Let us begin. Figure out what I'm sharing. Cord. Let's get started. Alright, so today's topic is data security. And data security is interesting. Interesting problem because it is both not something that you are going to have to. Focus on very much as a beginning developer. And it is true that it with all kinds of security, that it is everybody's job. OK, so you both need to. You probably will not be implementing security policies. You will probably not be doing that level of work. And yet, on the other hand, it's kind of like the the idea that every person who gets email in your organization. Needs to be aware not to click on on, you know random links that come through the mail. Because that's all related to, you know. Um. It's all all related to keeping the entire community. Save the entire organization, save. When you are programming, you have additional responsibilities to make sure that everything is safe. Policies are great. You know, policies are terrific and they they help people figure out what to do. But policies don't keep you safe because they're if they're not followed. Or if they're not complete enough. Or if there's something you do that is technically within the policy because they haven't foreseen the situation so data security is sort of a a big topic in all those ways. But you're also in a fortune physician because it is. The advantage of having lots of attackers and let's say there are lots and lots of people out there who are trying to hack into your system who are trying to to cause trouble or trying to steal data, all that kind of stuff. The advantage of all that is a lot of work has been done by other people. To make sure that. You have the tools that you need. To try to keep things as safe as you can manage. Right. So that is the big overarching thing. And. With that said, let's. From what I can see. Fair number of you got through the reading last night. But certainly not everybody. And so I'm going to quickly cover a couple of topics. Well, I really do want you to do the reading. I also want to just quickly cover a couple of the topics that were in boot camp OS for your reading to make sure that we're on the same page before we get into, you know, working through some code and seeing what happens. I do not want to do that on a regular basis because then that's why you're doing the reading yourself. Alright. Is oh. I don't know how many of you noticed because I sort of, you know, buried it a little bit, but there was a video that I got David put in for the C sharp class. I posted it right above in slack, right above the message I had which which told you which things were, you know, that I pushed. The exercises in which things were assigned by and all that kind of stuff. If you have not watched it, it is worth watching even after class, just because in it it's a pretty compelling sort of step-by-step through about how somebody would attack your system. I have to tell you I have a website. And. It only started happening relatively recently, but every once in awhile I would go in and there will be there are, you know, certain forms on my website that allow you to someone to request an evaluation license or something like that for software which nobody ever does now because the company is pretty much, you know, quiet. But when they do they have some. Feels that they can fill out. So some. Script bot thing. Comes along and submits. Proximately 1400 of these, all within a minute. That are all attempts at SQL injection that would work with Postgres or my SQL or you know. SQL Server or my carrier? You can just tell us a whole bunch of of SQL injection attempts. Since I do not have a secret or relational database at all, it they are useless but to me. But if I did and I and I were not protecting myself, they would have tried all the different ways that they could think of with this script and they didn't. It didn't cost them any effort. Nobody sat down and typed in those 14. 100 messages. They have some script that runs out and runs this on every website until it finds someone where they get back information that they need. So is theirs. Their cost is extremely low. Your cost if they break into your system can be extremely high. Fortunately, these things don't have good parallels. Is there a way to make it expensive for them then then you know they wouldn't do it, but there's no way to do that. Really. OK. So having said that. Basics of the SQL injection. I'm just gonna go ahead and put this down here for now. I don't want to do the whole thing, but I just want to show you the basics of it. Are Ben? Yeah. You're not sharing if you're moving too. Hey, ironically. Is that when people do something that causes an error? When you don't know the users, you don't know and trust the users. Part of the solution is to not give them useful errors. So we're gonna go totally against what we've done so far in terms of we're trying to give useful errors. And we're going to say we can't afford to give you a slayers. In fact, we're not. It's not clear we should give errors at all. Instead of what we should do is give them if they're trying. If this is a search situation, for instance. The appropriate thing to do is to just act as if there are no. Results. Right. Even that is dangerous. Can you see why it like? How would someone who was trying to find out whether you had a A, you know, a relational database using SQL unplanned parameterized SQL? They tried to search for O'Malley. And he gave them no users. I'm. What might they try next? Any ideas? They might just try and see if your search. They might try something like just typing, you know, searching for something like AR. And they are returns Maria. They may know you're out accepting substrates. Then they might try mallie. And if mallee returns something. I mean, if you're returned it, you know a something. If you're returned O'Malley, for instance, which might be a valid, you know, name in there. Then they are. They know that the non error you gave is an error. So it can be very tricky to think about how to respond appropriately when someone gives us part of the answer is. In different ways. Just think about do you really want to allow them to do? Partial matches on things. Now, maybe not. On the other hand, what are you going to do if they're trying? You know, we also that the other side of all this is with searches and everything else is we're trying to be friendly. We're trying to be helpful. What are we going to do if they are searching for Danilo and they don't spell it right? When do we try to be helpful there? So the the competition between the trying to help the user and trying not to help the hacker. Can be an ongoing battle of how you how best to handle these things. OK. So that's that's generally and we will actually go through in, in lecture to you know we will go through an exercise where we start actually playing with some of the SQL injection. So you can start you know getting a feel for how how to do this and. I would encourage you, you know, enough SQL. You know enough, whatever. I would encourage you to try this out. To see what starts to happen as you when we have problems and and what you can manage to do is really it's kind of fun to see what you can manage to get the system to do. And that's actually part of the problem. What really happened out here is there are malicious people who are trying to steal all your data. But a lot of the work that was done to figure out what to do was actually done by, you know. Just kids trying to figure out, and I think kids like all of us. But I just kids and people just playing around. Trying to see what fun things they could do once they learned enough, you know then that that stuff gets weaponized by somebody else. So here endless number of creative. You know hacker type people who are not necessarily inherently malicious, but they are feeding into this whole ecosystem, move of spammers and hackers and whatever else. And I mean, you know. I may have mentioned to you I I was at Swarthmore College and I was the first computer criminal that they had at at, you know, Swarthmore College. They didn't kick me out or anything. They just they they kicked me off the system for a semester and then they hired. But so yeah, which is often the way it works. A lot of the people who work for. Cyber security firms were originally hackers and then they they were good at it. So the cyber security firm hires them so they can go in and test things because they're the ones who are creative end and energetic enough to do. Whatever it takes to get into a system, correct? The second issue that we want to talk about is data hash and data hashing. This is the idea both data hashing, encryption are the general idea. That if we. Passwords are the example. We'll see them with. Password change is not the only example, it's where instead of storing data that might be retrieved. We store a hash value of the data and hash at that point is a an algorithm that goes through and looks at all the letters and come and does some transformation on them to come up with a single number which represents the entire. All the data in whatever The thing is. Now, there's no way that 132 bit number. Can represent all the data. Or what I can do is it is there are enough 32 bit. Numbers out there that if you run the same hash through something that is even tiny bit different, it will come up with a completely different number. There may be there are in fact many different things that would all result in the same number, but there are so many combinations that your chances of running into something else that will generate the same number are very, very low. Right. Give you one example. By the way 'cause I I find it reading that they do this. MD5 is one of these hash algorithms and the. It was so convenient as a way to track two to manage. Unique files that databases like Mongo DB. When they store a file, when you'll be actually say I want to save a file into my mongo DB database. The name that it gives is not the filename, they save the file name as a separate field. It is the MD5 hash. And the reason they do it is because that way, if there are, you know it's 100 different people. All have that same file. They often only store one of and then they keep a reference to where the other things are. This is also how I know that iTunes used to do it, and I'm really sure. I think Amazon uses it for music and video on things like that when it may look like you all have, you know you have your own version of the music. Uploaded to something like, I think iTunes is ancient, probably now or something. I I don't. I'm not modern in anyway, so I apologize if I'm using things that are no longer. You know, I I'm not talking about my space, but I'm I'm trying to stay at least that close. But. Well, the way iTunes would do it is you would upload your music so that you could have even things that you'd bought or that you recorded yourself or whatever else, and it would do an MD5 hash of the binary version of your song. And then just see does it already have that somewhere in this database? And if it does, there might be 100,000 people who have copies of that song. Enemy only be one of them that that hash maybe the like. There may only be one version actually stored on iTunes. It knows how to get it now, so hashing is used in lots of different ways, but. The advantage that we got for data security. Is that? Instead of storing the content, we store the hash. And if someone else you know password, this is the classic case. We have a password. Then instead of storing the password which would be