

**INDIVIDUAL ASSIGNMENT**

Course: Object Oriented Development with Java

CT038-3-2-OODJ

|  |  |
| --- | --- |
| **Student declaration:** | |
| *I declare that:* | 1. *We understand what is meant by plagiarism* 2. *The implication of plagiarism have been explained to us by our lecturer* 3. *This project is all our work and we have acknowledged any use of the published or unpublished works of other people.* |

|  |  |  |
| --- | --- | --- |
| **Project Title** | Bintang Veterinary Clinic and Boarding System (BVCB) | **Intake: UC2F1505SE** |
| **Lecturer’s Name** | KAU GUAN KIAT | |
| **Name** | Tan Wui Chin | |
| **Student ID** | TP034199 | |
| **Date Assigned** | **8th June 2015** | |
| **Date Completed** | **17th August 2015** | |

Contents

[1.0 Introduction 6](#_Toc427581828)

[2.0 Assumption 6](#_Toc427581829)

[3.0 UML Diagram 8](#_Toc427581830)

[3.1. Use Case Diagram 8](#_Toc427581831)

[3.1.1 Use Case Scenario 9](#_Toc427581832)

[3.1.2 Use Case Form 10](#_Toc427581833)

[3.2 Class Diagram 12](#_Toc427581834)

[3.3 Activity Diagram 13](#_Toc427581835)

[3.3.1 Make Appointment 13](#_Toc427581836)

[13](#_Toc427581837)

[3.3.2 Register Pet 14](#_Toc427581838)

[14](#_Toc427581839)

[3.3.3 Make Consultation 15](#_Toc427581840)

[3.3.4 Create Rota Table 16](#_Toc427581841)

[3.3.5 Update Last Fed 17](#_Toc427581842)

[4.0 Implementation of Object Oriented Programming 18](#_Toc427581843)

[4.1 Abstraction 18](#_Toc427581844)

[4.2 Encapsulation 18](#_Toc427581845)

[4.3 Modularity 18](#_Toc427581846)

[4.4 Inheritance 19](#_Toc427581847)

[4.5 Polymorphism 19](#_Toc427581848)

[4.6 Extra Features 21](#_Toc427581849)

[4.6.1 Abstract Class and Abstract Method 21](#_Toc427581850)

[4.6.2 GUI 22](#_Toc427581851)

[5.0 User Manual 22](#_Toc427581852)

[Login Screen 22](#_Toc427581853)

[1.0 Manager Menu 23](#_Toc427581854)

[1.1 Show Rota Table Form and Vet Table 23](#_Toc427581855)

[1.2 Display General Report 23](#_Toc427581856)

[2.0 Receptionist Menu 24](#_Toc427581857)

[2.1 View Customer and Pet List 24](#_Toc427581858)

[2.2 Register New Customer 24](#_Toc427581859)

[2.3 Register Extra Pet 25](#_Toc427581860)

[2.4 Show Appointment List 25](#_Toc427581861)

[2.5 Add Appointment 26](#_Toc427581862)

[3.0 Vet Menu 27](#_Toc427581863)

[3.1 Add Consultation 27](#_Toc427581864)

[3.2 View Boarding List 28](#_Toc427581865)

[3.3 View Rota Table 29](#_Toc427581866)

[4.0 Boarding Menu 30](#_Toc427581867)

[4.1 Update Last Fed and Current Condition 30](#_Toc427581868)

[6.0 Limitation 30](#_Toc427581869)

[7.0 Future enhancement 30](#_Toc427581870)

[8.0 Conclusion 31](#_Toc427581871)

[9.0 References 31](#_Toc427581872)

# 1.0 Introduction

This project is a computerize system that helps to manage Bintang Veterinary Clinic and Boarding (BVCB) daily activities due to the growing provider of veterinary care for pets. Activities such as registering customer, making appointments and others must include in the functions of the system itself. Several OOP concepts must be applied throughout the system to ensure its efficiency and effectiveness of codes.

# 2.0 Assumption

- The clinic operates every day.

-The clinic has 8 vets in total, only 3 will be working in each day.

- Other than that clinic employs 1 receptionist and 2 boarding staffs.

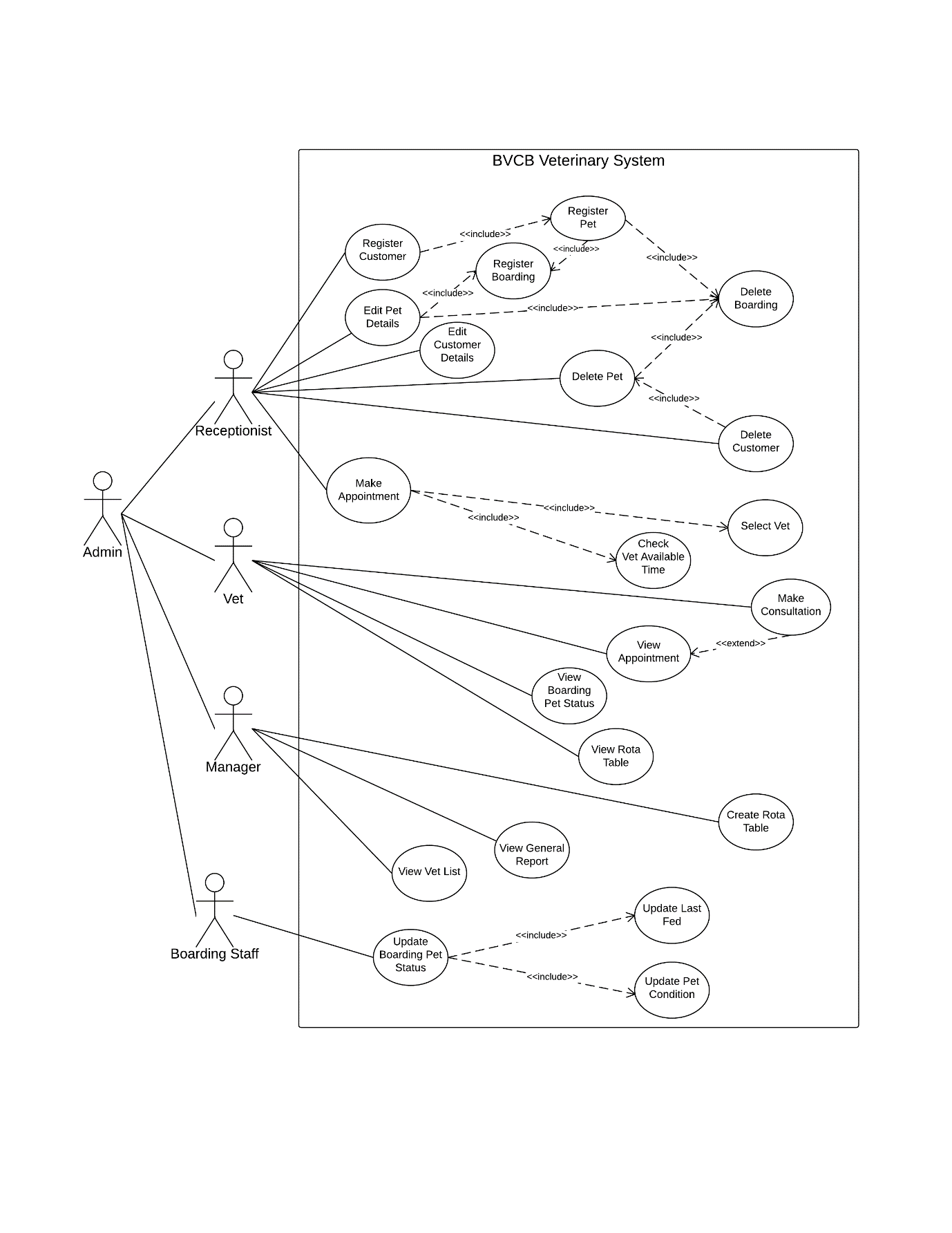
-Each vet two areas of expertise .

