More upon a raise. Yeah, I I don't. I'm kind of at almost a complete loss here. It's OK. Alright so. Break it down into simpler questions, right? That's that's usually when you're stuck is you can break it down. How are you going to get the last element of the second array? Just what is the last element of an array? Do you remember that? Uhm, vaguely. OK, So what you might put in a comment down below all this is. Something like. Get the last element of the second array. The last element of the second ray, right. And then you can sit and you can say, OK, so now I've got the last element. What do you generally need to do? You've got an array and you're gonna loop through it somehow. What are you gonna need on the first ray? Now that you've got that last element? You're gonna need to do something with the first two, right. So what are you gonna do? And again, you don't have to write the code, just hit enter and say and write a comment that says something like loop through the first array looking for that element. OK. So right, it's the last time. All the second ray that element I'm going to loop. Through the 1st. Array first, loop through the first array. Two to find the first element. Well to find the use the online 19 you got the last film. Now you're trying to find the matching element in the first array. Find the matching element in the first array. And so your what you're trying to do is even if you can't come up with the code. You're trying to come up with a thinking. Right. So and then. What happens when you find a matching element? Um. Like given that we do, we need to keep going once we find a matching up. Oh no. OK, so you're probably gonna return it, right? Right. Yeah. So if? Found then return the matching elements. Right. OK. Something like that gives you somewhere to start. Now I want to try to write just the method signature, so you're just gonna, right? Again, you're going to, you know, hit enter on the end of that line or whatever. Click on

Whatever you want to do, just try to write the method signature you would need. It's always going to start with the public. Right. We let me use notes during this. Or is this simply during an interview and actually I mean the reason I'm talking you through role of you. I want all of you to pay attention. This is these are actually the what you're going to face if I answer is don't use notes. Don't get, don't get tempted. It's it's hard on zoom to say. I've got all this right here. Right. And it's it's actually even slightly like when you're in person and you're on a whiteboard, you obviously can't use this. But there's another thing which you should be careful about, which is. And one of the things I will give you feedback on, but when he's not now during lecture 'cause in this situation when we get to a technical interview for instance, one of the things I mean, I'm going to technical mark interviews, right, one of the. Things I'm going to be watching for to tell you about. Is to be careful of things that make it look like you're looking up things. Even if you're not. Alright, I had someone who was interviewing balast cohort and. Their head was down low and it was always looked like they were looking back and forth and the person was actually just taking notes for himself. It just made it look like he was trying to Google for answers the whole time, right? And so we raised him up a little bit and got the reset, the camera, do whatever else, and then it was clear he wasn't doing that. Another thing which which be aware of, alright, if you wear glasses. Glasses reflect whatever is on your screen. They may not reflect what's at like. Somebody may not be able to read the text that is on your screen. But if it looks like pages are whipping by in your glasses, they're going to think that you're looking for something again. Don't worry about it. Don't try to look it up. Don't try to find an answer. Don't try to whatever else. Just say. Alright. I'm. I'm in the position I'm in. I can only do the best that I can do. That's all I'm going to worry about right now and and don't. Don't get caught up in being right. It caught up in being. Like in in just focus on the thing you're trying to do. OK, so cat. Yes, you're gonna write it, but what is it return? It's gonna return a. Indent. OK, so you write, so go ahead. OK, so it's gonna return. OK. And how is that going to show up in your in your method signature, OK. Uh. Public. Right array, we call it my array 1 Norway hold on. Is it going to return an array or is it returning? Oh, right, right. Actually, the answer is a leading question, because the answer is you're not quite sure 'cause you haven't asked. Oh, I see. OK, So what I want you to do is I want you to return the offset into that. First array, So what you're going to return is just integer as an offset. Is an integer, so you're going to return just an integer. OK, so in your in your signature. she Well, but that's going to be an image. So you're gonna return, Justin and not injure Arabian image. So your method signature, your return value using public and you're going to have to return value is just it. OK. Then you're gonna call it platter. You can either if you can. If you can't come up with something, the next thing should be the method name. If you can't come up with a good method name, call it my method like. I love you. We all be able to come up with good method names on the fly, but if your brain won't function. And they haven't given you a name. About on the spot method. Alright. And then what happens? What's next on your method signature? Calling the the array. Well, it's the parameters. So how do you make a parameter string? How do you pass in arguments? We've got to this is an array, so we're going to. Put in values. The array. So right now remove. We're trying to get the method itself, so you need that you have a method and a method is going to have arguments. By the way, the way I. If you're not on the spot, if you're not the one who's being put on the spot right now. You may think, oh, I could totally do this. And and be aware that you may be totally couldn't do this when everyone was watching or winter viewers sitting there, whatever else. Part of what I would recommend is practice, particularly practice the methods and nature. It's just sort of a the lowest bar that you need to pass. If you wrote nothing but logic after like you know, logical comments after this and you wrote your method signature pretty easily. You would be OK would be great for me. OK. But you should be able to get your message down. OK. So, yes, right here. You need two interets. So how do you define an int array? Oh, wait a minute and array. And what's it called? You said it was my array one or something. Like my array 1. And then my second one. Will be hopes. Mira 2. Something like that. That will do. Why rewind? I have some parameters right about use. You're not setting the values of my array. One already gave you the somebody passed it in. You're given to a race, so you don't have to worry about setting the values of my array one you just want to use it. Oh, I see. OK. So look at your comments from 19 to 21 because that will be the thing. Those are the reason you put those comments in. And again, this is really hard to remember when you're under the under pressure. So the reason you put the comments in is to remind you of the steps that you need to do. So just what's the first one? Get the last element of the second array. How do you do that? OK. So would I use? And if? Statement. Um if? Why is this where I call the actual array? Well, what do you what do you? You're going to get it. What are you going to put it into? What are you gonna do with that last element? I'm going to attempt to get the last element. Of the second array, right? What you probably want to do if you're getting something is you probably want to put it into a local variable. To what it probably create a local variable that's going to be stored. You don't have to, but that's that's one possibility, OK. So, but first of all, just what is the like? How do you what is the last element of an Iraq? I've got my erase. Cheat. That's what I'm trying to get. How do I how do I reference the last element in my array 2? Trying to think about that, um. Something like. Yeah, I can't remember exactly. I can see it, but I don't know how to explain it, but it's OK. I tell you what, we will need to move on. You've done fine. You're writing the comments down. This is why practice helps a lot. It's not that you don't know what to do, it's that your brain is not working under pressure. The reason we practice whiteboarding the reason to get into sessions with friends and to do it practice in front of other people, and whatever else is to be put on the spot. So that you can do what your brain certainly knows how to do when it is not under that pressure. It's the same reason why. Frankly, why, you know, people who are going to the Olympics practice a billion times before they do it is because when they're under the pressure of the Olympics, everything freezes and they have to, like, go on muscle memory. If you practice in front of people, you'll be able to go on muscle memory. You'll eat some of this will will be easier. Right. You define this is but this is. This is the message I'm trying to give you all is. This is what it will feel like. If you don't have a little more practice and I just say something, Ben. Sure, Kevin, we meet Patty and Kate. Coke does this with us every day, starting at three o'clock where he has us doing these technical. Problems if you wanna if you wanna join in a group. OK yeah, I'll definitely. OK. Alright. Well thank you Kevin for being there. Today is sacrificial. Yeah. Be happy that you won't have to do it again tomorrow, right? Alright. Speaking with, by the way tomorrow. Let me just briefly talk about scheduling here because. The scheduling here gets a little odd. Once again, I will first have to find you. OK, let us. Next week. well tomorrow is but this is your about to have a capstone after that. So this is your chance to sit down and say. What if all these pieces, maybe there's DAO stuff, didn't really get in as much as you thought, or maybe OK, you know how to select stuff, but you really don't know how to insert or update things you know and SQL or any of this stuff is fair game. If it would be easiest for you too. To work through a complete you know problem, we can do that. Think about what one of the days, whatever I think about what it is that you would need to help you. Get ready for a capsule which you know is coming and you know it's gonna involve all the things, right? All the stuff we've done up until now. The PA. Oh, the MVC pattern. The, you know, the different kinds of annotations. All this stuff is going to come together. You're going to be out on your own doing that with. You know, a couple of other people. Right. So think about what it is that you would need. On those days. OK. Alright. Yes. And I I mean I I really do wanna say I'm actually. Not the least bit surprised that it can be a real struggle going through a White Horse. This happened. It always happens. Some people will fall into it more naturally. Some people will have a harder time. It's not really how much you know, it's how much you freeze. Right. Part of the reason we're doing a little bit of practice is not just being used to, it's to find out for yourself how. You do it. Because if you find yourself fairly comfortable, then there may be other things you want to concentrate on more. If you find yourself like struggling with that, you may want to practice it more, just like Kida says. I think was key to I can't remember who said about practicing with Cole. If practicing every day, it's gonna be what's gonna help you get through this. Then practice it every day. Right? But it's partly self knowledge. It's knowing what you're going to struggle with the most. On that yesterday when I was going through it, freezing is the hardest thing for sure and I had to. I didn't talk too much when I was a. During the coding. But if you don't remember how to code something, comment what you need to do is just keep moving. 