there for somebody breaks in, they have your password. We saw the hash that was generated from your house word and from this salt value which is show assault value. Yeah, these are these are what the hash function may come up with, right? It shows it by the way as a digest insert it was all letters, it's not. That's just hexadecimal representation of a a 32 bit number usually. So it's just an int. It's just a very, very complicated end. I mean very very random like I guess so. OK, So what they're showing here, though, is that even very slight changes? Will lead to completely different results over here. So then what we have is we have your password, you enter your password. And it will do this and come up with a a hash. So all we if we store the hash, if you later come on, come in and try to log in. We generate the hash again. And all we have to do is compare captions. So when they say when when systems say we can't. We can't tell you your password. They literally don't store it. They don't have it. They know how to get this hash value, but if they don't store something now, this is again a big picture data security issue. Nobody can steal the thing that you're not saving. It's not in your system. Nobody can steal it. And the flip side of that is that anything you store anything at all that you store? Can get stolen. I part of the big change over the years has been is it OK to ask people for people's Social Security numbers? And many, many, many, many systems used to always. You used to ask for, used to use them. The problem is that that it is a. Think of it like the the primary cheat that we talked about. Is a value that is. It is a natural primary key in the sense that it is it is not auto generated by the database or something like that. But it's very, very hard to change. So if somebody leaks your Social Security number, you can't just go get another Social Security number. If someone leaves your your credit card number. You know they will generate a new credit card for you and send it out to you, and you have a new credit card. But what do you do once your Social Security number is out there on the web? You know, it's kind of out there. It's. I'm sure there is some hot process. I had interesting experience with this and my years ago when computers were not as sophisticated as they are now. My mother was a teacher. And she. Taught. And was he, you know, was paying Social Security? For several years, like six or seven years. The school screwed up and gave her and didn't put her Social Security number in. And she fought for years to try to get back to six or seven years worth of Social Security and never get it back. Right. The wrong number has been put in. They had no way of rectifying at that point. It was like, Oh well, you lost all that credit. Unfortunately, she worked for more years that it didn't matter that much. But it was like, you know, that that hurts. It's not an easy number to change. It's not an easy number to fix it, so it's just it's not what you want to be using casually as as information you store in your system. So from that point of view only store and this is a hard thing for companies to to do because companies love data. Companies want to absorb every bit of data that they can. And that is because in this day of big day big data, we we you know, we use things to figure out how we're going to advertise to people and how we're going to do this that the other. Leaving that aside, 'cause that is a bigger topic and it's for other people to worry about from your point of view. Only stored data that you. That. Then you will either protect as fiercely as possible. And that you actually need. Don't store data that you don't need 'cause. It could get stolen and you were less likely to fiercely protect anything which is which you don't desperately need. If you really, really need it, you will. You will protect it carefully. If you don't really need it, you're more likely to be LAX about it and then things happen. That was then back to the SQL injection. It feels like OK, how many? How many different places do I have to worry about it? Well, guess what? All over the place. Everywhere you have to worry about this, and that's hard worrying about something. Everywhere is hard, and companies being companies, Facebook, you know, Amazon. Jet hacked. Every every month. And they have experts at it. So you just do the best you can, right? Anyway, data hashing is the way of of one way storing information so that you have information you want to check whether it matches something else you you've done your hash, you you. You can just compare the hashes. Encryption happens when. You need the data to be out there. You need to be able to send it to people in another company or something, but you don't. Trying to figure out whether there's something I need to show you on this. I don't think so. OK. Be done with that. Alright, encryption is all about the idea that I can take data and I can make it unreadable by someone else. And. Not going in the details. You should do the reading on this. There's there are two kinds symmetric, which is where I cake. I have a password. You know my password. I send you something that is encrypted with that password you put in the password. You get the information. That's how zip files were. You have an encrypted zip file. You tell the person what you're going to use. Right there is. Symmetric is fine for lots of uses, but it is it always has the problem that both parties have to know what that password is. And that. Anybody who knows that password can get in? And that's where I'm gonna go back to hashing for a moment. One of the problems with hashing. Is. For people who use hash passwords. Right. Imagine that I my dogs name is Rover. Alright. And I live at, you know, 2021. Whatever, Rd. So I use Rover 2021 as my password. Patrick, not good password, but a bad password. Kept capital letters GO lowercase, letters got numbers and you know, not terrible, right? Probably take at least two minutes for a supercomputer to crash, you know. But Anna, 20 seconds Max, you know? But but it's there. It's OK. Here's the problem. If I use that on two websites or three websites or all the websites, right? Some of them are going to keep it. Some of them gonna be less secure than others. I I belong to something called writing.com. OK, whole bunch of writers, they go out and they write stuff on there. I would imagine that because of security, doesn't matter very much their security procedures with passwords are relatively. Less than perfect. Right. So they store, they do the right thing, they don't store my password, they store it as you know, whatever this wild, you know, cryptographic hashes. And they get hacked, right and. Somebody now has that hash associated with my name. And I tend to use the same username in a lot of different places, so I have my username and my email address 'cause. Those are usually the things that they're gonna store. Sometimes my phone number. Then another database, another website gets hacked or hacked. And that one gets hacked in such a way that they get not only the hash, but the password. Right. OK. Now they have the password for the first sight, but even if they don't, let's say they've never get the password. Now the place that accepts the. The hash can get hacked. Like my bank. Might have a much more secure system, but they might find a way to hack in so that they can push the end result of the hash value as opposed to in the password itself. They may never know my password. And they can get through subsystems. Probably not my bank, although my credit card has the worst security ever. You know, I I'm so tempted to just change around my credit card company because they do everything wrong. They they they send when they send bills out, they send the counts out they send. My full credit card number as part of the bill. Which you shouldn't do, you know. Their website doesn't allow two factor authentication, which why? Will get two factor authentication. Anyway. Encryption is the idea that I'm going to take this data that is important and I'm going to lock it up somehow and not give the other person a way to get access to it. Symmetric encryption is just where we both share a piece of information, like a password. Asymmetric is where we have this and you don't need to know a lot about it, but it's a lot of fun to write and read about it here. That kind of person, and you know who you are. If you. Each metric says there's a public key in a private key. I can send that public key to anybody I want. I can send it through email and it's not something I can publish. I probably don't want to publish it, but I could publish it. Someone can then use that to run and it basically I got the public key and I found the algorithm someone can can use that to encrypt it, but it needs the private key to unencrypt it. It's a one way encryption with. Public key. And one way decryption for the private key. How they do that? Is it pretty much magic? But you know, that's what it is. Encryption algorithms are only as good as they are. They used to be just, you know, used to have relatively simple encryption algorithm algorithms and super computers get faster and faster, able to breakthrough more and more. They have had to get. Bigger and longer and slower to calculate and everything else whole big topic. We don't need to deal with it a lot, but just so you know, when they keep telling you that I remember with my my Lotus notes password it used to be a 32 bit. Encryption and then it was 64 and then it was 256 and now I think it's 512, you know like make it bigger and longer and whatever else and because that's the only way that you can delete protect things in this day and age. OK. So that's that's all we that's a big picture of stuff that we need to talk about with data security. So let us now. Switch over to the lecture, find the lecture. And in the lecture folder. There is a database. And that database folder has your user manager SQL. So what you are going to do in first? If you are following along as you should as you should right click on databases. Create in PG admin. You should create a database now. I made light before about in the database in the. We assignment that database design about the name of your file. What I strongly suggest and what has really the norm here is call it the same name as your database, because then you can look at the the SQL file and you could say oh that's the new my database. I don't have to be telling you you should use this name even though it's that whatever you know. So in this case it should be user Dash manager. You do actually have to call it that because we are going to be connecting to it with Java and therefore we need to know what it's, you know, how to get to it. So when you were right click on that you had created your hoops. You have created your database now. Things I don't like about this. OK, create you've done your database. You've said user manager. I've already done it, so I'm not going to do that. OK, you are going to go in here and end user manager. You're going to double click on it. Right click on it, go to query manager and then you're going to open up that user dash manager dot SQL file which is under the lecture under big. You are now experts at DDL, so you will look at this and you go well. Wait a minute, I'm not putting any data in it. And that's right, this is going to do nothing but create a very simple table. You're gonna run the thing, so you create the very simple table. And we're gonna go over here. We're gonna go down to tables. We're gonna