- The users of the system are receptionist, manager, vet and boarding staff.

-Manager is responsible in sorting of rota table.

# 3.0 UML Diagram

## 3.1. Use Case Diagram



### 3.1.1 Use Case Scenario

#### 3.1.1.1 Make Appointment

Receptionist is requested by a customer who wants to make an appointment for his pet dog name Fluffie. The customer also mentioned that he is not a new customer for Bintang Veterinary Clinic and Boarding (BVCB). He then proceed to book his favourite vet, who is Mr. Henry, a dog expertise, on 10:00AM to 11:00AM on a date of 21th September 2015. The appointment is successfully made by the receptionist.

#### 3.1.1.2 Register Pet

A customer requested the receptionist to register her second pet hamster so that she can make an appointment for it. After she has filled up the form that presented by receptionist, the form is returned to the receptionist to do data recording.

#### 3.1.1.3 Make Consultation

On 10:00AM, a customer brought along his dog who named Fluffie to the vet consultation room. Mr. Henry the dog vet, made a prognosis for Fluffie , then recorded it in consultation form. Later the vet gently carry out a full body check-up for the owner’s dog. The diagnosis of the check-up is inserted along with the sickness level of the pet. The vet also entered the fees of this consultation into the form and saved it into database.

#### 3.1.1.4 Create Rota Table

Every month, manager create a new rota table for the vets in order to keep the working balance for each them. BVCB consists of 8 vets, and each of them have 2 expertizes. Manager decides to redesign the rota table while keeping balance of the vet expertizes of the week.

#### 3.1.1.5 Update Last Fed

Every day, 2 boarding staff are responsible to supervise the boarding pets’ condition time to time. They also need to supply them with food and water. Furthermore, pets’ feeding time must be recorded into database after the fed them with the food.

### 3.1.2 Use Case Form

#### 3.1.2.1 Make Appointment

|  |  |
| --- | --- |
| Name | Make Appointment |
| Actor | Receptionist |
| Priority | High |
| Extends | None |
| Includes | Select Vet, Check Vet Available Time |
| Pre-conditions | Customer and pets are already registered into database. |
| Post-conditions | Appointment table is updated. |
| Extension Points | None |
| Flow of Events | 1. Receptionist selects the customer in the table. 2. Receptionist select make appointment button. 3. Receptionist select vet. 4. Receptionist insert appointment time. 5. Receptionist select make appointment button. 6. If the reservation time is not available [A1] 7. Appointment is successfully made. |
| Alternative paths | 1. A1.Show reserved time unavailable, Go to 3. |

#### 3.1.2.2 Register Pet

|  |  |
| --- | --- |
| Name | Register Pet |
| Actor | Receptionist |
| Priority | High |
| Extends | None |
| Includes | Register Boarding, Delete Boarding |
| Pre-conditions | Customer details are already registered into database. |
| Post-conditions | The receptionist table is updated. |
| Extension Points | None |
| Flow of Events | 1. Receptionist selects the customer in the table. 2. Receptionist select register pet button. 3. Receptionist insert details. 4. Receptionist select add pet. 5. Pet data is successfully saved. |
| Alternative paths | None |

#### 3.1.2.3 Make Consultation

|  |  |
| --- | --- |
| Name | Make Consultation |
| Actor | Vet |
| Priority | High |
| Extends | View Appointment |
| Includes | None |
| Pre-conditions | An appointment of the vet is already made. |
| Post-conditions | The consultation is saved and the appointment record is marked as check. |
| Extension Points | None |
| Flow of Events | 1. Vet selects the appointment in the table. 2. Receptionist select make consultation button. 3. Vet inserts prognosis. 4. Vet inserts diagnosis. 5. Vet select sickness level. 6. Vet insert fees. 7. If fees input is invalid [A1] 8. Consultation is successfully saved. |
| Alternative paths | 1. A1 Show input error, Go to 6 |