'cause freezing is like the worst thing, at least when I was doing it. Just I need to make a loop if I don't remember how to write a loop, say make loop that does this and then just keep coding anything you can go back to it later. Absolutely. If you can't, freezing is bad 'cause. I froze up really bad yesterday. With cold. I mean, I was really like Stone faced and coat was trying to get me to talk through, and then I was just like. Clearly, and that'll happen. But but be aware that technical interview is probably 25 minutes long. And what's going to happen is if you freeze for too long. They're just gonna move on and nothing. Nothing failed or anything else. I mean, they remember their goal is not. To know that you know everything, their goal is to know which things you are comfortable with and which things you are able to do. But it obviously helps if you can get some of it down. So if you can't describe it in Java. Describe it in English. You know, and even describing it in English may be hard. Try to try to write down the stuff you know you do know. OK, if they say return it, same thing. I keep telling you guys 'cause. It's really important to keep in your head. They say returning integer array. Create the integer array and return the integer array. You can write those two things down even if you don't know what you're doing with the array. But if they say return this thing you know you can create a variable that does that. That is that thing and return it. And that's two comments you already have down in your code, so you know it's it's. It's partly just a practice and trying to figure out which you things you can do. Not wishing you can't do the reason we freeze. It's because we focus on what we can't do. Also sorry my camera off my my son is sleeping behind me and so he didn't want to wake up this morning and want to sleep on my bed. One thing I've been telling people to is if you're freezing up on stuff, ask clarifying questions like, even if it's stuff that doesn't really matter to how you solve the problem, but you know if somebody says taking a rate or take a make a method that accepts 2 numbers. They don't specify if their whole numbers or decimals, or if they're going to be big numbers or small numbers. You know, while you're thinking about how to solve the problem, you can ask ask these things to get more clarification which will help you solve it more properly, but also buy you some time to think about how you're going to, how you're going to figure it out. Yeah, excellent. Excellent. Then ask questions about what they're telling you. It will help you clarify and and it will help. Slow things down and it's putting the interviewer back in the position of having to come up with something as post to you coming up with something. So it's, you know, that's excellent advice. Alright. Today's topic is, well, actually it's a little it's I hate these part tubes. OK it is. Is more of the same but different. OK, there, there you go. Let me go ahead and share my. Screen and start the recording so I don't forget. so a couple of things that I wanted to to bring well. One of the things in being a software developer that you will all face. Is that you are going to work with teams of people you're gonna work with pairs. You're gonna work with groups. You're gonna work with whatever. And one of the things you will all face at some level is some form of code review. And. I I raised this because it's sort of a meta issue in that you will hear sometimes and you'll hear a couple of times in this. Black shirt where I am going to. Critique the way the curriculum does something. For example. I am not criticizing the curriculum team, which is excellent. It's an excellent team. They work really hard. They're constantly trying to get the right balance between how we, you know, force feed you all and incredible amount of information in a short time and also not. Not say things that are wrong or that are, you know, poorly done or whatever else it is a constant process of this. None of it's table, none of it. One thing I want to show you today. They well. In fact, I'll show you this right now because it's a good example of what I'm talking about. Well, we're we're gonna look at these custom exceptions today. Right. And one of the first things you'll see in this custom exception is private static final long serial version UID equals 1 L. Right as one of the instructors asked. Why are we doing that? We don't ever use that thing. Why do we have it? And. The curriculum team isn't doing anything wrong or curriculum team in this case is trying to follow a best practice which is no longer relevant. And that best practice is it you that it used to be that all exceptions? Well, it's still true that all of Java exceptions must implement at some level they must implement something called serializable. And it is a way for the system to keep track of which version of the API you're using, which version of the files you used, and serializable requires that you have a serial version UID that gets updated every time you change this this thing. Right, so this is the second part. One, there's the I'm not criticizing the code if I say let's get rid of this thing. It is not needed anymore, but the second is. You will be working on code that was not created yesterday. You're only working on code that was created over a period of years at different times by different groups of people doing different things with different requirements. So what that means is also when you do a code review. Sometimes you won't understand the reasoning for something. And. And maybe that means The thing is no longer needed, and maybe it means you just don't know what the reason is that is needed. The temptation can be to always leave all the the crud from all the years of things that were required, or the other temptation is to get rid of it. 'cause you don't know what it means, and then all of a sudden somebody elses stuff breaks right? It's always a balancing act. This is this is software development, it's. Balancing act between. We have to worry about this saying anymore. I don't know, you know? And So what you will see is certain amount is where when I bring things up I'm trying to get you used to the process that we we you should always look at things and say do we really need that. And you should also be accepting if someone says, yeah, I think we better leave it in 'cause. You're supposed to do it. 'cause we implement serializable, reality is now that we're using rest APIs and all this, we don't actually need it anymore. Fine, but that's solely because we're using rest APIs with our with our these exceptions. If we were using something else, we might still need it. It's, you know, it's good. It's considered good practice to have it fine. Leave it there. You'll see it a lot. Don't worry about it, OK? So having said that. I mostly want to give you the framework that when I. Bring things up. I am. I am trying to model the. Question. But also accept. Having said that, I mentioned it very briefly and I wanted to make sure that I. That it is documented, even if well. There are couple of things I want to mention just because I want them down in your class notes and you don't have to worry about the Great deal. But you should know that they exist. One is you're going to hear this term beings or Java beans in your reading last night 'cause we love that and you know this this happens where where something gets left in that is related to. A term that we never, never, definitely defined. When you turn, when you talk about beads or Java beads, right, that is just a fancy word for what we now refer to as the mom. It's the. plain ordinary java you know Plain ordinary Java. You know, objects that values the properties. And the getters and setters to be and maybe some additional rules, but have that kind of stuff. It's the model in what we're talking about, but you will see that you will definitely see the term being your Java. And they are because beans make coffee or what? Make up the coffee. That's why they came up with the term and everyone likes the term. But that's when you see being validation. That is not some, you know weird thing that just means validation of those those model objects. OK. I spent so long last night trying to figure out why they were called beans, couldn't get it. I thought. I mean, I I did reading myself and I thought no one's gonna know what that being valid I was like. I was like, are they being shaped? Are they like, are they shaped like beans? I don't get it. Why are they called beans? And now it's like, oh, Java beans. Yeah. Yep. So these are. Like these are the add notations the No, it's blank pattern, no, these are simply. I'm sorry, I'm going to go back to my. OK, when I look at the model and I have address. This whole class. Is considered a Java bean. Oh, it is just it is a thing that models the object the the data we're dealing. It was not just any Java class, but it's a class that is all about. Just encapsulate Ng one piece of information you know one piece of of your set of information. It's got its variables it must buy. Buy one of these weird rules they have. It must have a constructor with no arguments. And then it usually has a constructor that has all the arts, which I mean like all the things that need to get set. And then it will have getters and setters. It could have much more, but that is what makes it a job mean. It's following those rules. It's God, it's private variables. It's got a constructor with no arguments and it has getters and setters for all the things in for all the elements. Once we define the idea that there is this Java being, we can use it in a whole bunch of different ways. We know it's always going to have those attributes about it. So we use it for serialization and deserialization rest. That's the most obvious thing you're going to see, but it can be used in a number of different contexts in Java, and so that's the term. For our purposes, it is. It is one of these classes that is in the model that just I'm going to hold the basic information. About what kind of thing I'm dealing with. OK. The other thing which I wanted to put in here and I'm going to spend even less time on, is hideous, which looks like hate. Away. Away. Yes, I hate it. Yeah. I hate the acronym. This is about rest. This is one of the features about a rest API. Which? We do not use. In this course, because it just makes things look more complicated, right? But which is a thing? It is part of my personal obsession, because I've actually written commercial rest APIs that a rest API should always be. Uh. Something you can only understand if you read the documentation. The entire idea behind rest APIs includes this as one of its different constraints, which is that I should be able to go to the first end point. And find out every resource is available and if I go into that resource I should be able to find out what is available from there. Because they contain in them the self referential links and the example that I mentioned in class before, but I just didn't go into very much was what do you do when? Like you've got into a list of hotels. How do you then go to an individual hotel? What should happen and what does not happen? But what should happen is that each of the listings of the hotels should have a a. we refer to as in a trap that's actually underscore links and underscore self you know or coleman self thing it's this methodology i'll show you at some point but not now which which how do i file and once I'm at the individual resource if it allows you to get to something like when I went to H3. I should have a link which tells me how do I get to the reservations for H3. I don't