SMART PRESCRIPTION APPLICATION

Test Plan

By

Mr. Natthakan Kaeokanpai 552115020
Mr. Phithiwat Sitthitun 552115051

Department of Software Engineering
College of Arts, Media and Technology
Chiang Mai University

Project Advisor

_____
Dr.Prompong Sugunnasil

Document History

Document Name	Version	Status	Date	Viewable	Editable	Responsible
Documents						
Smart Prescription Application _Test Plan	Software Test Plan_V1.0	Reviewed	June 10th 2016	NAT,PHI, PROM	NAT,PHI	NAT,PHI
Smart Prescription Application _Test Plan	Software Test Plan_V1.1	Released	July 10th 2016	NAT,PHI, PROM	NAT,PHI	NAT,PHI
Smart Prescription Application _Test Plan	Software Test Plan_V2.0	Released	July 30th 2016	NAT,PHI, PROM	NAT,PHI	NAT,PHI

Acronym
NAT = Natthakan Kaeokanpai
PHI = Phithiwat Sitthitun
PROM = Prompong Sugunnasil

Table of Contents

Chapter One | Introduction 5
1.1 Objectives 5
1.2 Scope 5
1.3 Acronyms and Definitions 5
1.3.1 Acronyms 5
1.3.2 Definitions 5
Chapter Two | Test Plan and Test Procedure 6
2.1 Test Objectives 6
2.2 Scope of Testing 6
2.3 Test Duration 6
2.4 Test Responsibility 6
2.5 Test Strategy 6
2.6 Result of Testing 7
2.7 Test Environment 7
2.7.1 Hardware 7
2.7.2 Software 7
Chapter Three | Unit Testing 8
3.1 Class SiteController 8
3.1.1 Unit Test Case (UTC-01): login(String username, String password) : Boolean 8
3.1.2 Unit Test Case (UTC-02): logout() : Boolean 8
3.2 Class UserController 9
3.2.1 Unit Test Case (UTC-03): getUserList() : ArrayList 9
3.2.2 Unit Test Case (UTC-04): viewUser(Int userId) : Mixed 10
3.2.3 Unit Test Case (UTC-05): createUser(User user) : Boolean 10
3.2.4 Unit Test Case (UTC-06): updateUser(User user) : Boolean 12
3.2.5 Unit Test Case (UTC-07): deleteUser(Int userId) : Boolean 13
3.2.6 Unit Test Case (UTC-08): searchUser(String users) : ArrayList 14
3.3 Class PatientController 16
3.3.1 Unit Test Case (UTC-09): getPatientList() : ArrayList 16
3.3.2 Unit Test Case (UTC-10): viewPatient(Int personal_id) : Mixed 17
3.3.3 Unit Test Case (UTC-11): createPatient(Patient patient) : Boolean 18
3.3.4 Unit Test Case (UTC-12): updatePatient(Patient patient) : Boolean 20
3.3.5 Unit Test Case (UTC-13): deletePatient(Int personal_id) : Boolean 22
3.3.6 Unit Test Case (UTC-14): searchPatient(String patients) : ArrayList 22
3.4 Class AllergyController 24
3.4.1 Unit Test Case (UTC-15): getAllergyList() : ArrayList 24
3.4.2 Unit Test Case (UTC-16): viewAllergy(Int allergy_id) : Mixed 25
3.4.3 Unit Test Case (UTC-17): getAllergyList2() : ArrayList 26
3.4.4 Unit Test Case (UTC-18)viewAllergy2(Int allergy_id) : Mixed 27
3.4.5 Unit Test Case (UTC-19): createAllergy(Allergy allergy) : Boolean 28
3.4.6 Unit Test Case (UTC-20): updateAllergy(Allergy allergy) : Boolean 29
3.4.7 Unit Test Case (UTC-21): deleteAllergy(Int allergy_id) : Boolean 30
3.4.8 Unit Test Case (UTC-22): searchAllergy(String allergies) : ArrayList 31
3.5 Class LoginEvenManager 33
3.5.1 Unit Test Case (UTC-23): Login() : void 33
3.6 Class PreferencesManager 33
3.6.1 Unit Test Case (UTC-24): getUserInformation() : UserInformationModel 33
3.7 Class QRCodeEncoder 34
3.7.1 Unit Test Case (UTC-25): encode(String contents, BarcodeFormat format, int width, int height): BitMatrix 34

Chapter Four | System Testing 35
System Test Case (STC-01): Doctors can view a doctor home page on the web application. 35
System Test Case (STC-02): Doctors can create patient's profiles on the web application application by inputting personal id, name, surname, prescription or username. 37
System Test Case (STC-03): Doctors can update patient's profiles on the web application application by inputting personal id, name, surname, prescription and username. 42
System Test Case (STC-04): Doctors can delete patient's profiles on the web application. 47
System Test Case (STC-05): Doctors can search patient's profiles on the web application by using personal id, name or surname. 50
System Test Case (STC-06): Doctors can view patient's profiles on the web application. 54
System Test Case (STC-07): Doctors can view a list of patient's profiles on the web application. 56
System Test Case (STC-08): Administrators can view an admin home page on the web application. 58
System Test Case (STC-09): Administrators can add user's profiles on the web application by inputting name, surname, username, password, confirm-password and position. 59
System Test Case (STC-10): Administrators can update user's profiles on the web application by inputting name, surname, username, password, confirm-password and position. 64
System Test Case (STC-11): Administrators can delete user's profiles on the web application. 69
System Test Case (STC-12): Administrations can search user's profiles on the web application by using name, surname or username. 72
System Test Case (STC-13): Administrations can view user's profiles on the web application. 76
System Test Case (STC-14): Administrations can view a list of user's profiles on the web application. 77
System Test Case (STC-15): Doctors, Administrations, and FDAs can login to the web application. 79
System Test Case (STC-16): Doctors, Administrations, and FDAs can logout from the web application. 82
System Test Case (STC-17): Doctors can create allergy report on the web application by inputting personal id, name, surname, and allergy drug.....83
System Test Case (STC-18): Doctors can update an allergy report on the web application by inputting personal id, name, surname, and allergy drug.....87
System Test Case (STC-19): Doctors can delete an allergy report on the web application. 91
System Test Case (STC-20): Doctors can search allergy reports on the web application by using personal id, name or surname.....94
System Test Case (STC-21): Doctors can view an allergy report on the web application. 98
System Test Case (STC-22): Doctors can view list of allergy reports on the web application. 100
System Test Case (STC-23): FDAs can view a FDA home page on the web application. 102
System Test Case (STC-24): FDAs can view an allergy report on the web application. 103
System Test Case (STC-25): FDAs can view list of allergy reports on the web application. ...105
System Test Case (STC-26): Patients can login to the mobile application by using username and password. 107
System Test Case (STC-27): Patients can logout from the mobile application. 109
System Test Case (STC-28): Patients can view the QR code on the mobile application. 111
System Test Case (STC-29): Pharmacists can scan the QR code on the mobile application. ... 112
System Test Case (STC-30): Pharmacists can add the time of dispensation to patient's profile on the mobile application. 115
Chapter Five | Appendix.....116
Appendix A 116
Chapter Six | Reference.....118

Chapter One | Introduction

1.1 Objectives

The objective of the Software Design Document for "Smart prescription application" both Android mobile application and web application are to provide a description and explain of the design system including activity following requirements specification and before programming. This document is based on the project proposal, project plan and software requirement specification and help developers to understand how to work for this application.

1.2 Scope

Smart prescription application consists of a web application and a mobile application. Smart prescription application requires a web application for creating a patient's profile and allergy report by a doctor and the Smart prescription application require a mobile application to generate and scan the QR code on a mobile phone by a pharmacist.

The features of the web application consist of:

Feature#1: Prescription system
Feature#2: Account management system
Feature#3: Authentication system
Feature#5: Report allergy system
Feature#6: Allergy drug summary report system

The features of the mobile application consist of:

Feature#4: Prescription verification system

1.3 Acronyms and Definitions

1.3.1 Acronyms

UTC	Unit Test Case
STC	System Test Case
SRS	Software Requirement Specification
URS	User Requirement Specification
SDD	Software Design Document
UI	User Interface

1.3.2 Definitions

Feature — Transformation of input parameters to output parameters based on a specified algorithm. It describes the functionality of a product in the language of the product. Used for requirements analysis, design, coding, testing or maintenance.

Unit test — A unit of work is a single logical functional use case in the system that can be invoked by some public interface (in most cases). A unit of work can span a single method, a whole class or multiple classes working together to achieve one single logical purpose that can be verified. [1]

Chapter Two | Test Plan and Test Procedure

2.1 Test Objectives

The objectives of testing “Smart Prescription Application” application are:

- To detect and fix all bugs and defects.
- To cover all requirements that is functional, non-function, or user interface-related.
- To gain the acceptance and approval of the stakeholders in the test effort.
- To direct, guide and constrain the test effort, focusing the work on the useful and necessary deliverables.

2.2 Scope of Testing

“Smart Prescription Application” application will be tested using both white-box testing technique (i.e. unit testing) and black-box testing technique (i.e. system testing). The test results will be documented in the Test Record.

2.3 Test Duration

Date
Perform date: July, 16th 2016 **Duration:** July, 30th 2016

2.4 Test Responsibility

Item	Responsibility
Unit test of “Smart Prescription Application” web application	Natthakan Kaeokanpai, Phithiwat Sitthitun
System test of “Smart Prescription Application” web application	Natthakan Kaeokanpai, Phithiwat Sitthitun
Unit test of “Smart Prescription Application” mobile application	Natthakan Kaeokanpai, Phithiwat Sitthitun
System test of “Smart Prescription Application” mobile application	Natthakan Kaeokanpai, Phithiwat Sitthitun

2.5 Test Strategy

“Smart Precription Application” application testing is performed in following steps:

- Design test case for each feature.
- Prepare test data for each feature.
- Determine expected results.
- Perform testing on individual features.
- Record testing result.
- Store all test files in the project repository.

2.6 Result of Testing

All the test results will be documented in the Test Record and separated into two parts, which are:

1. Actual output: The actual output that is produced by each test case.
2. Test Pass/Fail criteria:
   2.1 Test Pass: The actual result is same as expected result.
   2.2 Test Fail: The actual result is not same as expected result.

2.7 Test Environment

2.7.1 Hardware

- Internet

- Laptops

1. Dell inspiron 5220
   Processor: Intel Core i5 3210M (3.1 GHz)
   Processor Memory: 8GB DDR3
   Memory Hard Disk: 1000GB
   Operating System: Window 7 Home Basic
2. Asus X550DP-DS101
   AMD A-Series A10-5750M (2.50GHz) 8GB
   Memory 1TB
   HDD AMD Radeon
   Windows 8 64-Bit

- Mobile phone: Android Operating System with camera

1. Asus zenfone 4.5
   Android 4.4.2 (KitKat)
   Processor: Intel Atom Z2520DualCore
   CPU Speed: 1.2 GHz
   Memory 8GB (Internal)
   RAM 1GB
2. Samsung galaxy J7
   OS: Android OS, v5.1 (Lollipop)
   Processor: Quad-core 1.4 GHz Cortex-A53 & quad-core 1.0 GHz Cortex-A53
   Octa-core 1.5 GHz Memories 16 GB, 1.5 GB RAM 8GB (Internal)

2.7.2 Software

1. Php Storm 8.3
2. Google Chrome
3. Android studio 2.1
4. Visual paradigm
5. Xampp

Chapter Three | Unit Testing

3.1 Class SiteController

3.1.1 Unit Test Case (UTC-01): login(String username, String password) : Boolean

Description: Test login(String username, String password) method for checking username and password that are exist in the user table.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result
1	Test method by input username and password are exist in the user table.	username = “admin” password = “admin”	True
2	Test method by input username and password do not exist in the user table.	username = “admin12” password = “admin12”	False
3	Test method by input only username is exist in the user table.	username = “admin” password = “admin12”	False
4	Test method by input only password is exist in the user table.	username = “admin12” password = “admin”	False
5	Test method by input username without password.	username = “admin” password = Null	False
6	Test method by input password without username.	username = Null password = “admin”	False

3.1.2 Unit Test Case (UTC-02): logout() : Boolean

Description: Test logout() method for checking web application can action to log out the current user.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result
1	Test logout is active when logout() is invoked.	-	True

3.2 Class UserController

3.2.1 Unit Test Case (UTC-03): getUserList() : ArrayList

Description: Test getUserList() method for checking whether the method get right list of user on the web application. The expected result will be arraylist of user.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	The method should return the right arraylist of user from the database.	-	ArrayList<User>{
			User{
			id = 15,
			name = “Phithiwat”,
			surname = “Sitthitun”,
			username = “phithi”,
			password = “1234”,
			pssword2 = “1234”,
			roleId = 1
			}
			User{
			id = 2,
			name = “Natthakan”,
			surname = “Kaeokanpai”,
			username = “natthakan”,
			password = “man”,
			password2 = “man”,
			roleId = 2
			}
			User{
			id = 3,
			name = “Tony”,
			surname = “Stark”,
			username = “tony”,
			password = “tony”,
			password2 = “tony”,
			roleId = 3
			}
			}

3.2.2 Unit Test Case (UTC-04): viewUser(Int userId) : Mixed

Description: Test viewUser(Int userId) method for checking whether the method get right user model object releted to user id. The expected result will be user model object.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input user id that exist in the databse. The method should return the right user object related to user id.	userId = 15	User{
			id = 15,
			name = “Phithiwat”,
			surname = “Sitthitun”,
			username = “phithi”,
			password = “1234”,
			pssword2 = “1234”,
			roleId = 1
			}