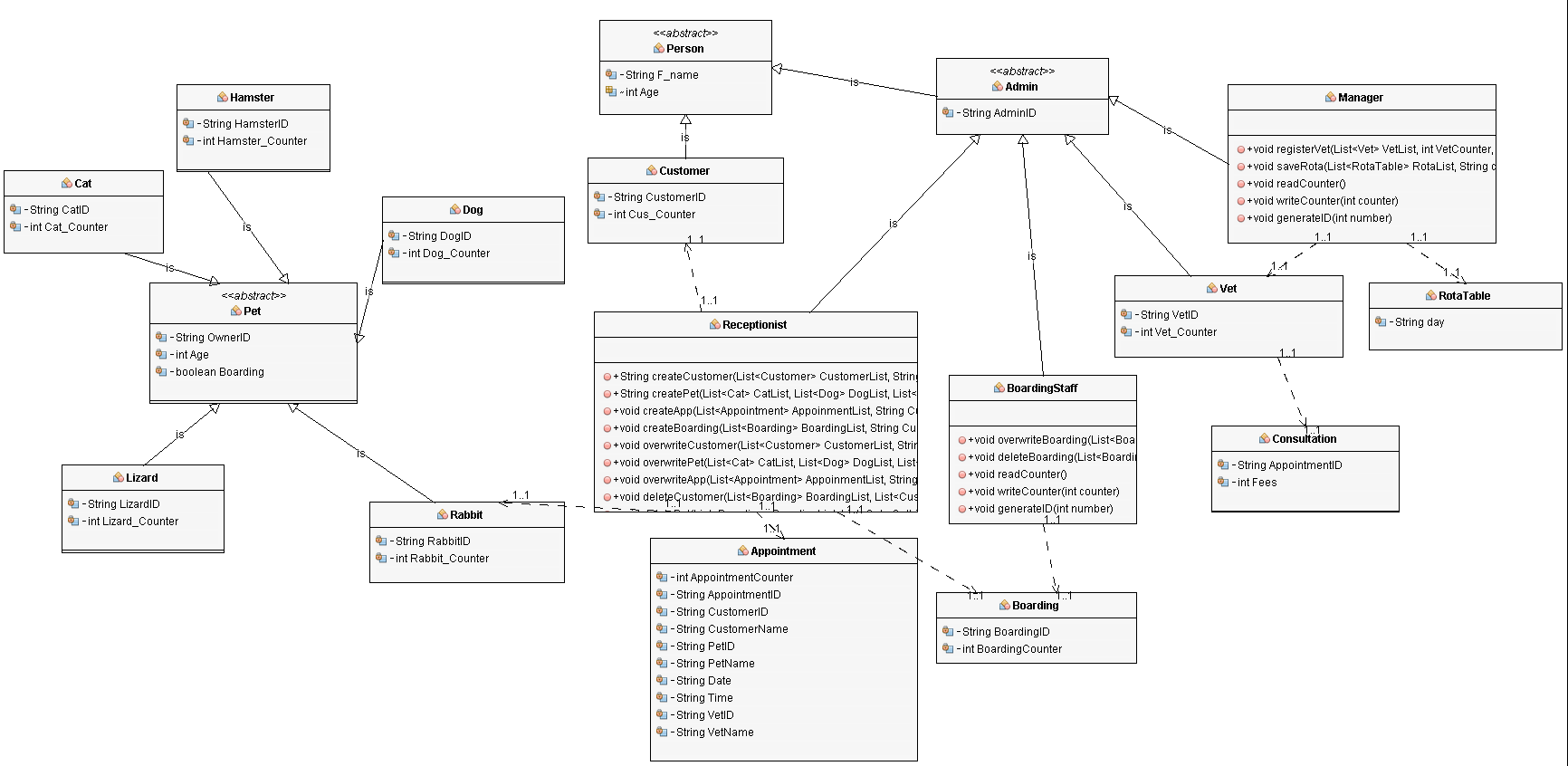
#### 3.1.2.4 Create Rota Table

|  |  |
| --- | --- |
| Name | Create Rota Table |
| Actor | Manager |
| Priority | High |
| Extends | None |
| Includes | None |
| Pre-conditions | All of the vet are registered in database. |
| Post-conditions | The rota table is updated. |
| Extension Points | None |
| Flow of Events | 1. Manager selects 3 vet for every day 2. Manager selects check rotatable button. 3. If any day is not balance [A1] 4. The rota table is balanced. 5. Save the rota table. |
| Alternative paths | 1. A1 Show imbalanced day , Go to 1 |

#### 3.1.2.5 Update Last Fed

|  |  |
| --- | --- |
| Name | Update Last Fed |
| Actor | Boarding Staff |
| Priority | High |
| Extends | None |
| Includes | None |
| Pre-conditions | All of the boarding pets are registered in database. |
| Post-conditions | The boarding pets’ last fed is updated. |
| Extension Points | None |
| Flow of Events | 1. Boarding staff select boarding pet.  2. Boarding staff click feed pet.  3. Boarding staff click save.  4. The fed time is saved into the database. |
| Alternative paths | None |

