Transcript: I gave Claude root access to my server... Model Context Protocol explained

Video ID: HyzlYwjoXOQ

Extraction Date: 2025-04-03 02:14:57

**[00:00:00]** it seems like every developer in the

**[00:00:01]** world is getting down with mCP right now

**[00:00:04]** model context protocol is the hot new

**[00:00:06]** way to build apis and if you don't know

**[00:00:08]** what that is you're ngmi people are

**[00:00:10]** doing crazy things with it like this guy

**[00:00:12]** got clawed to design 3d art and blender

**[00:00:14]** powered entirely on Vibes and just a few

**[00:00:16]** days ago it became an official standard

**[00:00:18]** in the open AI agents SDK if you're an

**[00:00:21]** OG subscriber to this channel you

**[00:00:23]** probably know what a rest API is you

**[00:00:25]** might even know about graphql or RPC or

**[00:00:27]** maybe many years ago you use soap when I

**[00:00:29]** was a kid the software engineering

**[00:00:31]** Gatekeepers told me I couldn't be a web

**[00:00:32]** developer unless I could explain the

**[00:00:34]** difference between these architectures

**[00:00:35]** and protocols well now the turns have

**[00:00:37]** tabled and these Gatekeepers have been

**[00:00:38]** utterly demolished because we're all

**[00:00:40]** just Vibe coders now embracing the

**[00:00:42]** exponentials pretending code doesn't

**[00:00:44]** even exist and just chilling with llms

**[00:00:46]** until we get the end result we're

**[00:00:47]** looking for that being said you can't

**[00:00:49]** call yourself a true vibe coder unless

**[00:00:51]** you know about model context protocol

**[00:00:53]** which is basically a new standard for

**[00:00:55]** building apis that you can think of like

**[00:00:57]** a USBC port for AI application it was

**[00:01:00]** designed by anthropic the team behind

**[00:01:02]** Claud and provides a standard way to

**[00:01:04]** give large language models context and

**[00:01:06]** they're so bullish on this technology

**[00:01:07]** that the CEO of anthropic said he

**[00:01:09]** expects virtually all code to be written

**[00:01:11]** by AI by the end of the year in today's

**[00:01:13]** video we'll actually build an mCP server

**[00:01:16]** and find out if it can truly make the

**[00:01:17]** world a better place by eliminating all

**[00:01:19]** white collar jobs it is March 31st 2025

**[00:01:22]** and you're watching the code report

**[00:01:24]** contrary to popular belief a Fire Ship

**[00:01:26]** is still a tutorial channel in today's

**[00:01:27]** video we'll take a storage bucket a

**[00:01:29]** postgress database and a regular rest

**[00:01:31]** API and then connect them all together

**[00:01:33]** with the model context protocol not only

**[00:01:36]** will this allow Claude to have access to

**[00:01:37]** data it didn't have before but it can

**[00:01:39]** also execute code on our server like

**[00:01:41]** write to the database or upload files

**[00:01:43]** and people of the internet are already

**[00:01:44]** using it to do crazy stuff like

**[00:01:46]** automated ston and shitcoin trading

**[00:01:48]** industrial scale web scraping and as a

**[00:01:50]** tool to manage Cloud infrastructure like

**[00:01:52]** your kubernetes cluster speaking of

**[00:01:54]** which to build our own mCP server we'll

**[00:01:56]** need some Cloud infrastructure and one

**[00:01:57]** of the best places to do that is savola

**[00:02:00]** which itself is powered by Google

**[00:02:01]** kubernetes engine and cloudflare under

**[00:02:03]** the hood they were nice enough to

**[00:02:04]** sponsor this video but the reason I like

**[00:02:06]** their platform so much is that it's far

**[00:02:08]** easier to use than something like AWS

**[00:02:10]** but provides linear predictable pricing

**[00:02:12]** unlike most of the application and

**[00:02:14]** database hosting startups out there and

**[00:02:16]** it's free to get started which makes it

**[00:02:17]** perfect for this project now like other

**[00:02:19]** API architectures mCP has a client in a

**[00:02:22]** server the client in our case will be

**[00:02:24]** claw desktop then will'll develop a

**[00:02:26]** server that maintains a connection with

**[00:02:27]** that client so the client and server can

**[00:02:29]** pass information back and forth via the

**[00:02:31]** transport layer now in arrest API you

**[00:02:34]** have a bunch of different HTTP verbs

**[00:02:36]** that you can send requests to Via

**[00:02:37]** different URLs but in the model context

**[00:02:40]** protocol we're really only concerned

**[00:02:41]** with