2	Test method by input user id that not exist in the database	userId = 99	False
3	Test method by not input user id.	Null	False
4	Test method by input user id with string value.	userId = “phithiwat”	False

3.2.3 Unit Test Case (UTC-05): createUser(User user) : Boolean

Description: Test createUser(User user) method for checking whether the method add the right data into the database. The data is includes name, surname, username, password, password2 and roleId.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input new data.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = “1234”,
		password2 = “1234”,
		roleId = 1
		}
			True

2	Test method by input the data that already exist. (username)	User{
		name = “Tony”,
		surname = “Stark”,
		username = “steve”,
		password = “tony”,
		password2 = “tony”,
		roleId = 2
		}
			False

3	Test method by input without name.	User{
		name = Null,
		surname = “Rogers”,
		username = “steve”,
		password = “1234”,
		password2 = “1234”,
		roleId = 1
		}
			False

4	Test method by input without surname.	User{
		name = “Steve”,
		surname = Null,
		username = “steve”,
		password = “1234”,
		password2 = “1234”,
		roleId = 1
		}
			False

5	Test method by input without username.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = Null,
		password = “1234”,
		password2 = “1234”,
		roleId = 1
		}
			False

6	Test method by input without password.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = Null,
		password2 = “1234”,
		roleId = 1
		}
			False

7	Test method by input without password2.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = “1234”,
		password2 = Null,
		roleId = 1
		}
			False

8	Test method by input without roleId.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = “1234”,
		password2 = “1234”,
		roleId = Null
		}
			False

9	Test method by input string.	“test”	False
10	Test method by input empty string.	“”	False
11	Test method by input an integer.	00	False
12	Test method by nothing input data.	Null	False

3.2.4 Unit Test Case (UTC-06): updateUser(User user) : Boolean

Description: Test createUser(User user) method for checking whether the method update the right data into the database. The data is includes name, surname, username, password, password2 and roleId.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input new data.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = “rogers”,
		password2 = “rogers”,
		roleId = 2
		}
			True

2	Test method by input the new data that already exist. (username)	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “phithi”,
		password = “rogers”,
		password2 = “rogers”,
		roleId = 2
		}
			False

3	Test method by input without name.	User{
		name = Null,
		surname = “Rogers”,
		username = “steve”,
		password = “rogers”,
		password2 = “rogers”,
		roleId = 2
		}
			False

4	Test method by input without surname.	User{
		name = “Steve”,
		surname = Null,
		username = “steve”,
		password = “rogers”,
		password2 = “rogers”,
		roleId = 2
		}
			False

5	Test method by input without username.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = Null,
		password = “rogers”,
		password2 = “rogers”,
		roleId = 2
		}
			False

6	Test method by input without password.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = Null,
		password2 = “rogers”,
		roleId = 2
		}
			False

7	Test method by input without password2.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = “rogers”,
		password2 = Null,
		roleId = 2
		}
			False

8	Test method by input without roleId.	User{
		name = “Steve”,
		surname = “Rogers”,
		username = “steve”,
		password = “rogers”,
		password2 = “rogers”,
		roleId = Null
		}
			False

9	Test method by input string.	“test”	False
10	Test method by input empty string.	“”	False
11	Test method by input an integer.	00	False
12	Test method by nothing input data.	Null	False

3.2.5 Unit Test Case (UTC-07): deleteUser(Int userId) : Boolean

Description: Test deleteUser(Int userId) method for checking whether the method delete the user from the database.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result
1	Test method by input use id that exist in the databasw.	userId = 15	True
2	Test method by input user id that not exist in the database.	userId = 99	False

3.2.6 Unit Test Case (UTC-08): searchUser(String users) : ArrayList

Description: Test searchUser(String users) method for checking whether the method can search the user from the user table in the database by using name, surname ro username. The expected result will be arraylist of user.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input name that exist in the user table.	users = “Phithiwat”	ArrayList<User>{
			User{
			id = 15
			name = “Phithiwat”,
			surname = “Sitthitun”,
			username = “phithi”,
			password = “1234”,
			pssword2 = “1234”,
			roleId = 1
			}
			}

2	Test method by input surname that exist in the user table	users = “Stark”	ArrayList<User>{
			User{
			id = 3
			name = “Tony”,
			surname = “Stark”,
			username = “tony”,
			password = “tony”,
			password2 = “tony”,
			roleId = 3
			}
			}

3	Test method by input username that exist in the user table.	users = “natthakan”	ArrayList<User>{
			User{
			id = 2
			name = “Natthakan”,
			surname = “Kaeokanpai”,
			username = “natthakan”,
			password = “man”,
			password2 = “man”,
			roleId = 2
			}
			}

4	Test method by input empty data.	users = “”	ArrayList<User>{
			User{
			id = 15,
			name = “Phithiwat”,
			surname = “Sitthitun”,
			username = “phithi”,
			password = “1234”,
			pssword2 = “1234”,

			roleId = 1
			}
			User{
			id = 2,
			name = “Natthakan”,
			surname = “Kaeokanpai”,
			username = “natthakan”,
			password = “man”,
			password2 = “man”,
			roleId = 2
			}
			User{
			id = 3,
			name = “Tony”,
			surname = “Stark”,
			username = “tony”,
			password = “tony”,
			password2 = “tony”,
			roleId = 3
			}
			}

5	Test method by input data does not exist in the user table.	users = “abcdefg”	Null

3.3 Class PatientController

3.3.1 Unit Test Case (UTC-09): getPatientList() : ArrayList

Description: Test getPatientList() method for checking whether the method get right list of patient on the web application. The expected result will be arraylist of patient.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	The method should return the right arraylist of patient from the database.	-	ArrayList<Patient>{
			Patient{
			personal_id = 1719900241047,
			name = “Somkiat”,
			surname = “Warahala”,
			prescription = “Paracetamal 500ml.”,
			create_by = 31,
			username = “somkiat”,
			password = “54139”,
			create_at = 2016-07-31 18:36:28,
			update_at = 2016-07-31 18:36:28,
			}
			Patient{
			personal_id = 1769900241047,
			name = “Dangdee”,
			surname = “Rakna”,
			prescription = “Vitamin C 90 ml”,
			create_by = 32
			username = “dang”,
			password = “67045”,
			create_at = 2016-07-30 23:55:17,
			update_at = 2016-07-30 23:59:23,
			}
			Patient{
			personal_id = 1660100485726,
			name = “Vikom”,
			surname = “Chomjai”,
			prescription = “Dextrometrophan”,
			create_by = 32,
			username = “vikom”,
			password = “58706”,
			create_at = 2016-08-01 03:39:11,
			update_at = 2016-08-01 03:39:11,
			}
			}

3.3.2 Unit Test Case (UTC-10): viewPatient(Int personal_id) : Mixed

Description: Test viewPatient(Int personal_id) method for checking whether the method get right patient model object releted to personal id. The expected result will be patient model object.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input personal id that exist in the databse.
	The method should return the right patient object related to personal id.
		personal_id = 1719900241047	Patient{
			personal_id = 1719900241047,
			name = “Somkiat”,
			surname = “Warahala”,
			prescription = “Paracetamal 500ml.”,
			create_by = 31,
			username = “somkiat”,
			password = “54139”,
			create_at = 2016-07-31 18:36:28,
			update_at = 2016-07-31 18:36:28
			}

2	Test method by input personal id that not exist in the database	int personal_id = 1111111111111	False
3	Test method by not input personal id.	Null	False
4	Test method by input personal id with string value.	string personal_id = “personal id”	False

3.3.3 Unit Test Case (UTC-11): createPatient(Patient patient) : Boolean

Description: Test createPatient(Patient patient) method for checking whether the method add the right data into the database. The data is includes personal id, name, surname, prescription, and username.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input new data.	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml.”,
		username = “vikom”,
		}
			True

2	Test method by input the data that already exist. (personal id)	Patients{
		personal_id = 1660100485726
		name = “Mana”,
		surname = “Sornjai”,
		prescription = “Aspirin 75ml.”,
		username = “mana”,
		}
			False

3	Test method by input the data that already exist. (username)	Patients{
		personal_id = 1660100485726,
		name = “Mana”,
		surname = “Sornjai”,
		prescription = “Aspirin 750ml.”,
		username = “vikom”,
		}
			False

4	Test method by input without personal id.	Patients{
		personal_id = Null,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml.”,
		username = “vikom”,
		}
			False

5	Test method by input without name.	Patients{
		personal_id = 1660100485726,
		name = Null,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml.”,
		username = “vikom”,
		}
			False

6	Test method by input without surname.	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = Null,
		prescription = “Paracetamal 500ml.”,
		username = “vikom”,
		}
			False

7	Test method by input without prescription.	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = Null,
		username = “vikom”,
		}
			Fasle

8	Test method by input without username.	Patients{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml.”,
		username = Null,
		}
			False

9	Test method by input string.	“test”	False
10	Test method by input empty string.	“”	Flase
11	Test method by input an integer.	123	False
12	Test method by nothing input data.	Null	False

3.3.4 Unit Test Case (UTC-12): updatePatient(Patient patient) : Boolean

Description: Test updatePatient(Patient patient) method for checking whether the method update the right data into the database. The data is includes personal id, name, surname, prescription, and username.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input new data.	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml. Multivitamin, B Complex”,
		username = “vikom”,
		}
			True

2	Test method by input the data that already exist. (personal id)	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml. Multivitamin, B Complex”,
		username = “vikom”,
		}
			False

3	Test method by input the data that already exist. (username)	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml., Multivitamin, B Complex”,
		username = “somkiat”,
		}
			False

4	Test method by input without personal id.	Patient{
		personal_id = Null,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml., Multivitamin, B Complex”,
		username = “vikom”,
		}
			False

5	Test method by input without name.	Patient{
		personal_id = 1660100485726,
		name = Null,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml., Multivitamin, B Complex”,
		username = “vikom”,
		}
			False

6	Test method by input without surname.	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = Null,
		prescription = “Paracetamal 500ml., Multivitamin, B Complex”,
		username = “vikom”,
		}
			False

7	Test method by input without prescription.	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = Null,
		username = “vikom”,
		}
			Fasle

8	Test method by input without username.	Patient{
		personal_id = 1660100485726,
		name = “Vikom”,
		surname = “Chomjai”,
		prescription = “Paracetamal 500ml., Multivitamin, B Complex”,
		username = Null,
		}
			False

9	Test method by input string.	“test”	False
10	Test method by input empty string.	“”	Flase
11	Test method by input an integer.	123	False
12	Test method by nothing input data.	Null	False

3.3.5 Unit Test Case (UTC-13): deletePatient(Int personal_id) : Boolean

Description: Test deletePatient(Int personal_id) method for checking whether the method delete the patient from the database.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result
1	Test method by input personal id that exist in the database.	personal_id = 179900241047	True
2	Test method by input user id that not exist in the database.	personal_id = 1234567890987	False

3.3.6 Unit Test Case (UTC-14): searchPatient(String patients) : ArrayList

Description: Test searchPatient(String patients) method for checking whether the method can search the patient from the patient table in the database by using personal id, name or surname. The expected result will be arraylist of patient.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input personal id that exist in the patient table.	patients = “1660100485726”	ArrayList<Patient>{
			Patient{
			personal_id = 1660100485726
			name = “Vikom”,
			surname = “Chomjai”,
			prescription = “Dextrometrophan”,
			create_by = 32
			username = “vikom”,
			password = “58706”,
			create_at = 2016-08-01 03:39:11
			update_at = 2016-08-01 03:39:11
			}
			}

2	Test method by input name that exist in the user table.	patients = “Somkiat”	ArrayList<Patient>{
			Patient{
			personal_id = 1719900241047
			name = “Somkiat”,
			surname = “Warahala”,
			prescription = “Paracetamal 500ml.”,
			create_by = 31
			username = “somkiat”,
			password = “54139”,
			create_at = 2016-07-31 18:36:28
			update_at = 2016-07-31 18:36:28
			}
			}

3	Test method by input surname that exist in the user table	patients = “Rakna”	ArrayList<Patient>{
			Patient{
			personal_id = 1769900241047
			name = “Dangdee”,
			surname = “Rakna”,
			prescription = “Vitamin C 90 ml”,
			create_by = 32
			username = “dang”,
			password = “67045”,
			create_at = 2016-07-30 23:55:17
			update_at = 2016-07-30 23:59:23
			}
			}

4	Test method by input empty data.	patiens = “”	ArrayList<Patient>{
			Patient{
			personal_id = 1719900241047,
			name = “Somkiat”,
			surname = “Warahala”,
			prescription = “Paracetamal 500ml.”,
			create_by = 31,
			username = “somkiat”,
			password = “54139”,
			create_at = 2016-07-31 18:36:28,
			update_at = 2016-07-31 18:36:28,
			}
			Patient{
			personal_id = 1769900241047,
			name = “Dangdee”,
			surname = “Rakna”,
			prescription = “Vitamin C 90 ml”,
			create_by = 32
			username = “dang”,
			password = “67045”,
			create_at = 2016-07-30 23:55:17,
			update_at = 2016-07-30 23:59:23,
			}
			Patient{
			personal_id = 1660100485726,
			name = “Vikom”,
			surname = “Chomjai”,
			prescription = “Dextrometrophan”,
			create_by = 32,
			username = “vikom”,
			password = “58706”,
			create_at = 2016-08-01 03:39:11,
			update_at = 2016-08-01 03:39:11,
			}
			}

