OK. The really important thing that we will get into today is the whole HTTP request to response. And I wanna emphasize it first because I want you to understand that. Aside from needing to know how it works, it is a very, very common thing for technical interviews to ask. So think of pay attention to what the winks when someone comes and says. What is a a request and you know HTTP request and response? Cycle or or the protocol or something like that. They are talking about this idea that in the web we go out and we send a message we we post a message and we get a response. And that is the key part to how we interact over II with with the web, but not even just with the web, just in general. So all of the stuff we're going to talk about now is related to that, that request and response cycle. I will stop occasionally as we go through this and I'll say this is the sort of thing you need to actually respond to or mention when when you're when you're talking in a technical interview. But this is a very common thing 'cause I want to make sure you understand the concept here more than the terms you. So let's go back a little bit. You probably rattled some about it and you probably understand a certain amount about it. I just wanna make sure everyone is is clear on some of these things. When we see a URL and let's go to something like the URL we've got right here, when you see a URL. It has several parts to it and you can read them about it. I think there's a chart here that now just cut it back on the earlier one, but you can read about what they parts are. But the important thing that you need to know is. What? Pull it up, 'cause. It's easier to see on the big. There it goes. The important thing is you need to know the protocol. Now protocol is the ones you all will have seen. A lot will be HTTP and HTTPS and they are basically the same thing, except that the HTTPS is encrypted between the browser and the client which. Prevents people from stealing your credentials and things like that. Right. But if you looked in the? When we set up the the string which said where our databases were. OK, back, you know, last week when we talked about when we get made a connection. To the specific database we're going to deal with the United States database or whatever else. If you look in that code, it will have a protocol and the colon in the slash slash. In that case it was JDBC. So there are lots of possible protocols and they are used in different ways. Another one that you might see sometimes is nail too. Mail 2: and then you'll have an address and that basically says when someone clicks on that link and tries to go through that place, the protocol is that it's an email. And so we'll put you in whatever your default email. Program is going to that address, so these protocols are. You'll see more and more as you get out of the just. I'm browsing the web into actually interacting with different services and different things. You actually set up your own protocols to different things. Postgres or Java or whatever set up the JDBC protocol as a way to connect to. Some resource. And that goes into what the entire thing is the entire thing is called a universal resource locator. When we were furta, all the things we contact, whether it's a web page or a database or whatever else we refer to those as resources. So all of this the purpose of this whole URL is to get at some resource that is something other than where we are. It may be on my computer, it may be on a server in India it may be whatever, but it's it's somewhere. But computers, just the way in Java. Computer doesn't really understand your numbers the way they are we we it's actually a series of bits and all that sort of thing. Well, there's no way for the computer to really know where example.com is or in the URL I have up here, the lms.techelevator.com. Something has to find what that is, and at its core these networks only understand IP addresses. So I want to make sure that you all get the idea of. IP addresses and what they what they are because sometimes you will use them explicitly in a way they are unlikely to have when you just browsing the web. Right. So does anyone have any? Does everyone get what an IP address is? Like how it works then being not understand that. Or wanna little more clarification. It's basically a series of numbers. Is most of them are IP addresses are four. Up to three digit numbers which are used to get closer and closer to the resource you're looking for. So they're gonna work from left to right. And it's gonna the first one is going to narrow down to one, you know, set of things and then supposedly at least go down. And each one narrows it in further and to get to this specific one, you need, right? Part of the reason we talk about it is because I there are special IP addresses and those IP addresses are that you will deal with are well, the most classic one that you will deal with is 127.0 point 0.1. And that is. Whatever I'm on, it's the device. It's the thing that I am actually on. It is also called and you'll see it there, sometimes called localhost. But the reason it's called localhost is not because computers magically understand localhost anymore than they understand example.com. There's a resolution process that we have to go through to go from a URL to what resource it actually connects to. And when you are going out to the web. And there's going to be a domain naming service that is going to go out at DNS, and that's going to be what resolves this for you. But there is also on your computer something a hosts file and the host file is what tells it how to resolve it. How to resolve URLs that don't ever leave your computer that are on your computer. So your host file is what's going to have you know the JDBC address. You could have a URL and go to it. It's going to be resolved in your local host file and localhost when we call that. It's really just in your host file, it's it resolves to 127.0 point 0.1 when at some point. When I was early and when I was using the, you know the the web very early on or actually before it was the web and I was using the Internet before the web came around. People used IP addresses much more often directly, and so programmers were all comfortable with just, you know, using particular ones and. But now nobody does offer almost anything, so almost everything has some name. Even if it is widely known as a specific and universal number now scuse me man. Yeah. I understand a little bit of it, but I'm just lost on what system creates the actual IP address. What creates it? Yeah. There they are assigned. OK, so well, there are a few of them. Like, like the local ones that are that are just kind of built into the system, but the. The addresses are assigned by a global body. That is, you know, an international body that actually they assign the top level. All right, so. Whatever your country has, it probably has certain ID's. We've ever seen those things where you're able to restrict and say I don't want any email you may not deal with it, but in some companies they'll do. They say like we don't want anything from Russia. They can do that because Russia has certain of those with the four different numbers. It has some of those numbers that it has the first one and you can identify where in the world those numbers are assigned. So you can tell all the sudden Russian IP addresses. But there's this body that basically says every time somebody needs an IP address, they have to go and and request it. But it keeps narrowing down. They don't want to deal with every single person who's IP address. So what they do is they do blocks of IP addresses, and then there's a local domain register that gives a smaller one, and so forth, so that when I first went out and got myself a website. My Internet provider for the person who is providing me the website assigned me a specific IP address so that people would be able to get to it. OK that's what I was wondering is that I did a website but I'm like I don't know nothing about how it had it, IP address or any. I just created the domain name. They keep hiding this stuff further and further away from so part of the part of the reason why you don't see it is because the usually your Internet provider, whoever provides the website you know if you want to. I did it through simplified. They managed their own IP addresses, So what they're gonna do is they're gonna assign them. You'll never know. You'll just have. You'll just know what your yours is called, OK? It is an important point though, that. The. The IP address the sub domain here like the WWW. What happens more and more and this this is a as they. When you get a service like Spotify or, you know word press or something like that. They use the the sub domain to really identify within their their domain. They use that to identify where the IP addresses is. So even though it looks like. Everything under example.com might have the same group of IP addresses. That isn't true at all. What happens is whoever runs example.com manages. The IP addresses for sub domain. So for instance when I look up at lms.techelevator.com. If I wanted to find out what the IP address is, it's really whoever is managing tech elevator that says where is the LMS part of this and it might be a completely different set of IP addresses than you know. I think the bootcamp. I'm sorry, I think Git lab might also be. Let's see if. Our Git lab is alright, so Git lab is also done with tech elevator.com. That means that whoever is managing the domain. We'll be able to look up where the sub domain is, but they may be in different parts of the world. Whoever GitLab could be hosted, you know in in Cleveland and LMS it I think is hosted in Colorado. Excuse me, Ben. Yeah. So in the URL, the part where where the LMS that's equivalent to www.thatsright. Oh, OK. alright and www.isthe but I could if I actually wanted to, I could actually assign if you just go to example.com I could have it go to a different place than www.example.com. It's just that it is common custom for those to always go to the same place. That is the that I when I set up and again I probably have a dozen websites. When I set up my websites. That one of the first things I do is I make sure that both www.youknowwhateveritisgeniusoff.com and Jeannie actsoft.com go to the same place, so that if you're trying to go to again something like ibm.com. Or microsoft.com. Your what you're going to do is going to be the same place as whatever www.microsoft.com, but there are. Millions of subdomains under any of those major ones, and that's like again something like WordPress which hosts blogs. The sub domain is what's going to control which blog it is or which website it is OK, but that's yes. So the LMS in this case is the sub domain. Tech elevator is the domain name and com is the TLD. The top level domain. In theory, the top level domain is supposed to manage what country you're in. Right. To the victors go the spoils or whatever. Since we created the system, we we picked.com and then tried to get everyone else to use.com or.com dot, you know, see A for instance, which is Canada. So all the other countries were supposed to use.com we and we were supposed to be.com dot US, but we never actually decided to use it. So you can have one more level here which is like. Forsomes.co.ukyouknowithinkits.com dot UK and it's. Co.de for Germany and things like that. Different ones use different things, but in the real world, and I'm telling you this, so you sort of understand how the dynamics of things were. In the real world, everyone who is anyone wanted a.com address. If they had a company. And so for the longest time, even if you were in a different country, you would try to grab one of the.com addresses and we'd let you do it. So you know. I I had a my company was was international was hosted in one. In one country, which wasn't the United States had a domain registry in another country, which was not the United States and I still got.com because at the time I could so it it it. The web comes up with lots of great intentions, but the intentions are not always followed. Doesn't really matter. That's sort of the just. So you know why you'll see the variety that you see sometimes in all this. The one critical change that has happened in the past, you know, ten years is that. Everyone has finally moved from HTTP to HTTPS. And I do want to make sure that you understand the the critical part of that right. The idea is that more and more people were having problems because there was important information that were being carried through in their URL. And was then getting stolen by somebody else who was able to just. If you don't use HTTPS. And all of your whole request is sent out to be resolved by somebody in cleartext. If you're sitting in a coffee shop. It's actually incredibly easy for someone else to to to tap into the the interactions that you're having with servers from your coffee shop. But there are different the web works in this very and this is where we get back to the whole request response thing. The web has certain rules about how this works. BURL. Was always sent in clear text. And it is logged lots and lots of systems will log every single you know request that they get including the IP address that it came from. So you have an IP address when you're when you are, which they can recognize, which is how they get back and identify you. Now your IP address is probably not. It is probably actually your your Internet provider. It's not you, but they can still get back relatively close to you if you're in a company, they might be able to get back even more close because like when I track IP addresses for people who come to my websites, I keep track like I I associate them with the names. People, if I know what the names are, so that if I get a hit from that IP address later, I know who the person is and I do it because I like. I've done that when I when I've had a company and I had to support customers, sometimes I would get requests and I needed to know more about the customers. So I wanted to go back and find out their information and if they wouldn't always request information, they just come and search on my website. Or something, but I would then know, oh, this customer is searching because they have this problem and I may be able to reach out to things like that. So it's very common to log the URLs. And log The Who is making the request. The problem with this is that it was also very common for people to to put important critical confidential information as part of the URL. So one of the things and and I want to emphasize why, that's why this is such a critical thing. The web is by and large, stateless. The entire HTTP request in response assumes that you that the server is not aware of the last thing you did. All they know is the one single thing you are doing right this minute. Think about that. You go to the web, you go to your bank. Can you log in? And you're trying to access your stuff, so you first of all, you may deposit a check and then you wanna go check your balance. Every single time you interact. Is it is a whole brand new thing? But you don't have to log in for every single time you you make a request, so there has to be an entire system