refresh to say. My table there. Users. A user's table is here ID, username, password and salt. The salt you should know if you did the reading mode again. Just reiterate this one time. Salt is an extra thing that you add on. Two. Whatever the password that someone gives. To to make the hash much harder to keep track of. Like when I said that thing where I use my whatever Rover 2021. I'm gonna use that if I am just encrypting that. I'm sorry. It just hashing that. And I use it on two different websites where two different users use the magical password you know as their password, which people still do in stannage. They were going to get the same hashes. We don't want the same hashes. So what gets happens is that we generate a random ish. Everything is random ish and the real world he random ish salt which is a an extra string to add the beginning or end of our password. So and it is stored. And save for each individual user. What it means is that my Rover 2021 using exactly the same algorithm is going to have a different cryptographic hash in one website than another. I can no longer look at them and say or even in the same website I can look at two different users have the same password. We're going to look at it next is. And see what their their hash value is and see whether or not it is the same. Yes, I already done this. OK, so now I'm going to go for you guys. When you go to the lecture. Where are you going? I don't think so. OK. And I'm going to go to my pond XML and open it up. I will say hello watching. So here is going to be my. Data security. Should be used to. It has maintained tests, although I think test is actually just placeholders. Yeah, we're just doing that so that we have the places where we live with this stuff. I'm sorry I've lost my cursor. Well, there you go. OK. What's more important, though, for this one, we will get to you know, all of tomorrow. We will get to actually how we we we do the tests like we're not talking about months out here tomorrow we're going to talk about doing these tests but for today you don't have to worry about it. So we're going to instead look at. Maine. And you will get used to this pattern now where there are. You've got the general package of. Just calm down tech elevator. And then we'll have different sub packages which are like for the individual items. So this is like we used to. There's a user. That user is going to be in Comm dot tech elevator dot model. The model is where we put our, you know, plain old Java files that keep track of our. The. Pieces of information for our entity and the getters and setters. And then in our DNA, oh, like we did yesterday, we will have our user DNA. Oh, this is very simple. There's only one table. So there's one user and that is the interface and it's just going to say, what can we do now? If you're wondering whether all these great big huge comments we talked about it before a little bit, but just as a reminder, this is now the comments that will show up when I'm when I when I use this value in Intellij is trying to be super smart. It will use the javadoc information. Which is stored here. Javadoc will give the description. It will give the parameters and the return value. So that when I am in here my. And somewhere I'm gonna see if I can show it somewhere. If I go to. OK. Yeah. When I when I mouse over this and I have this kind of information, generate random salt is going to say where is it stored? It is in com dot tech dot security which is the package. In the password hasher. And it's got a description which is generating new random salt and it's got it has no parameters but it says what it returns. You guys should be. Starting to do these more. As you get into projects where it gets harder and harder to know what everything does, doing the job once and putting it here means that later on when you're trying to use your stuff, you'll know what parameters mean and what they're there for you. You're going to try to use the best names you can, but you're still going to. This is a useful thing to get into so. I think from now on you will start seeing examples that tend to use more of it when we're generating for you so that you have that information when you're trying to create it. In any case, there's the user DL, which is just the the. Generic interface. It says we need to do this kind of stuff. So what do we have here? We have save user. We have his username and password ballot. And we have get all the users. From a data security point, well, from a data access point of view. When you look at this, you should say OK, these are the things we can do with the user. These are the methods that we can do with the user. From a data security point of view, you could have should look at these and say these are the vulnerable spots. Because it's a DAO. Using JDBC or when you get into the JDBC class list, this is where the vulnerabilities happen. You should check each of these to make sure they're not vulnerable. Guess what? I'm betting that something is not bold, that something is wrong because you know we're doing a lesson on data security. You know, spoiler alert. So we go into here and again, putting your DA, oh learning about data access hat on. This is where we're going to implement the save user. That. This stuff just happens. Yeah, this is there now is actually a. Constructer here. But we don't have to worry about it, so we can handle it for us. These are the vulnerabilities because these are the things that are going to call SQL so so you've got these saved user. You've got the is username and password valid. And you've got to get all users and each one of those. Somehow it's going to now notice, by the way, they didn't even bother to do a map. Because they were only doing it in one place, which I had through here. This is where you would normally put a nap. Now map row two user and and and isolate this information out they just said hey they only have, you know a little bit of information. Let's go ahead. Do it in life. So let us go out now. We should. Let's go back to our yes. We're gonna say let's create a new. When I say OK select star from user. What are we gonna see here? Doodily. Hi. So host grass and treat, did it really have a value for it or is that just? Oh. Users. It went ahead and tried to get things from actually. I'll make a point about that. That is. This dish is shown in the video, but it's just something worth noting. And this is where your vulnerability can also come in. My table is called users. Shouldn't be, but it is. Shouldn't be because they should be singular. You know that whole idea? Forget that. I don't care. Whatever your company will do, whatever it does, do whatever they do. I don't. When I say select star from users it is using this table. What happened when I said select star from user? It gave me a table that is not in here. That's because it's the Postgres table. Of all the users. In my system. What does that mean? Alright, that means first of all that there is now that's just usernames. That's not terribly useful, but that means I have access to the system tables which are probably in and again. I'm no expert in this, but in here. Alright. So I'm gonna look at my tables now, alright? God knows what I have in here. I have random stuff you should all recognize this. You know why you should recognize it? It's what you've all done. I was in the Postgres database and I went ahead and added the, you know, the campground stuff where I added something else. 'cause I do the same stuff you guys do, which is that I I casually am in the wrong database and I import a whole bunch of stuff into it. And then I have, you know, parks in my Postgres system database. Whatever. Leaving aside the minor embarrassment of that. You presumably have things in here which may or may not be system stuff. So what is this saying now? That's that really all appears, ma'am, I really in the Postgres right go into the other one now. I really am in Postgres. God knows what I did somewhere in here. There's it may not be in here. Maybe it maybe in a hidden one there's a user. People who know Postgres know that they can get a list of all the users if they can manage to fool you into calling us. So we should bear that in mind that that is a possible vulnerability that I didn't know. But. That somebody else knew? Because they know more about Postgres, right? So you may think that what they can do in a query is only deal with the things that you directly are having your database and maybe not. Just an interesting point in any case. What we actually meant to run is this. And he's got no data. I. What does it show us? Theoretically, if we if we do that query, what do we see down there? You see columns you see columns with names that tell us what those things are. So someone querying our database might be able to find out what values are in there. If they can get through to the point where they can do an injection and and read the columns, we may not want them to know what's in there, but this gives them information that they can then use as they try to query it. In any case, that's a different point, but we'll get there. So let's go over to our. I lecture let's say, OK, so we have. Let's look briefly at our security 'cause. That's the last of our sub packages. Security has this thing called past password hasher. Hey. When I was talking to you about how long my notes password was, I it you had this key length? She left is how much? How many bits of data I'm going to store. And the longer it is, the harder it is to crack in general. If you use the word password, it's easy to crack, no matter what you do. Or 12345 which also seems to be extremely popular. Work factor is 100,000. OK, work factor is this thing that says how many? Essentially, I'm going to hash my thing and then I'm going to hash it again. I'm gonna hash the hash value. And I'm gonna keep hashing the damn thing over and over again. That there's there's a little more to it and that's my simplified version of it, but what it really means is it makes it harder and harder for someone to track down what the value is. I will tell you that very quickly, as you get more and more secure. Easiest way for someone to get you your username password is to send an email which looks like your boss saying I need to get into your system. Can you give me your username and password? And people will go ahead and send their username and password to their boss because you know well, you know, you know what? Take off the boss so. I I will warn you ahead of time, the chances are extremely unlikely that your boss will ever do that. And if so, you should be working for a different boss. Nobody should be asking for username and password. Your manager can already get into all your stuff. There's like you don't need to do that. OK, I will say it. A tech elevator. Some of you may be aware, Anthony and and embarrassed to admit, I can't even remember his last name. He used first names only here. But Anthony is the CEO of Tech Keller. Right. And one of the two founders. And. All of us instructors and occasionally use students. We'll get emails and things from Anthony asking for various things. There's not because Anthony is testing your, your prowess is status, security people. This is because other spammers will know that the person you're most likely did not argue with is the CEO of your company. And so they will. You know, that's if social engineering is going to say they're going to go out and make look up the publicly available name of your, we just we we we refer to these as. I can't even say it. Basically spamming