## 3.2 Class Diagram



## 3.3 Activity Diagram

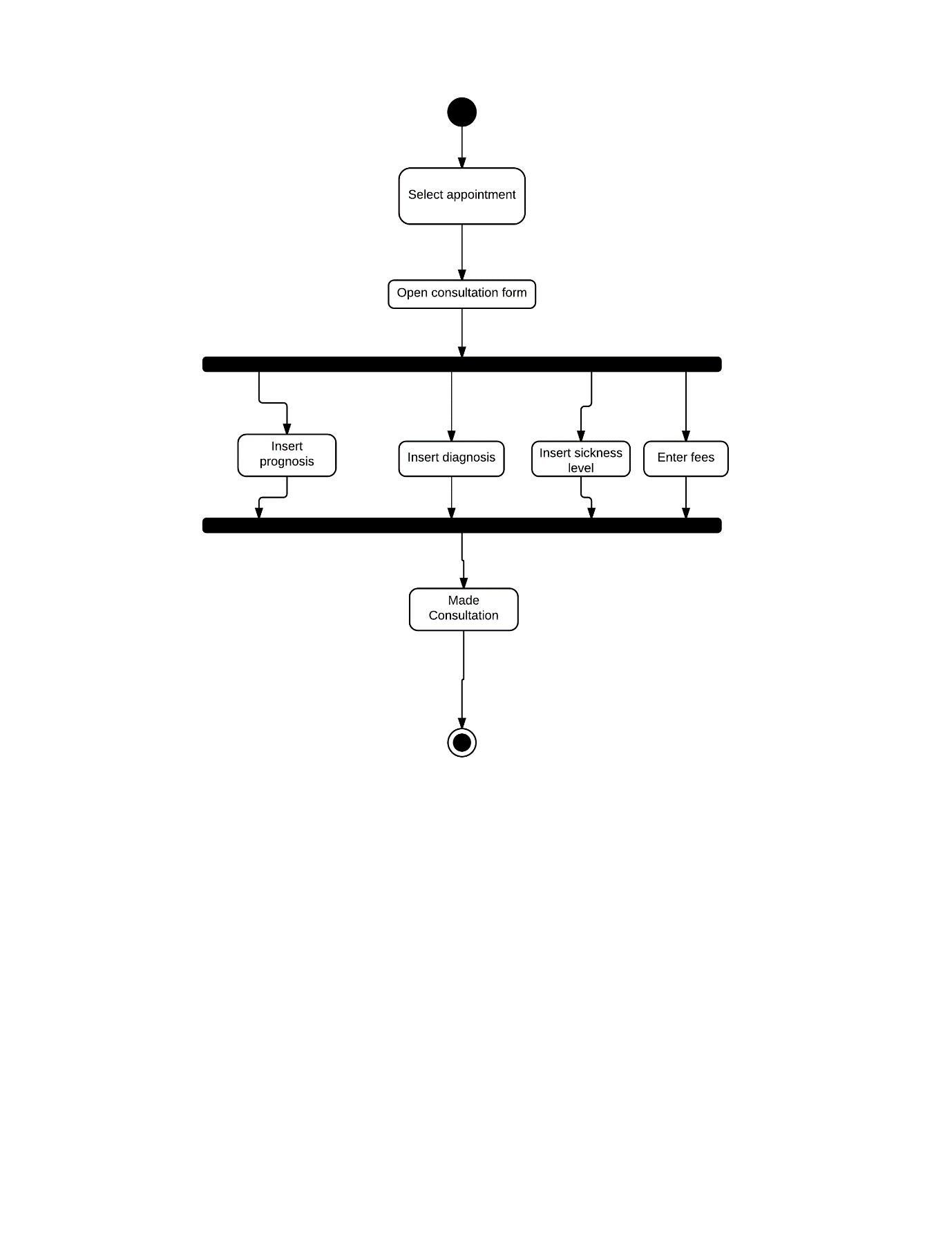
### 3.3.1 Make Appointment

### 

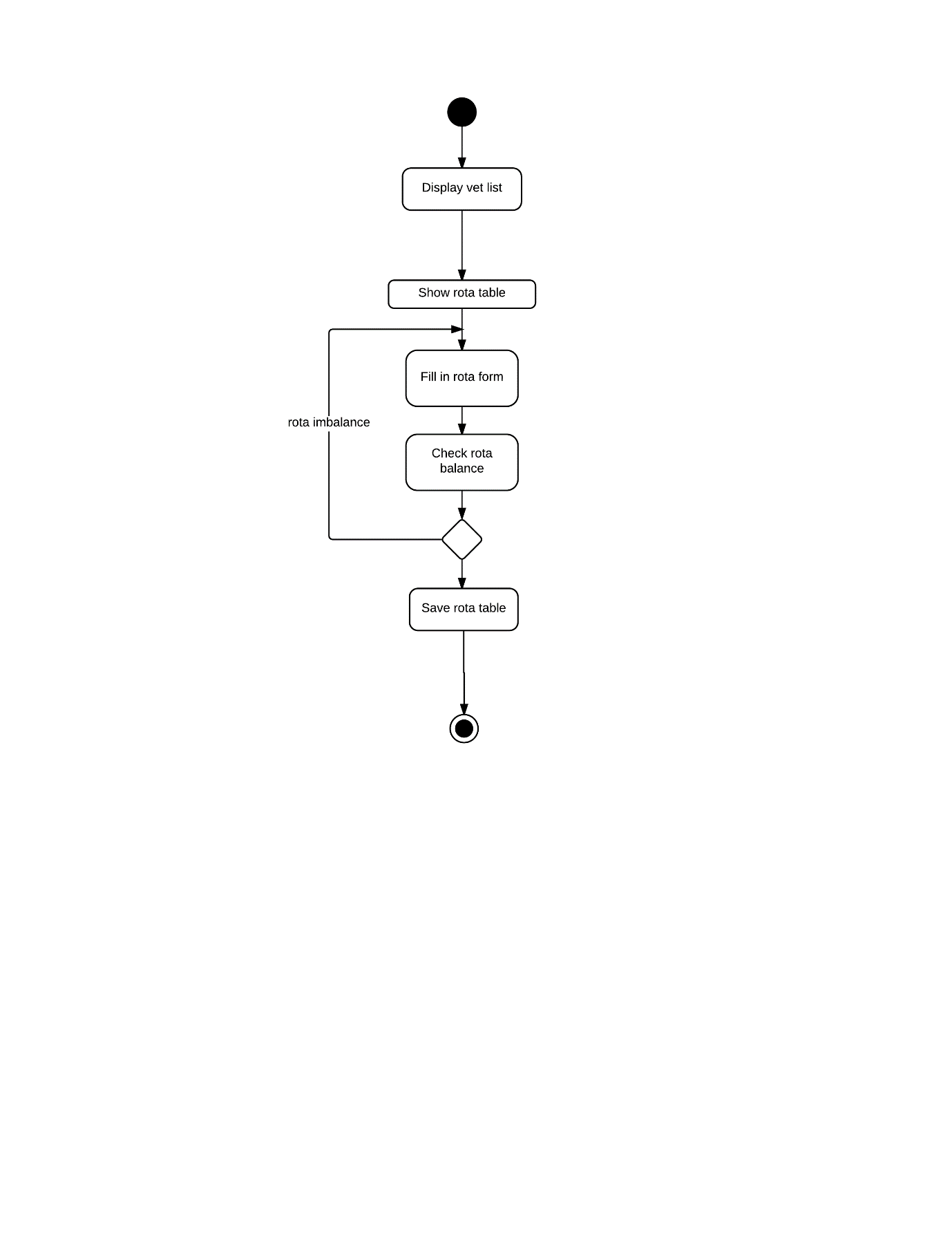
### 3.3.2 Register Pet

### 

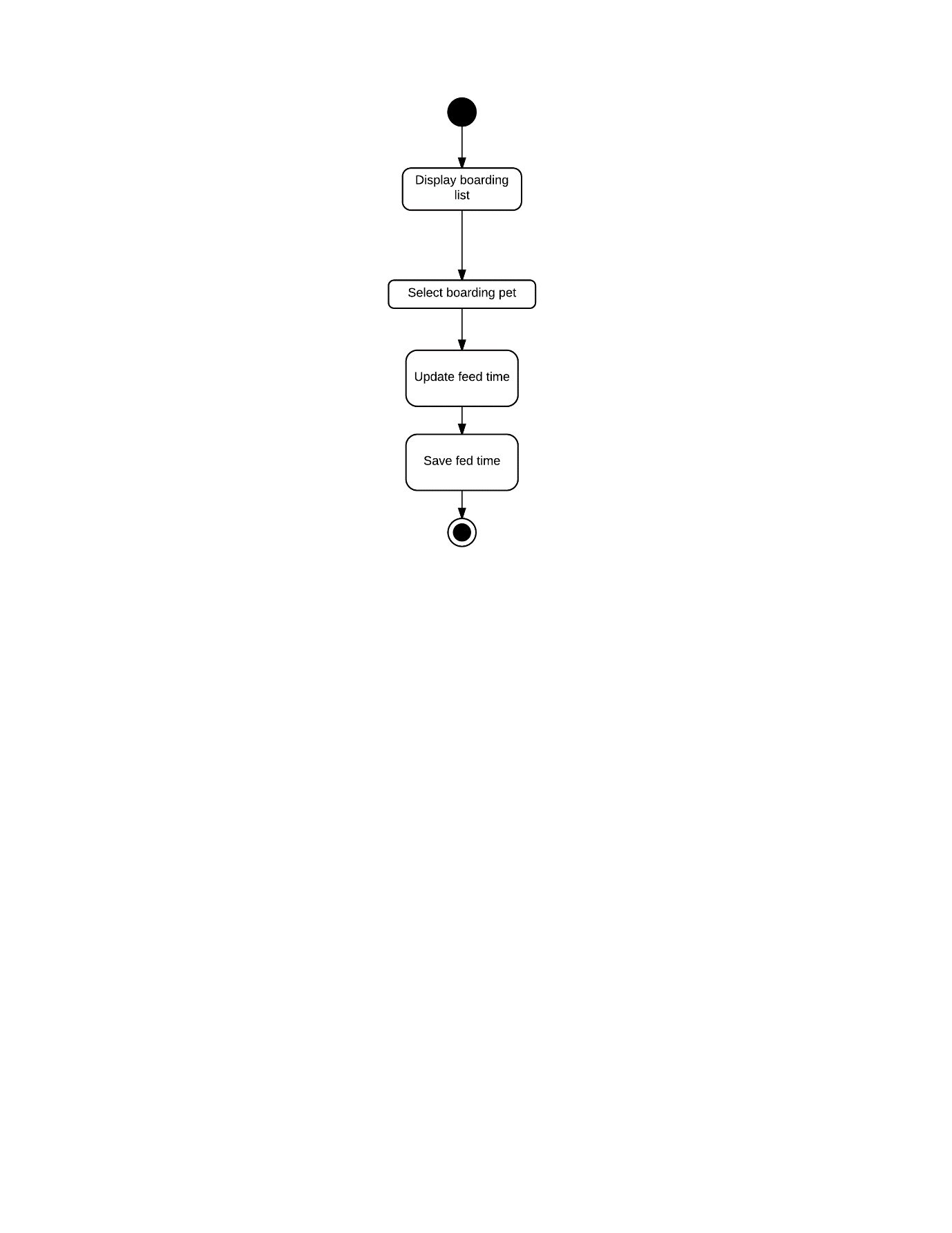
### 3.3.3 Make Consultation



### 3.3.4 Create Rota Table



### 3.3.5 Update Last Fed



# 4.0 Implementation of Object Oriented Programming

## 4.1 Abstraction

Abstraction is one of the best methodology use when dealing with complex system. (Anon., 2015) By giving meaningful names to the classes, it helps to bring out the characteristic and functionality of the class, furthermore understanding the relationship among different classes. Abstraction is largely implemented in this project. Several examples are show as example below:

Sample code:

1. Animal examples

Cat class: public class Cat extends Pet implements Serializable {…}

Dog class: public class Dog extends Pet implements Serializable {…}

Rabbit class: public class Rabbit extends Pet implements Serializable {...}

2. Human examples

BoardingStaff class: public class BoardingStaff extends Admin implements Serializable {…}

Receptionist class: public class Receptionist extends Admin implements Serializable {…}

Vet class: public class Vet extends Admin implements Serializable {…}

## 4.2 Encapsulation

Encapsulation is one of the technique of making private fields in a class and providing access to the fields via public methods. (tutorialspoint, n.d.) Encapsulation can also know as data hiding, which often used to limit the accessibility of other outside classes toward some particular data. Encapsulation can be metaphor as a protective shield that prevents the code being randomly accessed by other code defined outside the class. (tutorialspoint, n.d.) There are a number of private field data in this project and 1 of them selected as an example.

