Patel** 1:55:26

I have added it to chat.

I think or like.

 **Tarun Jain** 1:55:33

Oh, I can't see it.

 **Tirth** 1:55:33

You have added the collab to chat. I can't see it yet.

 **Hardip Patel** 1:55:43

It is still stand sending some of.

 **Tirth** 1:55:47

OK.

OK.

 **Hardip Patel** 1:55:48

OK.

 **Tarun Jain** 1:55:52

OK, it's a guest link.

 **Tirth** 1:55:57

OK.

 **Tarun Jain** 1:56:06

Now this is the same thing what we did.

 **Hardip Patel** 1:56:07

Please, please, please ignore the flash. Uh, flash is not working.

I mean, uh, for those who are facing issue.

 **Tarun Jain** 1:56:24

OK, but the code is same I guess when we have we actually improvise.

 **Hardip Patel** 1:56:27

Yeah, I just extracted the code that was working for me for.

OK.

 **Tarun Jain** 1:56:35

Cool. We can also update this one. We are we are also adding description and title and I added this region equals to India.

 **Tirth** 1:56:45

Oh.

 **Tarun Jain** 1:56:46

So I'll just comment this.

What the?

Yeah, good.

If you want to try tably with custom function, you can also try that.

 **Tirth** 1:57:21

OK, I think we'll try with multiple things. We'll see how goes.

 **Tarun Jain** 1:57:25

And one more thing, what you can do is you can add 2 tools here, one will be Tabley and one more will be Yahoo Finance.

So if you use stock price, just make sure you add debug to be true. Ask any stock price related question and check this line. When you use function call it should use name as Yahoo Finance. If you use anything related to search, it should be tably search.

Tably existing code from Lantern and Yahoo Finance.

 **Tirth** 1:57:57

But.

 **Tarun Jain** 1:57:59

So use two tools inside the list.

 **Tirth** 1:58:15

Thank you. Thank you a little.

 **Tarun Jain** 1:58:17

Yeah.

 **Tirth** 1:58:19

I don't. I just need a few minutes of your time if you have the gap, not more than 5 minutes, but I think we can end the session. We'll be left.

 **Hardip Patel** 1:58:29

Thank you.

 **Mitesh Rathod** 1:58:32

OK.

 **Tirth** 1:58:33

Thank you, everyone.

 **Ishan Chavda** stopped transcription