5	Test method by input data does not exist in the patient table.	patients = “qwert”	Null

3.4 Class AllergyController

3.4.1 Unit Test Case (UTC-15): getAllergyList() : ArrayList

Description: Test getAllergyList() method for checking whether the method get right list of allergy report on the web application. The expected result will be arraylist of allergy.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	The method should return the right arraylist of allergy report from the database.	-	ArrayList<Allergy>{
			Allergy{
			allergy_id = 8,
			personal_id = 1759900241047,
			name = “Phithiwat”,
			surname = “Sitthitun”,
			allergy_drug = “Pyramidone, co-trimoxazole, aspirin”,
			reporter = 32,
			create_at = 2016-07-30 04:15:11,
			update_at = 2016-07-31 04:42:21,
			}
			Allergy{
			allergy_id = 10,
			personal_id = 1600100485726,
			name = “Natthakan”,
			surname = “Kaeokanpai”,
			allergy_drug = “Vitamin C”,
			reporter = 32,
			create_at = 2016-07-31 04:20:43,
			update_at = 2016-07-31 04:20:43,
			}
			Allergy{
			allergy_id = 11,
			personal_id = 1779900241047,
			name = “Somjai”,
			surname = “Ngaila”,
			allergy_drug = “Multivitamins”,
			reporter = 31,
			create_at = 2016-07-31 18:47:11,
			update_at = 2016-07-31 18:47:11,
			}
			}

3.4.2 Unit Test Case (UTC-16): viewAllergy(Int allergy_id) : Mixed

Description: Test viewAllergy(Int allergy_id) method for checking whether the method get right allergy model object releted to allergy id. The expected result will be allergy model object.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input allergy id that exist in the databse. The method should return the right allergy object related to allergy id.	int allergy_id = 11	Allergy{
			allergy_id = 11,
			personal_id = 1779900241047,
			name = “Somjai”,
			surname = “Ngaila”,
			allergy_drug = “Multivitamins”,
			reporter = 31,
			create_at = 2016-07-31 18:47:11,
			update_at = 2016-07-31 18:47:11
			}

2	Test method by input allergy id that not exist in the database	int allergy_id = 99	False
3	Test method by not input allergy id.	Null	False
4	Test method by input allergy id with string value.	string allergy_id = “allergyid”	False

3.4.3 Unit Test Case (UTC-17): getAllergyList2() : ArrayList

Description: Test getAllergyList2() method for checking whether the method get right list of allergy report on the web application. The expected result will be arraylist of allergy.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	The method should return the right arraylist of allergy report from the database.	-	ArrayList<Allergy>{
			Allergy{
			allergy_id = 8,
			personal_id = 1759900241047,
			name = “Phithiwat”,
			surname = “Sitthitun”,
			allergy_drug = “Pyramidone, co-trimoxazole, aspirin”,
			reporter = 32,
			create_at = 2016-07-30 04:15:11,
			update_at = 2016-07-31 04:42:21
			}
			Allergy{
			allergy_id = 10,
			personal_id = 1600100485726,
			name = “Natthakan”,
			surname = “Kaeokanpai”,
			allergy_drug = “Vitamin C”,
			reporter = 32,
			create_at = 2016-07-31 04:20:43,
			update_at = 2016-07-31 04:20:43
			}
			Allergy{
			allergy_id = 11,
			personal_id = 1779900241047,
			name = “Somjai”,
			surname = “Ngaila”,
			allergy_drug = “Multivitamins”,
			reporter = 31,
			create_at = 2016-07-31 18:47:11,
			update_at = 2016-07-31 18:47:11,
			}
			}

3.4.4 Unit Test Case (UTC-18)viewAllergy2(Int allergy_id) : Mixed

Description: Test viewAllergy2(Int allergy_id) method for checking whether the method get right allergy model object releted to allergy id. The expected result will be allergy model object.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input allergy id that exist in the databse. The method should return the right allergy object related to allergy id.	allergy_id = 8	Allergy{
			allergy_id = 8,
			personal_id = 1759900241047,
			name = “Phithiwat”,
			surname = “Sitthitun”,
			allergy_drug = “Pyramidone, co-trimoxazole, aspirin”,
			reporter = 32,
			create_at = 2016-07-30 04:15:11,
			update_at = 2016-07-31 04:42:21,
			}

2	Test method by input allergy id that not exist in the database	allergy_id = 00	False
3	Test method by not input allergy id.	Null	False
4	Test method by input allergy id with string value.	allergy_id = “allergyreport”	False

3.4.5 Unit Test Case (UTC-19): createAllergy(Allergy allergy) : Boolean

Description: Test createAllergy(Allergy allergy) method for checking whether the method add the right data into the database. The data is includes personal id, name, surname, and allery drugs.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input new data.	Allergy{
		personal_id = 1600100485726,
		name = “Natthakan”,
		surname = “Kaeokanpai”,
		allergy_drug = “Vitamin C”,
		}
			True

2	Test method by input the data that already exist. (personal id)	Allergy{
		personal_id = 1600100485726,
		name = “Phithiwat”,
		surname = “Sitthitun”,
		allergy_drug = “Pyramidone, Co-Trimoxazole, Aspirin”,
		}
			False

3	Test method by input without personal id.	Allergy{
		personal_id = Null,
		name = “Natthakan”,
		surname = “Kaeokanpai”,
		allergy_drug = “Vitamin C”,
		}
			False

4	Test method by input without name.	Allergy{
		personal_id = 1600100485726,
		name = Null,
		surname = “Kaeokanpai”,
		allergy_drug = “Vitamin C”,
		}
			False

5	Test method by input without surname.	Allergy{
		personal_id = 1600100485726,
		name = “Natthakan”,
		surname = Null,
		allergy_drug = “Vitamin C”,
		}
			False

6	Test method by input without allergy drug.	Allergy{
		personal_id = 1600100485726
		name = “Natthakan”,
		surname = “Kaeokanpai”,
		allergy_drug = Null
		}
			False

7	Test method by input string.	“Allergy”	False
8	Test method by input empty string.	“”	Flase

9	Test method by input an integer.	500	False
10	Test method by nothing input data.	Null	False

3.4.6 Unit Test Case (UTC-20): updateAllergy(Allergy allergy) : Boolean

Description: Test updateAllergy(Allergy allergy) method for checking whether the method update the right data into the database. The data is includes personal id, name, surname, and allergy drug.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input new data.	Allergy{
		personal_id = 1600100485726,
		name = “Natthakan”,
		surname = “Kaeokanpai”,
		allergy_drug = “Vitamin C, Multivitamins”,
		}
			True

2	Test method by input the data that already exist. (personal id)	Allergy{
		personal_id = 1759900241047,
		name = “Natthakan”,
		surname = “Kaeokanpai”,
		allergy_drug = “Vitamin C, Multivitamins”,
		}
			False

3	Test method by input without personal id.	Allergy{
		personal_id = Null,
		name = “Natthakan”,
		surname = “Kaeokanpai”,
		allergy_drug = “Vitamin C, Multivitamins”,
		}
			False

4	Test method by input without name.	Allergy{
		personal_id = 1600100485726,
		name = Null,
		surname = “Kaeokanpai”,
		allergy_drug = “Vitamin C, Multivitamins”,
		}
			False

5	Test method by input without surname.	Allergy{
		personal_id = 1600100485726,
		name = “Natthakan”,
		surname = Null,
		allergy_drug = “Vitamin C”,
		}
			False

6	Test method by input without allergy drug.	Allergy{
		personal_id = 1600100485726
		name = “Natthakan”,
		surname = “Kaeokanpai”,
		allergy_drug = Null
		}
			False

7	Test method by input string.	“Allergy”	False
8	Test method by input empty string.	“”	Flase
9	Test method by input an integer.	500	False
10	Test method by nothing input data.	Null	False

3.4.7 Unit Test Case (UTC-21): deleteAllergy(Int allergy_id) : Boolean

Description: Test deleteAllergy(Int allergy_id) method for checking whether the method delete the allergy from the database.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result
1	Test method by input allergy id that exist in the database.	allergy_id = 8	True
2	Test method by input allergy id that not exist in the database.	allergy_id = 789	False

3.4.8 Unit Test Case (UTC-22): searchAllergy(String allergies) : ArrayList

Description: Test searchAllergy(String allergies) method for checking whether the method can search the allergy report from the allergy table in the database by using personal id, name or surname. The expected result will be arraylist of allergy report.

Prerequisites or Test input: The test data is available at Appendix A.

Test cases:

Test case ID	Description	Input	Expected Result

1	Test method by input personal id that exist in the allergy table.	allergies = “1759900241047”	ArrayList<Allergy>{
			Allergy{
			allergy_id = 8
			personal_id = 1759900241047
			name = “Phithiwat”,
			surname = “Sitthitun”,
			allergy_drug = “Pyramidone, co-trimoxazole, aspirin”,
			reporter = 32
			create_at = 2016-07-30 04:15:11
			update_at = 2016-07-31 04:42:21
			}
			}

2	Test method by input name that exist in the user table.	allergies = “Somjai”	ArrayList<Allergy>{
			Allergy{
			allergy_id = 11
			personal_id = 1779900241047
			name = “Somjai”,
			surname = “Ngaila”,
			allergy_drug = “Multivitamins”,
			reporter = 31
			create_at = 2016-07-31 18:47:11
			update_at = 2016-07-31 18:47:11
			}
			}

3	Test method by input surname that exist in the user table	allergies = “Kaeokanpai”	ArrayList<Allergy>{
			Allergy{
			allergy_id = 10
			personal_id = 1600100485726
			name = “Natthakan”,
			surname = “Kaeokanpai”,
			allergy_drug = “Vitamin C”,
			reporter = 32
			create_at = 2016-07-31 04:20:43
			update_at = 2016-07-31 04:20:43
			}
			}

4	Test method by input empty data.	allergies = “”	ArrayList<Allergy>{
			Allergy{
			allergy_id = 8,
			personal_id = 1759900241047,
			name = “Phithiwat”,
			surname = “Sitthitun”,
			allergy_drug = “Pyramidone, co-trimoxazole, aspirin”,
			reporter = 32,
			create_at = 2016-07-30 04:15:11,
			update_at = 2016-07-31 04:42:21,
			}
			Allergy{
			allergy_id = 10,
			personal_id = 1600100485726,
			name = “Natthakan”,
			surname = “Kaeokanpai”,
			allergy_drug = “Vitamin C”,
			reporter = 32,
			create_at = 2016-07-31 04:20:43,
			update_at = 2016-07-31 04:20:43,
			}
			Allergy{
			allergy_id = 11,
			personal_id = 1779900241047,
			name = “Somjai”,
			surname = “Ngaila”,
			allergy_drug = “Multivitamins”,
			reporter = 31,
			create_at = 2016-07-31 18:47:11,
			update_at = 2016-07-31 18:47:11,
			}
			}

5	Test method by input data does not exist in the allergy table.	allergies = “aaaaa”	Null

3.5 Class LoginEvenManager

3.5.1 Unit Test Case (UTC-23): Login() : void

Description: Test for check evaluation result of procedure when the user login to the system, the system will return ['status'] = 1 when user login by input both username and password in the text field and return ['status'] = 0 when user login only username or password or did not input both of username and password.
Test cases:

Test case ID	Description	Input	Expected Result
1	Test method Login() by input a username and password.	username = "username"; password = "password";	['status'] = 1
2	Test method Login() by input a username without password.	username = "username"; password = Null	['status'] = 0
3	Test method Login() by input a password without username.	username = Null password = "password";	['status'] = 0
4	Test method Login() by input a username and password.	username = Null password = Null	['status'] = 0

3.6 Class PreferencesManager

3.6.1 Unit Test Case (UTC-24): getUserInformation() : UserInformationModel

Description: Test for check the result of getUserInformation () that can get the information by the result will be not Null.
Test cases:

Test case ID	Description	Input	Expected Result
1	Test method getUserInformation() that can get the information by input UserInformationModel.	UserInformationModel	UserInformationModel is not Null

3.7 Class QRCodeEncoder

3.7.1 Unit Test Case (UTC-25): encode(String contents, BarcodeFormat format, int width, int height): BitMatrix

Description: Test for check the method encode() can input the value and then return the same input value by input .