two main things resources and tools

**[00:02:44]** a resource might be a file a database

**[00:02:47]** query or some other information the

**[00:02:49]** model can use for context conceptually

**[00:02:51]** you can think of it like a git request

**[00:02:52]** and rest meanwhile a tool is an action

**[00:02:55]** that can be performed like writing

**[00:02:57]** something to a database so that'd be

**[00:02:58]** more like a poster request and rest what

**[00:03:01]** we do as developers is Define tools and

**[00:03:03]** resources on the server is so the llm

**[00:03:05]** can automatically identify and use them

**[00:03:08]** when they have a prompt that needs them

**[00:03:09]** in my life I've been working on an app I

**[00:03:11]** consider my magnum opus called horse

**[00:03:13]** tender but as it turns out swiping left

**[00:03:15]** and right was a bad feature idea because

**[00:03:17]** horses don't have fingers so like every

**[00:03:19]** other failing startup in Silicon Valley

**[00:03:21]** right now we're going to Pivot to

**[00:03:23]** artificial intelligence luckily we can

**[00:03:25]** leverage our existing data and servers

**[00:03:27]** like here in savola I have a storage

**[00:03:28]** bucket and it contains all of the photos

**[00:03:30]** that my users uploaded in addition for

**[00:03:32]** user data we have a postgress database

**[00:03:34]** it has all the profile data for each one

**[00:03:36]** of our horses as well as the

**[00:03:38]** relationships they form together and

**[00:03:39]** then finally I have a traditional rest

**[00:03:41]** API written in typescript that fetches

**[00:03:43]** this data for my web IOS and Android

**[00:03:45]** apps and what's especially cool about my

**[00:03:47]** code is that it's in a git repo hooked

**[00:03:49]** up to a cicd pipeline that means after

**[00:03:52]** we write our model context protocol

**[00:03:53]** server we can push our code to the dev

**[00:03:55]** or staging branches to test it before it

**[00:03:57]** actually goes into production while

**[00:04:00]** automatically handles all the

**[00:04:01]** deployments and cash busting for us

**[00:04:02]** automatically and now we're ready to

**[00:04:04]** jump into some code here I have a dino

**[00:04:06]** project and the first thing you'll

**[00:04:07]** notice is that I'm importing a class

**[00:04:09]** called mCP server this comes from the

**[00:04:11]** official SDK but if you're not using

**[00:04:13]** typescript they have a bunch of other

**[00:04:15]** languages like python Java and so on

**[00:04:17]** we'll also be using Zod here which is a

**[00:04:19]** tool used for schema validation which

**[00:04:21]** allows us to provide a specific data

**[00:04:23]** shape to the llm is so it doesn't just

**[00:04:25]** hallucinate a bunch of random crap now

**[00:04:27]** after we create a server we can start

**[00:04:28]** adding resources to it the resource will

**[00:04:31]** first have a name like horses looking

**[00:04:33]** for love and then the second argument is

**[00:04:34]** a URI for the resource then finally the

**[00:04:37]** third argument is a callback function

**[00:04:39]** that we can use to fetch the data in

**[00:04:41]** this example I'm writing a query to our

**[00:04:42]** postgress database which is hosted in

**[00:04:44]** the cloud on savola then accessed on the

**[00:04:46]** server with the postgress JS library but

**[00:04:49]** you could access any data here when

**[00:04:50]** something is a resource though it should

**[00:04:52]** only be used for fetching data where

**[00:04:54]** there's no side effects or computations

**[00:04:56]** if you do have a side effect or

**[00:04:57]** computation you should instead use a

**[00:04:59]** tool like for horse tender we might want

**[00:05:01]** the AI to automatically create matches

**[00:05:03]** and set up dates between horses we

**[00:05:05]** already have a restful API endpoint that

**[00:05:07]** can handle that and we could actually

**[00:05:09]** leverage that code here essentially

**[00:05:10]** creating an API for our API in fact many

**[00:05:13]** of these mCP servers are actually just

**[00:05:16]** apis for apis and that sounds like dumb

**[00:05:18]** over engineering but having a protocol

**[00:05:21]** like this makes it a lot easier to Plug

**[00:05:22]** and Play between different models and

**[00:05:24]** just makes llm apps more reliable in

**[00:05:26]** general the case in point notice how I'm

**[00:05:28]** using Zod here to validate the shape of

**[00:05:30]** the data going into this function that

