This exercise has two parts:

* The first part is a practical coding exercise where you will modify a solution to improve the way it is tested, and in doing so flush out defects with the solution. You are asked to complete this **before** the interview.
* The second part will take the form of follow up questions at interview.

The “TflShop” solution represents is a simple checkout process through which customer can pay for items from a pre-paid balance.

The code to be tested is in CheckoutService and PriceCalculator.

NotificationProvider contains incomplete code and does not need to be tested as part of this exercise. The three classes in the Models namespace are used by these services, but do not need to be modified.

A unit test has been written, and it passes, but there are defects in the code. The current unit test is not a very good test. You may wish to refactor it, or scrap it entirely and start from scratch, it’s up to you.

NUnit, FluentAssertions and Moq NuGet packages have already been added to the unit testing project, but you are free to add any additional or alternative frameworks you choose.

The object of the exercise is:

1. To refactor the code and tests to improve the test coverage such that the defects are identified by failing tests.
2. Once a failing test has been implemented, fix the defects.

Identifying the bugs through code review is not the primary purpose of the exercise. Good tests should identify the defects for you, so focus on comprehensive testing.

Note that although the focus is on writing a failing test before fixing the defect, you may need to change the structure of the classes being tested in order to implement those failing tests; if you feel you need to please do.

You should come to interview with a set of tests and refactored code which you feel provides comprehensive coverage of CheckoutService and PriceCalculator. You may also wish to fix the defects that you identify.