have to know that that's there just by magic or by documentation. I should be able to see that there's a document that part of the object that gets passed is. What else can I get to from here? Alright, so the sort of a self referential API, that's when I refer to hideous or hate OAS or whatever you wanna call it. I'm going to leave the document. The definition here. I made an add-on example of it, but that's what I'm referring. We don't use it. It's just too complicated for what we. You know to implement spring boot and whatever else, it's not worth it for for your purposes, but you should know that it's true. Last, my little you know, rain haven't left feedback yet. Leave feedback. Let's take a look at. Let's take a look. The topic for today, as I said, is just more of the same. But. No skipping, but ever. Oh my gosh, all the things. The most important thing I think we're going to cover today, even though it's not all that we're going to cover is validation. We're going to handle what they call being validation. I've never actually heard the term before, but we're going to cover it validation and it's it's how do we bundle up all of the serialization and deserialization and making sure that everything is valid and returning the right error codes to. The thing calling to the to the client. How do we bundle all that up in the in a simpler way possible OK. I will tell you if you looked at the tutorial. That the code that they have in here and then we're going to talk about today that I have in one of my commercial rest API's which was built long before all of this stuff was available and I wasn't in Java either. Takes hundreds of lines to do what it is that we do in like a line. Here and you know, it's just it's. It makes my my soul hurt. How much work I put into that? It now can be done so easily. But you know that's that's progress. You like that progress. You should never have to do all the work that I did. So let's take a look at let's go through the. Almost the the not too challenging part first. Just to fill it out, remember that we did that create and we did. We get in whatever we did that yesterday. So the what we're supposed to do today? Not hard to do from what we've already done. I'm sorry. Here's our create. Hi, it's gotten a little bit more. I think we. That's the create reservation. We've we've added some more to the model here. There's a little bit more in here than was in here before, but I think what we're supposed to update is OK, but how do we update something? We now know how to. That was our create. This is how we create a new reservation. But how do we update? Our reservation. Just to sort of talk about the. How similar it is and what the slight differences are. What I also want you to see though is we do this fancy thing and this has become very important. We now throw custom exceptions. Custom exception is that we are creating. Are things that wrap up both irrelevant message to the user 'cause remember the users on another computer in posts and or in their web client, or whatever else? How do we think back to when you were doing your vending machine and you had an error? How do you deal with the error? If you listen to me, then you return that error as a string. And then something logged it or printed it out or wrote to the console or whatever they did right? In this world, in the Web API world. There are two critical things that you need. One is the message. And the other is the status code. Right. We could take a minute. I wanna show you. And remind you about status codes. OK, we've all talked about them and let's let's just go to what is the status like the normal ones that we will see if something is successful are going to be in the 200 range 200, two, 99. Don't worry about the three hundreds for right now. 'cause, it's not. You're not going to run into much. The four hundreds mean I did something wrong. Meaning the client gave incorrect information or did something that was not valid. And the five hundreds mean. That the server is having a problem of its own. It's probably something the client can't fix. So when we throw an error here like hotel not found. That is because the client. Has tried to give it a hotel that does not exist. That is a client error, so we would expect to see something in the 400 range. In fact, what we're going to see is is just 400 and and an error. So when when we create the. Hotel not found exception. Right. We're going to do the. Well, actually it's probably a 404 for this one, but we're gonna do the. We're gonna set the status code. That we want to return. And then we're gonna give it a reason. And that reason? Is gonna be. I mean, this is a little weird the way they do it 'cause they're actually using the reason both in here and in here. Sometimes this is the the shorthand version that you should send to the website and this will also show up as as part of your exception but. It doesn't really matter. You need to have the. The reason up here. That's all I care about. What's done down here is so that we can implement an exception. I don't think it really matters. But this is really important. You want to make sure that the error you're giving. Is the appropriate kind of 400 error. For the problem at hand. What if the problem was not that the hotel was not found? But the hotel is full. I can't make a reservation there because there are no more. Rooms at the end. Alright, so that is not the same error. Now I will show you there's a handy little website. That in a somewhat entertaining way. We'll show you the way to remember this. I think Elliot has probably seen this website. OK, it's called HVP cats. And it's at HTTP dot cat. That's actually the URL, right? And it will show you. Something like this, which is essentially each of the error codes or the success codes. With a cat themed picture to help you know what those things are. Right. So if I go down to my four hundreds. Is an easy one to find to go play with you if you want and you start looking, 400 means that there's something wrong with the fundamental request that I'm sending to you. It's, it's just structured wrong, right? 403 forbidden is a is a frequent one you'll see because it means that you tried to get there, but you weren't allowed to do it, OK. And not found is probably the most common one on the Internet, which is like OK I I can't find this resource. So in our case because it does not found this is a big clue and it says it's this unauthorized or payment required. Whatever. These are not just text. They look like just text, but they're actually part of the message in the sense that if I go out here and I look at my code. If you're looking for HTTP status dot. It's going to be in This thing. It's gonna be HTTP status dot unauthorized, all in caps are not found or forbidden or not acceptable or whatever. Hi. So you were trying to make the the error messages match and don't get fancy, OK? Some of you may have noticed I'm going to say. Four 18418 is is one of those popular weird Internet things. It means I'm a teapot, right? And it was written way back in the day to basically say. I'm not the kind of thing that can do the kind of thing that you think I should do. I'm a teapot. Alright, I'm in a in an Iranic twist. Hackers. Recently managed to change the Russian military, the main Russian military site. Which is like mil.ru. They they changed it. So we always return a 418 year. I'm a teapot. hi it's just one of those things OK don't ever return i'm a teapot except i OK He only returned 420. And Hans, your call. When knows where that came from. But what you mostly you're gonna return are going to be these first few. The the bad request. 404 for not found. Maybe 403 forbidden and possibly. Yeah, sometimes this one right here is a crazy picture, but 405, which means that a thing that I'm trying to execute is not valid on the on the thing I'm trying to validate executed on. Alright. In any case. Enough for that. Let's go back to our. So. Stuff up. Alright then next thing, we're gonna go through and we've got another one. By the way, reservation not found is very, very similar. OK. It's still gonna say that. Alright, let us take a break until 10:05 and then we're going to dive into actually modifying the controller and and doing some validation.

Recording in progress. Alright, so our next topic. Add issue things go hand in hand. As a reminder, and I, I feel like the the reading is in in a slightly odd order. We already talked about the whole that crud and create retrieve II tend to say read instead of retrieve, that's what it used to be and at some point people realize that reading just felt too much like it was a a. User doing it as opposed to a an application so they change to retreat, so it's create, retrieve, update and delete are the different kinds of things that we do with an API. And. The two that we're going to. At least show today our incredibly simple code, so like the by themselves, they're just. How do I do an update? How do I do it delete? The only real point of that? I mean it's like implementing them in class is a about the validation and error handling. So let's let's tackle them. But realizing that there's so little to tackle that. As I said yesterday, don't bother to type this in. Just watch. If you wanna do it later, it'll be in lecture final. It's just there. It's mostly worth paying attention to. What's going on? OK, so if we go back, if we go into our HH controller. And we go to create a new reservation. We're going to. Update a reservation. I could take it all out, but really I'm just it's easier for me to go ahead and say create. Get a reservation, create a reservation. We update our reservation. Are the reason I want to do this is because I want to emphasize the differences between the create and and update. And it's easier when I'm making the changes than if I'm typing it from scratch. So that's like. It's short enough that I could certainly write it by hand easily. OK, so I'm going to update this. I'm going to have another parameter. Which is the ID for the reservation. And then I'm going to add something in here which. This is a new annotation for you guys for Java docs, so it's not a a Java annotation, but it's a new symbol or whatever for Java jocks, which is to say, what is a kind of error that it throws. What is the error that it throws when something goes wrong? And we're going to have a reservation not found exception. That again, remembering that the point of these is both when you print out document, you go look at the documentation for API or when you were just mousing over the you know the. That thing you might call this gives you the information that you need. OK, so now. What should be the response status for an update? Enchiladas. I'm not sure we even need. It should just be a normal 200. Normal 200 is default. So we're just going to delete that. We don't really need. So that's again one difference #1. If I'm creating one, I need to explicitly say it's going to return the 2201 when it works. I don't need to do that because it's going to be a normal success, which is 200. Right. Now. When I am updating a reservation. I'm actually gonna do it inside reservations, not inside. OK. So, but what I'm going to do is I'm going to put the ID as part of the path. OK. And the request is not going to be a post, it's going to be output. So in other words, I'm going to go directly to when I was creating, I want to know where I'm creating. But when but I don't need to know the hotel at this point. Do I need a hotel at all? I don't really need the hotel at all, do I? So let's get rid of the hotel ID from here. OK. I don't deal with just the reservations as individual resources. I'm gonna do a quick method on it. I'm gonna say let's update reservation. Try. I'm still gonna have a reservation path passed in here. And I'm still gonna pass and ID, but now the ID instead of the hotel ID is going to be just an ID. And I'm gonna have a throws