Sample Code:

Vet class

private String VetID, Expertise1, Expertise2;

private int Vet\_Counter;

## 4.3 Modularity

Modularity is a technique that subdividing a huge computer programs into separate and small sub-programs. By doing so, similar functions of sub programs are grouped in the same place in the programming world, as separate and special functions are developed as separate unit of code to increase the usability of other applications. (techopedia, 2015)

Sample Code:

Customer Class:

public void readList(List<Customer> list) {...}

ReceptionistMenu Jframe:

public void populateList() {….

customer.readList(CustomerList);

…}

## 4.4 Inheritance

Inheritance in java is a type of mechanism in which one object acquires all the properties and behaviours of parent object. (JavaTpoint, n.d.) In this project, example like Pet Class is the parent of Cat Class, Dog Class and others, every attributes, methods and variables from Pet Class is shared among the classes that inheriting it, hence causing them to have common behaviours and characteristic.

For example:

Sample Code:

public abstract class Pet implements Serializable {…}

Cat Class: public class Cat extends Pet implements Serializable {…}

Dog Class: public class Dog extends Pet implements Serializable {…}

## 4.5 Polymorphism

Polymorphism is defined as an object that can take on multiple forms. The most common polymorphism in OOP is often refers to a usage of a parent class reference to a child class object. Any Java object that can pass more than one IS-A test is considered to be polymorphic. (tutorialspoint, n.d.)There are many type of polymorphism, which are overriding, overloading, casting and others. Types of polymorphism that apply in this project is shown as below:

Sample Code:

i) Overloading

PetInfo Jframe: showing different screen design of PetInfo Screen.

public PetInfo() {…}

public PetInfo(ReceptionistMenu repMenu, String CusID, String Cus\_Name) {…}

public PetInfo(ReceptionistMenu repMenu, String OwnFirstName, String OwnLastName, String OwnGender, String OwnAddress, String OwnDOB, String OwnContactNo, String OwnEmail) {…}

public PetInfo(ReceptionistMenu repMenu, String OwnerName, String petID, String PetName, String Sex, String Species, int PetAge, String Allergies, boolean Boarding) {…}

ii) Overriding:

Pet Class:

public abstract void readCounter();

Cat Class:

public void readCounter() {…}

iii) Casting:

Cat Class:

public void readList(List<Cat> CatList) { …

Cat tmp = (Cat) objectInputFile.readObject();

…}

iv) Upcasting:

VetMenu Jframe:

public static List<Vet> VetList = new LinkedList<>();

public static List<Appointment> AppointmentList = new LinkedList<>();

public static List<Consultation> ConsultList = new LinkedList<>();

public static List<Boarding> BoardingList = new LinkedList<>();

public static List<RotaTable> RotaList = new LinkedList<>();

## 4.6 Extra Features

### 4.6.1 Abstract Class and Abstract Method

An abstract class has a special characteristic which differentiate it from other ordinary classes, which is that it cannot be instantiated. Although it cannot be instantiated, but its other functionality, fields, methods, and constructors are all still able to be accessed. (tutorialspoint, n.d.) Other than that, an abstract class also has the capability to store normal method and abstract method. By having an abstract method, classes that inherit it must have the same method header in their class too. Abstract class and abstract method is used in this project.

Sample Code:

Pet Class:

public abstract class Pet implements Serializable {…

public abstract void readCounter();

public abstract void writeCounter(int counter);

public abstract void generateID(int number);

}

Cat Class:

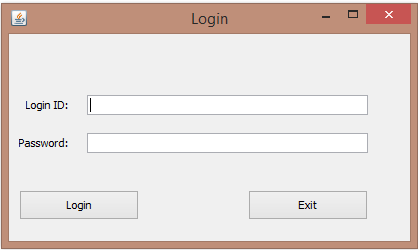
public class Cat extends Pet implements Serializable {

**public void readCounter() {…}**

**public void writeCounter(int counter) {…}**

**public void generateID(int number) {…}**

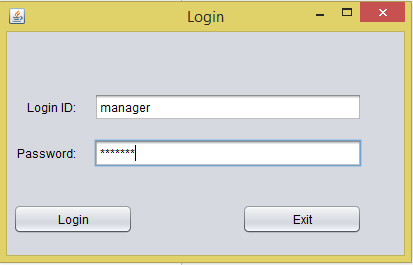
### 4.6.2 GUI



All of the data representation is carry out in graphical user interface, which is more convenient to use.

# 5.0 User Manual

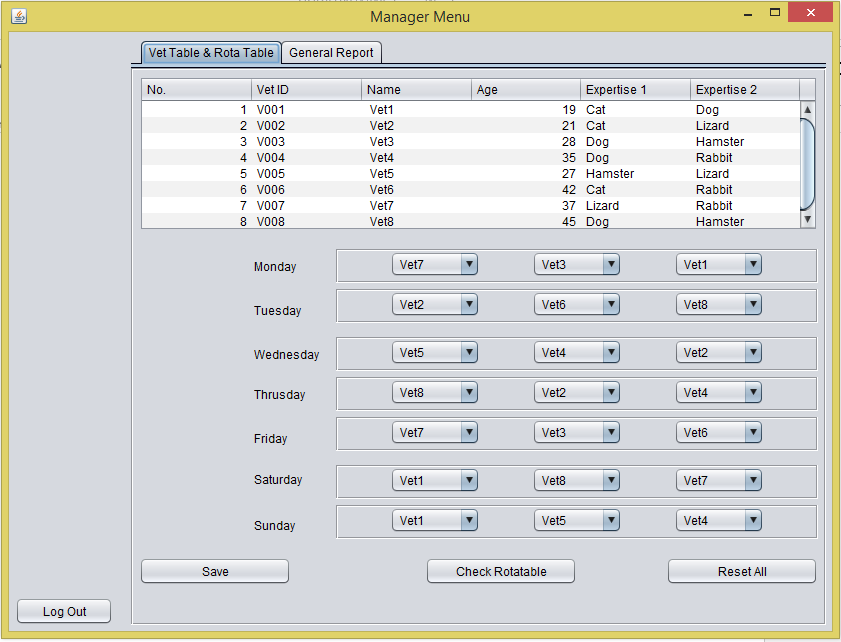
## Login Screen



The users of this system only consists of manager, receptionist, vets and boarding staffs. Each of them are given a special login ID and password that only reserved for them. Security is ensured at this point, so that all of the unauthorized access can be denied. Since each of the user have their specific screen, user must log out after finishes their business in the system.