built up so that you are carrying all your authentication information with you for every single request. Every time I refresh a page, every time I go to a different page on the website. Within reason it you are, you are going back and forth. You're making a request and you're carrying all your authentication information with you. Well, if someone steals your authentication information. Then in theory, they can go in and beat you and take a lot of money out of your account, right? Because of that, because of that stateless nature, because of the fact that each message has to contain contain all the information. At some point it became too hard to manage unless we encrypted that information so that other people couldn't so easily steal. And that the big change from and you'll see some websites use HTTP and some use HTTP, HTTPS, HTTPS is a its encryption algorithm that is used. It'll it uses an encryption algorithm so that your browser. Interacting with the server, well, it's more the client interacting with the browser that isn't. That message is encrypted so that it's the the person sitting next to you in Starbucks. Who is able to tap into your communication won't be able to read, your communication won't be able to read the URL on it. It doesn't stop the server from logging that information. Because the server needs that it like they they're the ones who actually need it so. If you, you may not know this, but for a lot of things where you login, you can actually pass the username and password as part of the URL. In the early age of the web, that was a super cool thing. Oh wow, we can just log in right now. We don't have to do it separately. It's a terrible practice because because they get logged, it means that somebody has on their website at full log to all those different addresses and username passwords. And if somebody hacks into their system, which happens fairly frequently, then they'll have you username, password. So don't. The big rule is. You should never ever put your username, password or that or other credentials of that sort. Anything which, which is too confidential or. Identifying as part of your URL. OK. When you look at here, this is a. When we look again at our LMS, right. So this is the we've got this sub domain, we've got the, the domain, we've got the TLD, the top level domain and then we have the path. And the path is everything after the.com, or.infoor.net or whatever is after that top level domain is everything after that before the question mark. If there is a question mark. The path is everything before the question mark. Everything after the question mark is called the query string. Again, originally developed just so that you could actually do, you know, Google queries or you know, you could actually just search it, but it now is really when you think of a Java method. See if you can. You can call him Etihad. That has no parameters. That is like calling this this website up until the list with without parameters. In a Java method, you may have a specific number of parameters. The the rules are much looser on the web, but the idea behind all of these key value pairs. Remember, we talked about key value pairs before. Each of these key value pairs is a parameter that is being sent to help the the website that you're calling or whatever it is page resource that you're calling to help it do its job. So when we talk about the query string, it is made up of querystring parameters and those are the the value pairs. The key value pairs. So. That's all theory. But let's go out and sort of look at some examples where. And to do that, we are going to introduce our new tool for the day which is postman. Postman is this little icon that I don't know has a little postman flying or a lightning bolt or something. I can't. My eyes are not good enough to read it. You people who actually use emojis probably are used to these tiny little things. But you know, I'm too old for that in any case. If you open that up. You were going to see I was playing with this last night, so I will show you that in just a moment. But you're going to open up in the scratchpad. And scratch pad is sort of my local version of Postman and just so you know, Postman was originally developed to be a local person, you know like just my ability to go out and make these HTTP requests and interact with them and debug them and all that kind of stuff. It very quickly became a more social not in the sense that we use social media so much as a social in that it is used by an organization or group or whatever else. The way slack is OK, Slack has channels. You go to a channel and all the people in the channel can can share. Whatever information or postman is set up so that you can very quickly if you want, you can create an account. You can make your own workspace and then you can share your workspace with someone else and you can both see the same request. We're not going to worry about that right now. You do not need to create an account. This is one thing where like you would never for the entire cohort need to create account. If you didn't want to. If you want to, that's fine. I don't care. You can do it either way. Some people do before the capstone because they wanna share things, but I honestly I'm not sure that ever helps anybody that much so. I usually just click the relax, which stops telling me that all the time I post man gives you the ability. To make requests and get responses. Well, let's go back to what those requests and responses are. So the web there are a number of ways to interact. Through this HTTP response request, there are number ways to interact with. The website or the database server or the other program or whatever it is? And those being the most common one and the one that you have all used all the time without ever thinking about it, is to get and get says I'm going to go out and get something and get just means I'm gonna. I'm gonna give it a URL. It's gonna give me back information. I'm going to use it. So if I want to go to a website that is a that has HTML. Alright then I will get back the contents of of that website. Whatever it gives me back. Let's try that for a moment. Just say, look, we know a website to look at. Let's do that one right. I'm gonna go back to my. It'll let me do this. I'm not sure it'll let me do this because and this is where I want it. Because when I do this it says oh, it's encrypted, which means I need to well it it is. LMS requires you to log in. So I have to authenticate. And if I don't want to put the authentication in here, it's gonna give me an error. OK, so if I go ahead and try to get my my URL. And it turns out it's got something it gives me. So what does it give me? You know what? It's keeping a session, so it will allow me to, but. Here we go. Whole lot of stuff somewhere in there it's gonna say, Oh yeah, this is where it's gonna say you want to authenticate. So let me put in a login form. That's what all this HTML is is that when I go to the lens that I'm not logged in. It's going to send back a page of HTML information. Right. But the key part of what matters to you is that where is I start pulling down. I sent it. A request here which had all those pieces for my URL that we've talked about, it has the protocol, it has the IT didn't actually have any querystring yet, but it has the protocol, it has the path and it got resolved. It knew what resources was needed. And it said OK, I have all this. Let me give you back information and that's the way and this is again this is the kind of thing that they want to understand when you wear mask and technical interview and they say what is the HTTP request response? It's like a message comes through which may have a URL. Well it it will have a URL always have a URL, it's called in and I'll go into that. It's got a method. And the method is one of these things, yet just says I wanted to. I want to get that thing. But you have things like delete I could say I want to delete that resource. I mean I have rights to delete that resource, but that's what it the delete method says I want to do this thing to it. And if I have the right to do it then it will go out and delete the resource, right? Post is the one that you will use. You will use get and post most often. Yeah. Is the one you already ready or using post is the one where when you submit any information. And it creates something on the other side. So when you go to. Twitter or Facebook or. Twitch or whatever it is that you use. I don't even know what most people are using these days, but it's probably none of those tick tock when you go and you want to put out a post when you want to create, even call it a post. But if you want to create a post or a tweet or a or whatever. You were gonna put information out. And that is going to and they are then going to respond. The response is different if it's a post because I don't. I'm not trying to get information, I'm trying to it. It will tell me what happened. OK. So as part of that, what you need to see in the response, these are the things you need to pay attention to in the response. I forget it's very simple. I usually just have a a URL. And I have headers and you don't even think about the headers whose most of the time you don't need to do much with them. I'm going to see if I can. Like what headers do I have headers that I have? I want to see all the hidden ones too. OK. the headers, you don't necessarily think about very much the headers are. The user agent is usually what thing is calling this. So when I if I'm using the Firefox browser. My Firefox version is going to get sent through as part of my book. Every time you you send you go out there and hit a web page or refresh or open something up for any of those things. It is sending a request to get request and it is telling it that you're going from. Firefox or whatever. And that information then can get tracked. As someone can say how many times you know how many of requests that come in are coming from this browser or that browser, and they may determine at some point that we're never getting any requests from this kind of browser. We are from those who do things special for those browsers. Most of that doesn't matter, just so you're aware, you're actually sending a lot of information that you can't see. Every time you are on in, in a webpage, interacting with with a resource, the web server you are actually passing a lot of information. What you will deal with in this is the accept. Header. And the accept says. I'm making a request, but what kind of information am I going to get back? It might be like normally you're thinking about. I'm going to get HTML back. 'cause that's what the web runs on. But let's say that you were URL was actually the action that the direct path to a JPEG file. To an image. If you say start a star, that means I can accept anything, but I can also say I will only accept JPEG files. Right. And so if I did that, you know, I would have to say that. Oh, I can't do it here. Let me create now I'm going to create a. I can't do it. Yeah, never mind. I'll get back to it. But I could go in and I can say I wanna accept only JPEG files and then if I would give it the address of a jiff file or a PNG file, it would fail. It wouldn't be able to give it back. So I'm telling you what? Can I accept this is gonna become really important, but we're working with Java and we're interacting with Web API. You're going to set what it is that you want to get back. Hi. When you get it back. We've sent your request and it's got that the URL and it's got the the headers not what you get back is always the status. The one thing you know you will get back is a status code. And those status codes you will quickly learn there are few of them that are that are really obvious. You'll get all the time and then there are really unusual ones, right? So I got back a 200 and they always tell you that it says like 200, OK, that means it's OK. I I got back something acceptable. Right. All of the two hundreds. R. Something went right. I'm getting something that I expected. That thing happened the way you know and 200 is the most common. It just means yes, it is OK. I did the thing I want and I got back. Some body of information. Whether it's a web page or something or the the image bytes or whatever it is I got back, I got something back in my what they call the body the request, the response body. Sometimes you will get it OK, which has no body. There's nothing to return. If I send. There's there's nothing to return, and then you get a 204 and 204 just means it's an OK. But I I don't have any information for you. So it just you know, so those are the you will, there's a chart of them in the LMS. You should look at it or whatever. But the ones that you will use and you will look for all the time are two hundred 201 which means I created a new thing and 204 which is means that you know. It was OK, but I didn't have any information for you. If something goes wrong, the status code is what's going to tell you kind of what went wrong. If I decide, let's say that I you've gone in, you've logged into your, you know, bank account. I need to your bank site. You've gone to your bank account. You're looking at the balance, and you want to go to the transfer page. Well, what happens if you copy the URL? It's at the top of that line. It's got all the gibberish stuff that they have and. And someone else tried to run it. Well, if they haven't logged in, what they are going to get is either a login page or more likely they're going to get an error which says you're not not authorized. There, there's some way to get authorized, but right now you're not authorized to see that page because they haven't gone through the entire process of logging, getting pretty credentials and all that sort of thing. So that status code is very important for us to look at because it is the equivalent. In Java of like raising an error or giving back a. Like is how you respond when you say no, you can't do that or or I was unable to do that. I was playing last night with looking around for something I could do that would you know, for a public API to to get something from there was one mentioned I think for exchange rates in the LMS. And this is just OK. So there's a website out there that hasn't a public API. And he. And it has an endpoint. Now the end point is how do I get access to my API? How do I get access to the to resources that it needs and it's the IT is all of the URL up to. The point where things start to vary. Right. And so if I look at this and I say the. The end point here is going to be the the URL we're used to. They often are like slash API or whatever. It usually has a version. And then one is this is actually my, it's sort of like a authentication code shouldn't really be in the URL, but that's what they use. One just says I'm a guest, so if I go out and say OK, how do I get whatever information it has on margaritas? It is going to go out and it's going to give me something. Alright, this is the response I got back. I got back at 200. OK. Right, that's important. I got back and this is the other thing that that you you don't need to know most of the time. But I got it back a whole bunch of headers. So just like you could send in your request you could