at the meets Pamplin. Yeah, whatever it is, we we have a way that we refer to it, but it is just all the random spam we get trying to get. Suppose I don't. I think that if Anthony actually sends any of us in email, we're unlikely to respond no matter what he asks us to matter what? That's why we use Slack so much because we will never respond to an email from Anthony. All got scam, you know, asking us to do things so. OK, anyway, so this is gonna go do it stuff. I'm just seeing if we can get to the place where. You get to things where you start seeing like, OK, this is the, you know, Pbe chi spec thing that calls or whatever. What this says is you are never gonna roll your own. You know, hash on you should never. Highly sophisticated people know what they're doing, and there are libraries to do all the things you take a trusted library you have it do whatever random stuff it's gonna do. And then you use that right. Only downside to this big picture wise and totally irrelevant to this conversation, except that I think find it humorous is that because of this, if somebody finds a vulnerability and this just happened in Java recently, if somebody finds a vulnerability in one of these core things. And all of a sudden, everybody is vulnerable to it because everyone is using the same libraries. Still, you are more likely to get hacked and and to lose things if you try to roll your own that if you use one of the standard ones. Now let let you be vulnerable when everybody else is also. Your boss is never gonna fire you because you used a well respected. Generally known algorithm, even if it causes trouble. They are certainly going to fire you if you roll your own and then it got broken. So you know, partly just for CIA. Purposes use a well respected hashing algorithm, whatever else, and and if you ever get to that position where you're doing anything like this at all, which many of you won't. You should understand the concept. OK, so basic thing is it goes, it does it's magic. It comes up with a hash. We don't much share about that. We are on the user user D AO and JDBC user deo end things. That is our role. So let's go try to run. Actually, you know what? Let us take a break and we'll come back at 10:05 and we will try running this and stepping through it, OK. Alright. Bear bear. Now, now, gonna we're gonna work with work with I. I will. I will apologize that this is this is one of one of these things, things things, some, some instrument instructor doctors who were were more more. Ambitious. Just that I that I am. And and this this includes cludes, David, David over and over and see the dot net net class. Had set up that up it how? How quickly the vailable. Status database base. So so you you can all interact, interact with that with the same name, date. And if and if I even had even have. So I have so much going on in my own in my life I life, I would I would have just done done that. And I and I just, I just didn't have that. The band bandwidth width to to do that that this this year here. So you will have will have to. You will have will have to be to be your own own. Test test subjects objects text it is. It is kind of fun. When you, when you all all have a have a shared data database based euro year old cream user users there's there popping, hopping up and oping you see there see things things happening happening and not just just you so you so you have you have to imagine that that scenario. For this years here so. Alright, so so when when I run run and you guys guys can try try it 2/2 and I right right click on user manager CLI. Will you be used to that as our as our our security guarding the point point? Let's let's put it very, very briefly actually added. Add it just to see what what it's gonna it's gonna do new. There's a there's a main mean. I really don't really don't like when think when they do they do this, they really should be create created in create the app the application and separate. That's neither. Neither. You're gonna enter there. There's a there's a mean mean it does. Does this list this stop stuff which which is? The hard hard work that I did. I wanna find focus. Focus on for a moment. Moment. And and. We have to set that that URL rail for the day of the day. This works first. And then that's that's why it was so, so important, important that you that you really, really do you call your day your database user data Dash manager. That that is. How is how you gonna be gonna find find it breaking right here and then we ended. We had 77 that we use music pass for this word. So this is this is our. Is our connection to to that coast Rest rest database states. And and stuff. Stuff you don't. You don't need to worry about that stuff in here. I and I and I mostly point the point that this help is help because because this this is the even even though I would, I wouldn't put it, put it in a nest nest. File this. This is the the the the pattern that you will often see in these. Please command me online my program. And we will include some of some of the things, some, some of you did this this way didn't in in your your capstone zones. So this this is where where you generally generally printed reading reading of you know some to some description. Then. And you have your rear loop and you you go through and do and do you do your stuff? And then often then, there is a a like in this and this. When they don't, they don't really, really have it. Have it. But often there is. There is you, you pretreat print the print that exit kind of kind of thing. It says, you know, thank you. Thank you for participating or whatever else. That's that's common pattern pattern and have have peer peer is to is to have have. And then initialize was initiation. Same thing in beginning giving of row. And then and then I'll loop that just it just goes, goes through and through and dazzle yourself first, long long as you do it, you do it, but and then some clothes that was out the thing thing. Ian Benberry many machine you could have just had decided in this this run run. Print print to do the load. Here here. Because it hasn't happened superscript before the loop loop. When, when, after, after, but that would, that would be weird. Do it. Do it here, here. What I I think I think several several of you love you did did, which is just a. It it. Perfectly appropriate way 2/2 is is that when you when you go out, go out to to create create. Be the constructor call calls. Also Momo does does the thing that that that that start starts it up set up so that it is soon as soon as you have. On on Jack jacked it is, it is it is. It is an initial initialized. Both both are valid. In which features? What we're doing is doing here is we're doing, I'm doing our menu, asking, asking, I think things and and our. Comedians that we are, we are running. Are this add new user, show users, logging user etc. I I. So if we if we look at at logging in users are 'cause, that's the one the 1099 chair chair the most, most spelled Bell right now. It's going. It's gonna ask from my barriers. PHP is zoom zoom information. So I would say I want to point after you after you guys that because she came up, come up some dumb before it is. System. Blush. Or anything, anything slash. When you're out, put, put. Big Bang. If you flush flush it. It actually right writes all the reps to the to the content and maybe maybe buffer offered it up top. Hey. Input output puts this distance. Spend spend a lot of lot of energy. Trying and be efficient. Not fast fish fishing. And so they don't. Don't do a lot of writing to disk. I think I think once once told you about this, I had this had this thing thing where they can do can do thousand process that thousand dozens of. I've documents. You know every second. And if you put you put 11 print print line line in there then it goes it goes out to the to the screen screen. That drops down into to, you know, you know. Hundreds of speed. I always was very expensive. And because I know and I and I I was, I was input output I it was it was very expensive. So when you, when you're going to read feed from the discourse for right, right to the disk. If it does, does something probably called buffer it free and which which is C thieves it up? I I may say say when I when I do assist assistant print Line III normally see that do that. Because as I'm interacting direct, who can't store. But if I if I were printing printing out to a file I may I may stay pretty printed to that to that if I if I or any any mean I mean not actually actually haven't gone gone under that under that file if if I'm look I'm looking at at file file for someone somewhere else it's not it's not gonna show show yet yet. Bring up If you want, you want to force force that that to go go into the to the. Play play. You would either either close the file which has to put it on, or if you don't wanna go, wanna close hoses. You need feed file. Flush flush. So the general term flush. It's it's not. Not like toilet toilet, it's it's it is like is like taking the stuff out of the buffer, right? Put it. Put it. You know. You know where we go, we go. Wait, wait. Maybe maybe a little. Like the white toilet, you know? You know? Guess this is stuff out there. Alright, alright. So then, so then we're gonna. We're gonna call. This is this is where we where we start start getting into everything we recorded call are you used? Now this this is a first first to do. It is just username and password valid. I. So so let let's go ahead and just just first of our short, just just create create, we're actually running running right down, down what we what we like to do. I am welcome a new user. That's just make me. Right, right. So add. And we'll say say, you use your name, your name, name. Billy. Billy. Bob. Hi. Hi. WW Baba has. I'm gonna. I'm gonna. OK, great. First, instance security every day. You should. You should never, never. He allowing it to print print this way this way in a real interface as much as much as possible. You should should never, never show the past rats word as someone who's done this type. Underwear wear. But this is this is really, really easy. Is Chef, chef something? Everything's password for field. Yeah, it will just will just put it. Put it they ask. Next list, gender, whatever. Whatever else. Even even that is considered somewhat what it is here. If considering someone, someone secure, sure, because you if you are letting someone know how how many, many better letters there's there are in in the in the past, that's word. Then they may be able to better gas gas what perhaps password is. When you type your typing. But but it is is last less it is. It is much more secure. You actually have to turn that turn that on but but. So anyway, anyway, I'm gonna use move my Rover to over twenty 2021 one because because you know, you know that's that's that's the. Thing and I came. I came up with stop my top of my head. I really, really have never, never used new set as a path. Last word. I don't I don't have. I I did. I did. I actually, we we had a dog Rover which. Yeah, yeah. OK, up. What was it really, really. My my dog comic. I I was living for college. I don't. I don't. I don't know. Alright, so so now now if we we. Watch it to show see if they. Only logged in users, users can see other the other users users. Alright, so I'm I'm I'm I'm just trying to figure figure out a little bit about about. What what I can do with you? Right. Let me let me add add another user. Just just have a least List 2. And we're gonna try to do. Oh. Right, right. And then also gonna gonna use for the moment moment row Grover. 