Test cases:

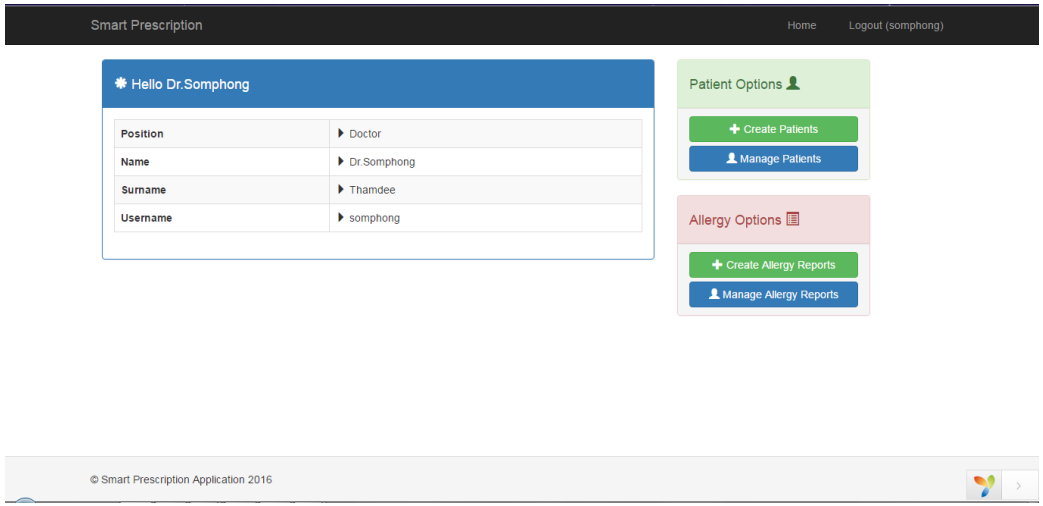
Test case ID	Description	Input	Expected Result
1	Test method encode() that can return the same value by input the value (String = "contents", BarcodeFormat = “format”, int = “9”,int = “9”)	BitMatrix {String = "contents", BarcodeFormat = “format”, int = “9”,int = “9”)}	BitMatrix {contents = "contents", BarcodeFormat = “format”, int = “9”,int = “9”)}

Chapter Four | System Testing

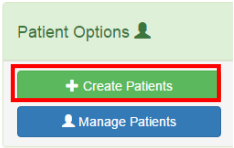
System Test Case (STC-01): Doctors can view a doctor home page on the web application.

Test Description	This test case is a system testing for URS-01. Doctors view a doctor home page on the web application.
Prerequisite	Doctor must login.
Prepared Data	The system prepare the information from the database which includes position, name, surname and username.
Test Steps	1. Doctor selects to “Home” on menu bar.

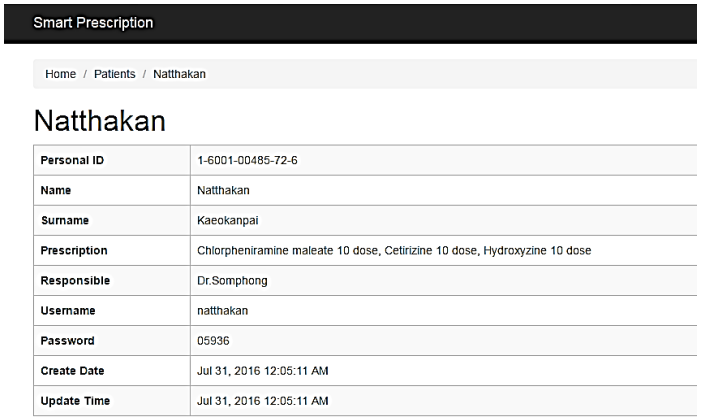
Test cases

ID	Description	Input	Expected Result
1.1	Test that doctor login to the web application successful.	-	The system shall display the doctor home page which includes information of current doctor, patient options menu box and allergy options menu box. Result : 

System Test Case (STC-02): Doctors can create patient's profiles on the web application application by inputting personal id, name, surname, prescription or username.

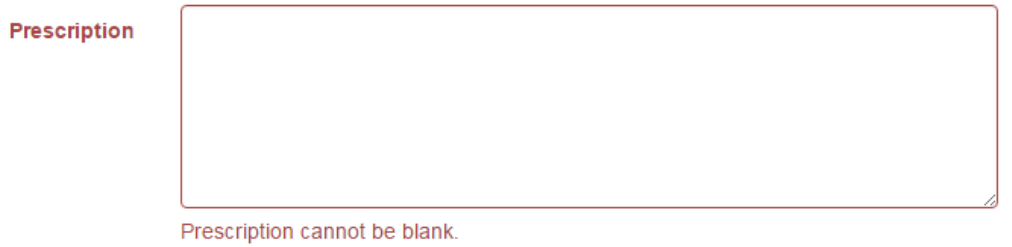
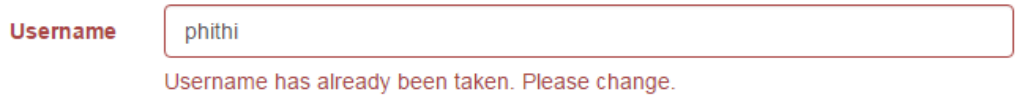
Test Description	This test case is a system testing for URS-02. Doctors can create patient's profiles on the web application application by inputting personal id, name, surname, prescription or username.
Prerequisite	Doctor must login. The system shall display a doctor home page.
Prepared Data	The system prepare the create form which includes personal id, name, surname, prescription, and username.
Test Steps	1. Doctor selects to "Create Patient" on patient options menu box.  2. Input information into the create form for create the patient profile. 3. Doctor clicks on "Create" button.

Test cases

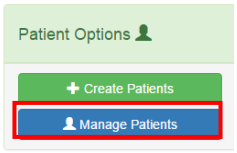
ID	Description	Input	Expected Result
1.1	Test that doctor inputs personal id is number 13 digits. Input name, surname, and responsible are English letters and input prescription.	personal_id = “1610010025485”, name = “Natthakan”, surname = “Kaeokanpai”, prescription = “Chlorpheniramine maleate 10 dose, Cetirizine 10 dose, Hydroxyzine 10 dose”, username = “natthakan”	System saves the patient’s information to the database and also display the patient’s profile. Result : 
1.2	Test that doctor inputs personal id that already exist.	personal_id = “1759900241047”	System displays error message “Personal ID has already been exist.”. Result :

1.2	Test that doctor inputs personal id is invalid.	personal_id = “1111111111111”	System displays error message “Personal ID is Invalid”. Result :
1.3	Test that doctor inputs personal id is number but less than 13 digits.	personal_id = “16100185”	System displays error message “Personal ID is Invalid”. Result :
1.4	Test that doctor do not input personal id.	—	System displays error message “Personal ID cannot be blank”. Result :
1.5	Test that doctor do not input name.	—	System displays error message “Name cannot be blank”. Result :

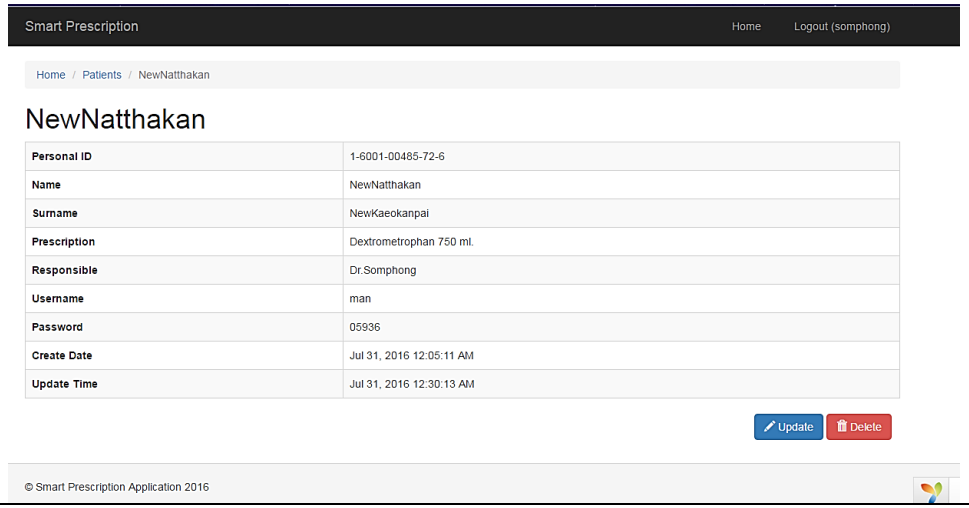
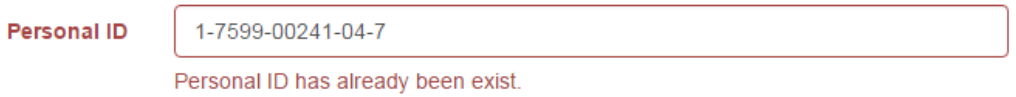
1.6	Test that doctor inputs name is not English letters.	name = “ณัฐกานต์”	System displays error message “Name is invalid Result :
1.7	Test that doctor input name is number.	name = “1234”	System displays error message “Name is invalid Result :
1.8	Test that doctor do not inputs surname.	—	System displays error message “Surname cannot be blank”. Result :
1.9	Test that doctor inputs surname is not English letters.	surname = “แก้วกันภัย”	System displays error message “Surname is invalid”. Result :

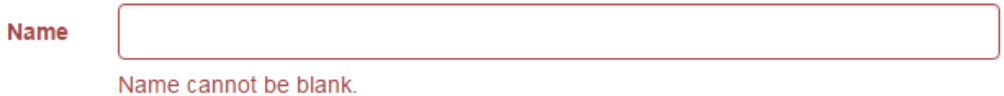
1.10	Test that doctor inputs surname is number.	surname = “1113”	System displays error message “Surname is invalid”. Result :
1.9	Test that doctor do not input prescription.	—	System displays error message “Prescription cannot be blank”. Result : 
1.12	Test that doctor do not input username.	—	System displays error message “Username cannot be blank”. Result :
1.13	Test that doctor inputs username that exist on database.	username = “phithi”	System displays error message “Username has already been taken. Please change”. Result : 

System Test Case (STC-03): Doctors can update patient's profiles on the web application application by inputting personal id, name, surname, prescription and username.

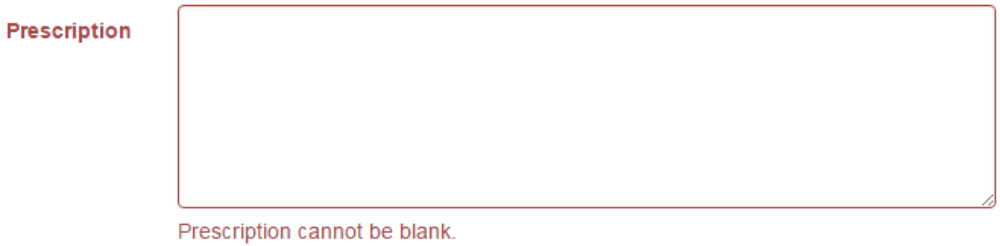
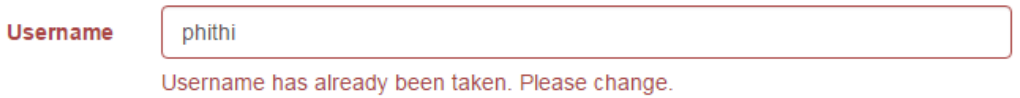
Test Description	This test case is a system testing for URS-03. Doctors can update patient's profiles on the web application application by inputting personal id, name, surname, prescription and username.
Prerequisite	Doctor has create patient's profile successfully before. The system shall display a doctor home page.
Prepared Data	The system prepare the old information and update form which includes personal id, name, surname, prescription, and username.
Test Steps	1. Doctor selects "Manage Patients" button on patient options menu box.  2. Doctor selects "Update" button ( ) in the options column of table. 3. Input new information into update form for update patient's profile. 4. Doctor selects "Update" button.

Test cases

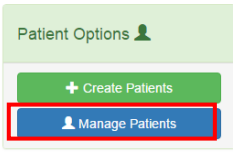
ID	Description	Input	Expected Result
2.1	Test that input the new information to update the defult information. Input personal id are number 13 digits. Input name, surname in English language and input prescription, and username.	Personal_id = “1610010025485”, name = “NewNatthakan”, surname = “NewKaeokanpai”, prescription = “Dextrometrophan 750 ml”, username = “man”	System saves the new patient’s information to the database and also displays the patient’s profile. Result : 
2.3	Test that doctor inputs personal id that already exist.	Personal_id = “1759900241047”	System displays error message “Personal ID has already been exist.”. Result : 

2.2	Test that doctor inputs personal id is invalid.	Personal_id = “1111111111111”	System displays error message “Personal ID is Invalid”. Result :
2.3	Test that doctor inputs personal id is number but less than 13 digits.	Personal_id = “16100185”	System displays error message “Personal ID is Invalid”. Result :
2.4	Test that doctor do not input personal id.	—	System displays error message “Personal ID cannot be blank”. Result :
2.4	Test that doctor do not input name.	—	System displays error message “Name cannot be blank”. Result : 
2.5	Test that doctor inputs name is not English letters.	Name = “ณัฐกานต์”	System displays error message “Name is invalid Result :

