# Bolser

## Unattended Programming Test

### Scenario

A UK based mobile phone operator has decided to reward its existing customers of over 12 months. A software engineering team, developing the customer account website, is working on the story below.

***Display customer's available rewards***

*As a customer, if I am eligible for rewards, then I want to see which rewards are available based on my tariff and loyalty.*

### Instructions

You are required to provide an implementation of a RewardsService. You may use any programming language you like.

The service accepts as input a customer account number and an associated tariff code. If the customer is eligible for rewards the RewardsService should return a list of all the rewards available according to the subscriptions on the portfolio.

### 

### Acceptance Criteria

The following table describes the codes for customer tariffs and the associated rewards.

|  |  |
| --- | --- |
| **Tariff** | **Reward** |
| 3G100MB | EXTRA50MB |
| 3G250MB | EXTRA100MB |
| 3G500MB | EXTRA250MB |
| 3G1000MB | EXTRA500MB |
| 3G2000MB | EXTRA1000MB |

The Customer Status team is currently developing the EligibilityService which accepts the customer’s account number as an input. You are required to provide a mock or stub of the EligibilityService interface.

This is a simple diagram of the interaction between the services:



The following table describes the EligibiityService output and the expected result:

|  |  |  |
| --- | --- | --- |
| **EligibilityService output** | **Description** | **RewardsService result** |
| CUSTOMER\_ELIGIBLE | Customer is eligible | Return relevant rewards according to the customer's billing history |
| CUSTOMER\_INELIGIBLE | Customer is not eligible | Return no rewards |
| Technical failure exception | Service technical failure | Return no rewards |
| Invalid account number exception | The supplied account number is invalid | Return no rewards and notify the client that the account number is invalid |

### 

### 

### What we look for

We are especially interested in how you structure your code so that it's fully testable, easily extensible, complies with best object-oriented practices and is easy to modify / understand by others. Please supply us with your source code and any tests you have written.