reservation not found. Because that's the thing that is variable in this III. I'm looking for a resource. I'm gonna emphasize this idea. Rest is all about dealing with resources, so the resource is a reservation and the ID is the ID. So when I can't find that resource, what I'm going to throw is a reservation not found, not a hotel that found when I was creating. The hotel was the idea was the thing that I was trying. The resource I was trying to find was a specific hotel. Right. So that's why I threw hotel not found here. It's 'cause. That's the thing I'm trying to find. And what I'm trying to find one down here which is just the reservation. That's the thing I'm trying to find, so that I'm going to throw a reservation not found. So. We now have. I'm sorry, we update reservation. We have the reservation, we have the path ID, we have it throws reservation not found. What do we actually do in here? Right. We're going to do a reservation day. Oh, we go back. We say. I don't know for sure what these are. So I go back to my reservation. Yale. Look for what is update. Update is right here. Now notice, by the way 'cause this is not. This is a thing that people frequently miss. Is you should have the throws in here in your interface. As well as our in your implementation. But I can't see you. I'm sorry. Looking at two but I got three screens up there, cheap bouncing. I'll go back. OK, sorry about that. So when we go into our. DL reservation DL thank you for pointing that out. So here. Thank you. I do update right here. I should specify the parameters or whatever else I should specify the throws reservation italics here. So that tells me anytime I implement this in any class that implements this, it should be able to throw that. So I know I am calling it. Oh yes, that's right. I was looking at the. It should take a reservation taken ID, so I'm going to go into here and I'm going to say I'm going to update. Take a reservation and it's gonna take an ID. Alright. So the error this is that the whole quote, I mean like there's not a lot to us, right, that's the whole code but. Two things I want to mention on here related to it. First one is about just the javadocs. Some of you even commented on this. I could put this here. But the other choice that I could have and you and in our in our code that we give to you guys, we sometimes do it one way and sometimes do it the other way. But I could say, why put it here? Why not put it in here? Right. And then everyone who does it will have the information because it's going to. This part is going to be standard. And I don't muddy up. The place is when I call things out here like put it all right, just for what it's worth. Now we put a bug. Alright. This works, but now this assumes that the only error I could throw. Is that the reservation ID that that ID could not find a reservation? But there are more rules on reservations. It turns out I can't make a reservation. A highly objective list, but I cannot make a reservation for more than five people. Beyoncé is never staying at this hotel. Her entire entourage is never going to be that small, but. Be that as it may. If I go and I want to put in that limitation, I now want to say. But wait, I'm gonna create a reservation or I'm going to. Actually I can do it for either one. I'm gonna create a reservation. And I want to create that reservation for. With some limitations. Right now what I'm gonna do is I'm gonna put the the the rules, the rules of outer reservation. I'm going to put into the model. OK so in here. If my rules are here. That I am going to go ahead and make make the rules right and I will show you it was in the reading, but I'll show you what we're going to put in here at some of our rules. Let's put one of them in just so you get the idea, and then we'll go back to the other code. So let's say the one we want here is the number of guests. Now we're gonna say. I'm gonna just say there should never be fewer than one guest. Right. And I want that to raise an error. But I already it's not a reservation not found there. So if I do this and I say OK, I want admin value equals one, so the minimum number is value is one and then I have a message and this is the message that happens. If it doesn't work. So just think about what we try to make it a positive message as opposed to you screwed up. It should be in a positive way as. Match. You know what the rule is? A minimum is this not. You can't do that, or that's too little or whatever, just that's a general that's less of a. It's a good conduct way of doing it where they pee eyes. Other error messages throughout your system that you have done have often been in the negative. You can't do that. But when we do a Web API, weird, we are inviting guests into our home. So we're gonna be more polite to them than we might be to our kids. Alright, so with our kid, we might say don't throw your, you know, coke on the floor. And with a guest who comes in, we are going to say something more like. Coats go in the coat closet. We tried to do with our kids, but you know, we hate crappy. Alright, so we have we'll say like the minimum number of guests is 1. And sort of a positive, upbeat wave demand. And let's say and this is where they are making the rules that I object to but. I would do what I am supposed to do. Spy message. It close actually. Number Oh yes. Is fine. So we put these in. And as one student notice from doing the tutorial last night. As is, this won't do anything. Like, it's great to put these in, but if I ran my system and I put in 22 guests. It's nothing's gonna happen. That's because putting the validation in the model is only the first step. And this is where I'm at and I'm trying to step back and make you understand that this is. We teach you one set of code that works. But often the things we're teaching you are valid in more than one place. These validation rules right now I'm going to show you how we would trigger their being used in a rest API. But there are other ways that spring boot can can handle them in other scenarios where you're not doing Web API. So we just set up the rules here and a message. And then in our hotel controller, so we we set our rules here in a message. We go back to our hotel controller and now if we want this to be true. We have to, we are looking for the request body. That is the request body is where we do the serialization or in this case. We we do serialization or DC deserialization OK, it's where we go from the Java object of the khujo khujo to Jason. That's the time when we would be able to say this data isn't valid. So in this case we are we have to put in at valid. It's a separate. I think it's actually a case sensitive, but add valid. OK it is a. Separate annotation that goes in front of at request button. And. If you remember our ancient TP codes. This is a bad request. For some reason it is a bad request. So we're going to put it right there because it's going to be true of creating new one and we're going to put it right here because it should also be true of. When we update something, in other words, maybe we could originally created it and we created a perfectly valid one and now we want to say that we want to stay. You know, we we want to have, you know, five more guests show up in our room and we're going to crowd everybody in. And so now it was three before and now it's gonna be eight. We still need to have the validation number. OK, so now we try to find out what I have messed up. I noticed that valid is coming back and read and then min and Max is also in red. Yeah, everything is in Med. It's all terrible. OK, so Maxwell, I guess is that. My guess, and I I say this because I never remember but. Intelligent always helps. Like point helpfully points this out. That I show my content. Put class show my contacts here class. My guess is that was the problem. OK, so in other words it needs to have that constraints. Now. Anyway that? Intelligence is really handy. I never remember the actually have to import the individual wants, but because by the time I've done one of them, I've done that. I then worked on it for another six months and like I don't remember that I did that way back when, you know. So yeah. And then the the valid in the hotel controller also needs to be imported. Has the same problem, right? Context. So now I should be able to run. And I mean luck. OK. So now we have our our thing. We're going to go back to postman. I strongly recommend that you all your testing at, you know through posting it is just. That's where you start. Don't worry, because otherwise what will happen is it. We give you, we hopefully give you a working version of the client. So you can import the server and test, but that's not always going to be a chase if you've got both, things might be wrong. What you want to do is take out the if you if like. If you are trying to figure out why a lamp is not working in your living room. Then you and OK. When you turn on the switch, it doesn't go off. Debugging is often like this. What is the first thing you do? Do you throw out the lamp and get a new lamp? No, no, you don't do that. No, no, no, don't nod your heads. Instead, what you do is you try changing the lightbulb. Right. If the light bulb. What if a new light bulb works? It wasn't your lamp, it was your light bulb. If a new light bulb doesn't work. Do you throw out your laptop now? You still don't throw out your lamp. You try plugging it into a plug that you know works, because maybe that plug in the circuit breaker it slow. And on Lee, after you have checked each of those things, you get back to throwing out the lamp by anything unless you desperately want a new lamp and you look for any excuse III. Remember this Car Talk show and how many of you ever listen to Car Talk where this person was calling you and complaining because they had, you know, uh, I don't remember what it was. A Volvo, I guess 'cause Volvos are super reliable. And she had been letting it run out of oil. You know, she had tried to run it completely dry of oil because she desperately wanted a new car and she could not get that one to die. So she had an excuse to buy a new car. That's an odd response, but. But it was an interesting car. explanation first, if you wrote the server and you wrote the code as much as I admire you all as developers. it is probably your fault that something isn't working so go to the thing that is working and It is probably your fault that something isn't working, so go to the thing that is working and try to figure out well if you think your server might be failing, go to postman, which you didn't right? And no, it's probably pretty solid and see just like when you were trying to write your code with your PA. Oh, and you're trying to find out. Is the database broken or are you broken, you know? You know which one do we think so we got a PG admin and we try to run the request there first because it's more likely and then if it works at my code still isn't work. I've put the code in. If it doesn't work in PG admin, my database is set up wrong. I mean like it's it's trying to eliminate the things that it might be OK. So having said that, we're going to go through here and we'll go back and look and we'll see if our see if we can find. One of our fun ones. I may have deleted them all so I don't know. Hey, what was the party here? No body was good. Alright, so this is this is creating a new reservation here. If I run this and it works well. Name check in date, guess whatever it is. If it works then I should be able to actually get back what is a response right? Posting it. So I'm creating. What should be the status group? If it were you are 1201? Cat in the concrete? Yeah, got a 201. And there's my mind, I. It's going to be 201 not because I use post in this case I like. That's why I wanted to do it. But that's not why it's giving me a 201. It's giving me a 201 because I as a good developer. Did the right thing and put in