send headers. In your response there are headers. And headers are not part of the. They've sort of like the the the metadata that that just if you send an email. I am so someone else. The body of that email is going to be whatever you wrote it. Whatever. You know, the information that you put in or or you send a tweet, the same thing, it's it's going to be the context. It's the stuff you wrote. The header is or. The metadata is all the stuff that's like who's it from? Was it to what is the date? The time stamp on it? You know, for an email. What is this subject? All that kind of stuff is called. Is is what in HTTP is the headers. And most of it. You won't understand much about it and you shouldn't it. Most of it is things like OK, what, where you might have an anti spam type thing that is out there and it is interpreting stuff and it gives you a spam score that tells you how likely it is this is spam or things like that. It's going to do stuff like that. It's also going to tell you if there is any particular. Way that it is encoded. If it were a. Things that you might see, for instance, if I when I I give you the example, I went out and what it's retrieving is an image. They don't wanna send you all the random bytes that make up an image because it's so easy to to mess with the. Like to make it hard for the program using it. Uh to to to use that information. So usually when you send a a binary kind of data, there will be some sort of encoding and the most common form is going to be base 64 encoding. Age 64 encoding you may have seen before. It just looks like all it's all the upper and lowercase letters and the digits and also equal signs to fill things out. It's just a way of turning. Stuff that is all kinds of bits and bytes into letters that will are easy to pass around. So there may be encoding. That's another thing that you will deal with some of the time and it's like what server is it on? And you know a bunch of stuff that we don't care. But there are headers and there are few who we will use in our Java programming. Organist. But the body is the content of this stuff. Right. Alright. That's a lot of content to throw at you, even if it was supported by the reading. So let's step back for a minute and say. Where did I lose? You know if if assuming I lost somebody weird it, I lose. Or are you all you know? Peter. It's always me. Um, someone. Some of this makes sense, but I'm just trying to understand. I'm sorry, this might sound weird. What's the point of postman? Whatever. If we if we if we're just able to pull up other people, websites and stuff, we can't really do anything to him. What are we pulling them up for now or is this something is for us when we're out? I'm I'm just a little lost. No, it's an excellent question actually. If we step back a minute to what we are doing at tech elevator, like what is the the point of all this? OK, what are we getting to right? Besides getting you all jobs being all, you know, find me fancy drinks when you Take Me Out afterwards. What is the? What is the goal here? OK, the goal is to be able to develop. Complex applications for for big companies usually alright and. There are different parts to that. To a web application, it is not just on the web, it's got a database behind you. Go to Amazon. It's got vast databases of information, so you know what products are there. Right. It's got middleware which is the Java which drives that which which handles the data from that database like we're going to be doing this. What you doing very shortly is starting to interact with our database. From our Java. And then it's gonna turn around. It's gonna be. How do I drive the website? How do I interact with the website? Well, the two ways in which we need things right now is we need to interact with our. Database from Java and the way we are doing that is essentially with this whole kind of response request cycle, but that's the database. Then we're going to turn around, I'm going to do the same thing with the web. So the reason word why we're doing all this is so that you can interact your program. Can interact with the website and with the server and with the database it can interact in all those ways. And it does it through API's. API is an application programming interface. So if you think of an API right now, you have been building. Something that is entirely self contained. It it is. All of your Java, all your data, all of it has been in the same place. Whatever you are doing is in your program, you control it completely. This entire module is about. Breaking free of that. So that we have client server which is despite the name really means one program interacting with another program. Whether that one piece of software interacting with another. Whether that software is on my same computer or whether it is across the world on some other computer. So the reason for this entire thing and learning how to do it is to start that communication, because HTTP is how that communication happens. In the in in the world today, this is add. There are actually other ways to do it, but this is the most common way that people do the interaction with other software programs. So the question being, yeah, sure is that kind of how our bash works with like how it sends the information to boot camp. I meant. Is it? Is it kind of how it like when we do our pushing and our pulling? Yeah, I mean that is get is a. It's a form of communication like that. It is a very specific one, but it is. It is like that end. It is OK. I'm taking my stuff that is here and interacting with some other system and I'm sending it messages and it's giving the responses back. So if you look at it, it's not actually using HTTP. HTTP, But it's a similar system or may actually be using it under the covers, but essentially what it is doing is it is still I am trying to push something up is giving me response that response might be an error sometimes when you guys try to push what do you get you get read text and you go by you know and and. Then you have to figure out what the right things messages are and what to do to resolve those errors. That's actually exactly what's going to happen here, and you're going to go the same way because it's going to give errors and you won't know why and you haven't done your you know, something connection right or wrong endpoint. That's all perfectly normal, right? But. Yes, it is very similar to that and that this is how you were gonna push and pull information from some other software. So in that sense it's very similar. OK. And then I had a question. OK, I was following along I think until we got to headers, could you just go over headers what they do and how they work one more time again? Thank you. Yeah, the headers, but the challenging thing about headers is that they're they're a little bit. There. Again, I'm mostly can only ever think of it in terms of like if you receive not even just an email, but if you receive a package in the mail you know or like a literal package in the mail, you order something from somebody and. All you care about is that inside it has the book you've been waiting to read. But there's a shipping label that has information and it's got maybe a postage. You know, his thing has been stamped on it and it's got information that that rides along with your package that helps it get from one place to another and help people who might need to know what's in it. I deal with it appropriately. If it has a warning label on it that says, you know, this is highly toxic. They might not send it through the same, you know, they might not put it on the same truck as as they would otherwise. They may still get it to you, but it's going to some other way. Headers are the equivalent of that. It's when I send the information I have to give it some information so it knows what to do. And it tells the other person, imagining your package on it said this side up. You know, don't don't put this upside down. I always get those upside down. Yeah, like I I think that they go out of their way. It says this side up, it turned it upside down. So I think that has the wrong message. But the message is there as a like how do I handle it? These messages similarly have. How do I handle it? Right, that's what headers are. The message that comes back is the both how I hint but and how the person getting their response should be able what they can do. So if you send off something and then somebody sends something back to you, it's almost like the sent. The thing is, if every package you ever sent, so I know that my grandmother long ago she really wanted me to write letters. Now this was back before email anyway, so she really wanted me to write letters. And the deal that we had was that. If I wrote a letter to her, she would always write back. And I tested this at some point I think I sent her 50 letters over about two months and she sent back a letter for every one of them. She was dedicated. I was insane. Whatever. But it worked, right? The request responses like that it's every single message that goes out must have response. The reason it must have a response is let's say that you put a request out there. And the Internet died on you. You know, your Wi-Fi went away or something like that. I don't know whether that message got where it was going. So the response is to say it's the, it's the constant, you know. Yes, I actually got your letter. I'm going to send one back to you and that's. in this case it's a yes, the package was received or or they do actually do it. Sometimes the packages where if I send it, it's gotta get signed for. And I really I get something back that says this hand was handled. It's really kind of like that the header that comes back is going to be. What? You know what is the what is the response? Was it OK? But the other really important thing about the header that comes back and this is mostly to where you're going to care about it. Is it says? What did I send you back? Like what if I said I want? A bottle of your best wine. And it might come back and say I decided my best wine was Chardonnay, you know, or whatever you like to decide how to deal with it. Based on what I said. Came back. Well, in here when I look at my header coming back alright with my header. This is my header going out my head are going out. It's saying. What can I? Yeah, back. So I go to my header coming back and it says somewhere probably says OK, what is the content type that is the most important thing you will see. And it says, what is the stuff you sent me? Like maybe I went and requested an image, but what you sent me back was HTML with an error page saying that image is unavailable. If I think I'm getting an image back and I try to, you know, use an image which is actually an HTML page. Everything is gonna go bad. It's not gonna work. So one of the most important things that always comes back is I went out and said I would accept again so that you understand these two things I said I would accept anything. But when it comes back, I don't. I can't deal with anything. I need to deal with something. So what I'm dealing with is Jason. And the content type for Jason is application Jason. Right. This is my. Oh, this is what I actually put in the box that I sent back to you. And then as my primary Member, this is not when we were using postman and this is really important when we're using postman. We are a person looking at what happens on the interaction, what we're doing is mimicking what's going to happen in your software, where your software has to deal with it. Your software can look at it and say, well, you know. No, Doug, it looks like Jason, which we haven't even talked about Jason yet but you, but you quickly recognize Jason said well as Jason. That's great because I happen to be able to look at it, but my software program, when I've made a request to get something back, it needs to know what it's dealing with so it knows how to process it. Right, so the header is the thing that tells me. How do I interpret the stuff coming back? Occasionally, for instance, it could be that would come back. Looks like Jason or looks like HTML, but what it'll tell me is that it's plain text. Imagine that you sent an email and remember emails virtually always get sent as HTML. You don't see that you see the content. Well, imagine that I pasted some HTML in the beginning of my email. It needs to know that even though it looks like HTML, it's not an HTML page. I want them to see that HTML. I don't want to see it interpreted into it, you know, rendered into something. So the header is a very important way of telling it. What am I actually sending back to you so that you as a software program, not as a person but as a software program? Can do the right thing with my \*\*\*\*. Anybody else have any questions on this because we're going to take a break after that, but there's I wanted. I can't want to leave anybody hanging. If you're just dying to know some other piece of this. Everyone sort of kind of at least doing good or Patty have something. So is this similar to this whole purpose of this? I understand like if you have like a website or something and you're you have like restaurants and you want to use, you know, put in directions, you could go to an API because you obviously you know don't have like all the maps of the country or whatever. But is this similar this function is postman similar to what we're doing? In a sequel, but just outside of it like you're you're pulling in like I want all the cities and States and whatever, and then you you get back results. Is this similar to similar in a way function of equal is? I mean SQL is a language. OK. In the sense that language that interacts with the software like it, it does do that. There is that where the API comes in is not the sequel itself, it's I'm calling my SQL from PG Admin. PG Admin is a client. It needs to send a message with that SQL included two Postgres. Postgres is then going to do its job and send a response back, so the response request is not the getting the tables or whatever else the response is. I am sending a message with two Postgres. And I've attached some sequel to it and it's going to respond either with an error or it's going to respond with some content and tell me what it is. So it's kind of what you're saying, but it's not the it's not the the, you know, select whatever, it's not the. That part of the language, it's that I take the a bundle up, all the stuff that I want to do when I pass it off to Postgres and say can you handle this? And maybe Postgres and says no, I can't handle that. And that's when you get in there, but it's PG admin actually doesn't interact with the data at all, it just passes that they're all off. And the reason why that's an important concept is that's the way your Java programs are going to be. They're not going to interact with the stuff they're going to pass some sort of request to someone else who handles it. And then they're going to handle the result. The response. But they're not actually touching the stuff themselves. It's all it's all. It's all part of a, you know, that message response process. Does that help by? Sort. OK. Let's take a break and we'll come back and tackle some more of this and we'll get into some Java code. Who? Yeah.