2020-2021. Boston just just you might be might be able to get a guess so that we can we can have the hash value values what what not. Now, now before I do, try doing anything else. Let us let us go over here, here. We should be should be connected to the same same database. So we have, we have none. None of it is made is made restrictions say online you know you know people, people who can't, can't do through their their things because we're logged in then. So we can we can see the. Because we can see. See Billy Bob and getting Gander. We can see their their past faster risk. There's now this this is not is not there. Not there. Clear clear text caps word. This is the hash hash sign. Rover Rover 2020 21 you want as soon as I finish typing typing. And he went enter. It's going away away. It's thrown away way we don't. Don't store it at all at all. We don't. We don't pass the rest around. Now we can do nothing with me. With it. We we just. She say this? Super handy, dandy, dandy Andy. You know now? 33 two. Character basis in encoded in base 6064, encoded quoted strength, strength. I I see this. See this only be only because. When I say sixty 6044 uncoded coded and I'm not sure how sure how many you are, are you familiar Miller with that to that chart chart? If you if you want to want to take. Binary data. And represent it with characters. Then there is. There is a 8 and algorithm that just turns something nothing and they they call it, call it basically 6044 encoding coding because because literally literally if you if you have like like if you have if you have a base space 1010 number number. If it uses a dear, dear digit digit 003399. This is 616 nineteen number. It uses zero throw through 99 and also so eight through F. Right, right. What Hughes use a A bases 6064 Ford number number it uses uses all the level letters at the end of the alphabet plus plus some. OK, OK. And so you you get that. This kind of you know, plus the plus the number numbers, so you so you get to get this thing being in here that that is is always is in plain plain text reader readable but and can be can be easily converted, converted back back she will she will see baby basics 64. Or a lot lot in your in your program programming career careers. The reason the reason I I personally know that it that it just just being six, 6044 or 'cause it could there could be other other things. Is is that that? There are, there are certain certain things things where where if there are left or left over religious center neither needed it. It's it's going to going to put an equals equals equals equals at the end at the end. 44 shooting legally, so sitting legally rule it will it will feel like the end end of a base basically 44 number of numbers. So to get the right the right thing and with equal equal signs. Right, right. I mean, I have a quick question question. Sure, sure. Up so. So you know how sometimes when when you have you have to do undo a new password or. If, if if this is this was same name instead of the actual path past where where which those those two are the same pain password. That's word what they would they have different friend. Cash. Why is it? It's so good, so good telefile. If I use old sword again, again 'cause. Sometimes I have that I do that. I do that. I try to try to use all these old. Yeah, sure. And that's actually a really, really good question because and and any any it is is. I don't know 100% percent of the answer. I know that the part of the part of the answer answer. What they do is this. Think about it. If you use the same same name. Diffuse pass through the you used usually usually look at. Look at the past, you know 2025 or 5050 or whatever. They don't. They don't compare them. They may look at look at that hash. But in order to do that. If they say you use difference and salt salt, they have to have to store or longer the 5050 salt values. She should have salt salt value value in the salty, something that is just just it is is created. On on the fly and randomized. And you do. They say they saved faithful last 5th 50 of them. Run, run the hash through those 50 when you're trying your training machine to change your password. Losing, losing the salt, salt and if any of them match, match up this hash hash password password. They know they know you using a repeat. They don't. They don't have to know what to know. What is? It's they. They just have to have to know, know, know that when once they've had hash that \*\*\*\*. They got they got a match, so they will. They will have happy table people. It's really, really interesting. Interesting way of doing it. I mean let's say say you have. Check and make sure make sure it's not, it's not something done done with the last 2020 pound pass routes routes used. You're there. Then we would have a have a table. Which which would have. 2020 dirty difference insults just just a personal table table, not we're not one one by you by using. But I personally, for me, or meet a table. That that has hashtag. And and so sold out Guy. Add will just will just store, store and so so as I as I add another new one one I I was when I when I when I tried adding a new one. Generated assaults all. Which which is gonna need to do to do with book book first. First thing we will do is before I even even generate my my new hash hash is I will, I will say might be go and try try. Hash hashing it with this. This salt value value. And see if it matches this this past password. No, no salt salt with this this band and had had, assuming we had, we have a table people here which is which is my my previous these things. So it would, it would just just go through and through and and compare the hash. Does that does that actually keep track of the past passwords just as the hashtag? Because it does. It does that for that. For all that it can consult it can. It can say, oh, you know what, what you use that, that 15 team having passwords for the zigzag go. If it's not, it's not secure. Right, right. It want people, people not not to be to be repeating, repeating repeatedly using this in the same pain password password because because anytime you repeat it if if somebody somebody has has stolen it it then then. You know, at at that time. They would be would be able table to use it use it. Hi. Hi. Remember Remember that values actually are relatively easily available to anyone. It just makes make it hard, hard to crack crack because it it it's because the hashes but but. But but if you have a match hash and and the soul soul then then they need. They need you stole the past. That help help for me from that that cleared up moron moron. More or less less, yes, yes. Yeah, yeah. Quests. Alright, alright so. I now I now have had these things, things they have completely different hand hash, hash, hash words, words. Even though sustaining packing, passwords word. You do because they have completely different that soft tops. How did you come to come up with some salt care care if it happened in the Mets Mets security password hash orchard thing thing it came up came up with this with assault that I used, but I all I know, I know it was. Wasn't there conducting completely gibberish. Yes, yes, string strings of alpha alphanumeric American shared. Hey. Hey. Did you guys, you guys III was a history major, major align Latin American start studies? So I did a lot of bottle puzzle science last history, and one of the one of the things that they frequently when they like saying hey in history history is is that you know you know. The victors rewrite right history. St victors set the rules. Similarly, in the in the computer world. So, so much of early programming was done on in the United States states. If we if we got to have to set that the term terms and everything is. Every everything that it is this saved on the Internet. In in this this may make sure sure. Is is going to be done in ASCII. Which, which is just West of the 1st 2027 living care or hundred hundred and 2027 seven characters. Of the. That is. Upper and lower case. It's 830Z. And the digits. If some other country tree had to do this, we we would, we would, we would we we would have rules on how how these all show showed up. We may maybe those those rules and now everybody uses uswest ASCII as as their different default for things like. Basically 64 on code coding for the salt. All the all these kind of kind of stuff. Because we we did that and that's why you're not not gonna CC. Accents end on characters and fish and things like that. And there's there's no reason they can make couldn't, couldn't have set up Internet to use those servers. There's not. There's not even some special about that, right? If it happens to us, to workout, workout well, well, because it's it's, you know, you know. You can fit it all into into one. You know. Bye bye tight \*\*\*\*. They're even planning planning. Other things are things using the one bye bye. And and we just made us made the rules. So you have our users here. See here here. And I came over and I got program and I I'm running running. It's still still says up it's. So let's let's say we want we want to log in again. So we're gonna log in again. By the way, this is. Is he he, you know, like allow me lowercase questelles. When they just show it that that way I have to do it at that rule follower over that one hour. OK, so I'm gonna log, log in and and and I wanna use scandal. And I put it in. Wrong. Wrong passwords word. It's it says logging in is is not not now. See these strike. Try try again. I want to log log again. Log in. Then there's this game. Spell. Spell it right? Right. Runs to say they're over. 2020 to 2021. Well, welcome. Yeah, now. Now what it did at that point. Was it it? Created and hash. It it looked ticket candle and also user record over here here, here. Looked at the Gandalf. Look at the salt. Trudy. Trudy, gather other put. Put the salt before or after. After. Does does something? Nothing with it. And then it it generated. I said that that that hash match. Jack. That's what it did. Did did I will. I will say usernames are virtually always does any any same thing system. She's insensitive. So. So what I'm really do, if we look back at our code, is this just kinda gonna capitalize as it is? Save it capeline live at source. Just gonna compare them. It's just case insensitive. Stop something like like that. Right, right. In SQL terms, it means your means you're gonna do it. Do it on I I like. To me, you're comparing. Your. Yeah, yeah. Your your username and then this. You're gonna do undo it. And I like consist of align like because you want to be. You want one big kiss kiss. Cat. Cat. So now we should be able to show all use. Jelly belly. Bobby. Bobby. Right, right. Alright, right. Now is where we start start getting getting me list. Exit. Let's let's quit, right? So. Quick quick. Their First off I want 1.2 because well, this is really really really obvious from our current. Running running out of it. This is his new group for us during the coke over or or relatively new last two years, she did. When I run into it again and. So his guys Sam. No, I'm not. I'm still. Not logged in, so I'm gonna log again. OK, again. I'm gonna win again. And as Billy, Billy Bob. Time. How? How? Spell the thing? I think probably. OK. Save the row river. Well, welcome, Billy. Billy Bob. I'm gonna show you. And then there we are. I see this. This is there. I I didn't even. It's not not like like the the the. The very many many machine which restarts from scratch. Everything I microcystin stores says whether I have, I have two users or or 