that HTTP status created. The reason I emphasize that is if you don't. Chances are good. Now I I I haven't tested this, so this is completely. You know. The way it should work versus, I don't know David and Spring boot are also brilliant. But like sometimes they figure stuff out on their own. But so this demonstration may fall flat on its face. Not I don't care with that. So we're in that let. We're going to go ahead and run this again. Alright, we'll say. Ted Smith, because we really don't want. You know, John, stick to make two reservations at the same time. So if I write Ted Smith, it's going to give a 200, OK, because that's the default. Everything went smoothly. So it depends on my being the good programmer that I am. And putting in the right response code for a create. Just something to remember. Hey, Ben, can you pull up postman real quick again? Sure. I notice when you did the the post last time, it wasn't posted to the right hotel. Yeah. See, it's posting a hotel ID zero. Even though you're in free. Good point. It's also giving you the idea of four, which is what the last one had as well. Right. OK. So let's go find out why. OK. So I'm doing hotels ID, ID, reservation path variable is ID. And Bally was hotel ID. OK. When I go back and I look at my. Postman so many things running. Going to hotel Surrey. And I'm trying to post it. To reservations. So I go back just checking my my logic here. OK, so it should be calling this and it should be calling this right here. In my past aerial, which is ID, that all looks correct. Right. But it does not seem to be working. So when we don't know we set a brief point, we stop our thing. Is it I say this because you may be sitting there running your application doing this same thing going by? And you have two choices at that point you go look for an instructor and we may be busy off making chocolate chip cookies, or you can try debug. Yeah, I went to watch a movie last night and I thought, well, I mean popcorn. I looked around, I could not find my popcorn maker. I'm still not finding things after this move, so I could not find my popcorn maker. So I wound up having to make sure I just. That is the life of a of an instructor. I'm going to go to. Spanish in debug mode. May or may not give me an answer, but let's let's find out what what it is doing when we do this. So we are going to go ahead and post posting here. We're going to send it. Hot bradenville here. And what does it have now? The hotel ID is 3. So it certainly looks like it should be doing the right thing, right? So then we say well. The reservation. What is the reservation show? Reservation shows an ID of three or zero hotel ID of zero and these other things. So I didn't set the hotel ID in my. Postman, but I did specify. So let's go back and look at our code again. I do model good behavior for you. I'm going to go into it. I'm going to say, OK, I should get the. Yes, and I add my reservation. What's wrong? I never bothered to set my hotel. I like if I'm gonna pass it in. You wouldn't really think that it should use it. So what should be happening right here? Is it should be setting the ID and it isn't. So good catch there. That is clearly an oversight. I am. Where is that gonna be? That's gonna be in the. Memory hotel. Sorry where? Where am I right now? I am in the. Memory reservation day. Oh, which I'm not really. Yeah, memory reservation. That's where I am. Alright, let's stop this because we don't really care. Let it fail. Now let's say. Reservation. That. Set. Hotel I eat. Equals to hotel ID. By the way, I imagine that the reason why that didn't get caught when they were writing this code. But again, we're back to nobody's perfect. We're already in code. We're doing our thing. The people who wrote this example. On our lovely curriculum team, right? When they were calling it, they were calling it from something that. From the client and on the client side. When it built this out, it added the hotel ID. Since that worked, they never noticed that it didn't pay attention to this hotel, I think. It's very easy and this is all part of testing. It's very easy to miss that when the code works, it's probably why you have to think like a. You know, I tend to say, think like a hacker. What's gonna happen when you don't do the thing that it says you should do? Alright, so in this case the hotel ID wasn't set. We fixed it. Presumably we'll see. Presumably now that I have set the hotel ID as well as setting the ID. Now, why didn't the other one? I don't know. It's a good question, but I would guess it had to do with. Something about this get Max ID plus one not. Doing what it was supposed to do, because the hotel idea was zero or something like that, but we'll see. Let's go ahead and run it. What are we doing? OK, but it's. Let's run it in debug mode one more time. Just because we might want to stop there and just see where doing it right and do something wrong or right for the wrong place. Start up my server again. Go back. Run. So I got to here now, so I would step in. And I'm here my I hotel. I my reservation right now has an ID of 0. I don't know how to get Max ID plus one works, but in theory we should at least be beyond 4, right? Because that was the last thing we got. So let's see what it does when we step over that. Still gets 4. Something is wrong here. I think it's actually that's OK because the server was closed and reopened, so that's previous one was posing and reopening the server and it keeps resetting it back to the original. You're right, that's what it is. Well, we'll try to get. We'll run another one and we'll make sure it gets to five at that. So this this is OK. Alright, I keep forgetting that we're not running again. Any persistent data store or whatever. Alright, so now we're going to set over this in our hotel. ID is 3. So when we step over this, we should sure enough have a hotel idea free. So when we continue on. We're going to have. We're going to get our tour 201 back. And we're going to get our thing with an idea for our hotel ID of three, all this stuff, right? What happens? Let's let's run it one more time. Just because we're curious what happens. When do we get a 5 this time? So we're gonna go ahead and run it again. It's going to get here again. I'm just let it run. OK. We go back here. Now it is a hotel at 5. By the way, does everyone because Elias brought that up and they didn't, you know? I wanna make sure everybody else is on the same page. There. You understand why I keep going back to Ivy 4. Right, I kept stopping the server and because we have everything in memory entirely with, you know, the entire memory storage in memory, it starts over with the same three reservations every time I restart. It doesn't keep track of anything else because I'm doing this memory. If I were saving it to a database. Which we all get there. But if I were saving it to a database then it wouldn't do that it would, it would when I stopped and started it again. It would. We would still have the reservation to be created and the reason. But the reason why is that there would actually be my server would be caught, would be the client to Postgres which is another server which would act. It is actually storing their mission. So be that as a bank. Yeah. I'm sorry. Somehow I missed how we create corrected the hotel ID to make it 3. The hotel ID what was happening was I was shutting down the server and restarted it. And the server the way we have it right now is starting with three reservations every time I restart. So because I was turning it on, stopping it and starting again in debug mode. Or stopping and starting again every time I did that, it's going to go back to the first three reservations. No, I mean the hotel ID. We remember the hotel ID. Yeah. Sorry. Different. Different issue. The problem we had there was that they thought I say they being the curriculum team they thought. That because this is the way that client would call it, that we had the three up here, but we also had hotel ID. You know. Three hours, they thought we were calling. This this way. If we had, this would have all worked. We were not calling it this way. We were only putting the three up here, so the fix that we need for that. Hey, go out of here. The fix that we made is we go into our code. Um. Inside, where we actually create the reservation. We added this line right here. Now, if we didn't, we didn't assume that the reservation that they were giving us had all the values that needed. We went ahead and. Manually set the reservation or the hotel ID before we added it to the list. That makes sense, OK. any other questions on on this logic right now I would guess that if we put it in. OK. If we do it like this, I would guess it will work fine. it's a valid question, so if we do this then we do this. It is going to be fine. It will put in the right hotel at if we. And I'm actually going to do two at the same time. Let's say that I said this is going to H1. And most say that I'm gonna say the ID. Is 450. Flying 40, OK, right. If I go do this. I'm gonna try to create it. And it's going to I'm going to step into this. Now, right now the hotel the the Deserialized version of the hotel. So I took taken it out of Jason, put it into the hotel object. It has 440 in hotel of one and all this time. And then I'm going to step over. Set the ID that's gonna change the ID. And the step over the hotel ID is gonna change it to three. Right. But that's a really important thing to recognize it. Sorry, wrong place. OK. Really important to recognize that what it's doing is not just forcing all this in. The reason why the create returns of value is because this may not be the same and things like that ID number always give her placed in this case if I if I'm trying to put a resource for H3. Then it's may. It's going to override. You can either put it in or not, but whatever you do is keep gonna ignore that value and it's going to make needs, but so that is why in earlier code, remember how when we did a create. Of the object. And we didn't just update the ID, we went ahead and got it again from. Alright, the reason why we jotted again was because it could have changed like this. That was the reason why getting it by ID was worth doing. Alright, so now we have done. Jump through hoops, we've done a lot of stuff. We've we've, you know, that's great. No, this is the process that you go through when you're coding. So I don't really need the hotel idea. I certainly don't need the wrong hotel ID, but we'll leave it here for now. What about if I say I want? 10 yes. So now this is the situation where there's nothing wrong with my my other logic, but I have violated any a constraint that I had built in. So is this valid anymore? And this code right here? So where's my create? Creating new reservation for a given hotel, right? That is valid. Is going to test any of the annotations that I have built into my model. So if I run down here as kind. Check the validations and notice the way the annotations such wanna remind you that annotations I can have as many as I want. And they all apply to the the thing that is underneath to the one variable underneath. So we should violate this. So let's see what happens when we do this. Right back here I am violating my maximum. Send that. I am going to get a 400 a bad request. Because that's what spring boot does. When, when that is add is valid, doesn't work. When valid, it will do this and now look at what else it does. It gives it trace values. It gives a bunch of stuff. It says bad request, but somewhere down at the bottom of all this nonsense it will see the message. And the message is validation failed, object reservation physically it's one error. Now it's possible to have. It could violate multiple errors if you hit like it's going to do all the validation. It's not just the one. It's unlike Java, where as soon as it gets one error, the rest of them you should probably ignore because