In this section, I'm going to give you a more challenging exercise. So, inside this Mocking folder, look at this BookingHelper class. Here we have one method called OverlappingBookingsExist. So it takes a booking object, first it checks if this booking is cancelled or not. If it's cancelled it's going to return an empty string. Otherwise, it's going to run a query to get all the existing bookings that are not cancelled, and their ID does not equal the ID of this booking object that is passed here.

Now in this implementation, this code is using some wrappers around entity framework. In your applications, you may have a different mechanism to query your database. You may use entity framework, you may use a different ORM. Or you may use plain ADO.NET objects like SQL connection, SQL command and so on.

What matters here is that this code is accessing some external resource, in this case, a database. And as you learned, we need to refactor this code, and make it loosely coupled and testable. So next, once we get all the existing bookings that are not cancelled, excluding the booking that is passed to this method, you want to check if this booking object we passed here, overlaps with any of the existing bookings. And finally, if there's no overlap, this method is going to return an empty string. Otherwise it's going to return the reference for the first booking that over laps with this existing booking. So, go ahead and spend some time, and write all the required unit tests for this method.

Make sure to test all the execution paths, make sure your tests are clean, maintainable and trustworthy, so all the best practices you have learned in this course I want you to apply them in your solution. So pause the video now, do this exercise, and when you're done, come back, resume and see how I write unit tests for this method.