**[00:05:32]** prevents the llm from hallucinating

**[00:05:34]** random stuff here basically when you

**[00:05:36]** prompt Claud it's going to need to

**[00:05:37]** figure out the proper arguments to this

**[00:05:39]** function is so providing data types

**[00:05:41]** along with a description will make your

**[00:05:42]** mCP server far more reliable and then

**[00:05:44]** the final step is to run the server in

**[00:05:47]** this case I'm going to use standard IO

**[00:05:48]** as the transport layer to use it locally

**[00:05:50]** but if deploy to the cloud you can also

**[00:05:52]** use server sent events or HTTP

**[00:05:54]** congratulations you just built an mCP

**[00:05:57]** server but now the question is how do we

**[00:05:59]** Act actually use it to use it you'll now

**[00:06:01]** need a client that supports the model

**[00:06:02]** context protocol like Claude desktop

**[00:06:05]** there are many other mCP clients out

**[00:06:06]** there if you don't want to use CLA

**[00:06:08]** desktop like cursor and wisor for

**[00:06:10]** example and you could even develop your

**[00:06:11]** own client but that's an entirely

**[00:06:13]** separate topic altogether once installed

**[00:06:15]** you can go to the developer settings

**[00:06:17]** which will bring you to a config file

**[00:06:19]** where you can add multiple mCP servers

**[00:06:21]** in the config file all you have to do is

**[00:06:23]** provide a command to run the actual

**[00:06:25]** server which in our case would be the

**[00:06:26]** doo command for the main.ts file where

**[00:06:29]** we find our server code you'll need to

**[00:06:31]** restart clad but then it should show

**[00:06:32]** your mCP server is running in this case

**[00:06:34]** my horse is running which means I should

**[00:06:36]** probably go and catch

**[00:06:40]** it then you can go back to the prompt

**[00:06:42]** screen to attach it that's going to

**[00:06:44]** fetch the resource from the server so

**[00:06:45]** CLA can use it as context in the next

**[00:06:47]** prompt and because clae is multimodal

**[00:06:49]** you could also add PDFs images or

**[00:06:52]** anything else to the context really like

**[00:06:54]** all the horse images in our Savala

**[00:06:55]** storage bucket and now magically you can

**[00:06:57]** prompt Claude about things specific to

**[00:06:59]** your application like if we want to find

**[00:07:01]** out which horses are single and ready to

**[00:07:02]** mingle we can make a prompt like this

**[00:07:05]** and it will use the context that we just

**[00:07:06]** fetched from our database then if we

**[00:07:08]** want Claude to write to the database we

**[00:07:09]** could make a prompt like this where

**[00:07:11]** it'll connect two horses from the

**[00:07:12]** context on a date you'll need to Grant a

**[00:07:14]** permission to do this and then Claude

**[00:07:15]** will automatically figure out which data

**[00:07:17]** to send it but based on the schema we

**[00:07:19]** validated with Zod and it'll use our

**[00:07:21]** server tool to mutate data in the actual

**[00:07:23]** application I can't imagine anything

**[00:07:25]** ever possibly going wrong here and

**[00:07:27]** anthropic is extremely bullish on this

**[00:07:28]** being the future like their CEO just

**[00:07:30]** said that 90% of coding will be done

**[00:07:32]** entirely by AI within the next 6 months

**[00:07:35]** and nearly all code will be AI generated

**[00:07:37]** within a year I'm going to go ahead and

**[00:07:38]** press X to doubt there and I think it's

**[00:07:40]** only a matter of time before some AI

**[00:07:41]** agent accidentally wipes out billions of

**[00:07:43]** dollars in customer data or becomes

**[00:07:45]** self-aware and just deletes the data for

**[00:07:47]** fun that being said though there's all

**[00:07:49]** kinds of amazing tools being built with

**[00:07:51]** mCP right now and you can check those

**[00:07:52]** out on the awesome mCP repo just please

**[00:07:55]** make sure to Vibe code responsibly huge

**[00:07:57]** thanks to Savala for making this video

**[00:07:59]** possible and enjoy this $50 stimulus

**[00:08:01]** check to try out their awesome platform

**[00:08:03]** this has been the code report thanks for

**[00:08:05]** watching and I will see you in the next

**[00:08:07]** one

# Full Text (without timestamps)

it seems like every developer in the world is getting down with mCP right now model context protocol is the hot new way to build apis and if you don't know what that is you're ngmi people are doing crazy things with it like this guy got