10,000 users all just this exists for next next year. My start struggle program. At Xbox, the whole advantage to. I databases is a persistent and store store. So go when I come in and I I can do, I can do that, that this is making that in that point because. We have we have not been new in doing that. Throw away. Let's let's try try quitting. Getting. Let's let's now run through deep debug bug and see what what is happening. Yep, Yep. So, so uh, can I ask, is is there reason why Billy, Billy Bob you use served erpt Gandalf and didn't and number to to come number to two. Is this this like a last in first list out kind of thing? As an album. Well, I wanna list. Oh, in, in, in there. And well, well in. Should say yeah, yeah. Question and I III would imagine. Well, well, OK how how would you make sure? Sure. That that it's stated in that in that order. To be honest, I said rather have the oldest list 11 from first. Well, if you have two users, but if you have you have 1000. Random border you're gonna have all kinds of times stamp, time stamp stamp would be the only thing I think I could think think until so. Right. So so my my, my guess is that what they do is they said that that. Look, look. Is that where we show? OK. Where where we do do this? It's it's just just doing it select like bye bye order. But what if you did watch? Say, OK, let let let's let's try this. Right, right in the beginning again, man. Because this. It's an interesting point that that you would think, think about that is it said if I go in and run, run my. Same thing, one more time and I said let's add in new new users. We we will. Not not do it. Do it up here. Here. You're even though we all do. K I'm gonna add, how do you use user? I'm gonna add a. Jenn. Jenny. So James Leo is with I actually has a more interesting seeing password which is you know. I don't someone like that. I would never be able to log in again. There's this. Gently. Oh yeah. OK, OK. So James Leo is now. Now I'm gonna add the end. Had. Don't know. Who is gonna hear your ears? Password word. Hi, dad. So now if I get in here, I've now added four one when I show. I have to log in. Log N stand handoff again, then. Yeah, I'm gonna do. Actually, that was an interesting to me. I I did get banned off that I thought I typed space out after a group. One interesting thing thing thing. Your system will, will they accept that that is your username? Does does trim trim the spaces beginning? Sometimes this system gives you or do it sometime down down, but it doesn't just take you to get out there or how much. From security pointing one give you the only reasons I've mentioned in that that is some, sometimes people will. Yes. Yeah, yeah. Because if it if you you put it in this in this page it doesn't find it yet. And does find it without without the space. What that says is there there probably not doing much massage ending or validation of your users. It's same thing. I mean uppercase some lowercase, sort of you you're you're trying find. Find. Find out. You're testing the the. What does the system do without without being able to see what it does, we can actually code. You user wouldn't wouldn't be able to. So I can. Show all users using again. So now it's Billy. Billy Bob. Jenny, it's just testing him. Whatever. Or I I've had. Sometimes. What if I didn't want that? But if I said, you know that that, that. It's going to be a terrible water interface for stricter getting along longer, so I might say hey. But I just let it go and and and changing jar thing and say say what we need to do. We have very great get all users. How how would I fix this this so so that it didn't? I was just saying in tech elevator terms. Different than slightly different than non non tech tech. Cal Richard, I would just say. Bye bye. Is your name name. Alright. That way they just said she was going to return it. In that. So now when I running again, yeah. I log in and as. Show your users. Now, now it's gonna show them in the mail now that quarter by five first. That member, remember that you are the your, your, your. Or your sister. You you can just say. I I think think it should show up in this order. You could show send me start with you know. You know, Jenny, I like you. Could you are the one who gets to just decide. And you have a lot of control and it's all about your your CC people still say this. This is why we do this in SQL statements first and then then come, come and then start doing doing this. This is just. You should be familiar now. Now you can do things with. If you have had an espresso seeded table. And these were in some somehow groups some other way you you could sort by something wasn't even even table able by joining another table for or anything. All kinds of things. You can then do. Do you have all of the cheap abilities of SQL in in here? That said, we're here are our vulnerability weirdo we add in in values that we might be able to put for around. Just just from from from just us from running it here here. Like like we we can try certain things. It's like like do we get an error to get us to? Sing, sing, sing, sing. we got

Now But you know or whatever value put in here here is is in my main menu. I'm not gonna go to Boehner. Boehner abilities come about. Something to do with doing something? What about when I the the the two two places one one place when I when I show we use similarly I and I'm not having any at all. If I gave it the ability to. Limited to, you know, show by first you know by first letter. I would. I would have ulnar nerve, Illinois. But I'm I'm not that actually. quitting quitting in any information in in in in in in in in the

Where do they use that? Uses the password as part of the computer. Had the compute hash. If the compute hash is not going to the database, it is not a vulnerability. So when we look at logging in the user, it's the username that's a vulnerability, not the password. But it could be either one in there there times. That's why I'm saying it's it's sort of a alright, so the simplest of all things is what what happens when we get through a single phone. Will this break our thing? And if we can tell that it breaks it, how does it break? So we'll go ahead and say we're gonna try to create a user named single quote and we're going to get a password of Rover 2021 'cause we don't really care. What do you say? It went ahead and added a user with ID 6. Wow. That's perhaps risk. Like how is he gonna deal with that when that is used in a query? And now we we have maybe added a vulnerability. Into our into our queries. It didn't crash, it didn't break alright. But now when we show all users. Deserves. Let's say we want to log in as handoff again. OK. We are gonna say I don't care about that. That's fine. We say show all users. Easy enough to do, but let's run through our. Everywhere that runs in debug. I'll try to do it this way. You guys are all probably better at. Doing this kind of thing that I am at this point because. Too many different things. OK, so we're going to try it again. We're gonna add a new user. We've got up add. We've got a break point here. We're going to try to give it a user name of. K. OK, well, it did before here. So what is our problem? Paired statement callback SQL insert into. just doesn't feel like it is oh this is logging in i'm looking at the wrong thing Which is why it's not stopping from. But yes, you are right. So I am trying to add a new user but I am actually stopping on the wrong thing. So let's let's stop here and we'll say the user name is this. I'm gonna step over, and it's gonna wait for me to to give input here, which I'm going to do my 1234. So now I am here. So now I am going to user V aossey views. Right. So I step in to save user. What is the save user going to do? It is going to call the the generate random assault 'cause it's a new user so they want to create a brand new salt value which is what you should do every time. OK, I'm going to step over that 'cause I don't care. Kind of hash my thing, which similarly don't share, although sit, wait and wait and wait as it's hashing. Alright, I'm back here and I will step over that. So now what do I have? I have my. I have my. Show my debugger. OK, I have my username which is a single quote. I have my password 1234. I have my salt. I have my hash password. I'm all set and now I am trying to call this thing right here. And it is. Saying insert into whatever. And it goes and it tries to insert and what it's going to run into is. The let's see that username is going to. Make it think that it's starting a brand new query. But he can't start the brand new query. You know, like if I the the username is supposed to be a value so it's going to have a single quote and it's going to try to use it. This should not have been allowed in the first place. So even though in this code. And in this code I am parameterized but the value that's in my table is bad. In my table should not have stored it that way, but it should have sorted as is a single quote single. But that's an escape And it stored a single quote, so then it breaks this query and I crash. Thinking that's what's happening, but let me make sure 'cause we'll see maybe. I step over this again. It's also possible that I'm wrong about where it's going, OK. Crasher. Yeah, it did crash. I just couldn't see it because it wasn't like that. OK, so all of a sudden there I have a problem. Let me go ahead and now get rid of this user for a moment, because it's going to be hard to even clean it up from the other side. There's no way to believe it's so let's say delete. From. From. User users. Where? ID. Equals. Sex. Alright, so I managed to make it crash by by doing that and now when we try to run this. OK, so we now don't have that, so make sure we're back to a stable place. Let's try doing something more interesting. Now. We we know that there's a problem in the is username or password. Uh one. We know there's a problem there. Line 68, I think, Elliott said. So here's our problem. We know there's a problem here. So let's try to do something fancier. Let's try saying what if? I run this again. I'm gonna add new user. We know that single quote is going to do something in here. So let's take for a moment. This string right here. And let's go over to. Here and this is the way a hacker thinks, even if they don't like, they may know enough to do it on their own. But this is what they they do. I'm going to go ahead and create something and try to make it fail. OK, so. I know that it's gonna put my value in here. OK, so that what that means is because I'm generating the full query. Is that it's going to look like? It's gonna look like that. OK, if I do a single quote. What else can I put this? Is this this right? Here is what's on the outside. There's something that's going to go on the inside. What can I put in here that will end this? And create a new query and keep going. Well, First off, I probably want just something. OK. Actually I probably want something that is really, really unlikely. Alright, because I don't want it to fail because there already is a user that name. And then I want to put in my End Quote to get rid of it. Debbie, I'm sorry, PG admin is gonna make it very hard for me to do this 'cause. It keeps wanting to create the quotes for me, so I apologize. I like trying to fight PG admin which is trying to to help me here, but OK so I know that I. If I do that, I should have ended this. Now what if I put a; Here? That should have ended this select statement. Now I want to say what else? What do I want? Alright. So I might say. I wanna say. Insert. Let's