INF SCI 2955 - DELIVERABLE 3: System Testing with Behavior Driven Test (BDD) for Animal Shelter

Weichuan Hong

Jiawei Xu

July 2014

Contents

1. Summary 3

2. Description 3

2.1 Design of the scenarios for three user stories 3

2.1.1 Feature: Manager 3

2.1.2 Feature: Staff 4

2.1.3 Feature: User 4

2.2 Translation of the English scenarios to Java code 5

2.3 Java code 5

2.4 Location of the Java code 6

3. Test Results (Screen shot) 6

3.1 Summary (11 scenarios) 6

3.2 Manager (3 scenario in details) 7

3.3 Staff (5 scenario in details) 7

3.4 User (3 scenario in details) 8

# 1. Summary

The Animal Shelter program is a Java application written for animal shelters with dogs and cats to manage their animals that are brought in and those that are adopted, which are then removed from the list. The program is designed with the FIFO (First in, First out) methodology.

Here we are designing a system test with BDD for this Animal Shelter application. We have considered three groups of people who are directly related to this application: the manager of the animal shelter, the staff of the animal shelter, and the people who would like to adopt an animal from the shelter. Because of their different role, those three groups have different requirement and interest for the application:

* Manager of the Shelter

As manager of the shelter

I want to know the statistical data of whole situation of the shelter

So that I can analyze the whole situation and manage the shelter effectively

* Staff of the Shelter

As a staff of the shelter

I want to register an animal into or remove it from our system correctly

So that the system can work steadily

* User who want to adopt animals

As a user

I want to adopt one or more animals

So that I would have either dogs or cats

There for, it is necessary to design different test plans for three different groups to satisfy their requirements.

# 2. Description

## 2.1 Design of the scenarios for three user stories

The first issue we met is to design the scenarios for three different user stories. We tired to “discuss things in a way that an intelligent but non-technical user of the software would understand”, as well as to discuss things also in a way that is easy to translate to the Java code. The following are our 11 scenarios:

### 2.1.1 Feature: Manager

Scenario: Current how many cats do we have?

Given a shelter for cats with cat1,cat2,cat3 in order

When the manager check the quantity of cats

Then the system should display 3 in the shelter

Scenario: Current how many dogs do we have?

Given a shelter for dogs with dog1,dog2,dog3 in order

When the manager check the quantity of dogs

Then the system should display 3 in the shelter

Scenario: Current how many animals do we have?

Given a shelter with dog1,cat1,dog2,dog3,cat2 in order

When the manager check the quantity of animals

Then the system should display 5 in the shelter

### 2.1.2 Feature: Staff

Scenario: First come first go for cat

Given a shelter for cats with cat1,cat2,cat3 in order

When someone wants to adopt a cat

Then the system should output the cat cat1 who has been here longest

Scenario: First come first go for dog

Given a shelter for dogs with dog1,dog2,dog3 in order

When someone wants to adopt a dog

Then the system should output the dog dog1 who has been here longest

Scenario: First come first go for animal

Given a shelter with dog1,cat1,dog2,dog3,cat2 in order

When someone wants to adopt an animal

Then the system should output the animal dog1 who has been here longest

Scenario: Be able to register a coming dog

Given a shelter

When a dog dog1 comes in

Then the system should be able to register the dog dog1 into the system

Scenario: Be able to register a coming cat

Given a shelter

When a cat cat1 comes in

Then the system should be able to register the cat cat1 into the system

### 2.1.3 Feature: User

Scenario: I should have a dog if I ask for a dog

Given a shelter for dogs with dog1,dog2,dog3 in order

When I want to adopt a dog

Then the system should be able to give me a dog

Scenario: I should have a dog if I ask for a cat

Given a shelter for cats with cat1,cat2,cat3 in order

When I want to adopt a cat

Then the system should be able to give me a cat

Scenario: I should have either a cat or a dog if I ask for an animal

Given a shelter with dog1,cat1,dog2,dog3,cat2 in order

When I want to adopt a animal

Then the system should be able to give me a animal

## 2.2 Translation of the English scenarios to Java code

The second issue we have met is how to translate the English scenarios to Java codes.