they're all just cascading. If you have, if you violate multiple constraints, it wants to let you know all of the things you might need to fix at once. Since there is only one, it is. I'm going to show you the errors here. Cafe. Gibberish stuff, and somewhere down in here is going to have. The default message, which is the maximum number of guests is 1. Now when you sit and look at all this gibberish. Right. It's a bit of a pain to do something with it. It's a bit of a pain to find out what you're looking at where and whatever else I one of the things I want to point out to you is that a lot of the clients that are going to be reading this are JavaScript. They're gonna be on the web. And in JavaScript I can say make this a Java object a JavaScript object. So I think the reason why it's called JavaScript object notation is that serialization and deserialization in JavaScript is incredibly simple. I can just make this a JavaScript object and then I can point directly to the things I need to do. That's that's sort of the power. If I'm in Java, it's a little more complicated. Not a lot more complicated, but I can go ahead and deserialize it in my own way and say go to the right thing. I'm just saying that in JavaScript, which is where was it supposed to sign for, and what is often calling it? It is really easy to go cool the pieces out and use it because you just turn it into a JavaScript object and work with it. And JavaScript, unlike Java, I don't have to have created the whole model. It's just the objects that are there. We'll get to JavaScript a little later, not even that far from now. But we will get two JavaScript soon and when we do one of the things you're going to see is that I don't have strict like there isn't a class definition. I just have an object. It has whatever it has in it. If I happen to know what that is, I can use that stuff, so that's JavaScript is very, very loosey Goosey on that stuff anyway. The validation failed and I got my 400 which is really the most that I generally need to know something went wrong. But let's talk about. More about so we can now let's talk about the fact that that means that we can allow people to put whatever random gibberish stuff they do. We can validate this before we start adding it to our system. We can filter out the stuff that doesn't match our business rules. And by business rules I mean I am a hotel running, I am a hotel chain running a business. I don't want bad data in my system. I don't want people making reservations that are invalid. I can do all that validation this way. Back in here for a minute, we'll talk about. What would you think? Are some other validations that would make sense? Given this is supposed to be a hotel reservation, right? So what are some validations that would make sense here to you? Is it possible we could deal with the whole hotel ID not matching situation in this? I think that's an excellent idea. We should definitely not allow for a 0. OK, so in that case we at the very minimum you would think we would do a Amin. OK. And just like the other one will do, a value equals one. Acid. People say you know. Wow field. Tell. ID. Let's be positive it is required. I will say is required because the most likely thing is that someone hasn't specified. The problem with this. Is. And this is what they suggest in the curriculum code. This is what they suggest my problem with this is I don't want people to have this specified hotel ID. I already know the hotel. I why in the world am I forcing people to put the hotel ID in when I already know of that information? Right. So. You know, I I, on the other hand. The flip side of that logic is remember that this same validation rule is used for updates. And on updates I don't know the hotel ID and I don't want someone changing this to 0. So you know, whenever I'll, I'll, I'll. I don't know so far of away. To make these only apply to certain calls, if you had to do that, I think you would have to have your own test. You would, I mean remember you can always have your own customer, you know errors. But I think that will leave this that I want you to put in all the information. You should have a valid hotel ID. I will not. I will ignore the fact that I may override it. But I should have something. OK, what else might we have? Should definitely remain right. After what? You should definitely have a name. We should definitely have a name. So how do we do that? Remember what would be a good way of? Uh, I think it's not blank. Yeah, let's do add, not blank. OK. No, we don't have to have a value now because it's just there's just a message. Despite, OK, Yeah, I'm going to reiterate this 'cause. It's really obvious when I'm writing it and saying here, but you'll forget about it later that the message is assuming that it has failed. So you always write the message not as and you need to not be blank when you need to say someone. Someone has given you a blank value. OK, I'm going to do this the same way, which is the main reason that someone is likely to have given you this is they didn't include it at all, and the default value is blank. So I am probably going to just say the field. No full name. Is required. Telling him it should be blank, whatever, whether or not they did this in here by. Doing this. Or. This either way, it's kind of the same thing you haven't given me a full name, so there is always a certain amount of. It's not a right way or a right message. Try to come up with something that generally will handle the situation people have, which is we need to have a real full name, right? So what about? This is a little different. What about the check in date? Yeah, I was just gonna ask about that. Why wouldn't? Why are they strings? And why wouldn't it be like annotated with past and future? That's a good question. If it were, end it ated with past and future, you know with checking dates, we probably need to give it some sort of a default value. But yeah, I II would agree with you. Why in the world are we doing these as strings? We should really be dates. That is, that is a code review question. Because you might say to yourself this is gonna be a pain in the neck if I'm trying to do any business logic with this object. Like what I wanna know is when you know when is it overlapping dates and all that kind of stuff. I think you guys had a test, you know, uh exercise example already on dates being you know, do we have an open spot? Excellent question. I don't dare change it right this because I don't know where what other things do it, but, but as a as a code review question I would say yeah, should we be storing this as a strength? That is the kind of thing you should look at and say question things question why are we doing this in this strike given that we are doing it as a string. Be about the best we can do is say. I mean, there are things we could do. We could, we could say we could use the regular expression logic if we knew. Red Jacks well enough. Which I'm afraid to do it, but right off the top of my head. But let's say that I want to say you can only make reservations for this year and we'll just we'll do something like that, OK. The check in date is going to have to be this year. So at that point we could do. Oh, and we remember from last name what the what the term is. I don't know. You go pull it up. Pattern. Pattern. That's weird. OK, so we let's do let me look up pattern.

Do it from there. Team last answer like put it in here. I just have it on a different window. OK, so I could do my regex. I could say. Let's say it has to start with. 2022. And we'll say. Idiotic is this is must be a valid date. Right, but it probably should be able to do is here is do something along the lines of, you know zero through 9. Show my juicy row three one through 12 and I can't do that. OK. I I could do this kind of logic. I'm not going to do it. I'm just gonna say it has to start with that format. That's the only format I align, right. Or I could say the other way around. I could say I must end in 2022. Assuming the other ones I'm going to say Slash 2022$ right? If it isn't, then it's not a valid date. I'll do that for both the checking and checking update. OK. Should say check-in must be a valid date. I will leave it as an exercise for you who are curious. Get the validation for a date that you accept. You can even say I want in this format or this format. Regular expressions are lots of fun to play with, but if you do that, you can actually set up OK something like this, but is any of this bad like the other thing, I want to emphasize here is now we're going back to that, that. I can make a validation that works. But should I be doing that job? Is that the right way to handle the problem? No. The right way to handle the problem is to make it a date and you can say is that after today and it's this is one of them before the other. I mean like that is the right way to do it. But it can be very easy to get caught in this pattern of I can do something. Right. So let's try that, see if it works. Just going I'm curious whether my. I don't know whether I have to slash. I better just. I better not assume that even barely shoot that will work. OK, so let's see if this will work with our with our code. What we're complaining about? So now if we. Go back to our reservation. Start. Remember, we've just last. I just stopped at which means we've just lost any reservations. We just created that. Gonna say class not blank is not there because I haven't imported it. By the way. I'm fairly sure not positive, but I'm curious here. 'cause I want to test it I'm fairly sure. But I should be able to. Do that and it should do the entire thing. I think it's you have to do dot\* to import all the constraints. OK. Let's see if that works. He seems happier. That way it's not going to costly as I try different things. What I should mention, by the way, is that I should be able if these were dates. That I should be able to in my after I should be able to say or past or future or whatever. I should be able to say past the future of the other one so that I could say like the checkout date has to be lated in the check in date. That's another thing I I will try to find that code. We put that in which is curious about that, but I think there's a way to do that too. And what I mean by that is I could say. I could have a value here. And in my value I could use a variable that's already defined in here. So for instance, if I wanted to see my hotel ID must always be greater than my, you know reservation ID, which makes no sense whatsoever. I could say my my value was my value down here. For Hotel ID was ID. That was the minimum value. Here there. Let's see if that this works. So when we try to do this right now it is going to. It's gonna fail, which we knew it would. Is it gonna feel with two errors? I'm hoping it'll fills with three errors I've managed to. We play knock this one for a loop. Yeah. So what are my errors? OK. Checking must be a valid date in 2022. Checkout it must be a valid date in 2022. Oh yeah, that's great. This example that I came up with is from an old one. So let's see if we can fix that. Say. Three people. Still has an error. It probably has an error with my syntax is wrong for the regex. Yeah, it's still gonna have an error here. 'cause. I've got the wrong red jacks. I did one of the downsides of Red Jacks is that. I know there are at least three different well accepted regex systems that all have different rules. And I don't know, so I don't even know which one is using and there's the Python one. There's the JavaScript one, and there's the. It's called SQL one and they all have different rules. I will instead say fine. He is the cheesiest possible one here. I would say not blank. I'll say the field. Back here. Let's do this. Yeah, we're gonna. We're gonna get really cheesy here. We're going to do this. Are there any other tests you guys can see that we might want to do? Put in another right. You see a reason why we're not going to put. A test on the ID. K may explain to me why we're not going to put a test on the ID. Isn't IT system generated? It's system generated? We don't want to force somebody to put in a value that we're going to override anyway because. It just confuses matters, so don't put in validation where you don't want it. I do put it in where you do want it. Can anybody see in our model so far? Why we do not have validation on the hotel or the address? 'cause it could. But we don't. Well, the reason we don't is because our model so far does not allow you to create or update. Hotels. So if it doesn't allow you to do that thing, we have no place to check that validation. So there's no point in doing it. So if you have a system set up which is perfectly legitimate, I might have a Web API. It's like, say, why in the world would I ever give anybody out there the ability to add a new hotel? Hotels are an internal process. They're not extra process so. Your API does not simply implement everything that is possible to implement. It implements everything that you want people to be able to do. Now one thing you do want to be able to have people do like I should point out 'cause it's relatively simple is you know what actually we'll do it after a break. But but one thing we which we can do is we can delete a hotel and and come up with an error on that one. So why don't we come back at 11:20? Then we will implement that last thing. i'm pretty rigid about these breaks by the way some instructors just go and some instructors take them whenever one seems tired i like i will never remember to take a break unless i do it around the top of the hour so i would i'm just let it go