get fancy here. I want to insert something into my users table, so why? But I wanna do something or I wanna. Maybe even, yeah, let's say insert into. I don't wanna think I didn't tell you guys, but it's true. Is that my insert does not require. Insert into does not require that you specify the columns. You can say I already know that. So I could shorten this, but let's go ahead and do it. We're gonna say user name. And we're going to say. K Now use. OK, I will say I don't know, Fred. Yeah, I still have a trailing single post there. So I'm going to put a; and then I'm going to put. Dash, dash. Because that's a comment. So that will then get rid of the rest of anything that is in there. Right. This is what I am gonna try to do. We'll see if it works, but where I'm gonna try to do this well, where does all this start? It starts right here. Ends right here. I don't want to do the quotes on the outside. The quotes would get added. OK, so if I go back to my thing, knowing what I am trying to do. And I put in my username. I'm now gonna put in. Let's see if it'll let me paste or. Whether I'm gonna be having to try to type this, let's just control V. Alright folks, long and ugly. But this is how SQL injections happen. So I do this. And then I get to hear and I go ahead and step over. I wanna step through and see what it's doing. So get back to my console. I wanna password. I'm gonna use my Rover 2021. Now I'm going to step over it again. So we have to hit enter, OK. Now I am here now. I'm going into my saved user. So I'm going to step into that. I'm going to. Step over, step over and remember the password that has got in here. Is, UM. The password is just for over 2021. The real user that is kind of trying to do is ZXC. But hopefully it will also do this. Fred. George. Salt. That's what I think is going to happen. I don't know. That I am coming back with is this insert into returning. Alright. OK I have a. Check it here. This may not work 'cause I may be looking at the wrong one I maybe I. Maybe it's the login where I have to do this, but let's go ahead and try this and see what happens. I'm going to step over 'cause instructive. Alright, step over. I'm gonna do this. And I am going to come back with a new user. I have an Ivy and the username is all that stuff. Maybe. OK. OK. Now let's look here. So we're going to try to see what we're doing here. We're going to select this. And we're going to see. Yes, it really did let me create this entire thing in there. Well, it's kind of dangerous. Now I have my SQL injection actually inserted into a column in my table. So anytime anybody does something with this, it may insert this person. Alright, it's not even just me anymore. It's like somebody else doing something innocent in here who does their select star from user might get that thing. OK, but let's let's step through our logic again. Alright, so I'm gonna return my new user. OK, shades on, I got my username. Stale. So what's gonna happen when I try to order? First of all, it showed it here. What's going to happen when I try to show all users now? Maybe it'll insert this, I don't know. Actually, you know why this is going to work. By the way, I forgot to say insert into users, so this is gonna fail with a mistake as opposed to what I will fix that in a second. But OK, so let's go ahead and see still shoot me. I can't do that, so let's let's log in and we'll try our hold on a second. Insert into users. Alright. We're going to try see if we can get this to work. Let's log in. And we're going to log in. We're gonna do this a little differently. Actually, we're not going to log in as this 'cause. We're actually going to try to log in. Let's say Gandalf. Just where you were there in log in. We want to use a real valid one there and see if we can get that to take first. So we're going to log in as Gandalf. We're missing the second thing in the comment. The second one, at the end of the line, I think you want, we want the double dash here, OK. Do that where I said password is Rover so we are putting in the appropriate stuff for for Gandalf. It is going to get to its username and password. But. Wow, that still crashed wildly, but I'm a little afraid it may have crashed because of the other one that was in there. I'm not really sure. OK. Let me see what we have over here. We'll try it again if it. If I just made a mistake there. OK. Let's get rid of it, added Fred George Salt. He crashed, but it also added the value that I wanted. Yes, the crash is happening because it can't thing about unique keys or something. Is that because it can't have duplicates in this table? Is that where the crash is coming from? Probably, although let me see what the crash said because it said insert into Rover exception thread whatever SQL select I'm gonna see if it shows me the SQL statement. OK, select star from users where upper username equals Gandalf. Right. And then. Damaged by insert. Multiple result sets were returned by the query. So whenever he got a result from this and a result from this and it couldn't handle it if I actually had better error handling, I would have not crashed, but I would have still done my injection. OK. So everyone see that even though I crashed and I'm OK 'cause remember I'm running the program, I don't care where the right crash, it's just a front end crash. All I want I can this trial and error thing is exactly how well hackers you know a lot more don't need you as much trial and error. But if you're in the system you don't know much about it. You can find out. I just kind of trying to put these things in. And seeing the pieces of information you get again, that's where the video doesn't a more clear thing to us. They know more about what they're doing, but OK, this is still fine. We got our value, which is not useful now. Alright, let's go back to what I said before. How would I use this in a more real sense? Like he doesn't help me in the least to have a password of George 'cause. I'll never match a hash with George. But what can I do? If I am setting the salt. And I want to set a blank password, so I created a new user. With a blank password. OK, I can do that or actually any password I want because I know what the value is. Now I can go ahead and create the values that the hash value of knowing what my salt is, knowing what I want is a hash value. I can go ahead and just take any password I want. Use the salt that I am specifying and put in the hash password that I am specifying all this and I have a values are with a valid password and a valid solve. It could be no password at all. Now it may be that your system wouldn't normally allow you to put that in. This is also, yeah. Remember, we're talking about data security here. Where does the restriction happen? It may be that like that, a wise system would say the user would not allow. A password that was blank. But how can you check that in your SQL? Remember I had. I can't remember which one. Banana Republic or whatever was a blank password. It comes up with a hash value. There's no way to tell us to blank password, so there's no way for the SQL database to know. To restrict anything about the password at all. So if I want to create a user and this becomes a particularly powerful because this is a very random user thing. What if I have in here a role in my users table which says I am an admin versus a normal user? All of a sudden I may be able to create a superuser kind of person. within own password and no and and the salt that i've specified and then i can even if i crash the system i go back in i log in again now i surf. With no password. OK, so surf head is no password at all. So now I'm going to log in as surf. With no password. OK. Now surf with no password. Get part of data security is there should be really strict controls in your database. We have not talked about privileges that different people have. Surf has no privileges, surf is a guest user. Surf has Zippo, surf can do nothing. Right. So in theory, so in our example, when I look at surf here OK, I have admin, admin has super powers. I have served surf does not have superpowers. Right. To Andrew's point of view, what if I want to change admin password? I don't have any easy way to know what the cryptographic hashes they're using it. I'm an assistant foreign system, I don't know. How do I change admin password to something that I can use easily? Look at our thing here again. OK, how about? We're gonna insert into users now. We're going to do this differently. We're going to update. Users. Right. Now update users island on separate lines and then we'll put it back together and we're going to have what is set. And we're going to have password. Equals. Select. User. Or password. From. Users. Where? Username. Equals surf. OK, that will give me the password. And then I do. I can actually even do both. I can do password which will probably work. I can do password and salt. OK. I know I have to work that I'm sorry. My dear. Password miss. Man, I'm going to say, alright. Salt. Equals. Right now what I have is a big long query, but what can you tell what I'm doing here? I am updating a record. Update this. Updating. A this. And I'm gonna say now I'm gonna say on the outside of this, I'm gonna say where. Username. Equals admin. So if I have done this correctly, solved. We are username, so real quick there's extra equals on those salts. When you assign the salt. Brett, thank you. So what I'm doing here is I'm cramming together this great big long thing, and then I have to get the end here. So I have my update where I'm gonna put my; And this. I now know. How to log in? And hopefully. Run this update and what it should be doing is by using subqueries. It's going to set the password to be the password that serve has. And it's going to set the salt to be the salt that serve as. So what I should wind up is being able to have admin had now has an empty password. I don't even hear what the salt is. I don't care what the cryptographic hash is, I just know that when the system did this to me before. It worked. So now it should do it here. So in theory might try this and we'll see if we can do this, what we should wind up if we get this right, we may crash the system. Again, I don't care if I locked myself in the right position, we should wind up with the same password and solved in admin. We haven't served. But the difference is that that admin has superpower rights and surf doesn't. So then we log in as admin and all of a sudden we have control of this. Hunt. We are now going to do this. We're gonna try to log in. We only use our super amazing thing here. Hit enter. We're gonna do Rover 2021 'cause. That is the password for Gandalf. It's gonna crash, but did it do its job first? Right, so let's run this. Yeah, I should just testing the system, but let's let's go ahead and see what it gives us. What happened? Doesn't look like we managed on that one. I got something wrong with the with the error was maybe. OK. Again set users whatever selects. Well, toggle results sets for return next week ception. But this time it didn't set it. Process, although we could, but this is the logic you're doing here happens because of two different problems that have to do with data security. One of the problems is that I should be doing some validation on my usernames. Is it legitimate to have? Certain characters in my username maybe, maybe not. What are those characters that are on the gitim it to have. Alright, so the first, the first mind of defenses before you even get into things like parameterized, which is to say I should be saying you can't have. Quotes