At the beginning, we tried to use Jbehave, it uses Gherkin grammar as well as Java coding. However, we can not find a full guide of Jbehave from installation to execution. We have tested some sample code with Jbehave but failed because of some libraries or classes missing.

Then we changed to Cucumber, which is based on Ruby. This time we found all of the necessary libraries to make our test run successfully. The following are the libraries we have implemented in our project:



## 2.3 Java code

The third issue is how to write Java code to complete these tests. Here are problems we met and solved:

1. Cucumber .feature file has very strict format.

2. The Regular Expression is somehow tricky for us.

3. Sometimes it is easier to describe in English but more difficult in Java code for a scenario. For example, as a manager:

Scenario: Current how many cats do we have?

Given a shelter for cats with cat1,cat2,cat3 in order

When the manager check the quantity of cats

Then the system should display 3 in the shelter

It is difficult to test in this project because the Animal Shelter program does not provide a method to return those numbers. To finish this test, we have applied the size() method of the ArrayList class.

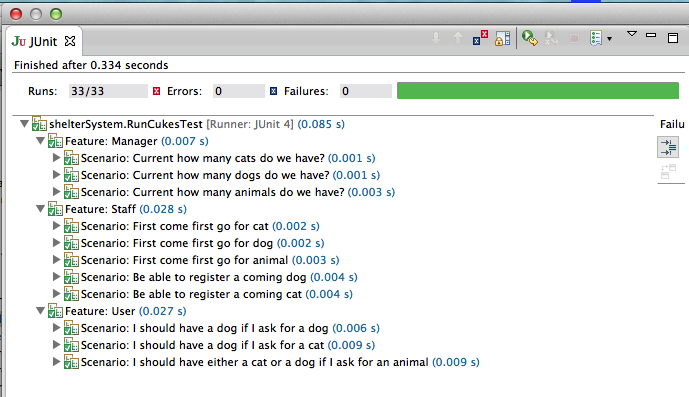
At the same time, a notice should be sent to the programmer of “missing some function”.

## 2.4 Location of the Java code

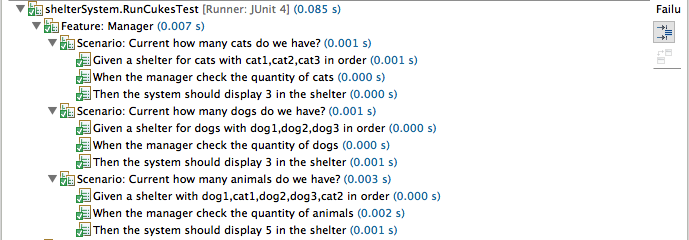
[https://github.com/lockTP/animalShelterBDD](https://github.com/lockTP/animalShelterBDD" \t "_blank)

# 3. Test Results (Screen shot)

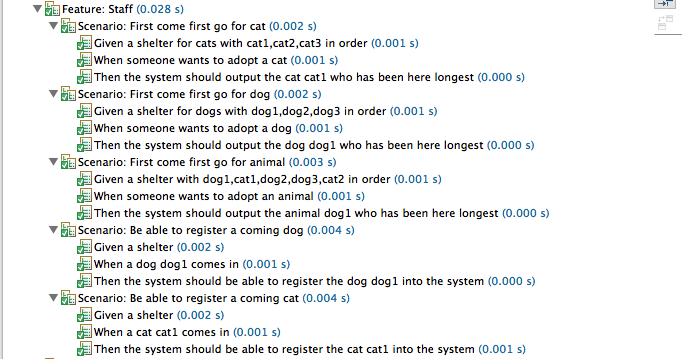
## 3.1 Summary (11 scenarios)



## 3.2 Manager (3 scenario in details)



## 3.3 Staff (5 scenario in details)



## 3.4 User (3 scenario in details)