Um. Can you just share it or do I have to stop sharing? Yeah, I have to stop sharing. Mine says the same thing. OK. Well, it's like Akita. Oops. OK. So. My guess is that if you go to your your get and you click at the very end of it. Now yeah, go to the very end. And if you hit end, does it go any further out? I wonder if there's an extremist character on it or something. It looks like a return symbol. Yeah. She that. Hit Yeah, whatever that extraneous character was, and the reason I wanted to show that is because. These are computers talking to other computers. They're not inherently smart. So. It's easy to make you to make a request that looks right and another person might look at it and they might get it. But if it's not exactly what they expect, they won't necessarily be able to process it. Right. The flip side of that, of course, is that it means that if you are a person, if you are writing a program, you want it to be very robust. The classic example is OK if I and I've seen this both ways. If you if somebody enters their password. On a site. And they include an extra space at the end. This happens a lot because people copy and paste the password from somewhere else, so they copy that extra space at the end or the extra

At the end. It's your program smart enough to. Trim the ends of it to make sure there's not a space before. After there's no password, should actually have a space at the beginning or end. So is it smart enough to do that or not some are some aren't, nor the same thing with user names. In this case you had a

Character. At the end and it which shows up as that little you know return thing. You had one at the end of your URL so it said postman is going to be really really you know it's going to do exactly what you tell it and you've sent it that and if you look Cedric do you have do you see the same thing as there a little character then? Yeah, I mean, I have, I'm so used to dealing with this that I deleted it without thinking about it. So I didn't tell you to do it, but that's you know, that's what you see. OK. But but. But it's a good reminder you guys are going to run into that exact kind of situation in your. In your code where you think something is, is is all right and you look at it, it looks absolutely right and you've got an extra space in there somewhere. You know, that's the kind of thing that happens a lot. I have a question, sure, so I also never used or seen postman before yesterday I guess or even now and I just I guess I wanted to ask, is there anything in postman other than maybe saving or batching commands that you can do that developer tools in Google Chrome cannot tell you? Or is this like does it have a lot more that? It just has a lot more than that. It has. It actually has an awful lot you can do we we won't use a ton of it, but it could. A good example of one of the things that you can do is one of the one of the things we will do is create a collection and you can create a collection where. Where you you have a log in as one of your items and then it saves that in a cookie and then it shares it with the next thing which will then you know allow it to pass that one thing I didn't mention to you is that along with the other thing that you have in your headers are cookies. And that is how we often handle what I said. You always pass back the authentication with every single request. It's often stored as a cook, a temporary cookie on your machine, and that's how you're, you know, it will keep going to pass it along, but. Postman is good. I mean you can automate tests, you can do all kinds of things. You post it for the purpose of what we're using it with it right now, we will mostly be. It's a good way of saving a series of different things that I can use to interact to make sure I know what is really happening. When I make a request and what I'm getting back before I have to go do it in Java. In Java it's really a pain in the neck to go out and change the code and make it make that request and then try to get a debugger to show you enough information about what's coming back 'cause you have to sort of tell it what to look at. You can't just go look at it. Postman is this fabulous tool then for saying how do I, how do I deal with both sides of that interaction? I can communicate with, with, with anybody with that and I I think I could. I don't know what I probably could use. I probably could use the JDBC strings if I knew what I was doing and make get requests on my, you know, on my Postgres server there's a ton you can do it. What we're going to do is mostly just. Just model. You know how when your your when we've had you in in Java and we showed you how to call the database to get information. Right. When I keep telling you is go to you know PG admin first and type it in and make sure you know what your SQL statement looks like because it's so easy in PG admin to see what's happening. And so hard in Java to do that same thing and know what's happening. Wedding pads are the same thing. Going the other direction, which is. It can be so hard to tell what's going on in other ways, but if you're in postman, you can. You can reproduce the exact sequence of things that you think is happening. And what often happens if it works in? If it doesn't work in postman. Then it's not going to work in Java, and it's easier to debug. If it does work in postman, it doesn't work in Java. There's something wrong with your Java code now, so there's it's a good way of sort of telling where is the problem I'm having. Sometimes I and I have had this experience we all have when you may even have it during this covert where you will be. In your Java code, you keep tweaking and tweaking and tweaking and and you're never getting back what you want and what you don't realize is the server has the problem. You don't have the problem. But you don't know it because you keep thinking. Well, I wrote the Java code. I'm the most likely thing to break. Therefore you know I'm the problem, and sometimes it's not. It's the server that's having the problem. This will be particularly true because, as you will see today, you are both the server and the client. And so you. You can easily make a mistake on either side. Yeah, little bit thick, right? So that leads us to. OK, so to your git bash. Alright. Go into your lecture. I'm gonna go ahead and do this, but trust me, it'll take long enough that you will have plenty of time to catch up. Go down into your server. And do NPM install. If you did the reading last night before I do the MPM install, I'll look at what's what else is in here. It's got a package lock dot Jason and package dot Jason. Those are what it uses to figure out what I'm installing. I just say and NPM install. I don't know what to install. Alright, so I save my MPM install, it reads those files now again, there's this magical Jason thing. Right. Jason stands for JavaScript object notation. And it go back to my postman for a moment. And when you look at something that they get. She went. That's the ping file. So let's look at this one, OK. JavaScript object notation was originally developed for Java. And it just takes JavaScript. I'm sorry, it was originally developed for JavaScript, hence the name. It has since become the way that a lot of things communicate. You know whether or not they're JavaScript or not. But. It is essentially it will look up awful lot. Like a JavaScript object and it has a key and it has a value. You guys don't use JavaScript last, so you won't have seen this, but a driver scripting it has arrays just like JavaScript and where you do a square bracket and then everything you know inside is part of the array. In JavaScript, object starts with the curly brace and ends with a curly brace, so it's gonna do that same thing. It's gonna end with this curly brace somewhere down here. Right. And the reason why it was created was they wanted a way to make a string out of a JavaScript object. And so it does very little except put quotes around the key. And and turn it all into a long one long string that they can then pass as information. I'm not going to go a lot more into it until we probably get to JavaScript and stuff, but Java, Java or Jason is our way of passing information around in a universal way. The other one you may see sometimes is XML, which is another way of passing information and one of the reasons I want to mention that is that. With some URLs, but you see this accept slash slash the accept header as the star slash star. I might tell some you are I I might have an endpoint that looks like this. And I might be able to say what I really want is text slash, slash H or XML. And it looks at that and says Oh my default is Jason, but I'll give you that to use XML because your system wants XML that is entirely up to the service that is providing the response. Maybe it knows the XML as well when I I've written several of these API's for my products and they all allow you to pick between you know CSV and XML and Jason and a number of other things because different people need different things. For a purpose of our course, we're just going to use Jason. Jason is by far the easiest nowadays to use, but I did want to be clear that sometimes you're gonna, if you got it back and it came back as XML, you might have an API that defaults to XML and then you have to specify the application slash Jason so you get the JSON that you need. So that's another use of the headers. All you really need to know in here and the part you will need to know is if you see see a square bracket. You have an array. If you seek early praise braces, you have an object. And there are fields within that object. So if you compare it to Java and you think about how that looks, so we will very quickly get to the point where you need to do that. This is as if I had a a Java field variable. Called Ivy drink and then a string value which was eleven 007. OK. Back to our Git bash, do your NPM install and you hit enter. And it takes way too long. It is going out there it is. NPM is a package manager. For JavaScript type stuff. And it's going to go out and look, and it's gonna find pendency's between different things, and it's going to pull things in. And it does all this magical stuff. Because everything. On the web nowadays is built from a dozens of different libraries, packages and whatever else, and what it does is it's going to go find all that stuff. It added 229 packages. From 124 contributors and it checked 229 the auditing that you don't really care about, but the fact that I'm add trying to add 1 little server theory, one really poky little server. And then it has 229 packages. Is kind of intense, so you have all these different people who have stuff that might have a new version and it's going to do this. I will warn you, it's going to come up with the scary looking thing that says found 19 vulnerabilities. Sometimes it'll tell you some of them are high vulnerabilities, right? For the purposes of this course, you do not need to worry about what those mean, but the idea. Yeah, I never was able to even get to the server. OK. Does that mean you didn't do a get pull? I did get. I'm sorry. Get pull up stream Maine or whatever you did. You did that. I did that when lecture again. When you told us to. OK. I said do you see the lecture? Um. Can I share my screen really quick? Sure. Let's do that thing. Hold on one second. Sorry. I figure out how to unshare mine. I'm sure I'll be set, by the way. Go ahead. I don't know what I did wrong. OK, so hold on a second you have. Yes, you're not. You need to go down into module 2. And then into. No, you need to go to the city and in module 2. OK. And then 11 hit enter or to tab? You said put 1111 and then it hit a tab. And then do lecture. And then, Sir? OK, hit enter. And now you should be able to do your NPM install. Because it uses that package dot Jason, you have to be in exactly the right place to do it. Is this both have that warning like that? Yeah, it has all the warnings that has you notice you'll have a different number of ulnar abilities you've got hired by 8 end PM is so mysterious that that, that, like, never nobody can ever really explain exactly what's happening. And you all may see slightly different numbers and slightly different, whatever else. And trust me, at the end you all wind up what you need with what you need to do, and part of the reason why that happened, I mean, just like the part that I do understand III joke about it. But I do understand is that. Other people may have installed other things on your machine, or you may have installed them without even realizing you were installing them and. So what set of things you have available as global installs versus like I'm installing in this directory is. Complex, right? So. Depending on exactly which kinds of things you've done with your system since you've got you all pretty much got the same thing, but since you got that, you might actually see different results regardless, you all see different results than I did because I have actually installed more things than you have and so I know that I I will have things, but you may still see different results. Depending on like, I don't know the time of day and in the middle of class somebody may have changed a package. It's just monstrous. But the nice thing about it doesn't make any difference 'cause the stuff we're doing really would work with all kinds of different versions of these packages. They just try to keep up to date. Hold on a second, I need to. Alright, the second thing you're going to do is what you're going to do is you're going to start your server. Now I want to be clear that this server is the. Is a dopey little server that was written intentionally for the purpose of. Using his examples more than anything else, it's it's a little JSON server and if you do an LS in this directory. Not free place anymore, but OK if you go to. If you do an LS, you're gonna see Hotel Dash reviews dot Jason. You could actually look in that and get all the information that you need from that. What you will notice also is this thing called node modules. That's new. Node modules is NPM is really node dot JS. Package manager. Node JS. Is an important thing for you to recognize the name of, even if you don't use it much right now. When JavaScript was written, it was written entirely to be within run within a browser. That was this whole purpose in life. But at some point, JavaScript became so popular and so widely used that people said, well, what if I want to run my JavaScript program? But I don't want to be in a browser, I want. I wanted to do stuff. I wanna write stuff in Java and JavaScript, but I don't want to have to be in it. I want to be like like I can't with Java where I can just go run it from the command line. And so even though you guys don't haven't been doing that, you can do that? And by the way, I went and tried just 'cause. I was curious how hard it would be and I like, like all you really do is go down to the Java file in one of your applications. You could try it in your caps tone or something like that and just say Java space and the name of the the, you know Java file and it will run it from the command line. So you could run your you know try it out one of these days capstones weren't. Because it's command line program, it tends to to do that, but it's it's pretty easy to do so. Where I was going with that, but in any case. When you run this server here it is good. It's it's a little server that serves up whatever is in this hotel reviews dot JSON as a so that