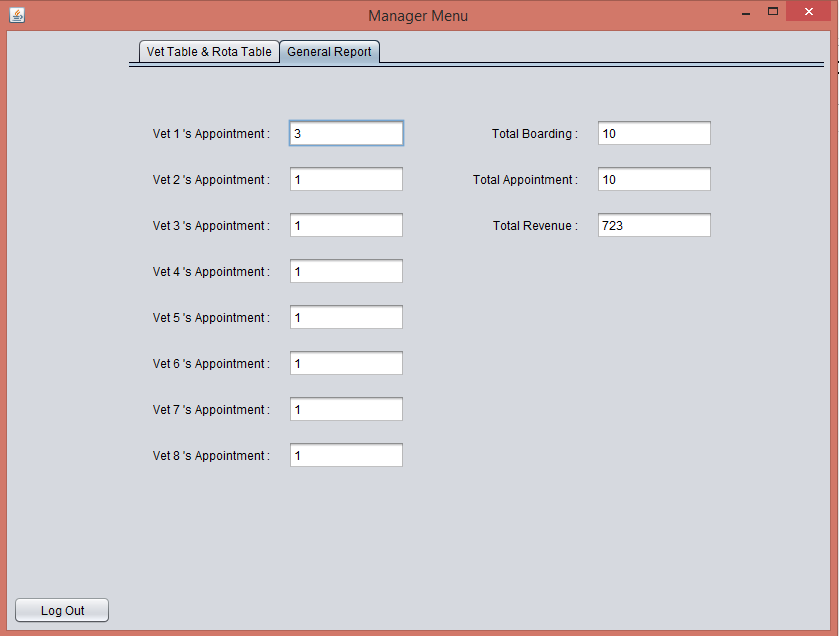
## 1.0 Manager Menu

### 1.1 Show Rota Table Form and Vet Table



After all of the combo boxes are sets with the vets that manager desired, he will need to click Check Rotatable to check the balance of expertizes in the form. A message will be prompted stating which day is not balanced with expertise. Reset all button is provided to reset the rota table back to default. Save button is only clicked when manager decide to set the rota table.

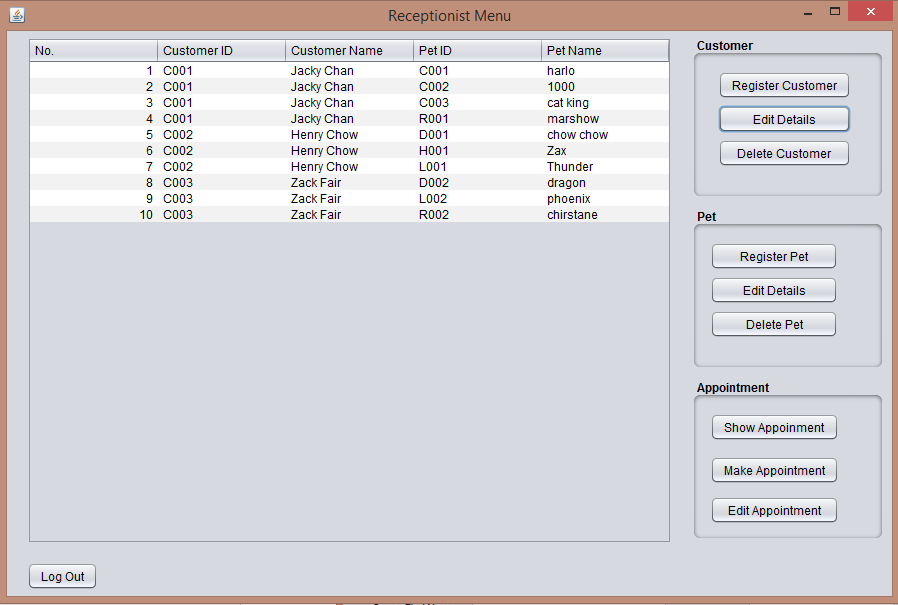
### 1.2 Display General Report



The general report consists of numbers of appointments that each of the vet has, the total appointment made in this systems, total number of boarding services in the system, also the total revenue of the system.

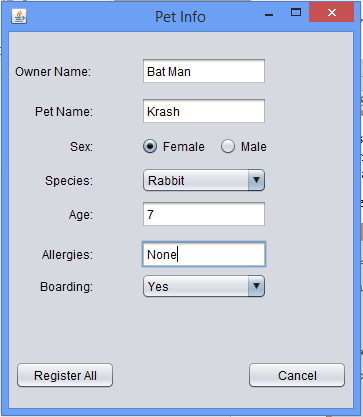
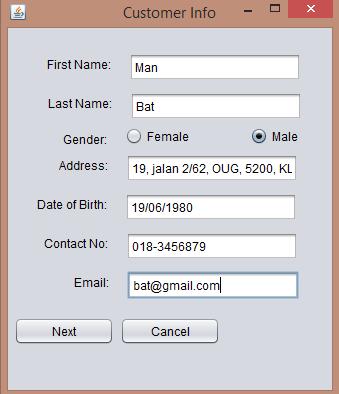
## 2.0 Receptionist Menu

### 2.1 View Customer and Pet List



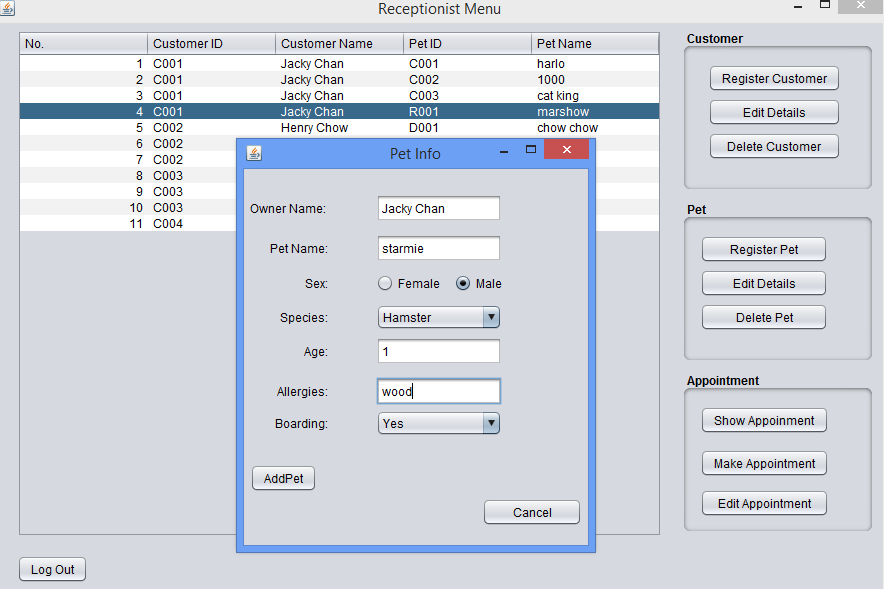
In this screen, every customers and pets of BVCB ‘s are show in the table. Receptionist can also perform activities such as register new customer and others by clicking buttons the right. For Pet panel and Appointment panel, a pet must be selected in the table before the button is clicked.

### 2.2 Register New Customer



After clicking register customer, a small window on the left is pop up. Receptionist must fill in every field so that he or she can proceed after clicking next. Then another window which is the left window will pop up, noticing the receptionist to fill in customer’s pet info. If cancel button is click in the pet info screen, both data of the customer data and pet will be destroyed without saving.