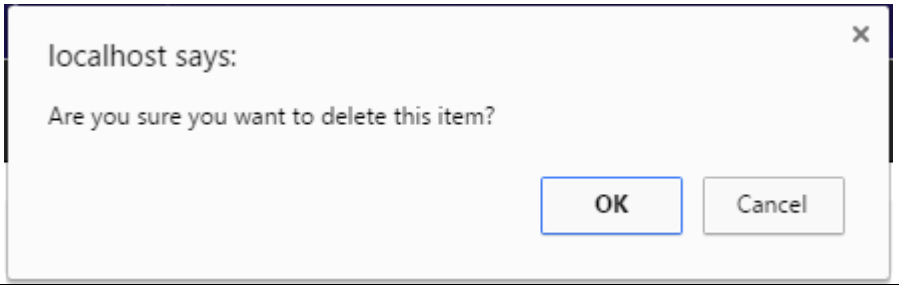
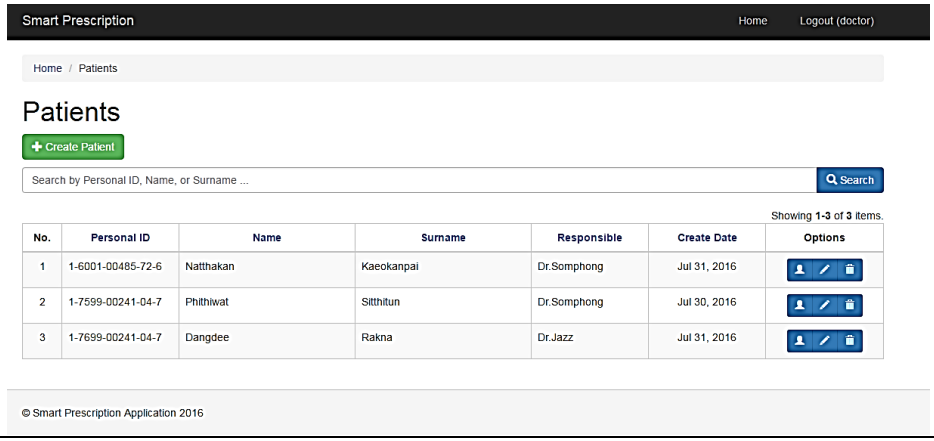
2.6	Test that doctor input name is number.	Name = “1234”	System displays error message “Name is invalid Result :
2.7	Test that doctor do not inputs surname.	—	System displays error message “Surname cannot be blank”. Result :
2.8	Test that doctor inputs surname is not English letters.	Surname = “แก้วกันภัย”	System displays error message “Surname is invalid”. Result :
2.9	Test that doctor inputs surname is number.	Surname = “1113”	System displays error message “Surname is invalid”. Result :

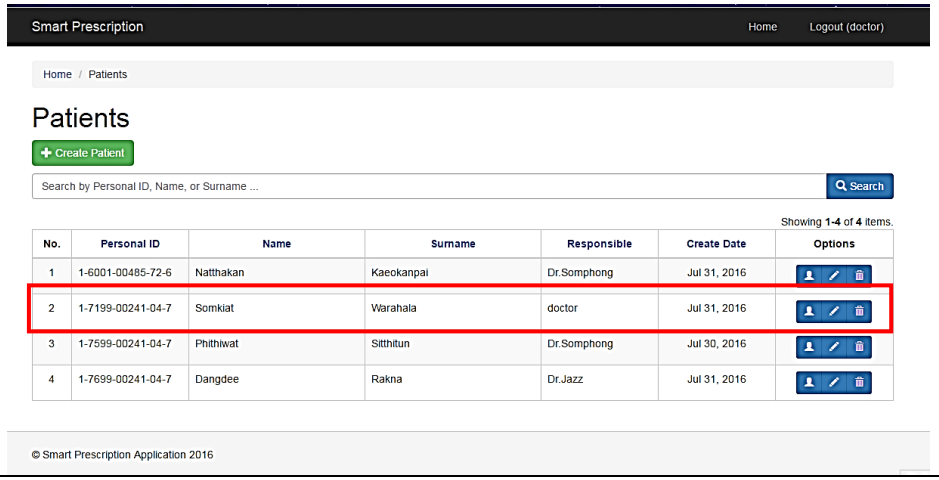
2.10	Test that doctor do not input prescription.	—	System displays error message “Prescription cannot be blank”. Result : 
2.11	Test that doctor do not input username.	—	System displays error message “Username cannot be blank”. Result :
2.12	Test that doctor inputs username that exist on database.	Username = “phithi”	System displays error message “Username has already been taken. Please change”. Result : 

System Test Case (STC-04): Doctors can delete patient's profiles on the web application.

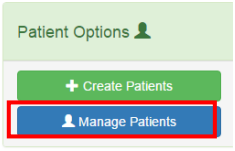
Test Description	This test case is a system testing for URS-04. Doctors can delete patient's profiles on the web application.
Prerequisite	Doctor has create patient's profile successfully before. The system shall display a doctor home page.
Prepared Data	The system prepare the patient's profiles.
Test Steps	1. Doctor selects "Manage Patient" button on patient options menu box.  2. Doctor selects "Delete" ( ) button in the options column of table. 3. Doctor select 'OK' button on the confirm box.

Test cases

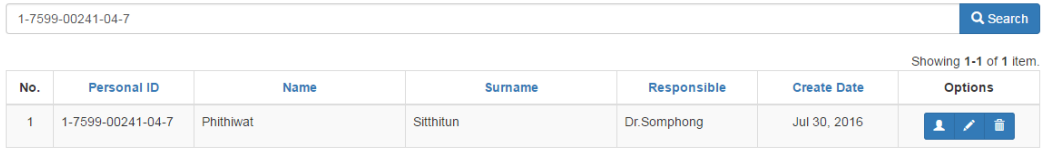
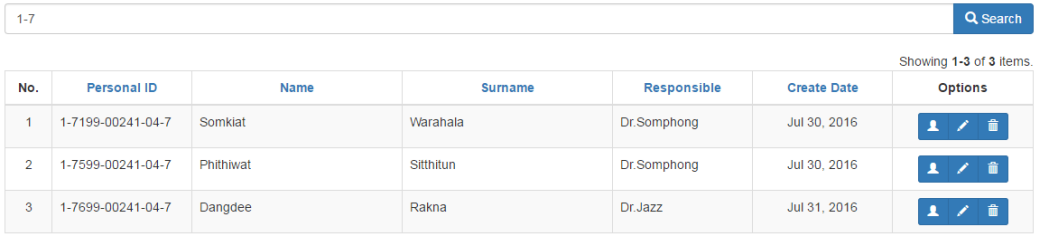
ID	Description	Input	Expected Result
3.1	Test that doctor clicks on ‘Delete’ button for delete patient’s profile from the database.	Click on ‘Delete’ button.	The system interface to show the confirm box “Are you sure you want to delete this item?” and delete from database after click button ‘OK’. Result: 
3.2	Test that doctor click on ‘OK’ button for confirm the deleteion.	Click on ‘Ok’ button.	The system display list of patient’s profile without profile that deleted. Result: 

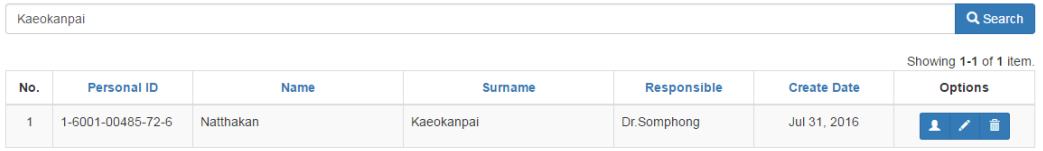
3.3	Test that doctor click on ‘Cancel’ button for cancel the deletetion.	Click on ‘Cancel’ button.	The system display list of patient’s profile that any patient’s profile not to deleted. Result: 

System Test Case (STC-05): Doctors can search patient's profiles on the web application by using personal id, name or surname.

Test Description	This test case is a system testing for URS-05. Doctors can search patient's profiles on the web application by using personal id, name, or surname.
Prerequisite	Doctor has create a patient's profile successfully. The system shall display a doctor home page.
Prepared Data	The system prepare the information from the database to searching about personal id, name and surname.
Test Steps	1. Doctor selects "Manage Patient" button on patient options menu box.  2. Input the data about personal ID, name, or surname for searching. 3. Doctor selects "Seach" button.

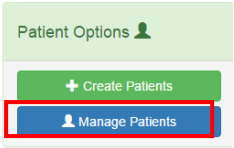
Test cases

ID	Description	Input	Expected Result
4.1	Test that input personal ID is exist in the database.	personal id = “1-7599-00241-04-7”	System displays patient’s informations. Result : 
4.2	Test that input some digits of personal ID are exist in the database.	personal id = “1-7”	System displays patient’s informations. Result : 
4.3	Test that input personal ID is not exist in the database.	personal id = “1-1111-11111-11-1”	System displays message “No results found”. Result :

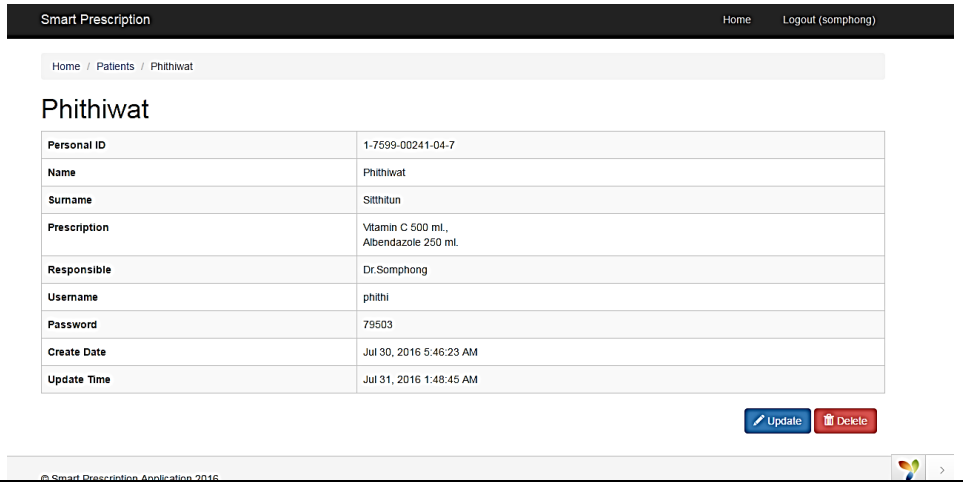
4.4	Test that input name is exist in the database.	name = “Phithiwat”	System displays patient’s information. Result :
4.5	Test that input some letters of name are exist in the database.	name = “wat”	System displays patient’s information. Result :
4.6	Test that input name is not exist in the database.	name = “Buggy”	System displays message “No results found.” Result :
4.7	Test that input surname is exist in the database.	surname = “Kaeokanpai”	System displays patient’s information. Result : 

4.8	Test that input some letters of surname is exist in the database.	surname = “kan”	System displays patient’s information. Result :
4.9	Test that input surname is not exist in the database.	surname = “Madee”	System displays message “No results found” Result :

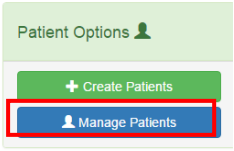
System Test Case (STC-06): Doctors can view patient's profiles on the web application.

Test Description	This test case is a system testing for URS-05. Doctors can view patient's profiles on the web application.
Prerequisite	Doctor has create patient's profile successful. The system shall display a doctor home page.
Prepared Data	The system prepare the information from the database which includes personal id, name, surname, prescription, responsible, username, password, create date time and update date time.
Test Steps	1. Doctor selects "Manage Patient" button on patient options menu box.  2. Doctor selects "View" button ( ) in the options column of table.

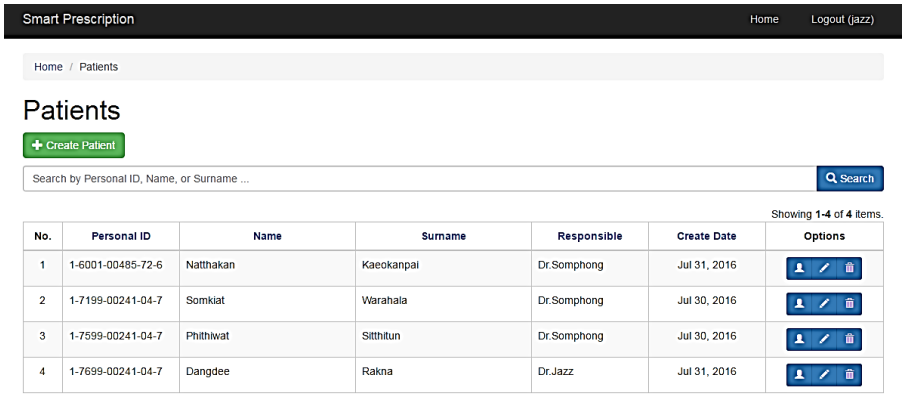
Test cases

ID	Description	Input	Expected Result
5.1	Test that doctor click on ‘View’ button to view the patient’s profile.	Click on ( ) ‘View’ button	System displays the patient’s profile, which includes personal ID, name, surname, prescription, responsible, username, password, create date and update date of patient’s profiles, update button and delete button also. Result : 

System Test Case (STC-07): Doctors can view a list of patient's profiles on the web application.

Test Description	This test case is a system testing for URS-06. Doctors can view a list of patient's profiles on the web application.
Prerequisite	Doctor must login and has create patient's profile successful. The system shall display a doctor home page.
Prepared Data	The system prepare the patient's profiles.
Test Steps	1. Doctor selects "Manage Patient" button on patient options menu box. 

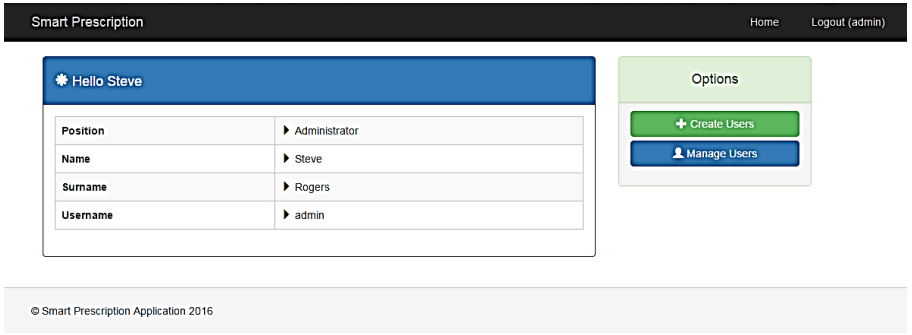
Test cases

ID	Description	Input	Expected Result
6.1	Test that doctor click ‘Manage Patients’ in the patient options menu box to view the list of patient’s profile.	Click ‘Manage Patient’ on patient options menu box.	System displays the lists of patient’s profile on the management page.. Result : 

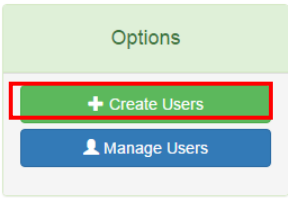
System Test Case (STC-08): Administrators can view an admin home page on the web application.

