So, I have renamed this test methods to make them more consistent and easier to read. So, we have three scenarios, admin cancelling, same user cancelling, and another user cancelling.

One thing you can see about this test is that they act as some kind of documentation about our source code, this is another benefit of writing tests, imagine you join a new team that gives you a new project, and your task is to make some modifications to a class.

If that class is properly supported by

a bunch of unit tests, you can easily figure out what the responsibility

of that class is. Now let me show you another benefit of writing tests.

Earlier in this section, I told you that once you have tests you can refactor your code with confidence. So back in our reservation class, I don't like the way this method is implemented, it looks very amateurish, so I'm going to refactor it, and then run all my tests, and see if I've broken anything or not. So the first change I'm going to make here is instead of having two if statements,

I'm going to have one if statement and use the logical or operator. So if user is admin, or MadeBy equals user. You're going to return true, otherwise, we're going to return false. Now, back in test explorer, run all the tests, so, my tests are verifying that our application is still working. I have not broken anything. I can take this to the next level. So, instead of having this if statement here, and then return false, I'm simply going to return the result of this expression. Now, we have only one line of code here. Once again, we run all the tests, all the tests are passing, our code is much cleaner and you can see another benefit of writing tests.