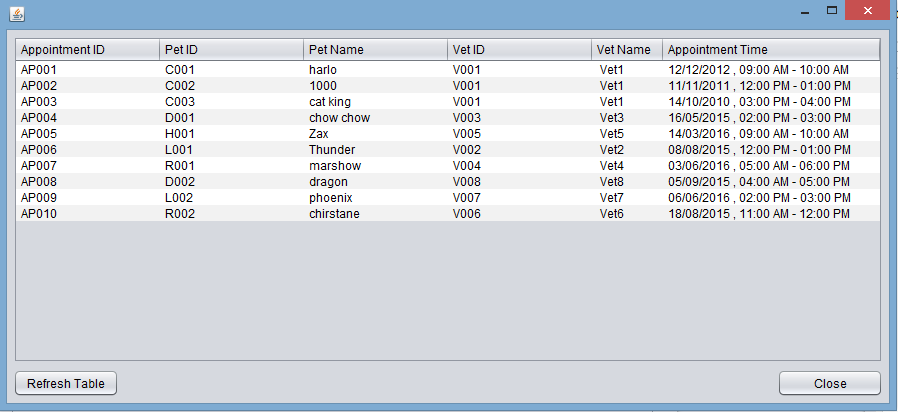
### 2.3 Register Extra Pet



After selecting a customer, click register pet to add another pet under the customer’s data profile.

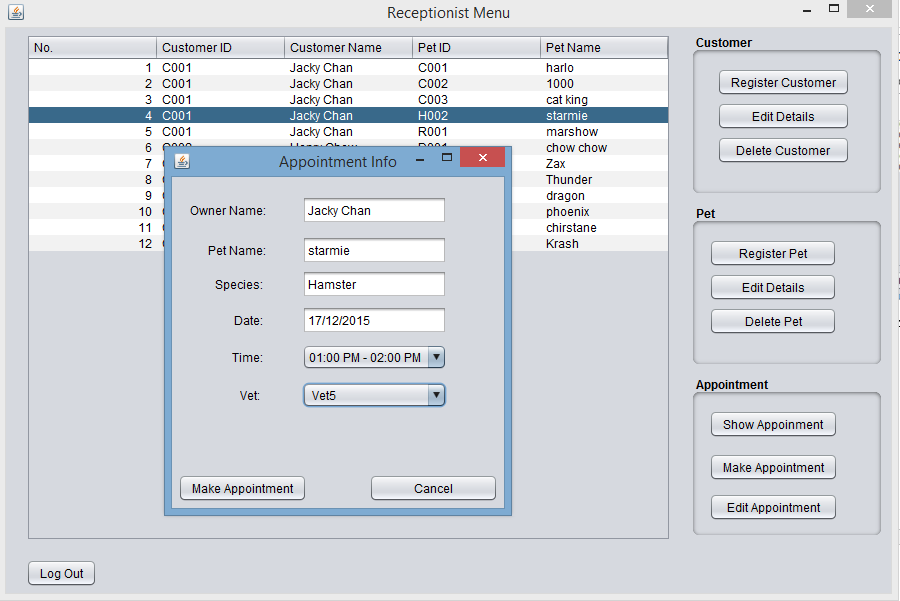
In the pet info screen, receptionist should also select boarding status according to the customer’s request. If boarding is set “Yes”, a boarding data is created; if “No”, the boarding service will not be registered, and if boarding service is preregistered, the record will be deleted.

### 2.4 Show Appointment List



When show appointment button is clicked, appointment list window is pop. Receptionist can check customer’s previous or current appointment of the customer.

### 2.5 Add Appointment

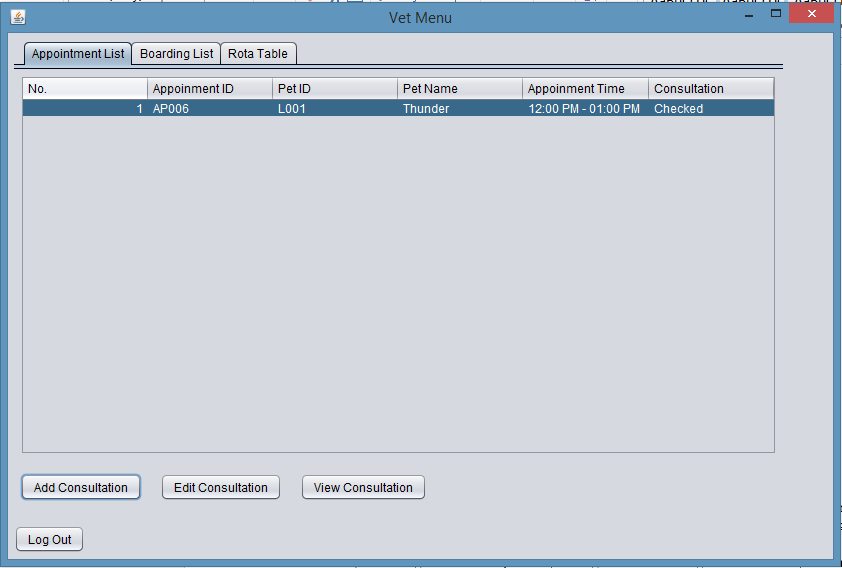


To add an appointment, receptionist must select a pet in the table, then click make appointment button. An appointment info will pop up, and receptionist should complete this form by filling in the date, select an appointment time frame, and then select a vet. After all is done, click make appointment button to finish this process.

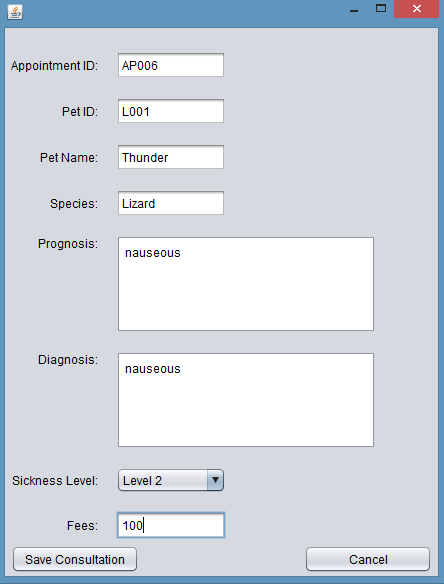
As shown on the right, appointment delete is not supported in this system, as the appointments are used to act as a references record for generating general report.

## 3.0 Vet Menu

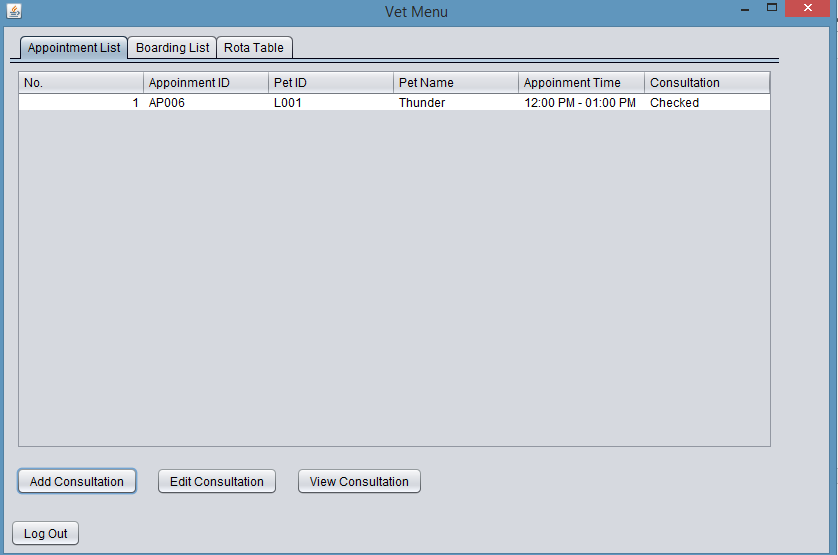
### 3.1 Add Consultation



This first image shows the vet menu. After selecting a pet in the appointment table, add, edit and view consultation is enabled. After selecting add consultation button, a consultation form as second image is pop up.