Test Description	This test case is a system testing for URS-08. Administrators can view a doctor home page on the web application.
Prerequisite	Administator must login.
Prepared Data	The system prepare the information from database which includes position, name, surname and username.
Test Steps	1. Administrator selects to “Home” on menu bar.

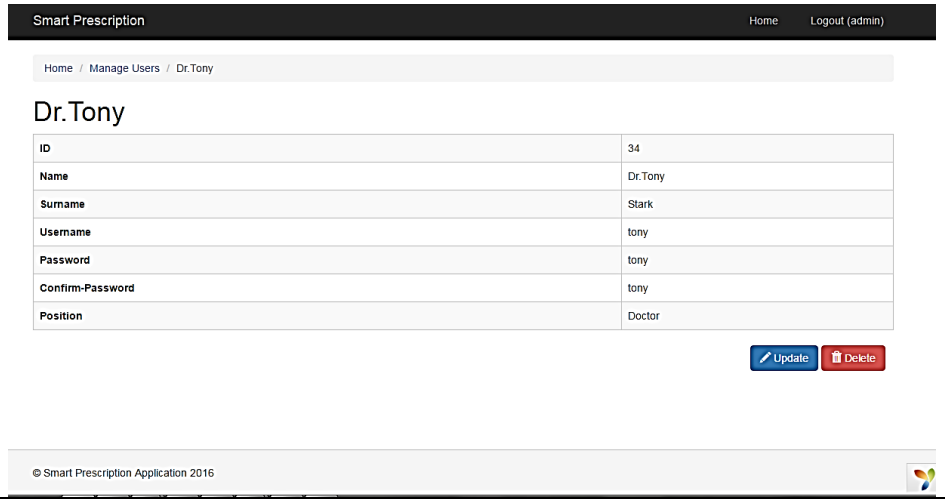
Test cases

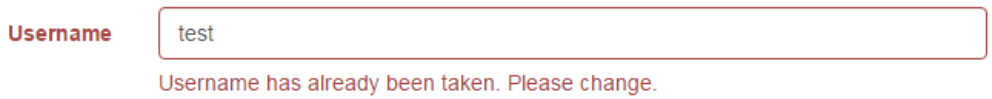
ID	Description	Input	Expected Result
1.1	Test that admin login to the web application successful.	-	The system shall display the doctor home page which includes information of current admin, user options menu box. Result : 

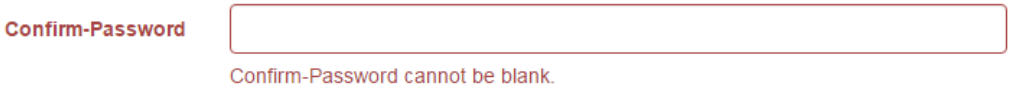
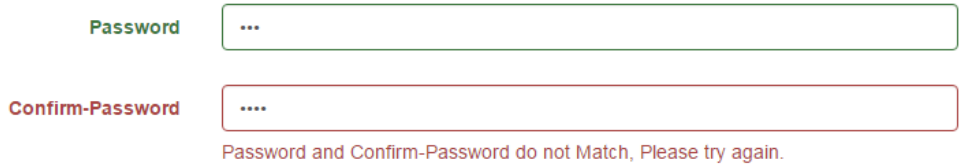
System Test Case (STC-09): Administrators can add user's profiles on the web application by inputting name, surname, username, password, confirm-password and position.

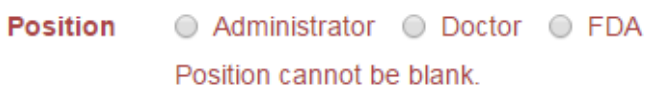
Test Description	This test case is a system testing for URS-09. Administrators can add user's profiles on the web application.
Prerequisite	Administrators must login. The system shall display an admin home page.
Prepared Data	The system prepare the create form which includes name, surname, username, password, confirm-password and position.
Test Steps	1. Administrator selects "Create User" in options menu box.  2. Input information into the create form for create the user. 3. Administrator clicks on "Create" button.

Test cases

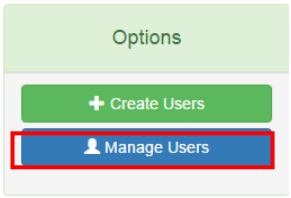
ID	Description	Input	Expected Result
7.1	Test that admin input name and surname in English language and input username, password, and password2. Selects position.	name = “Dr.Tony”, surname = “Stark” username = “tony”, password = “tony”, password2 = “tony” role = 2	System saves the user’s information to the database and also display the patient’s profile. Result : 
7.2	Test that admin do not input name.	—	System displays error message “Name cannot be blank”. Result :

7.3	Test that admin do not input surname.	—	System displays error message “Surname cannot be blank. Result :
7.4	Test that admin do not input username.	—	System displays error message “Username cannot be blank”. Result :
7.5	Test that admin inputs username that exist in database.	username = “Test”	System displays error message “Username has already been taken. Please change.” Result : 
7.6	Test that admin inputs name is not English letters.	name = “โทนี่”	System displays error message “Name is invalid Result :

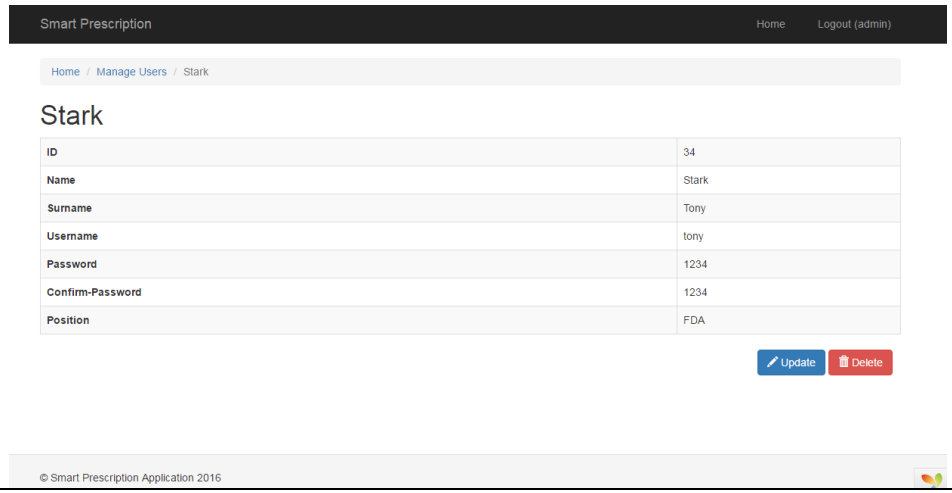
7.7	Test that admin inputs surname is not English letters.	surname = “123”	System displays error message “Surname is invalid”. Result :
7.8	Test that admin do not input password.	—	System displays error message “Password cannot be blank”. Result :
7.9	Test that admin do not input confirm-password.	—	System displays error message “Confirm-Password cannot be blank”. Result : 
7.10	Test that admin inputs password or confirm password are not match.	password = “test” confirm-password = “test1”	System displays error message “Password and Confirm-Password do not Match, Please try again.” Result : 

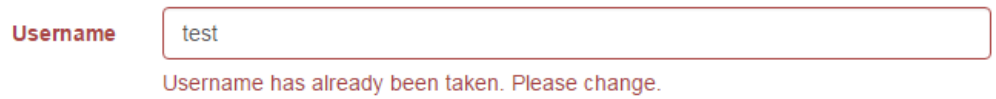
7.11	Test that admin do not select position.	—	System displays error message “Position cannot be blank.”. Result : 

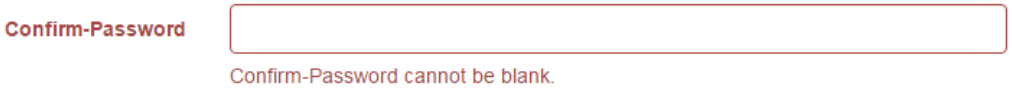
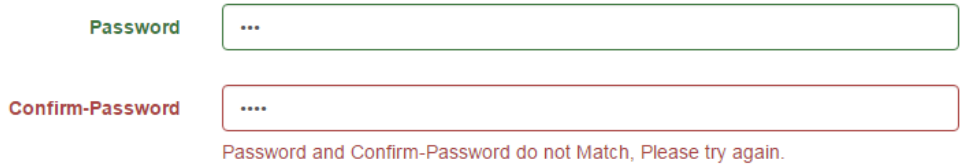
System Test Case (STC-10): Administrators can update user's profiles on the web application by inputting name, surname, username, password, confirm-password and position.

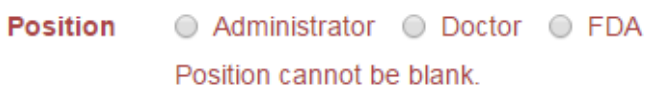
Test Description	This test case is a system testing for URS-10. Administrators can update user's profiles on the web application.
Prerequisite	Administrator has create user successfully before. The system shall display an admin home page.
Prepared Data	The system prepare the old information and update form which includes name, surname, username, password, confirm-password and position.
Test Steps	1. Admin selects "Manage Users" button on options menu box.  2. Admin selects "Update" button ( ) in the options column of table. 3. Input new information into update form for update user information. 4. Admin selects "Update" button.

Test cases

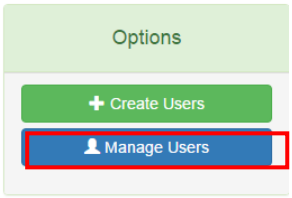
ID	Description	Input	Expected Result
8.1	Test that admin the new information to update the defult information. Select position and input name and surname in English language and input username, password, and password2. Selects group.	name = “Stark”, surname = “Tony” username = “tony”, password = “1234”, password2 = “1234” role = 3	System saves the user’s information to the database and also display the patient’s profile. Result : 
8.2	Test that admin do not input name.	—	System displays error message “Name cannot be blank”. Result :

8.3	Test that admin do not input surname.	—	System displays error message “Surname cannot be blank. Result :
8.4	Test that admin do not input username.	—	System displays error message “Username cannot be blank”. Result :
8.5	Test that admin inputs username that exist in database.	username = “Test”	System displays error message “Username has already been taken. Please change.” Result : 
8.6	Test that admin inputs name is not English letters.	name = “โทนี่”	System displays error message “Name is invalid Result :

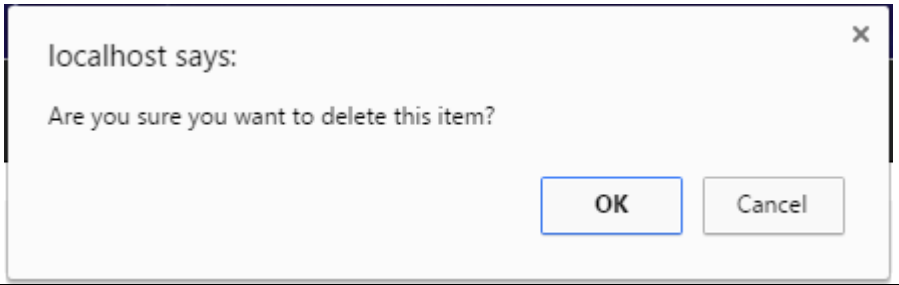
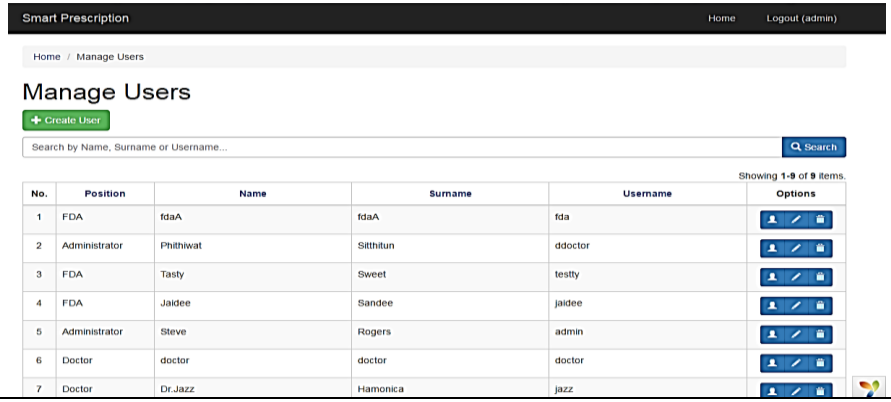
8.7	Test that admin inputs surname is not English letters.	surname = “123”	System displays error message “Surname is invalid”. Result :
8.8	Test that admin do not input password.	—	System displays error message “Password cannot be blank”. Result :
8.9	Test that admin do not input confirm-password.	—	System displays error message “Confirm-Password cannot be blank”. Result : 
8.10	Test that admin inputs password or confirm password are not match.	password = “test” confirm-password = “test1”	System displays error message “Password and Confirm-Password do not Match, Please try again.” Result : 

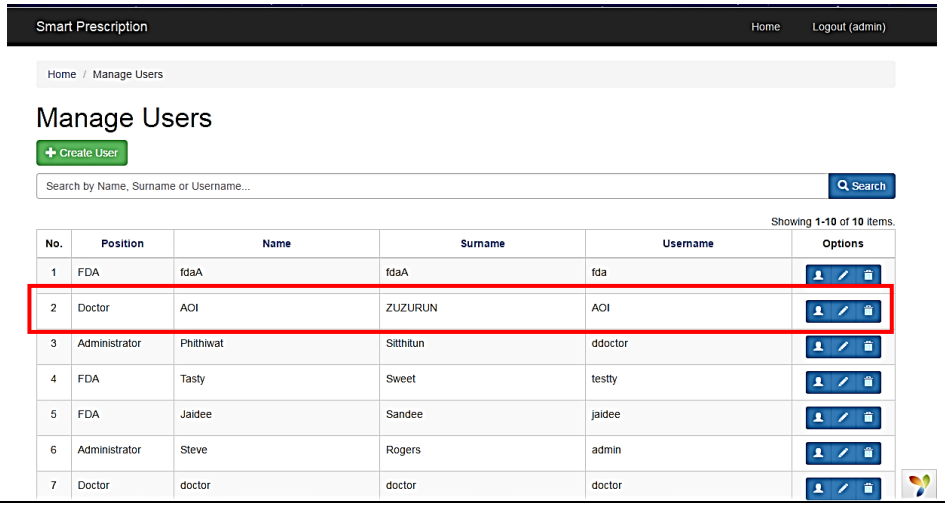
8.11	Test that admin do not select position.	—	System displays error message “Position cannot be blank.”. Result : 

System Test Case (STC-11): Administrators can delete user's profiles on the web application.