clawed to design 3d art and blender powered entirely on Vibes and just a few days ago it became an official standard in the open AI agents SDK if you're an OG subscriber to this channel you probably know what a rest API is you might even know about graphql or RPC or maybe many years ago you use soap when I was a kid the software engineering Gatekeepers told me I couldn't be a web developer unless I could explain the difference between these architectures and protocols well now the turns have tabled and these Gatekeepers have been utterly demolished because we're all just Vibe coders now embracing the exponentials pretending code doesn't even exist and just chilling with llms until we get the end result we're looking for that being said you can't call yourself a true vibe coder unless you know about model context protocol which is basically a new standard for building apis that you can think of like a USBC port for AI application it was designed by anthropic the team behind Claud and provides a standard way to give large language models context and they're so bullish on this technology that the CEO of anthropic said he expects virtually all code to be written by AI by the end of the year in today's video we'll actually build an mCP server and find out if it can truly make the world a better place by eliminating all white collar jobs it is March 31st 2025 and you're watching the code report contrary to popular belief a Fire Ship is still a tutorial channel in today's video we'll take a storage bucket a postgress database and a regular rest API and then connect them all together with the model context protocol not only will this allow Claude to have access to data it didn't have before but it can also execute code on our server like write to the database or upload files and people of the internet are already using it to do crazy stuff like automated ston and shitcoin trading industrial scale web scraping and as a tool to manage Cloud infrastructure like your kubernetes cluster speaking of which to build our own mCP server we'll need some Cloud infrastructure and one of the best places to do that is savola which itself is powered by Google kubernetes engine and cloudflare under the hood they were nice enough to sponsor this video but the reason I like their platform so much is that it's far easier to use than something like AWS but provides linear predictable pricing unlike most of the application and database hosting startups out there and it's free to get started which makes it perfect for this project now like other API architectures mCP has a client in a server the client in our case will be claw desktop then will'll develop a server that maintains a connection with that client so the client and server can pass information back and forth via the transport layer now in arrest API you have a bunch of different HTTP verbs that you can send requests to Via different URLs but in the model context protocol we're really only concerned with two main things resources and tools a resource might be a file a database query or some other information the model can use for context conceptually you can think of it like a git request and rest meanwhile a tool is an action that can be performed like writing something to a database so that'd be more like a poster request and rest what we do as developers is Define tools and resources on the server is so the llm can automatically identify and use them when they have a prompt that needs them in my life I've been working on an app I consider my magnum opus called horse tender but as it turns out swiping left and right was a bad feature idea because horses don't have fingers so like every other failing startup in Silicon Valley right now we're going to Pivot to artificial intelligence luckily we can leverage our existing data and servers like here in savola I have a storage bucket and it contains all of the photos that my users uploaded in addition for user data we have a postgress database it has all the profile data for each one of our horses as well as the relationships they form together and then finally I have a traditional rest API written in typescript that fetches this data for my web IOS and Android apps and what's especially cool about my code is that it's in a git repo hooked up to a cicd pipeline that means after we write our model context protocol server we can push our code to the dev or staging branches to test it before it actually goes into production while automatically handles all the deployments and cash busting for us automatically and now we're ready to jump into some code here I have a dino project and the first thing you'll notice is that I'm importing a class called mCP server this comes