The next test case we want to write here is where our booking starts and finishes in the middle of an existing booking. So again we have an overlap. Now, one more time, I'm going to select this code, and duplicate, so the scenario is where our BookinStartsAndFinishes InTheMiddleOfAnExistingBooking.

So here we can set the ArrivalDate to 1 day AfterTheArrivalDateofExistingBooking. So After, and similarly, we can set the departure date to 1 DayBefore the departure date of existing booking. So, let's run this test. It passed, beautiful, let's move onto the next test case. It's possible that our new booking starts in the middle of an existing booking but finishes after. So, kind of similar to this test case, I'm going to select this, duplicate and change the test name to StartsInTheMiddle OfAnExistingBookingButFinishesAfter. ButFinishesAfter. So, to simulate this, we simply need to replace Before, with After. Let's run this test. This is passing too, excellent. And, the last test case is where our booking starts and finishes after an existing booking, in this case we don't have an overlap.

So, let's select this code, duplicate and change the test name to BookingStartsAndFinishes AfterAnExistingBooking. And here we expect to Return EmptyString.

So here to change the logic, we should set the ArrivalDate after

DepartureDateOfTheExistingBooking, but we should set the departure date to let's say an extra day after. So, you need to modify our helper method. And how do we take an optional argument? So, IntegerDays, we set it to 1, and replace this 1 with Days. And then we can set the DepartureDate to 2 days after.

And, here we need to change our Assertion and ensure that result Is Empty. Let's run this test. This is passing too, excellent. So, we have covered various possibilities of two date ranges overlapping, we've fixed the bug we had in our BookingHelper class to make the logic shorter and

cleaner. But we still need to write two more tests here where we have an overlap but one of the bookings is cancelled. Either the new booking or the existing booking. If the existing booking is cancelled, it should not be returned from our repository.

Because inside this method we have encapsulated the logic to return

bookings that are not cancelled. So to test that scenario we need to write an integration test for this method we need to populate our database with a bunch of bookings, and make sure that this method only returns the bookings that are not cancelled. So the only other unit tests we need to write here is where this booking is cancelled, and it overlaps with an existing booking. So, back to our test class, I'm going to grab, one of these tests where we have an overlap, so it doesn't matter which one you can pick anyone really. Let's copy this, and paste it here, and change the method name to Bookings OverlapButNewBookingIsCancelled.

In this case we should Return EmptyString. So now with this logic we have overlapping bookings, but we should set the Status to Cancelled. And then Assert that result is Empty. Let's run this test. And this test is passing too. So, we have a total of 7 unit tests for this method, and this concludes this challenge.