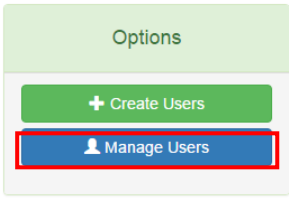
Test Description	This test case is a system testing for URS-11. Administrators can delete user's profiles on the web application.
Prerequisite	Admin has create user account successfully before. The system shall display an admin home page.
Prepared Data	The system prepare the user's profiles.
Test Steps	1. Admin selects "Manage User" button on operation menu of Manage User page.  2. Admin selects "Delete" button ( ) in the options column of table. 3. Admin select 'OK' button on the confirm box.

Test cases

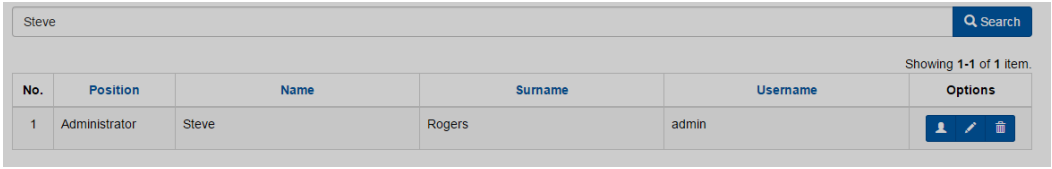
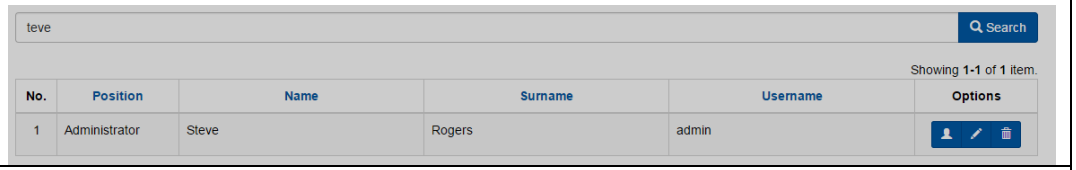
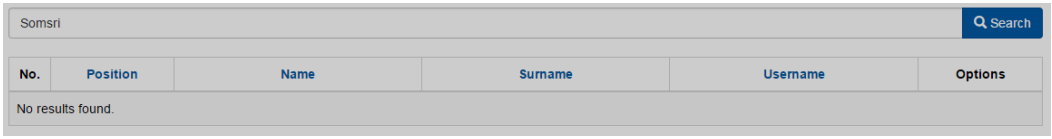
ID	Description	Input	Expected Result
11.1	Test that admin clicks on ‘Delete’ button for delete user from the database.	Click on ‘Delete’ button.	The system interface to show the confirm box “Are you sure you want to delete this item?” and delete from database after click button ‘OK’. Result: 
11.2	Test that admin click on ‘OK’ button for confirm the deleteion.	Click on ‘Ok’ button.	The system display list of user’s profile without profile that deleted. Result: 

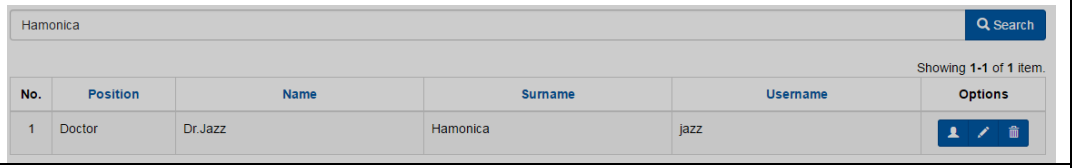
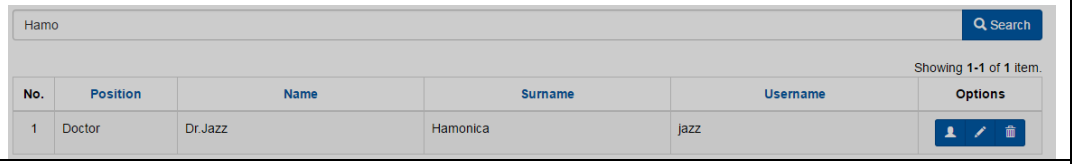
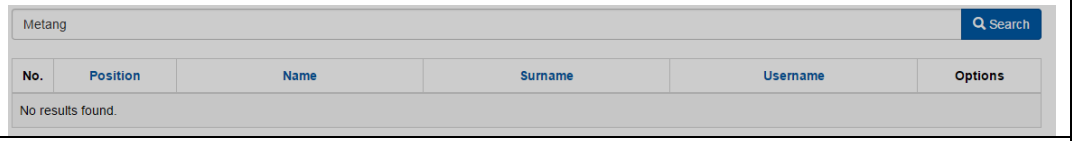
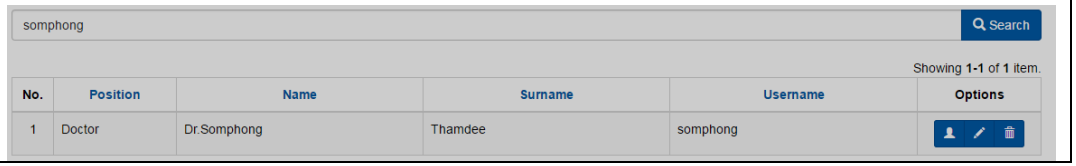
11.3	Test that admin click on ‘Cancel’ button for cancel the deletetion.	Click on ‘Cancel’ button.	The system display list of user’s profile that any user’s profile not to deleted. Result: 

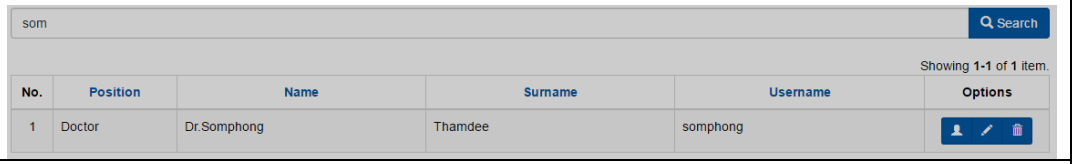
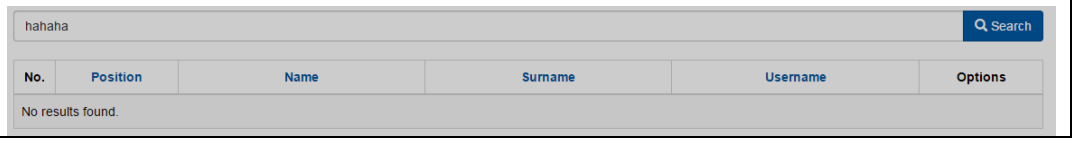
System Test Case (STC-12): Administrations can search user's profiles on the web application by using name, surname or username.

Test Description	This test case is a system testing for URS-12. Administrations can search user's profiles on the web application by using name, surname or username.
Prerequisite	Admin has create a user successfully. The system shall display an admin home page.
Prepared Data	The system prepare the information from the database to searching about name, surname and username.
Test Steps	1. Admin selects "Manage User" button on operation menu of Manage User page.  2. Input the data about name, surname or username for searching. 3. Admin selects "Seach" button.

Test cases

ID	Description	Input	Expected Result
4.1	Test that input name is exist in the database.	name = “Steve”	System displays user’s information. Result : 
4.2	Test that input some letters of name are exist in the database.	name = “teve”	System displays user’s information. Result : 
4.3	Test that input name is not exist in the database.	name = “Somsri”	System displays message “No results found.” Result : 

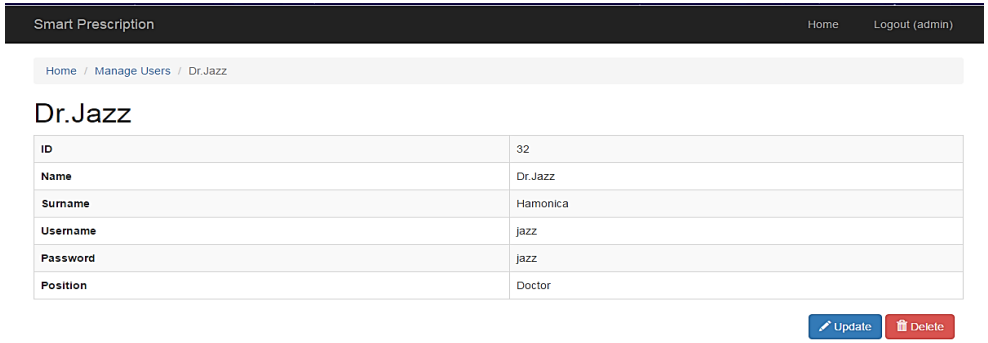
4.4	Test that input surname is exist in the database.	surname = “Hamonica”	System displays user’s information. Result : 
4.5	Test that input some letters of surname is exist in the database.	surname = “Hamo”	System displays patient’s information. Result : 
4.6	Test that input surname is not exist in the database.	surname = “Metang”	System displays message “No results found” Result : 
4.7	Test that input username is exist in the database.	username = “somphong”	System displays patient’s information. Result : 

4.8	Test that input some letters of username is exist in the database.	username = “som”	System displays patient’s information. Result : 
4.9	Test that input username is not exist in the database.	username = “hahaha”	System displays message “No results found” Result : 

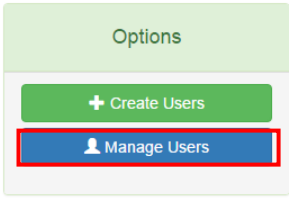
System Test Case (STC-13): Administrations can view user's profiles on the web application.

Test Description	This test case is a system testing for URS-13. Administrations can view user's profiles on the web application.
Prerequisite	Admin has create user successful. The system shall display an admin home page.
Prepared Data	The system prepare the information from the database which includes name, surname, username, password and position.
Test Steps	1. Admin selects "Manage User" button on options menu box. 2. Admin selects "View" button ( ) in the options column of table.

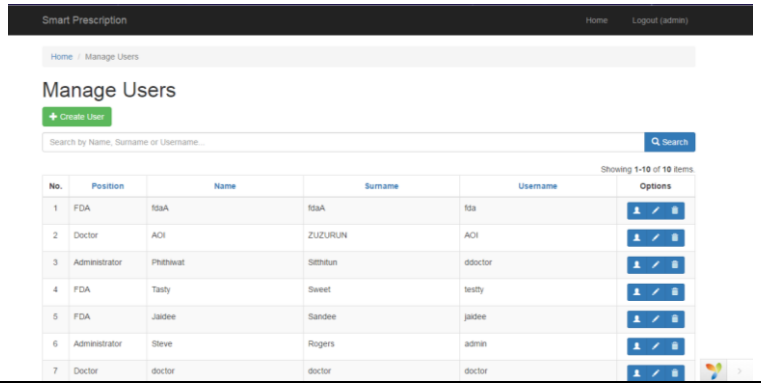
Test cases

ID	Description	Input	Expected Result
5.1	Test that admin click on 'View' button to view the user.	Click on 'View' button	System displays the user's information, which includes ID, position, name, surname, username and password of user, update button and delete button also. Result : 

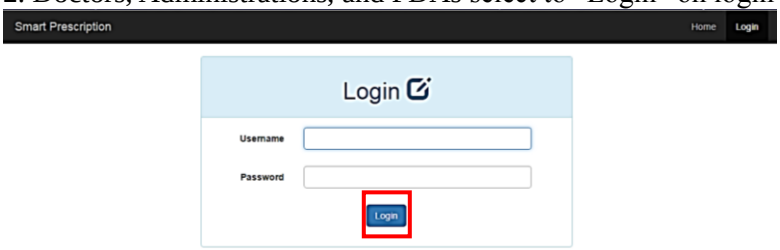
System Test Case (STC-14): Administrations can view a list of user's profiles on the web application.