or dashes like you'll notice this is not now a persons name where O'Leary or O'Malley or whatever with you know with an apostrophe would be acceptable. You're now saying usernames and usernames often are, they say only alphanumeric characters. So that would be the very first. We're not going to bother with that right now, but that is important to do. If you've done that, none of this will work. Couldn't save the the user at. The other thing is the one that they're talking about doing here, which is that let's go to the very limit minimal of this, which is we're going to say what is the error that we have. OK. How would we just test? Just the username it. You know here how do we test and see whether there's a quote in the username? OK, it's Java. So we say username. Dark teams. OK, character sequence windows say. Single quote. OK. So we'll do that because it's very hard to do anything with any of this without a single quote. So with nothing else then we should be able to to capture this. So if I have my login user and I do this and I say. Now remember. And this is back to the other thing. All you should tell them is that it wasn't valid. Don't tell them what the problem was. It's very tempting as programmers because we're trying to keep track of what's going on. I like to have a different message for each thing so that I know exactly what's wrong. Any interaction with the user, though, should actually avoid that. The operation didn't work or returned an empty result set. Give them as little information as possible, because then they won't know if I say you know quote is not allowed. You know in thin single quote it almost just advertises that I'm using a relational database. If I just say the login is not valid. Then I'm not even telling which characters wrong, I'm just name that's wrong. That won't work. OK. Then I would have to have a return. If I had justice added to my logic, let's see if that works. I run it again. When I go to log in now. I say. See if I still have it. Wrong thing, OK. How long is not valid? Well, that's true, but let's try to get this log. Let's try to get this login instead. OK, so log in. Add great long things there and it's just going to say that login is not valid. Please try again. I never have gotten I I've limited one part of the owner bill. But the other part I'm going to take this out of here now because I want you to run into the other part. But I will leave it in here in the comments so that you can see it afterward is to go and fix as username that and password that every single place in here where I do a SQL query I want to check and say am I ever doing my. Income ABBA query for object or a query for roadside or anything else? Is my SQL statement that is getting passed in? Does it have parameters like the question marks or does it use its own concatenation? This is done properly. This is not done properly. So how do I fix this? What do I change? Come on, wake up. What do I change to make this super amateurs input? What do you put into your SQL statement? Is this the question mark? Really my? So that's what you do here. And then how do I change the rest? I will go back and fix this, but then you gonna get rid of all this stuff. And then you put in an argument. Similar to like well, there's the second argument in the JBJC template. From wherever it is your query for rose at. You're now gonna have to quit in that argument, so I'm just gonna head. Is not going to be all of this 'cause. I don't want all of this. I just want. The user named at 2 uppercase. That simple change will ensure that even if I didn't do any input validation. It is trying to actually look for that entire username. So if I have done this correctly then I will. Or well, and similarly if I go back to my create user. Sorry my save user I think is now in here. OK. OK, actually no. Never mind. My point being OK, so now when I run this with with the change that I just made, no, I'm still the wrong place. OK, when I when I. Had my parameterised. Intruder name. Password. Now it's gonna search for a username with that full thing. Even if it finds it, it's not going to cause the action to happen. So let's try it one more time. We are going to. Quit. Just probably take you somewhere that I shouldn't have. I'm going to run again. Maybe a single single quote? And so and even the dashes it parameter, it takes care of all that stuff for you so that you won't cause a problem. So basically the punchline is all. I'm bored with 99% of at least of all SQL injection attacks are solved by the wild card thing that's already been. It's a solved problem as long as we use the question mark syntax. Well except and then yeah, I missed part of the punchline is the parameterisation used. The big single thing that helps the most? But it it raises the problem. That I caused inadvertently, which is parameterisation alone, is not enough. And the reason Parameterisation alone is not enough is the problem here. I have a fully parameterized. Saying it might save user. OK, I wish my saved user that's your login user. Where's your seat? Add new user. OK is add new user call saved user in here. You're going to save user, save user is parameterized. OK. So. By itself this looks great. And if I actually run this in my add my user. And now I'm going to say I'm going to change it slightly 'cause I don't want it to be. I want it to be hand off too, right? So now I'm gonna. Who's that? And now when I say row 2021. OK. I have now added a user in here. With a problematic. Strain. Right. So when I show users. Over here. Right. Let's log in as surf. Surf has no password. I wanna show all users. Maybe it did it, maybe it didn't. I don't know. Probably didn't do it. I'm gonna see. Alright. But I should not have. I mean, I have it here. Yes, I already have a friend, so it wouldn't you find it? Let's see if I can get it. The point being, whether or not action right now figure it out or not, if I have. This set of users right here. This was added with Parameterisation. It is problematic nonetheless. I can probably come up with a way to trigger this. To run to do the thing that I didn't want to do. So it says Parameterisation is the absolute requirement for all SQL statements. Right. And it says it's not enough to prevent putting values in your in your system. That may be problematic. For that you need user validation. So the two of them are both necessary. When you're adding new values, parameterisation actually will allow you to inject on the in your reading. It's called second level SQL injection. You all allow you to add them in because you're parameterized. So you have to user input to prevent that second level SQL injection. So as a generalization. Whatever flavor of SQL you're working with, maybe just make any sort of syntactic. Symbols that are not letters and numbers just completely illegal. Basically, that's generally, yeah. And you know, but you have to if you're in Ireland. So people really may object to the fact you can't put the the single Is not allowed in any. Username in almost any system. Right. That is a level of user interface that I would put in. Anything I would say, yeah, you can't have that when you start getting into JavaScript will be additional things that you wanna do it want to do, you just don't want to have certain characters. the simplest way yes i'd be alphanumeric sometimes in american space Yeah, I try. Whereas if I can try literally, there are things out there that can you try a billion combinations of, you know, a second. It's unbelievably fast. If you put in any sort of delay, then slows that down so that each one of those things takes a tenth of a second. And that billion combinations is not going to happen in any, you know, but anybody's lifetime. Yeah. So. Anyway, that's the be all and end all of. David Security, we should make sure there's nothing else that I am forgetting that I'm supposed to talk about because I have a tendency to forget topics at the end of the day. I don't think so, but OK. Yeah. The only other thing they want to talk about is certificate authorities and that sort of thing. It's not necessary for what you're doing there. There are going to be more security things as we get into the web. They are going to be more ways to use cheap avalue. Do you store it in a separate, you know thing that gets passed around or do you have a separate file that controls it? None of that really matters to you right now. We'll worry about it when we get to the web. Alright, so that's all I've got. I did just want to fill you guys in two different things that are unrelated to the topic. So let's just pay attention 'cause, these are whatever, right? One of them has to do with the parish project, right? I had generated the pairs, generated the stock. I'm going to give it to you tomorrow. Because I don't want you to distract yourself from the stuff you already have to do, but. I want to tell you these are actual pairs of one group of three. Some of them, some of you are going to look at your hair and say. I don't think we can do this on our own. Right. I have faith that all of you can do a fair part of what needs to get done in the pairs that I have assigned. I looked at every pair I thought about it. I believe you can do a fair amount of what used to get done. It's not graded. I do not care that much whether you complete every bit of what it is posed to be done. III care that you make it good faith effort for it, and for this particular project more than the general, I want you to try to work together with your partner. Mostly not. Go to anybody else. Stop. You can't ever ask you question. Don't sit in a session with other people and work through the whole thing or whatever else I want you to be doing it with your partner. That means all pair partners actually have to be involved. I don't want anybody sitting this one out or doing where else it is for all of you. I want to understand what you're doing, and I want you to work. I I I believe you can do it, but I also want you to just get as far as you can. Don't stress too much if you can't but but but believe you can. I'm now giving you something undoable. Alright, get as far as you can and can see how that goes 'cause I I want you. I think some people I think you get it. It's it's easy to get reliant on other people sometimes. And I think sometimes you need to actually say I gotta do this on my own. Right. Like if I can't do it, I I want to see how far I can do. I'm not judging you in anyway. I just want you to actually have the experience of seeing I. We gotta figure out as much as we can on our end. We'll do it right. And I had faith. The other thing is just this is more of a heads up. I am not a super secretive person, so there's like the reason I'm going to Boston is that my mom is going into Hospice Hospice, and I'm not sure how familiar you are with that basically means she's kind of at the end of the rope now. Does that mean that she's dying next week? Does that mean she's dying a year and a half from now? I don't. But I know that she is at the point where she is sick enough that you know my three brothers and I are all going to go there and we're going to try to work out what needs to happen from now on. Whatever it doesn't, I'm going to be back on Monday. Weather permitting, what it does mean, though, is that. There could come a time. During the rest of this cohort, when? we we come to the actual land and then it's possible that i will have to briefly go away and we'll have a substitute if that happens and you know we have other instructors who float around and do this i don't know