I can have an API that runs against it. For your purposes though, it makes no difference. Like you happen to know that the server could also be running over, you know it could, it could be running in Parma on a server that was doing it a completely different way. We don't care, we care that we can interact with it. So what we're going to do is is MPM start. I'm just gonna do that. And tell them look like it's doing very much now. There's just not a lot to this because all it is doing it is loading up. I think I went way back when I was going to the fact that node modules. That's the reason it's called that. That's where it's when you did the MPM install. That's where it put every. So now what I'm doing NPM start it tells me what to start and what is starting is something inside my node modules. If you're really excited, you can go down and you can look at it, but you know they'll be they'll be more stuff than you could ever believe stored under that folder somewhere because of those 229, you know, packages that install it just transfers huge amounts of the Internet down to your website. I mean down to your lap. OK, so anyway, it is watching, it is waving, is waiting for somebody to make a request. That is this whole purpose in life. So. we should fulfill its purpose in life because i don't think anything should just left We should fulfill its purpose in life because I don't think anything should just be left hanging. We're gonna go look, and we're gonna try to. Going to use postman because that is our thing that I can never find my cursor again. Cosan OK so. And let us get something from postman. Now remember I said that with a. When whenever you add call a Web API. It has what's called an end point. Which, ironically, is the start of your your thing that you're getting. The endpoint is where you land. And you start from there and you you you build on that. Most API's will have one general, like all the different commands you can do for that API will will start in the same place. OK, so their end point there will be a higher level endpoint and then what you have are you have endpoints that are for. Type of resource. So if it were bank accounts, I might have my whole initial part of my end point slash accounts. Alright. And that slash accounts is what's going to give you the resources. So the accounts within that. So I'm going to model that here, but the URL I'm going to use is that one that we talked about which is going to be the HTTP. It is not. I am not doing encryption. First of all because I am on my local machine and because it would be a pain in the neck to get certificates and things like that, I needed to make it work. So I'm using Justice CP and that's fine for our purposes here. 'cause I'm actually not leaving your life. You're not leaving your laptop. We're gonna do 127.0 point 0.1 the magical this, you know, here.: 3000. That is the port number. You don't normally need to worry about the port number right when you do an HTTP call, the normal the default port number that it uses is 80. So when your website is listening for, they call it listening for requests or doing like my server where it's waiting. It is listening at a particular port at a port is like a doorway that you get into the website. So. You might have different doors that go into the same you've got one URL at different doors for different purposes. So on the web, when you type your HTTP command, you know we you use URL and you go to your bank or you we go wherever you are. Virtually always you won't see a port number. That's because they all use port 80. Different kinds of applications we tend to allocate a particular port. Different types of applications. So if you have. Trying to think of Microsoft Teams or or zoom or whatever and they have certain kinds of things they're going to use a particular port that all video sharing programs use so they are defaults. We're going to actually specify it, and you just put a colon after your IP address or after your. The name of your website which says what port you're going to. Ironically, this means that if you went out to something like LMS, they did this thing. If you wanted to, you could. Put. You know 80. And it should work. And the reason it's going to work is because 80 is the default, so it knows what to do. You see if it does, it may be set to have a different default in here. Oh, it's HTTPS, so it's not. It's like 143 or something like that. Reason is sitting here, hanging is 'cause. I'm trying to go into the door for unencrypted HTTP, but the URL is HTTPS, so I'm on the wrong door and it just there's nobody listening at that door. OK. I never remember the one for this, but I think it might be 143. Somebody in the reading day, anybody remember the reading 'cause it showed you on the reading? What is it? I think Elias is angry. One forty 3443443. Let me try. Well, no dairy quickly. It'll either come back, then she add came back, so it defaulted to 443 because that's what HTTPS defaults to. So I am going to listen explicitly I or my server is out there listening explicitly to the local server at 3000. The reason why is because it may be listening to other to the port 80 for other purposes. When I go to a browser, I want to be able to get to the right place to the server that is doing this, so I'm going to have something listening at PORT 2000. Having said that, now I can put my past after that and my path. It's just gonna be hotels. Normally an API would have much longer than just the the this much of a path. It would actually have an endpoint that was slash API slash whatever. I don't need to. I don't my services only handling hotels and that's all you're ever going to that port for. So I don't need to do any more than that. And with any luck. It will come out. It will tell me the hotels. Look here. Alright. Let's start with in postman. If you if it worked OK and feel free to raise your hand if it did not work. If it worked, what you should have gotten is something like this. It has an array. And going to be in the raid because I'm asking for all the hotels. So it's going to be an array and it's going to be an array of objects, and each object is going to be one hotel. Object is going to have in this case an ID is going to have a name. It's gonna have an address which is another object. And that object you go down as deep as you want. That object is an address object and that has its own things. So if you think about this in Java. You might have a hotel. Object and it might have a field variable with which is an address object. OK. And so that's that's called address. When you do that, you could get to the the field values in there. So what else is there there's it's also got how many stars rooms available cost per night and a cover image right and it's got however many hotels it's got? So. My. If I go back out. I leave that going, but I go back out and I look. My server. Now normally it might cheat trap if it bothers to keep track of things you request like this, it probably does into a log file that nobody looks at, you know or on a regular basis. Because we're trying to show you what happens, it does it right here and it says the pieces of information that you care about. I have a get request. This is the path. It's beyond the end point. The result was a 200. I was able to do it successfully. And then it tells me how many milliseconds. It I have baffles me why they care how many milliseconds, but I I think it's because this JSON server is also set up to mimic some kinds of servers where if you start having delays like what if it takes you two seconds to get a response to something that's supposed to take, you know 24 milliseconds, there's a serious problem that somebody needs to get notified and so sometimes they react. To the number of dollars hikers movement, that's probably why they show it. We just use this. It's a free JSON server theme. We use it for the purpose of of lessons, but we don't need the number of bold seconds unless you're just obsessively concerned by how fast you think. OK. The fact that I can do this here. Tells me that I can also do it. Actually, let me just show you one thing right here. I mention localhost. What you will see a lot is nobody likes to type the IP anymore, so they'll type localhost and the. And it will do exactly the same thing. This is more. Older school people will tend to use the 127 -, 0.0 point one, because that's what everyone used to use and newer kind of people will use localhost, so you should probably use localhost because I don't know. You will look like you can come from this Millennium, unlike me. OK. If I have it and it works. Here I have it, yet I have nothing. I didn't do anything special. Authorization. My authorization is just, you know, they say inherit from parent which basically means I have no parents so that. Harant. We'll worry about later about how you get it, but there are all kinds of different authentications. I'm not doing any of them, I'm just leaving it alone. So if I can do this, what does that tell me? It tells me I can switch over to Chrome. And I can say fine, let me do it from Chrome. Right to do it from Chrome. Guess what? I'm gonna get the same thing. Your website is perfectly happy to show you the the Jason. I will tell you that it will look different depending on which browser you use, which can be confusing. It will have the same information, but if I go back to postman and look where I. Did all this OK? And you remember in Visual Studio code. Wait, where? There are places where you can go down and you can change the what type of content it is and it will have different syntax highlighting so that if it's just thinks it's plain text, it won't highlight the same things that it thinks it's Java. Right innocence Postman is it uses a general thing which they just call. Pretty. And then it has like visual if I were HTML, pretty would show the HTML formatted nicely, but if I clicked on visualize doesn't have a visual. I don't know what it does in here. Quite frankly visualize for Jason. Probably doesn't anything it. Yeah, it says you have to set up a visualizer there is from one. If it were HTML and I go back to this one. I'm seeing pretty which they pretty version of it RAW is. It looks like that. So you see how that's exactly the same code, it just kind of did the syntax highlighting and and indented and whatever else. So it made a pretty version of it. But if I go to visualize. OK, none of them are set up. If it went to visualize normally with an HTML page, it'll render it so you see what the page looks like. So be aware that when what you see as pretty. Is similar to what your browser might do when I go back out to here. OK, pretty is this. What is the raw? Well, it's pretty similar, but it's not exactly the same. This could have been all run together like you don't need space so it could be crammed together and it could look hideous in your pretty is still going to show you this kind of like oh, give me that ability and it also will often give you the ability which. In a in a web browser, but being able to collapse certain objects that you can go look at them. So the point being that when I see this, Firefox does a much prettier version of of this than Chrome, for reasons that I don't know. But now that I've got this. One of the things that I can do is I can go and I can look at the developer tools that are in Chrome to get some of the information and you'll see that it matches. I am not an expert on developer tools in Chrome, so I always have a little trouble getting to them, but they are aware ever more. Developer Tools Control shift I it says right. You've got bunches of stuff here that. OK, let's see if I can get to it again. It should. There's network. OK. So network right when I do my hotels. The network is going to tell me what requests were made. What the status was, what type of thing came back? The size the milliseconds all this kind of stuff. Well, notice what it came back with. It came back with two things. Even though we only thought we were looking for one thing right, the favicon is is sort of what icon I am going to show in my browser so that you notice that postman when it's up there has the has one little symbol. And Git lab has the tech elevator symbol. Yeah, and zoom has its own little symbol. Whatever it is, that actually, it's an awful lot like Postmates. Then maybe the default one that is the favicon. Um. And so if you on your website. Set up a default fabric count for your thing. Then you can have your logo show up in the browser and it will. You'll be identifiable and all that kind of stuff given that. I don't know what she, that's that's a great little fabric on that. It comes up there and it's got a little curly braces with eyes on it and very pretty I like that. So they actually went to the effort of choosing that when it can find it, you'll go get it. But what I did was to get requests and it got responses and I know the types. So you can see the same information you would have gotten. From. Postman. Alright, I will tell you the one difference is the postman doesn't bother to pull fabric ONS down because they're not showing the browser so they don't need it. That's a browser thing to go out and request it whenever I get up page I go and do a separate request for that. You know what is it? So I can show it it is. And so that's that's why. You can do much more, but I will tell you that sometimes it is very handy when you get to the point where you're doing web pages. Like you can mimic a certain amount of stuff in. You can mimic a certain amount of stuff. Inside postman. But once you get to JavaScript running and stuff, it's gonna put stuff on the console. In here there's the equivalent of. In Java we do the system out. You know. Print line or right line or whatever it is print line I guess isn't. When we do that in Java, it goes to the Java console or the command line when you do something the web, there is an equivalent and it goes to. This console nobody ever sees this console. The only time anybody ever sees this console is when they're debugging an application of these developer tools, so you don't normally want to put stuff out to adjust the way we don't normally want to put stuff out in Java once we're beyond command lines. But it is all there. And your tools are here and there are advantages. I guess. My point is there are advantages to. Using the Chrome tools at a time when you're really actually working with a web application. Until you were at the point of actually having web application is easier to use postman. and you can also the other advantage that the web application have is you can you can stop at any you can watch it as you go through a bunch of different steps where you don't want to set up every request in postman to to mimic all the stuff that you would get back and forth alright but postman is remove for setting up things and having told there's a way to get not terribly concerned. OK. So. If we want to make our own application tool, if we want to make our own Web API. In Java. We can do that. We want to be the ones in charge. We want to be the ones that give information out. So part of the thing that we're going to do today is we're going to go through an application that allows you to write your own server. 2. I'll accept a URL. And respond with resources. This is how we the way we're going to get into the whole working with Web API's is we want to be the Web API. We have the power so. let us take another break because at that time will come back at 11:15 in the world.