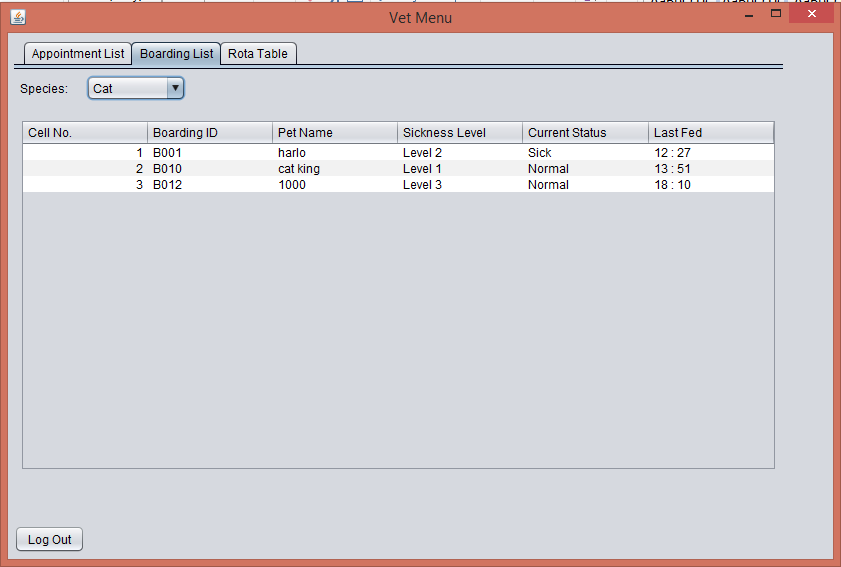


The vet need to fill up prognosis and diagnosis if necessary and rate the sickness of the pet, and also the fees for the consultation. Sickness level is updated in the boarding table after clicking save consultation.



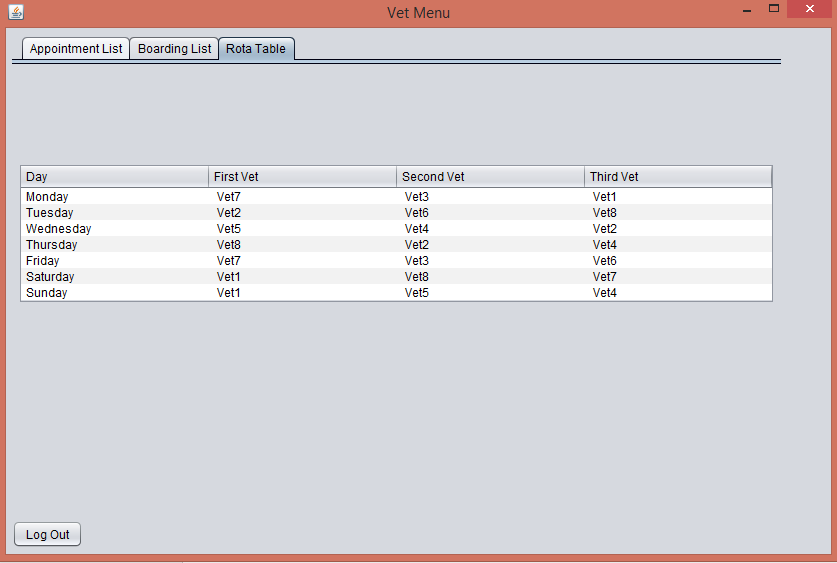
After that, the appointment table is immediately updated with “checked” at the consultation column.

### 3.2 View Boarding List



By selecting the species in the combo box, vet will be able to view different pet’s boarding status.

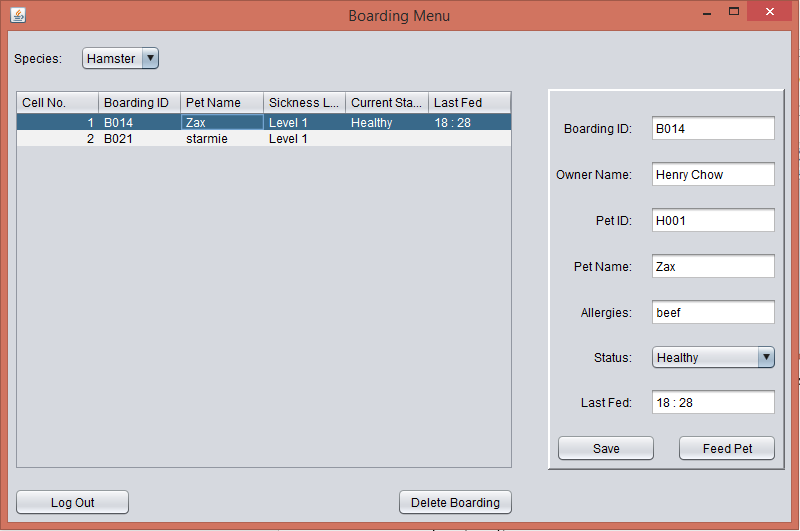
### 3.3 View Rota Table



The vet can also display the rota table, and acknowledging his and his colleague working time table.

## 4.0 Boarding Menu

### 4.1 Update Last Fed and Current Condition



In order to update boarding pet status, boarding staff need to select a pet. The form on the right will automatically fill itself but leaving the status and last fed text field empty. After feeding the pet, boarding staff should click feed pet to fill the last fed text field. Boarding staff should also select a status for the pet according to its current conditions. Finally, save button is clicked to update the boarding table and the boarding list database.

# 6.0 Limitation

1. GUI is poorly designed and full of flaws.

2. Lack of validation capability on user’s input.

3. Login function is not complete.

4. Incapability to generate monthly revenue according to month

5. OOP concept is not fully implemented to carry out the efficiency of the system.

# 7.0 Future enhancement

There are still a lot of effort needed in order to put this system to its final form. Future programmers might able to capture every input error, hence increasing the validity of date save into the database. Database can also introduced to substitute binary files as binary files have very low readability. Furthermore, GUI designed can be improve by adding colours and images to make the system more attractive.

# 8.0 Conclusion

Conclusion, implementing OOP concept into this project helps to improve understanding and implementation them in order to carry out more efficiency while constructing a system. Acquisition of these skill will absolutely reduce the difficulty and complexity in programming a system in the future.

# 9.0 References

Anon., 2015. *Beginner's Book.* [Online]   
Available at: http://beginnersbook.com/2013/03/oops-in-java-encapsulation-inheritance-polymorphism-abstraction/  
[Accessed 2015].

JavaTpoint, n.d. *Inheritance in Java.* [Online]   
Available at: http://www.javatpoint.com/inheritance-in-java  
[Accessed 2015].

techopedia, 2015. *Modular Programming.* [Online]   
Available at: http://www.techopedia.com/definition/25972/modular-programming  
[Accessed 2015].

tutorialspoint, n.d. *Java - Abstraction.* [Online]   
Available at: http://www.tutorialspoint.com/java/java\_abstraction.htm  
[Accessed 2015].

tutorialspoint, n.d. *Java - Encapsulation.* [Online]   
Available at: http://www.tutorialspoint.com/java/java\_encapsulation.htm  
[Accessed 2015].

tutorialspoint, n.d. *Java - Polymorphism.* [Online]   
Available at: http://www.tutorialspoint.com/java/java\_polymorphism.htm  
[Accessed 2015].