from the official SDK but if you're not using typescript they have a bunch of other languages like python Java and so on we'll also be using Zod here which is a tool used for schema validation which allows us to provide a specific data shape to the llm is so it doesn't just hallucinate a bunch of random crap now after we create a server we can start adding resources to it the resource will first have a name like horses looking for love and then the second argument is a URI for the resource then finally the third argument is a callback function that we can use to fetch the data in this example I'm writing a query to our postgress database which is hosted in the cloud on savola then accessed on the server with the postgress JS library but you could access any data here when something is a resource though it should only be used for fetching data where there's no side effects or computations if you do have a side effect or computation you should instead use a tool like for horse tender we might want the AI to automatically create matches and set up dates between horses we already have a restful API endpoint that can handle that and we could actually leverage that code here essentially creating an API for our API in fact many of these mCP servers are actually just apis for apis and that sounds like dumb over engineering but having a protocol like this makes it a lot easier to Plug and Play between different models and just makes llm apps more reliable in general the case in point notice how I'm using Zod here to validate the shape of the data going into this function that prevents the llm from hallucinating random stuff here basically when you prompt Claud it's going to need to figure out the proper arguments to this function is so providing data types along with a description will make your mCP server far more reliable and then the final step is to run the server in this case I'm going to use standard IO as the transport layer to use it locally but if deploy to the cloud you can also use server sent events or HTTP congratulations you just built an mCP server but now the question is how do we Act actually use it to use it you'll now need a client that supports the model context protocol like Claude desktop there are many other mCP clients out there if you don't want to use CLA desktop like cursor and wisor for example and you could even develop your own client but that's an entirely separate topic altogether once installed you can go to the developer settings which will bring you to a config file where you can add multiple mCP servers in the config file all you have to do is provide a command to run the actual server which in our case would be the doo command for the main.ts file where we find our server code you'll need to restart clad but then it should show your mCP server is running in this case my horse is running which means I should probably go and catch it then you can go back to the prompt screen to attach it that's going to fetch the resource from the server so CLA can use it as context in the next prompt and because clae is multimodal you could also add PDFs images or anything else to the context really like all the horse images in our Savala storage bucket and now magically you can prompt Claude about things specific to your application like if we want to find out which horses are single and ready to mingle we can make a prompt like this and it will use the context that we just fetched from our database then if we want Claude to write to the database we could make a prompt like this where it'll connect two horses from the context on a date you'll need to Grant a permission to do this and then Claude will automatically figure out which data to send it but based on the schema we validated with Zod and it'll use our server tool to mutate data in the actual application I can't imagine anything ever possibly going wrong here and anthropic is extremely bullish on this being the future like their CEO just said that 90% of coding will be done entirely by AI within the next 6 months and nearly all code will be AI generated within a year I'm going to go ahead and press X to doubt there and I think it's only a matter of time before some AI agent accidentally wipes out billions of dollars in customer data or becomes self-aware and just deletes the data for fun that being said though there's all kinds of amazing tools being built with mCP right now and you can check those out on the awesome mCP repo just please make sure to Vibe code responsibly huge thanks to Savala for making this video possible and enjoy this $50 stimulus check to try out their awesome platform this has been the code report thanks for watching and I will see you in the next one