So the idea behind a rest API was originally the idea of OK, if we're going to make it stateless if every request had to be individual, then we had to have a process and a an accepted way of handling. This essentially they transfer the state back and forth between the the the user, the client and the server. That's what it all stands for, right? What it really means in most cases is we have a we've come up with a particular way of setting up API's so that we interact with the resource that we need and the path of the URL will hand. Will tell it exactly which resource to deal with. Before or even now, when you don't have a Restful API, often what happens is you will go and you will start using in particular. Resource. You know, and then you will. The server will keep track of your your session and it will know which research you're on, so you don't have to be passing that information around, and that's more efficient in some ways, but it it leads to a tremendous amount of difficulty with dealing with large numbers of people interacting with the system at once. All of a sudden these. The server has to keep track of all these sessions and all the information for all these sessions and never lose it, and they don't know when people have walked away from their computer and they're never coming back, so they have to keep it and at some point they have to know when they can get rid of it. Even though Chris maybe walked away for an hour and they come back and cursed the computer 'cause it don't no longer remembers where they were. It's it's just it was a it was a bad scene, getting worse and worse as we got bigger and bigger systems. So rest APIs say we're going to handle all that we're going to pass it around. You're going to use the the the URL to identify exactly what you're interacting with. So when we interact with. We do this. Call us the postman I guess. When we are doing this call and we want to go to the hotels, I should then be able to get to a specific hotel using only the information that was passed back to me. I shouldn't have to know anything different than what is in that information. That was passed back and that's how I get down to the individual hotel. I mean, if I want to get to the address, I should be able to do it by by. That information and you walk down the tree, you get it. So we're going to do that in our Java code. And you will not be surprised because we we try hard to make life easy spring. Has a thing called a rest template. Right. It is their template that allows it. Makes it very easy for you to create your own rest API. And. At this point I think probably like 80% of all the API's that you will ever interact with will be rest and there may be more by the time you're interacting with pleasure. So the idea behind a rest API is what do we need to tell. We give certain information to the. Give certain information that allows. This rest template to make call to the correct endpoint to retrieve the information that we want. And remember we're now what can be very hard to understand here is that we are dealing. We are the ones actually dispensing the information. So somebody else is actually requesting it of us. We are being the server right now. What you will see in your model in in your over here it is the beginnings of what is later called an MVC model or a model view controller. But I just tell you that because you'll hear that term and we are we're starting down that path, right. But what you really have here is you have your model just like we did with the database. These are the actual objects, so I might have a city and the city has certain pieces of information. Hotel. Yeah. It's got an ID. It's got a name. It's got the number of stars. It's got the rooms available. It's got the coverage. OK. So this is the the job of model. And it does not have much beyond getters and setters, just like it did when we were dealing with the disease. And then we have what we call a service. So the service model is this is these are the the thing that that handles a particular kind of. Device or or way of handling the resource so our console service in this case is going to be the thing that actually deals with our console. So right now is the. The interaction between, say, a user typing something in and the server. Our hotel service is our API. So in your lecture you already have a full hotel service and if we went out and ran it, which we went to in a moment, then you would be able to see. That running OK, you'll be able to see interact with it. You could then fill in information from here. So. Sure. At the right pieces in the right place. So how do we do all this? And then we'll take a look at how it runs, right? Let's look for a minute at what we want to be able to do. And then we'll look at how we do it. OK, so we have a main menu and it's gonna we want to be able to list the hotels with the reviews, show the details for particular thing. Right here. We now want to say, OK, our hotel service should be able to call that that API. Our hotel service will have in it. Things that end this should look very familiar from what you do with the with. With the database it should have like. This is like the interface that you have. When the DM model. It says. What can I do? I get a list of hotels. And I then need to we don't do it with an interface, but I now need to implement this list of hotels. So that my console service can print them out in a way that makes sense. And the way we do them, they are. They make this this use this used to be a nightmare of a thing creating APIs and it's now a relatively simple thing but, but let me show you the model and then move on. So if I want to just list a hotel, there are a few things I need. I have an endpoint, my localhost 3000. I know where I'm starting. I'm going to need to use. Right. So I'm gonna call my not going to return this. I'm gonna call my rest template. Which remember, is up here as a class variable and I'm going to say rest template. God. Yet or object? And then my. Base URL. Is going to be. Can't stand it when it covers things up. My API base URL. Notice this in all caps like that with Snake case. That is, a final variable is a constant. OK. Put it in here. Alright. And then I'm going to say what is the kind of resource that I am dealing with. And the kind of resources hotels. Do quote else. And now? The next argument in this is going to be the. Um. Remember how when you did a query for object in for the DL, you have to give it the the class of the object. So that it knows what it is, what, how it is returning the information. So in this case, what we are doing is we are paying an array of hotels. The way you do that, this is not. I'm not in the right place, so I'm certainly not where you do it. When I'm not quite place, when I'm right here the way you do that is I just say my hotel bracket bracket dot class. What this does is it passes something and it says. What should I be? What objects should I be creating using the information? Now. That's the entire thing. It gets a list of hotels. The rest template is super compelling in all this, it's super powerful. And what it is going to do is it's gonna say I'm gonna take this thing that I saw out here or I'm calling the same URL. I'm gonna take this information and I'm gonna map it into a Java object. So we're that in charge of saying, OK, there's all this information, I want it to be mapped onto a Java object. I need to do the setup to allow that mapping. And. When I go I'm returning hotels, so when I look at my Java. Alright. Out of my hotel. I need to set up. That things here so that when it looks at the JSON which is what it gets back here. It knows what to do with it. Right. If I look, this is the the magic is is actually relatively straightforward, but I look and I do it very much like I did with the DAO with a database table. I look at the model and I look at the names of the of the various pieces. And they should, Mac. Directly to the names of. In the hotel. So I have my ID and my name and my stars and my available coverage. Notice it doesn't right now have the address and that's because it's harder to map and an object inside my object. But the idea is the the names are going to be almost exactly the same. It will allow you, in the reading showed an example of this, but we won't use it very often. It will allow you to use something in snake case. It will compress it and turn it into the right knee so it recognizes that there is some ability to do translation. We'll see when we get to the city that that you can also say I will tell you the translation. So it's the mapping is not incredibly obvious though television the information. But here's an important thing. The class object and this is different than what you saw in the DL model. I don't need to have all the pieces of information. Damien passed a ton of information. Now I may not want it all. My model says what part of the information do I want? OK, for a hotel, it's a good part of it. But it is not. Let's see what it has. We have the name, the ID, the name. We don't have the address with stars, rooms available cost per night in the coverage. Do we save the coverage? I don't remember. Yeah, we have the coverage. OK. So we have all those pieces of information because we're going to use those things. So the mapping, the rest template will then do all this. OK, let me take this stuff. Let me pull it out of JSON, store it in values in here. This is so much easier. Then what you were doing with DDA aware you have to figure out the translation of types of things like this. The rest template does it all for you. Yes, do is it does the rest template do it because the ID so the JSON object has a or of a field called ID and the. Add models is that it just maps is literally maps. Yeah, but if it were like ID or some other name ID under, I don't know well under score ID if you were using like in the Davis design. If you use Member ID and it was called ID. It doesn't know that those two are the same thing. OK, alright. I give you a flashforward 'cause. It doesn't really. It's not very secret or anything. There is an annotation that we use that will allow you to go from the Jason main. To the JavaScript name or the Java name. So if I had member ID out there and I wanted it to be ID here but it's versa, I would just do this annotation right above the variable. And it would know that that was the translation. So this this would not actually neither of these two would actually be necessary, because it already knows to do snake case and things like that. But. I always do if I'm burying it all. Between my external and internal name, I go ahead and put the Jason property so that I don't have to worry that you know, I know it's unambiguous and so the person who's looking at the model can say, you know, what is it? Where is it coming from, especially if it's if it's at all different. But but even if it's very mild and different, I will tend to do this. Now. One of the reasons why you wind up having to do this kind of mapping is with the database. When you were creating the the DLU. Whereas I like to say that God of both the database and the the the Java. You control both of them. By their very nature, web APIs, somebody else is in control. And what I want to use in my job, I may not be the thing that they want to call it or the way they want to call it or anything else. And so it can get more complicated than this, but the the core of what you're trying to do is you have a model, you give it IDs. I mean you give it sort of variables. With names that map to the Jason and then you pull the information in and the rest template will just just going to get the object and then this is not even just an object, it's an array of objects. I'm just gonna get this whole array of objects on the map, each object and what you will have at the end of this is a list of hotels that done it. Move right So. Forget how to. Start running this in. OK. List reviews. That's one is we have a review. Similar thing. You could change the hotel, see that there's an example where it is very particular about capitalization and things like that. So in the hotel have ID happened to be ID is is all capitalize, you'll see that a lot. I I don't know why honestly in. I would quibble about it. Over my lifetime, when I use, if there's an ID, it's almost always all caps. In our course of study, we tend to always do this. More strict camelcase. But if you see it on the web, there often it would be like this. It's very common to have the ID half, plus it will not do the translation automatically between those because remember Java is case sensitive. So what it's going to look for? It will get rid of the underscores for Snake case, but it will not look for and find that hotel ID with a capital D is the same as hotel ID without. But other than that, it's a very simple thing. It's going to say, OK, there are reviews out here I want to take those reviews and I want to run this, OK. So. I do. To build for a moment. Alright. So how would I end? This is fairly straightforward stuff. How would I go ahead and and write the list reviews that? Having just seen what we did up in most hotels. What is it gonna look like? What do you return rest