Now that we've created a Firebase account and created our first project through the Firebase website,　it's time to get set up for Android.

If you're only using Flutter to make us apps, then you can skip this lesson and proceed to the next lesson　where we set up for iOS.

So first let's add our Android app and you can see the little Android symbol here. And we're going to　have to figure out what our Android package name is.

So how do we find this?

Well as it suggests, it's in the app level　build gradle file and it's under something called application ID.

So if we head back into our project and we open up the android folder in our　project and make sure that　we're in the project view, then if we go into the app folder and we go into the build gradle that's　inside the app folder and not the one that's in the project,　well then we end up here.

And if we look through here then you'll find your application ID. And this is what we need to supply　to Firebase.

Now usually your application ID should be unique across the Android app store.

So I recommend changing this part to your name.

So it could be Angela Yu or it could be whatever your name is.

Now if you have a company, then be sure to put in your reverse domain name.

Now how did Flutter come up with this name?

Well whenever we create a new Flutter project one of the things that we have to supply is a company　domain in this setup wizard.

So say if your company was www.google.com, then it would create a package name of com.google.　the name of your app.

So this is how we came about the co.appbrewery.flash\_chat.

So once you've modified this to your own name or your own domain, then go ahead and copy the entire string　over to here and paste it in here.

Now it's really important that what you see here matches what you have as the key for the application　ID exactly, not including the quotation marks of course.

Now once you're done with that go ahead and click on register app.

And once the spinner stops, we'll get to download our google-services.json configuration file now.

It's really important that if you click on this once, you get the google-services.json downloaded　but on a lot of computers, if you click on it twice say maybe by accident, you'll get another file downloaded　but you'll see that it actually has one after the name of the file.

And it's really important that you only have one of these files in your Downloads folder so you don't　end up with a one added to the name of the file.

And the reason is because Firebase is going to look to find a file called google-services.json　and not something called google-services(1).json.

So this is a really common place where people trip up so it's important to be aware of that.

Now once you've gotten this file, go ahead and open up Android Studio and have your Finder or your Windows　Explorer open up where that file's located.

And we're going to drag it aiming for the app folders. So make sure the app folder is the one that's　highlighted.

So now, we get to go ahead and add it to the app folder and it doesn't matter if you want to add it to　Git or not.

It doesn't really affect your project.

So now we can see the google-services.json　json in its entirety inside our app folder and we're ready to move on to the next step of setup.

So let's go ahead and click next.

And now there's just a few more things we have to do.

So we have to open up our project level build gradle,　so it's under our project/build.gradle.

So that refers to the one that's inside our Android project under the name of build gradle.

And here we have something called dependencies.

And what it's asking us to do is to copy this line of codeand add it below any other dependencies.

but making sure that it's still inside the curly braces of our dependencies.

And then we have to open up the App-level build.gradle

So this is under our project/app/build.gradle and it's the one that's inside here.

So inside the app folder we've got our build.gradle.

And here we also have some dependencies down here.

And we're going to copy this line and add it into the dependencies inside our App-level build.gradle.

I'm going to paste it right here.

And we're also going to copy this line which implements our Google services file right at the bottom　of the file, below everything else and outside of all the other curly braces. Now once you're done, you　can go ahead and close up all of those files and make sure that you've got the android emulator selected.

Go ahead and run your app just to make sure that everything is working as expected.

All going well, you should see the app run in the emulator or in your device.

And if you take a look inside the run tab, you might see something about something being deprecated or　a couple of messages from Firebase.

But don't worry about this as long as you don't see anything in red inside your console under the run　tab and your app actually loads up, then that means you've setup Firebase to work with your Android app　successfully.

So now you can hit next and you're going to skip this last step and continue to console. And we're now　ready to move on to the next part which is setting up the iOS app.