In the last lesson we used Firebase authentication to register of very first user.

So now that all user's has been registered, they've been allowed to go over to the chat screen.

But what if I wanted to be able to login at a later date?

Well currently I login screen doesn't do anything yet.

So let's go ahead and fix that.

And I'd like to pose that to you as a challenge.

Can you use Firebase auth to login an existing user?

You'll need to capture the email and password from the text fields similar to how we did things on the　registration screen and then you'll need to use Firebase auth to sign in an existing user with their　email and password.

And therein lies the challenge. You'll have to figure out which method to use to sign a user in.

Now I'll give you a few seconds to pause the video and channel your inner Sherlock Holmes before I show　you the solution.

Ready?

Here's the solution.

Go ahead and open up the login\_screen.dart.

And as usual, we're going to need to import the auth package that will need to rely on and we are again　going to create a auth object.

So we're going to have it as a private auth object and it's gonna be set to equal a Firebase auth object.

And we'll need again to have a email variable and a password variable so that we can save the values　that are created inside our text fields.

So this one is going to be the email and this one is going to be the password.

All right.

So now that we've tied our values in our text fields to some variables, we can actually use it to log　our user in inside our button.

So right about here, we're going to create a new user which is going to be created using our auth object　and we're going to sign in with email and password.

So this is the method that we're going to use　and this is the counterpart to the create user with email and password. And email is of course set to　equal email and password is going to be password.

Now again, we have to make sure that we change our onPressed to async so that we can wait for this sign　in with email to finish before we go to the next step where we check to make sure that we actually　did get a actual user back.

So user is not equal to null.

Well in this case then we should probably take the user to the chat screen right?

They've gone through all the checks and we as the bouncer are going to admit them to the club.

So we're going to use our pushNamed and our routeName is going to be our ChatScreen.id and to　use it, we of course have to import the chat screen.

And now the last thing we have to do is to wrap all of this inside our good old try catch block just　to catch any exceptions that occur. And you might get exceptions such as if the user doesn't actually　exist or if they typed in the wrong password, all sorts of things that you can save and put into widgets　to show the user or you can create your own messages for them saying that something didn't go right,　maybe they should try again.

So now we're ready to check our log in.

So let's go ahead and do a hot restart again and let's go ahead and log in our user. So my email that　I registered where there was angela@email.com and the password was 123456.

So now if I click on log in and I get taken to the chat, then I should see that email being printed again　because whenever a new user comes over to chat, we always tried to print out the current logged in users　email.

So now that we've confirmed all of this, we can probably comment out or delete this line of code.

So that wraps up the solution to the challenge and we're ready to tackle the very last part of our log　in and registration flow, which is to be able to log the user out. Because you can only login once you　log out　right?

So we've got this little cross button here created with an icon in our chat screen and we can implement　the log out functionality really really easily.

So here's our icon button right here　and when our button gets pressed, we want our user to log out.

So we're going to tap in to that auth instance and we're simply going to use the method sign out.

And it's as simple as just telling it to sign out.

And once the users have signed out though, we'll need to use the navigator to pop them back to the previous　screen.

And now we can simply click on the cross button to log back out. And that takes us straight back to the　log in page if we want to go back into the chat app.

So in the next lesson, we're going to upgrade the log in experience because at the moment when we log　in, everything kind of just stops and then we see the chat screen.

We're going to add a loading indicator to show the user that something's happening in the background　and they should be patient until it's done when we'll take them to the chat screen.

So all of that and more, I'll see you on the next lesson.