template? Give her object. The API basically the same, right? So yeah, and then. Stop. Plus, sorry, no, go ahead. Well, yeah. What do you think will be? Plus reviews. Yeah plus. You know what? Plus reviews is. Like if I can right now I'm not controlling this if I control the other side, I could say what do I think it should be called? I I wanted this is a precursor to where we are, but. The norm for resources is that it should be a noun and it should be plural. So this is unlike in the DAO where the table should be singular except get used to these things. The norm is to have you know, whatever and. And plural alright, but every API will do the thing that it wants to do. That idea is all that always gonna be. Like. Can I can I read it? They're trying to make everything readable, and it turns out that when you're doing this, you'd say I want to get the reviews and so they made it plural. Whatever. Right now, what's what's after that at the end of the string at, and then capital review the object and that would be an array. DOC class) and then; And that that is the whole thing. So so far pretty simple. Now we're going to say now, what is the normal model for getting the root? What if we want a particular hotel, for instance? OK, let me. Let's go back to our thing and say this is our hotel. OK. So the normal thing given that we are passing our state with us, OK. Back to that idea is that if I want to get. The Hilton Cleveland downtown. Right. I want to use it is Hotel ID 2. The norm about how you do this is you just say I'm within the hotels, so I use the ID or the primary key sometimes, but usually it's a number, you know, but whatever that is, well, I I'll just say it's a primary chief. Think of it as the primary key. That's easiest way to look at. It happens that the primary key here. It's more like a serial number, you know? It's just the second one. File and when I do that and I go should get it just the information for that individual. Just the object. There's no array and it's just that one. The state transfer was that I looked at my array. I found the one I wanted. I got an ID and then I could drive dive down into it. Sometimes in API's and but they don't bother with it in this time when you do your list it won't give you as much information when you go to the individual one it will give you more information. That's a more traditional model, so they the list might just give you IDs and names. If you wanted to get the actual individual one you would then use. This dive down into this and the the way these are setup you there's a whole system of getting used to. What system of identifying? How do you actually get to your your individual resource within the JSON? But usually it's just the first field they usually it's called by B, so it's pretty easy. But anyway, so this is the the goal. So in my Java. If I want to do that same thing I just did here. I want to get a hotel by an Ivy. Alright. How am I gonna do that? What is it gonna look like? So it would be up return rest template dot get for object. For object and then it's going to be the API based URL. Plus hotels slash 2 or so hotel slash hotel slash and then plus ID. Slash. Plus. And then, hotel dot class. Hey. That's how it works. It's just that easy. Right. So. Now, jet reviews by Hotel Ivy. Right. This goes to the next level. And I'm having you fill these out 'cause we could try running up and like it doesn't really matter at this point. What really matters is you understand the pattern need go. So I'm how would I, if I look in post man? I wanna get the reviews for the Hilton Cleveland. Right. It is possible that there is a separate. In fact, it's quite likely that if I go to reviews for instance, it may be available. There are different things and they have the ID's, so I could go out and get a. Could get an array of all the reviews and then look for the reviews that were from this attempt, but the classic way that this happens is that even though I can go to reviews that way. Is that I can go to hotels slash 2 slash reviews. And it's going to give me the reviews that are just for that hotel. Well, there aren't any for that hotel because they use different example than I did. So will do for one. So. Cleveland. Hilton Cleveland Downtown has no reviews at all. It makes me question things, but that's what it is. So here's the idea you're going to say the resource within the resource. This chaining of resources is very common. But remember that it should always. You should think backward like I'm narrowing down at every level. Hotels would be a whole list. Hotels someone is, you know slash one is going to be that specific hotel. So if I said something here it better be within. H1. You can write it some other way, but it won't be a well formed API. The the goal of a well formed API is I should only put something at this level right, which is a well formed DPI. But what if I want to get to? This review. I mean, there's no ID in it. So I either can just say do I get the first review or do I do it like that? Do I need to get to an individual review? I this is again, it's beyond what we need for today. It's it's more that just the way you design the database heads up, you're going to be designing an API so when you design an API you're going to think about these things. Do you have a need to get to an individual review? And if you do, it should probably have some sort of way to get to it. For the purposes we have right now, this is what they're looking for, hotels. Individual hotel and the reviews. So. This is all of you who don't respond super often in class. This is your chance because, like, these are fairly straightforward. They follow a very exact model. The end result should look like this URL. And every one of them is gonna be, esentially call the rest template and call the they they get data for object. So can someone tell me how to fill out this one? Someone who does not respond very often. You know who you are. Or maybe you don't know you are not always self aware, so return rest template. Get for object rest template dot I'm gect URL is API base URL. And then hotels. Slash. And then plus ID., and then Hotel dot class. That's going to get you the hotel. That's not going to get the reviews for the hotel. So what is it we changed in here to get the reviews for this hotel? So you just added reviews. Just gonna add the reviews at the end of that, so you're gonna add slash reviews 'cause you need to. So you're just going to say, OK, this is almost right, but now what we need to add is slash reviews. You're literally just building the URL. That's all you're doing with this. And the final thing you do is you say, what am I getting? That is simply the same thing as your return value just for any of these. So what is the time? Getting. Right. Would it be reviewer a dot class? You. Right class. Right. Now this is also on 26. It should be a hotel ID, not just idea for this one. I'm sorry, should be hotel ID. Yeah, yeah. No, thank you for noticing. I didn't. Yeah. Didn't notice that at all, OK. Now it turns out what is Capital D is also capitalized in this particular example. Yeah, 'cause, we just can't be consistent. OK. I think that's actually, that's the reality. The reality is we are consistent and you have to pay a lot of attention. You will all capitalize it wrong and then it will give you a compile area. Fix it so. Is there? Is there a reason why somebody did hotel ID instead of ID? Nope. Whoever was typing it in, you know, was happened to be someone older than a certain age. Probably II. No, really. There's no good reason. Then we've ever heard of the the idea designed by committee. Anything that is designed by multiple people at different times or fixed by multiple people at different times, somebody is going to put it in and you know there's not. Unless somebody else happens to notice and complain about it, it'll sit there. And so when you actually, when you're in the real world, no matter what the rules are. I mean, you're in the real world. What you'll discover is you'll interact with lots of code that will have lots of different, even. You can sometimes tell who wrote certain kinds of code. Like you'll get used to like, oh, that. That must be. You know, that must be JS code, because she always tends to do this. And there's no way to enforce absolute regularity so. It is what it is. OK, so get hotel by star rating. Well, that's a little strange. Like, what is the thing we need to do here? Right? And it turns out that we don't have a. The star rating is not a resource, it's not like reviews, and so it is. In this kind of case, where there is some attribute, we usually are going to do it by some sort of query parameters. And a query parameters are going to give us the information, but I could try to figure it out here. And remember you often don't know very much, but you're right. Yeah. You're trying to find out if I were sitting after playing, I was saying, OK, I know that it's got stars and it's not part of well is it don't want the reviews. Do I want them now I want the hotels. OK, so let's say that I want the hotels and I want. So I know that I'm looking for the hotels, but I only want the ones that are four stars. I would tend to do something like OK stars equals 4. Right. Maybe if they're right, maybe I don't look at him and I say, are they foresters? Well, I did it right. What if I said? And star equals 4. Like I'm guessing wrong here. Hey, maybe. OK, so that says stars equals three that says stars equals four. I clearly have done this wrong. But it still gives me a perfectly reasonable answer. Right. Why? Well, when it processes query parameters. Unless you go to a lot of effort, it doesn't care if you put in query parameters that make no difference. And you know why? That's true, 'cause. There's a really good reason for that. It's not that people are just lazy or something like that. Try pasting a URL. Into Facebook sometime. Facebook is gonna add. A dozen query parameters that it uses for tracking things, and it will break your URLs. A lot of the time. Like, sometimes you realize won't work and things like that many systems will add their own query parameters. So instead of saying I know the parameter, I know the the the the parameters like I do in Java. The web is Wild West of you know, this kind of stuff. And so it will look for things it actually recognizes. And it will look for the first instance of things that actually Russia right recognized. So for instance, there's nothing illegitimate about. Except that it's idiotic, right? Equals two right? Chances are now again, depending on exactly how was invited, this will show you the four star thing, but not the two start thing. So it goes. Query parameter after parameter looking for the ones that it recognizes and it does the things it wants if you misspell it. Then it's the equivalent. When I do the star equals 4. Right. And and I say this because it's when you're sitting here, you can kind of see what's going on. You're running from Java and you're getting it completely different result than you expect. The reason could be something like you spelled stars, stars or stars as star. And all of a sudden you're getting all the hotels and it's not doing its job, you feel like. But I told that which stars to do. So this is another reason to go out and try your URL in postman and say can I figure out what it's doing because it's not getting the results I want. Go back into our. Alright, so if we're going to get hotels by star rating. Very, very, very slightly different, but still. Don't do your thing. Somebody tell me. What is this gonna look like? Would it be rest template or yeah. Target. For object. API base URL plus. Hotels slash. Plus hotel ID. Well, we don't have a we don't have a hotel ID in here. We only have a stars stars. So it's going to be close to this. It's not going to have the slash because if we look at and it's, is there absolutely fair jump back and forth between postman, figure out what worked and what worked was.? Stars equals 4. So I am literally gonna go over and say I'm gonna do that thing. Let's see? And I'm going to say, wait so? Stars. Equals. And then over here, I'm going to say I have the stars. Hey. That's going to work. Right. Do you need a hotel or a that class? I do all right. OK. Now get with customer ID. Get with customer. Custom query when you're looking for here. How their point here is they're saying if you wanted to go out and do call a completely different API. I was mentioning the. Let's say that what I had was a list of up. Series. Right now and we really use sheet music, CDs or music, whatever is the things that you know, whatever you do now, you know use that thing. So whatever it is you have that has a collection of music on it by articular artist and you are getting that back. And it has information, but it has no information. Well, one of the things that you is very important to you in your application is to know what the artists and ages. Hey, you wanna know their age? And it's not part of your API. So the idea here is what happens if I want to go out and I wanna get a completely different piece of information that might be using some of the information that I have. So this is not