My wife named this one Natasha, but then the cats didn't choose the right person, so whatever. They both showed a very clear preference, which are cats have not other cats. We've had not so necessarily done, but these two were each clearly. Our cats. So which was that? Was at least fortunate in terms of of if you're going to live different places, at least the cats want to be in. My cat chose my mother when I was a kid. It happens. They just choose someone and that's their purse. OK. But us? Recording in progress. I just wanted to point out one more thing, which is both about. Codesign or thinking about things and about? And then one more you know constraint that we could put on a validation that we could put on. I added. This one right here, which is if I wanted to force. The checking date to be 10 characters long, which does not in any way force it to be slash TV slash. Why? Why I couldn't do it like this? So I know it's gonna be 10 characters. Neither case. I could say I could give an error message which tells it that it should be a that. And the reality is I'm not really enforcing that, but I'm at least telling the right information. That point that I got first of all is you can use at size, you can set only the minimum or only the maximum. You can set both of them. That can be really handy for things like names and whatever else where you just want to set a maximum you want to set something else. You will sometimes see at length. Which is as far as I can tell, identical to ask size. I don't know. There's some reason that there's some technical reason I found on a website about why you should use one or the other, but like, I don't think it really makes any difference to you guys. What I was going to say though was the other thing is. Should I have both of these? And this is not there's not a right or wrong answer to this, but this is basically saying is I mean like the second one would work. The thing that the second one wouldn't handle is where they didn't include a. Checking it at all. So the nice thing about including this first one that says it's not blank is this. Unlike, they're actually if it's blank, is blank or, but if they don't include it, it's going to use this. And then this first message will come out. So it may be and actually they both come out if you if you don't include it. Both of these errors will be in there. So that is one thing other way I should definitely check because I I do these things and I don't always check. So make sure that this passes. This is my better at least allow me to do valid things. So when I. Do that. I got my 201 so it didn't really create it allowed this. If I had said. Yes, I did it again. It's going to give a 400 and it's going to say down here in the. Gibberish at the bottom. It's gonna say there's one error. It's gonna have failed and validation is gonna say check-in date much pre formatted as this and this OK. I mean it's kind of useful. You know, I can see that working the person wouldn't know that they could have put in you know a B slash E C slash EFF whenever. Anyway, OK. That was just a side note. So you both remember that you could do a size which is very, very valuable, just as a point to that, the place where you're going to start wanting to do this. The size is the way you might want to think about it is when you start tying together a Web API where you could input stuff. To Adieo where you come with a JDBC DL where you're going to save it into a database table and the database table has a bar bar card. You know size of 50. You could let the database fail when you put it in. But you're better off putting the size validation in here so that in your Web API it will get an error right away and you never try to shove it into your database incorrectly. All right, it gets harder and harder the more errors have to bubble up from different places, the harder it is to make them bubble up in any way that makes sense to the person calling. The calling application. Isn't gonna really understand why SQL is having a problem with something, and it's very hard to bubble up that error, so putting the validation in here is a good way to make sure that I'm not going to violate a constraint that is in my database, so that is actually. That's why I wanted to bring up size. Let's go back to our hotel controller. So we have now done our. Our update and our delete, do we ever do an update? We never try the update. Let's go ahead and try the update. We just created this right here. Oh wait, we we didn't create this one because this one failed. Create a new one so that we're, you know, we're gonna say. Isabel. Smith. OK, it's gonna have just one guest and we don't care what the idea is. And is there a hotel? The hotel is 3 now. So I'm gonna go ahead and mail. It's gonna fail. 'cause. I don't have a hotel, do I? No. You're in hotels three still and you have a hotel idea. Three, but the check in date still isn't gonna have the correct lane size there. So so we go ahead and do this, alright and. Presumably this will work. If we look down here, we're going to see our ID is fine. So now let us go to another. We're now going to take that information actually go back to wherever the post was. Take this information. Is this the easiest way I post to make sure I've got one. And we're going to go to. This. We're gonna say we're going to make this up put. There's a put is an update. HTTP, whatever. But we're now going to do it in relatin. In reservations. And we know it is ID 5 this is. That's the one we're dealing with right now. No wonder our body. Raw, Jason. Guy. So we're going to say Jezebels name is spelled wrong. So we're going to update that and everything else is going to stay the same. But when we put this. That is, if we do everything right. Updated, went ahead and returned the thing back to us, which is gonna look very similar to this one. Probably we could get away without putting the ID in because it's not required, but it would give it back to us anyway. But we should really put in all the values for put just in case anybody wonders. There is also another thing called Patch. Patch is where patches right there. Patch has slightly different rules and the format is not as obvious but Patch says I only wanna set one value. I don't want to set everything. So with patch you can go ahead and set and we're not implementing any here. This is more for your general knowledge that there is a way to do it. Think about the case where I don't want to give somebody all the information. I don't want them to overwrite it like for example we had said if they were going out to do something at the hotel and they don't have the address. Like what do I do? Well, Patch will allow you to update just the one value that you care about, or just a few values that you care about. It's just a little less. Clear what you are required to have, so that's why I want you to know that it does exist so that this work and let's say that I now wanted to update the same thing over again this time I wanted to update it to 20 guests. It's again should fail and it should fail. Now, because you know we've one error which was again the maximum number of guests is 5. so that's just pointing out that our update also works and our update also works not because it's magic but because when we did our update right here we put the add ballot I'm not gonna do this actually, 'cause, I would have to restart now I have to recreate the thing in like. Trust baby, open the app, valid. It just doesn't raise the errors. Some of you may have experienced that when you tried to do the tutorial because he doesn't ever specify to put the italic the reading it. But the tutorial OK. Last one that we have to do is delete and delete is just going to be like OK, the reservation. You know. Jezebel, really furious that she can't have 20 guests stay in her hotel room. So we're going to go ahead and. Fix this up. So it's delete a reservation. Now think about it. Tell me if we're going to delete. Leaving aside the javadocs, we're going to delete. What do we need to have? Like what is our request path likely to look like? We're trying to delete an individual. Reservation. Are we going to use a pattern more like? The hotels ID reservations. Or are we going to use something more like the update? What do you think? More likely opting yeah. Should look more like the update. We won't fact. Sorry. It will look a lot like it. We will say request. Mapping. Pad equals. Slash ID. Right. The method now. Some method equals request method dot Right, this is now the equivalent over here of making this. I delete. We're not going to do just yet. China so. Code. OK, so this is that's this. What do we want it to return as a status code? When it's done. Should it return a 200? That's a generic successful. Maybe two or two. No two. I'm not sure what 202 is even. 201 is create the correct thing says it's expected. Got your cats in front of you? What you want is 204. And the reason why 204 is 204 says I'm successful. I have nothing to give back to you. I have no information. There's nobody. So what you want in that case then is a. Remember, we look up here. To model on our create, create. That's how you do it response status. Is created. Well, what we're going to do is respond status is no content. Like PR where I am. Still in the wrong place. I saw. There we go. So we're going to say actually I think you need to have it first. I'm not positive and response status. Each cheaper each TP. Better make sure I've got the right. And I have to do this issue to be status stuff OK. Status dot. I looked down here, it will give me some of my things and what I looking for is no content. OK, so there's no content, so if you're not sure. You can look these up and you can probably get a pretty good idea. I don't even need to know what the number is, I just need to know. It means it's a it's a no content. So that will give me the. The response status of 204, which is no content, which means I don't include a body. When I. Does everybody get what it means by the way to say don't include a body? OK, it means that I am giving back a status code that says it's done, but there's nothing to give back. I don't need to give back the reservation anymore