a great example of what they're if, but the idea is that if I wanted to get information from 2 APIs, I could have another query that is not related to my API at all. I'll call something else. So the example they give which is you know. Super useful, useful. Right. I wanna paste this in because it's just not even worth. So they're actually saying, well, what happens? I actually have another API that could give me a city that I could use along with my city. So that's their example. So we call cities geonames, whatever boats. Because we don't know what it does and because it is our goal in life to always know other things do. We go over here and we say, OK, let us run this thing and see what it gives us. Right and see why we might pull this information in. See if it does pose information now. Maybe teleport is busy today? This is the other thing will happen. You will go out to APIs and they will sit there and your system will sit and sit and sit. You will have a timeout. One thing the rest template does have is a time out. This will have a timeout but it will be a very very long time out. Your timeout will usually be you know a second or two. And you can actually change it, but you don't. OK? So it finally came up with that and said. The status code is wow, that's a 500 internal server error. OK, this is back to your thing. What does 500 mean? Anything in the five hundreds? Means that it's an error on the server side. You didn't do anything wrong. A server screwed up. Most of the time it's just going to be 505 hundred is just like, I don't know if something add went wrong. My service is down, something is gone. All right, so when you get a response, code is in the five hundreds. There's nothing you did that can make it work better. It might work later, it might not, but it's not your fault. It's not something you're doing when you get something in the four hundreds, it means it's your problem. You're trying to get access to something you shouldn't or you know you you don't have the authorization of end ication you need. The classic one that you may actually be familiar with is 404404 says. I can't find the resource. Right. And it is just the classic thing, and they usually have a special page that they'll return. Now that's a point here. Notice that I did get something here. Alright, even though I got an error, I still got content. I have a header which tells me what is my kind of content. It's application Jason and it puts the characters right, so when you get something. Then you. Can you use it? It could have been HTML which wasn't when you go out to a website and they have these clever, you know you're looking for the wrong thing. And if it's, you know Starbucks, then it's got somebody drinking coffee while you look for the wrong thing or whatever that is, right. They're all returning a 404, which says you can't find it. They're just also returning an HTML page that they say if you want to, you can display the page for people and then it'll amuse them or something. That or it could in this case it could say. Here's an error and and we might know a little bit. More than just the 500, we don't know exactly who we know a little bit more, something is wrong and in this case we can't find this Geo name ID. Why can't we find this to your name ID? Now, unfortunately, I'm let me let me try this 'cause. This is the one thing is that some of them absolutely require a slash again. Now you still can't do it. Error connecting to this proxy. It cannot assign requested address blah blah blah whatever. OK, not a problem. But if you wanted to do a query. I wish I had another example to give you, so I will leave this in here and it may not work, but just so you know, this said it might be that I have a way to get. The information in the same form from a different URL I might have all my hotels that are Ohio use one API. And all my hotels that are in Pennsylvania use a different API, but I want to make an array of all the hotels in both. I could then do this. Hey, Ben. Yeah. A stupid question. Is it supposed to still be the rest template that get for object? It is. I just didn't type it. OK, OK, now that was. I was like, why are we settling changing it? No, I just copy and paste that forgot to copy paste. There we go. Yeah. Um, but notice it still did use my custom class. So the the big message to get out of all this is that you should be. Um. For the big Mr B, then the big message here is that you. Are creating a model. That is not necessarily as tightly controlled by the database as an adieo. Web service API's are you are often dealing with somebody elses service. Like, we're going to deal with our own little service, we actually know exactly what it does. You're going to be dealing in in real web applications, sometimes with half a dozen dozen different services given to you by half a dozen different vendors and a good example that you'll probably see in it. You know, I don't know if you don't see anywhere else, somebody will put it there, caps down, but it's all. It's fun. It's something that will go out. Then we'll use the Google Maps API. Because you can do so much with it if you go to all these websites that will tell you you know where their hotel, you know where their hotel is and they'll quote, do their own thing. They they are using the Google Maps API to show it and then if you actually try to dive in. They actually throw you out into Google Maps so that you know you can get more information. So that is why on our website you will see something that looks not like very much like the Google Maps that says where they are. But if you click on it, it goes into Google Maps, it shows that they're using the API to pulse like some information, use it, you will be doing that a lot. Because of that, you don't control, you kind of control what you have locally. And you do some sort of map to the external thing. It can be very distant, can be, you know, your goal is just to to map it somehow. I'm going to run this app briefly. Theoretically have done are different things. So here's our app alright. If we want to list the hotels it says it's not implemented. OK, what did I not do? Once again, I thought the app was all implemented so I didn't have to do that perfect. Give me one second while I figure out what they think we're doing here. Oh, they just want you to say they're not doing it until until we've actually done it. So I will show you what this is. Every one of these things, this you're going to do the you're going to use your console application, your console service. And it's got like print, hotel or whatever. Alright, so all we're going to do for the app is we're going to say. If menu selection is one we should do. Council service show you all them, but console service dot list, hotels or rooms probably print hotels. Hotels. Try and then what are we going to put in there? We're going to put in about thing that we just did. So we're going to do the hotel service.

Alright. Alright, that is going to. Then do it now. I will rerun this. Exit. I'm going to rerun my. Exit go away. To re run my app 'cause I just changed something so I'm gonna run it again. I mean exactly one of these. And then because standard class I would just go ahead and put the code in the lecture code that actually implements it. There's nothing to it. All you're doing is calling the things we just did using the console service, which we didn't, right. So like did not, but just so you can see what we're doing here is we do our want. Now actually one thing I can show you here. Remember we've got this thing running all the time. It's it's listing every one of these things we do. So last thing I did was this get hotels? You know stars equals 4 whatever. So when we run our application. However, we are we run our application and we list our hotels which hopefully will work this time it's going to list the hotels. But if you go and look at your server, you're going to see, it's gonna say get hotels. So you can follow the logic of what it's doing if you're doing the right things. It's going to call things correctly if you do the wrong things. It's gonna also record what they are. So when I say get, hotel, star, whatever notice it just gave me a 200 because it was OK with that. But whether it gives an error or not, it's going to tell you what it is that it actually called. So when you. Get to the point where you're writing homework or you're doing homework. You're actually trying to create this stuff. You're doing it, you're going to be able to go back and and do that level of debugging just by saying, what does it think I call? And then you realize, OK, it passed in the parameter of stars, and I never actually gave it the stars. So nothing happened. Right? So that's very common. That sort of like, make those mistakes. This is going to tell you what's going on. When you are finally done, this is just a warning because this is running in the background. If I close this window without stopping this. It's going to still be running in the background. You're gonna have trouble the next time you try to run. It's. You just do control C. Control C will stop it. You wanna start it again? You do NPM start again. OK, this is it. That's all you need to do. In the event that you close the window, how would you close it? Now, if you were to claim you were to closeout. No, I mean there's, there's, there's an easy way to do it. And I what I will do, I'll post in the the Java Channel, not that you normally need to do it, but I'll just post the the. Commands you use that go out and get the process and kill the thing you know with a process like, you know, always get to a process that's running on windows or or anything else. It's just not as straightforward. But if you're well, I will tell you what you do, OK? Like, let me say that if I this 'cause this, this is also quite common. You guys all run with like, 15 million windows. So what can be also very common is if I just go ahead and where is it? Which one is it? The terminal. If I went back into my thing. This this is the one thing you'll see that it won't make a lot of sense to you. Slash 11. Where am I? Ranger, right? Rush. Make sure. So let's say you went in again and you already had it running and you do it again. OK. It's gonna give this kind of thing. It's going to say that address is already in use. This is when you would have to do the thing that I'm going to put in the in the Java channel. You would have to do it when you get to this point you can't figure out what's going on. You may still have it open in another window, or you may have closed the window, but this is what you would see. Right. Alright. Anybody have any questions based on like all of the stuff 'cause, there's a lot going on here, but. Do you have any questions? They're all totally ready to to do this right? It will. It is honestly pretty straightforward at this level. It'll get a little more complicated, but just remember that all you're really doing with these calls is creating that URL. And. It is. I absolutely recommend that you keep caution open. You keep this thing open, you go back and forth. You type in the URL's, get used to postman postman. There's not a lot to post in that for what you're doing or remember the the little plastic will start a new thing. A new request. All you're doing is get, so you just put in the URL you want and then see what it comes back. It'll be an error or the wrong stuff for the right stuff. Now those are your choices, so keep this one open so that you can figure out what you're doing wrong on the other side, and then see why you're getting it. And remember that in here it's going to tell you what you've done, whether it was postman or the other. And remember, that's that's important. You've got a server out there. You've got two different things interacting with it. They're both going the same place. So if you're, if you're posting is working, but your job isn't, then look at what's different and what it's putting here. Plus there's probably something different. Frank. Alright, I am flying back tonight so I should be back in my regular, you know, place tomorrow. I will be around. Probably at least until about three years. Something if people have questions or you like I, it might not be a bad idea to open up the homework even if you're not going to do it all tonight and just figure out like, how to even get started. One of the things that that may not be obvious is that that MPM install is going to happen every time you have a different folder. So. With your exercise, you will need to go down into the server part and type your MPM install before you do your start MPM start. OK that is normal. It will not hurt by the way to ever do MPM install more than once. You're not going to NPM install. Can I go out? It's gonna look like he's doing stuff, and it's not, you know, might have even found something new by in the short time I've been doing that. So it doesn't have a did. But I love that it tells you how many packages are looking for funding 'cause. There are people out there like cool kids who pay us to do what the stuff we're doing, you know. So anyway, if you want to pay. Or go for it, alright. Bye all. Thank you. Safe travels. Safe flight. thank