because it seems to exist, so I don't need to know it anymore. And there's nothing else to give back. By the way, it would work just fine. Virtually every application, everywhere. If you return to 200 as well 204 is is nice to have but. You will see plenty of APIs that just don't bother. You should still do it, but bear that in mind as you may see ones that don't do it. OK, so now we need to do our delete. So public. Website doing the rock says yes, public. What's going to be the return value to this? Am I returning anything? Turn public void. And I'm gonna call it delete, delete reservation. Manager, patient and now I'm going now. I don't have to give it a body passed in either. Right 'cause. All I need to know is basically that URL. So what I'm going to do is and and since it's got that idea, I do need to handle this. So what do I need to in order to get that ID and remember? Like, how do I get this ID? Isn't that just pathway variable ID, path variable? Int ID? Right. So and then I'm going to say. But this is important. It throws. Reservation. Not found. We should not found exception. OK, that's important, because again, I'm expresser specify a reservation that doesn't exist. You know what, honestly, there are some API's and this is again just so you know how the world works out there. There's some APIs that would say, well, fine, you give me a reservation doesn't exist. It's obviously been deleted. Forget it. I guess that's actually generally not good form. People need to know that they really got rid of the thing they were getting rid of. But it's it's sometimes you will see that, OK. So anyway we have this and then we have our body and our body is incredibly simple. Here we're going to go back and we're going to look at our reservation. Deo, how do we delete something? It's a void. We delete the thing using the INT. So there's this thing that's by the other way, by the way. The other way that we would know what our return value was is what does it show in the. In the reservation DL, but OK, so we're gonna go in here, and we're gonna say reservation day O dot delete. And we're going to use that ID. And then we're done, right? That's, that's the whole shebang, right. I want to. I mean and and we can go test this and whatever. In fact I will, I will test it because you know we do this thing. Let's delete. Let's start out with an error. OK, we're gonna see reservations. We're going to say Slash 404. So we're going to delete reservation 404. We go ahead and we do it says method not allowed. Funky want to have everything. Request method delete not supported. Bizarre. OK, so let's go back, see what we did, what we did wrong on that one. Um. Did you restart the server? No, I didn't. That's why 'cause it's the previous version that doesn't have the server in the delete implemented. There you go. Y'all need to do critical thinking, especially when it gets late in the day and I'm I am tired. OK, I will forget things. Your job is to to figure those things out. OK, so good call there. So we are going to go back and do our. We're going to try again. We're going to still going in there, but notice that we got a different error. We got 405 because we didn't ever implemented delete. Now we should have implemented this, Willie. So we're going to get there, but this time we're going to get a 404, which is that it's not found. And the 404. That we get is reservation not found. That's it. Might be a nice one, and if we go back into our code and we look at our reservation not found exception, that's the error we got here and the error message that it sent back is issue status not found. Right. Again, you could create a custom exception that was not just not found, that was something else if you wanted to, but generally speaking, what you were going to do or trying to throw on all these is they're not found version of things. Alright. What if we actually were gonna jump back? We know we had this here, so let's do our post. We're going to go ahead and create. Something to delete. OK, so this is the one that we created now. So that's our response. So we didn't know that ID 4 is there. So if we switch back to our elite, we change it to just 4. OK, actually let's do our. Let's see if we have our get list of reservations. OK, so we're gonna send our list of reservations. So that's not gonna have anything. Why is it going to have anything? Because I didn't take it up the rest of it reservation. As your vacations. So when we get our list of reservations, it should give us 123 and four. So now we go do our delete. And delete done before. It says. OK, I had no content to give back to you. So it went through successfully. If we now go back to our get list of reservations and we do our send, it will get one and two and three but not 4. Right. So. Alright. Now, because it's the end of the day. This will make. Little fast for you anyway. I might as well talk about it now. I'm going to talk about two more things. One thing that changed in the code that we give you. Can you see what's different now about this? Then about yesterday's example. Is this the constructor dependency injection? Yes. We don't actually do the new memory Dio thing the way we did in yesterday's lecture. At that point we were creating. A new memory. I I knew DA. Oh, right in here. And remember, I mentioned to you that this meant that our controller was more aware than it should be with the implementation of how your your data was stored. It meant if I were starring in a database, I would have had to have a new JDBC. I would have had to do a connection. I had to do whatever else. Dependency injection. Is one of these very fancy words that people like to throw around, and it shows that you know what you're talking about when you're at a really geeky cocktail party. But dependency injection just says. That stuff's hidden over there. We'll go ahead and just put it in when we need it. With a with a controller and ADA. The fact that I'm a rest controller allows me to do independency in. His head. I don't go to those kind of cocktail parties. The fact that you can do it. Is just built into the way spring manages things. We could have actually done it yesterday was actually. There was nothing we actually set up differently, right? But. What you do need? Is this little magic annotation here? Fat at component in front of the the implementation thing. The Memory Hotel D AO is says that component says I can be used for the dependency injection. Right. You know, whatever. Do you understand that you understand it? If you don't, then OK, fine. Neither do I. Really. But it does. What it's supposed to do. And this is how you do it, OK? Anybody have any questions about that, which I almost certainly will not be able to answer. Alright, it's magic. There's a certain amount of magic in it all. I'm OK. I mean I, II when I went out to create. A software company. Right, I called him Genii software because I think software is pretty much magic and so. And that's the name of my company that I I just. I think this stuff is matched. You learn the the incantations that you need to learn that works, right? You don't have to know everything alright? Something I want to show you. This is back in the. Because the theme of this lesson is also all the things that I can say that like. About like code review, thinking about that, that thinking process and again part of the reason I'm bringing up code reviews and bringing up the the critical. Thinking process it's because you're going to have a capstone next week and then I'm going to do code reviews with you again and hopefully at that point I'm gonna give you more of this. Grandiose critical thinking. Maybe you could have done it this way. Kind of whatever. Alright, so let me point out for you. Something the curriculum team put together, which I have already put in a curriculum change request, which is how we I I could have slews of these things is how we say this question should have been better phrase for this thing should change or this whole entire lesson. You know, Mrs this context or whatever and we we throw in lots and lots of them and they track them all and then they have. Take all these sprints, which is you'll get into the agile world, which is where you you all get together and you have spend one particular time doing a whole bunch of things at once, fixing all the problems at once. That's the Sprint. You will do sprints in your final capstone. You won't particularly do them with this next one, but OK, so if I go down here. They put in for you in the code. This is a different version of the same thing that I did yesterday. When I was. When I was showing you how you could search by state. They put in. They wonder what they were trying to show. Is that when I was listing hotels I would do with my list. My list reservations now that's my list. Reservations with my list hotels. OK, right there. My list of hotels and what they were saying. When you do this you would use this path. If you wanted to use another thing with a jet. That would would give a filtered view of this. Let's give it a separate. You know URL which is slash hotels slash filter. And then we would use it looking like this, where it says Hotel slash hotel slash filter? Skate, whatever else. What I did for you yesterday was say. I didn't use this slash filter. Right. The reason is because this is just not what you do with rest. This is a a necktie pattern. It's not a good thing you don't create. Parts of URL that are about actions. Parts of your URL are all about resources, your method names, your get, post, whatever else. Those are about actions. If you want to filters the results of something, whether it's deleting, you know something based on a filter, or whether it's creating something, or whether it's getting something, it's usually getting something. You should not do it with a slash filter. You should just do it with slash filter with the way I did yesterday, which was to make parameters that are not required. But in any case, this is what they did. What they said was we're going to do a filter. It's gonna look like this. We're going to require the state, but if you want to, you could say there's an optional city so I could get all the ones for Ohio. Or I could just get the ones for Cleveland. This will be in your code. It doesn't look a whole lot different, but I want to use it as to jump up and down on my, you know, soapbox and say don't use. Things that have to do with actions. In general, there are certain very certain admin control things. Sometimes people have a shutdown type method where there's no valid way to. You're talking about the system, you're not talking about the resources, but anything you're doing with the resource. It's perfectly fine to have the question mark. You know, to have the query parameters, but if you're going to have the query parameters. Then they should just be on the resource or collection of resources. They should either be like this if it were going to be. I want a filtered list of the collection. Or it could be if I want H5. I could say which what is some information about how I want to give it back? Query parameters are about when we talk about MVC, the model, view, controller, query parameters are the thing that tells that are in the controller that says how should I present the view? OK. So. And myself. OK, with that said, I am gonna well, I don't. I didn't look at the. Exercise you did for yesterday. What other instructor told me this morning was that she wished she'd listened to my advice and implemented this the way I did in lecture yesterday, because it confused people when they got to the the. The exercise last night, I don't know what she's talking about, but you know. I I think that somehow it helped you to do this this way and then I can give myself pattern back and have a cookie and I will. I think we're done for the day. Does anybody have any questions about any of this? And and bear in mind.