Test Description	This test case is a system testing for URS-14. Administrations can view a list of user's profiles on the web application.
Prerequisite	Admin must login and has create user successful. The system shall display an admin home page.
Prepared Data	The system prepare the user's profiles.
Test Steps	1. Admin selects to "Manage Users" in options menu box. 

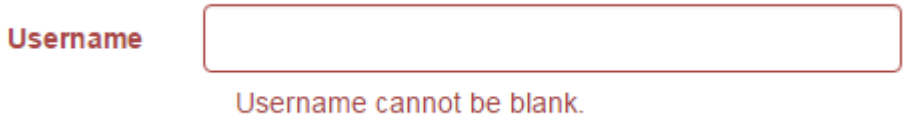
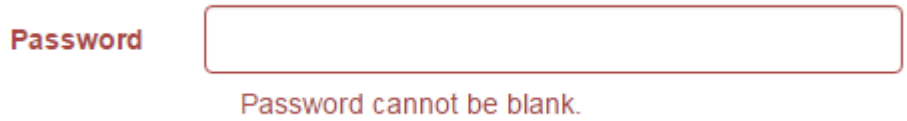
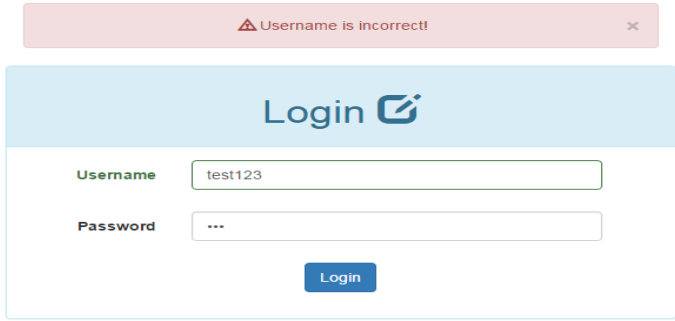
Test cases

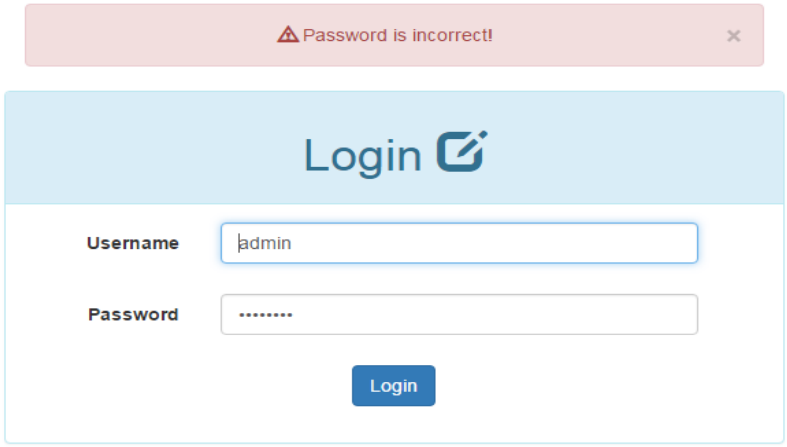
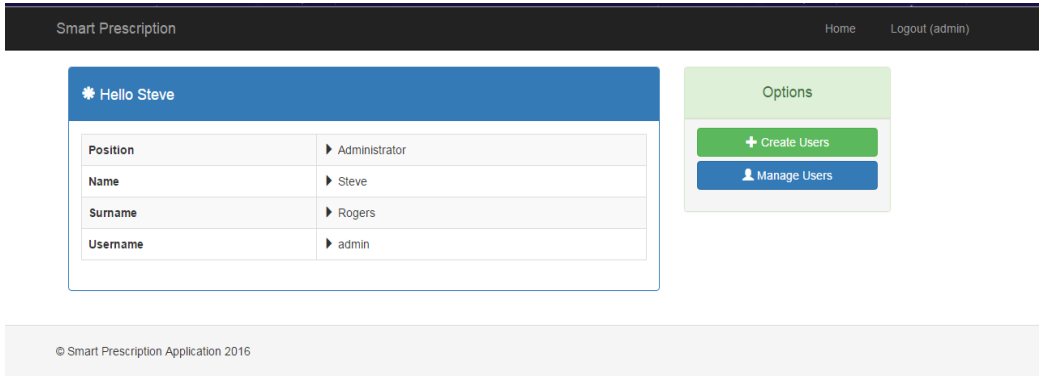
ID	Description	Input	Expected Result
14.1	Test that admin click ‘Manage Users’ in the options menu box to view the list of user.	Click ‘Manage Users’ on menu bar.	System displays the lists of user on the manage page. Result : 

System Test Case (STC-15): Doctors, Administrations, and FDAs can login to the web application.

Test Description	This test case is a system testing for URS-15. Doctors, Administrations, and FDAs can login to the web application.
Prerequisite	Doctors, Administrations, and FDAs must have the username and password.
Test Steps	1. Doctors, Administrations, and FDAs select to “Login” on menu bar. 2. Doctors, Administrations, and FDAs select to “Login” on login form. 

Test cases

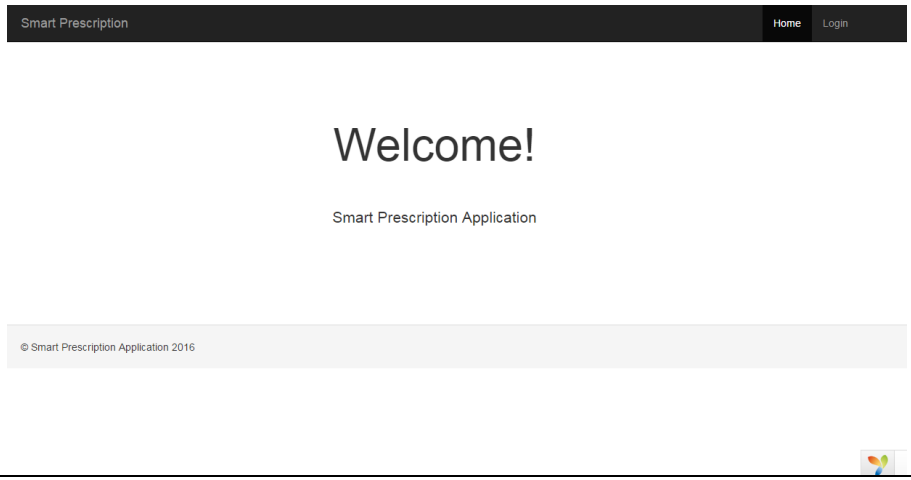
ID	Description	Input	Expected Result
15.1	Test that do not input username.	—	System displays error message “Username cannot be blank.” Result : 
15.2	Test that do not input password.	—	System displays error message “Password cannot be blank.” Result : 
15.3	Test that input username does not exist in the database.	username = “test123” password = “admin”	System displays error box message “Username is incorrect!” Result : 

15.4	Test that input password does not match the username.	username = “admin” password = “admin123”	System displays error box message “Password is incorrect!” Result : 
15.5	Test that input username and password are correct.	username = “admin” password = “admin” Result :	

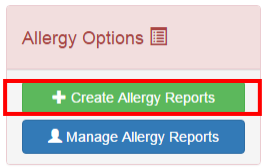
System Test Case (STC-16): Doctors, Administrations, and FDAs can logout from the web application.

Test Description	This test case is a system testing for URS-16. Doctors, Administrations, and FDAs can logout from the web application.
Prerequisite	Doctors, Administrations, and FDAs must login.
Test Steps	1. Doctors, Administrations, and FDAs selects to “Logout” on menu bar.

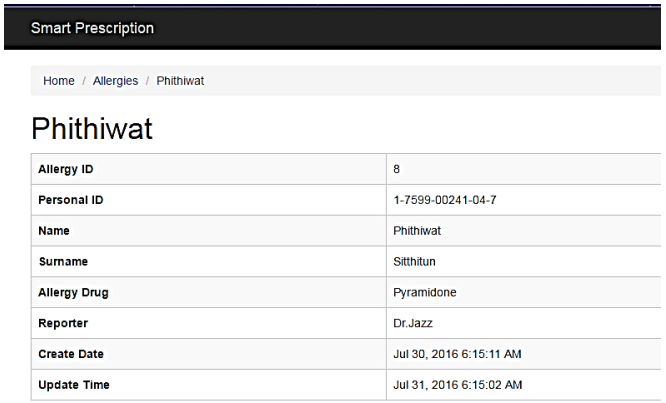
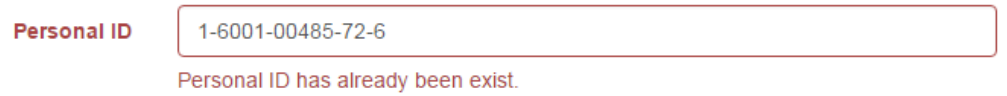
Test cases

ID	Description	Input	Expected Result
6.1	Test that click “Logout” on menu bar.	Click “Logout” on menu bar.	System redirect user to index web page. Result : 

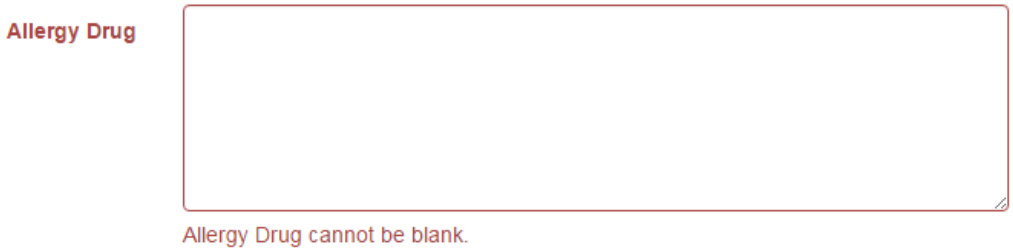
System Test Case (STC-17): Doctors can create allergy report on the web application by inputting personal id, name, surname, and allergy drug.

Test Description	This test case is a system testing for URS-22. Doctors can create allergy report on the web application.
Prerequisite	Doctor must login. The system shall display a doctor home page.
Prepared Data	The system prepare the create form which includes personal id, name, surname, and allergy drug.
Test Steps	1. Doctor selects to “Create Allergy Report” on allergy options menu box.  2. Input information into the create form for create the allergy report. 3. Doctor clicks on “Create” button.

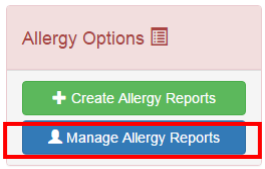
Test cases

ID	Description	Input	Expected Result
17.1	Test that doctor inputs personal id is number 13 digits. Input name, and surname are English letters and input allergy drug.	personal_id = “1759900241047”, name = “Phithiwat”, surname = “Sitthitun”, allergy_drug = “Pyramidone”	System saves the allergy report to the database and also display the allergy report. Result : 
17.2	Test that doctor inputs personal id that already exist.	personal_id = “1600100485726”	System displays error message “Personal ID has already been exist. Please change.”. Result : 
17.3	Test that doctor inputs personal id is incorrect.	personal_id = “1234567890987”	System displays error message “Personal ID is Invalid”. Result :

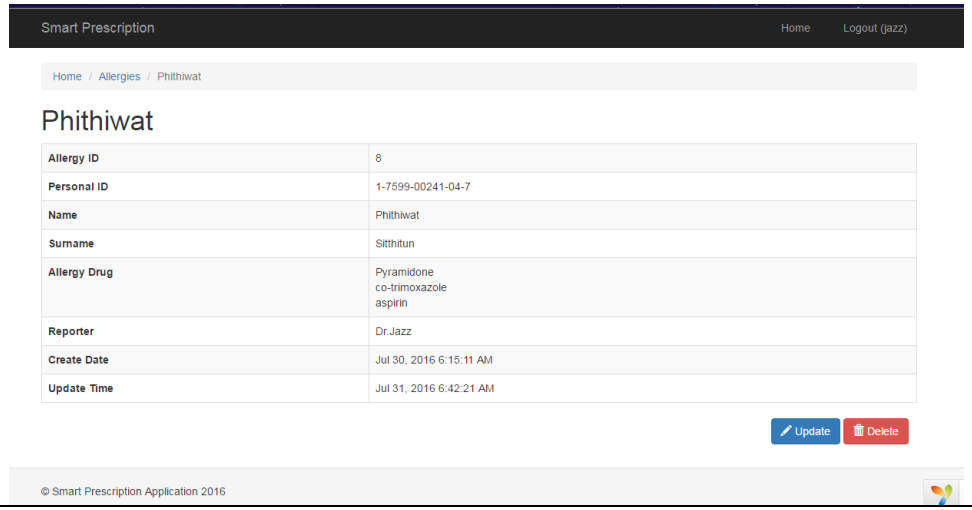
17.4	Test that doctor inputs personal id is number but less than 13 digits.	personal_id = “16100185”	System displays error message “Personal ID is Invalid”. Result :
17.5	Test that doctor do not input personal id.	—	System displays error message “Personal ID cannot be blank”. Result :
17.6	Test that doctor do not input name.	—	System displays error message “Name cannot be blank”. Result :
17.7	Test that doctor inputs name is not English letters.	name = “พิธิวัต”	System displays error message “Name is invalid Result :
17.8	Test that doctor do not inputs surname.	—	System displays error message “Surname cannot be blank”. Result :

17.9	Test that doctor inputs surname is not English letters.	surname = “Sitthitun123”	System displays error message “Surname is invalid”. Result :
17.10	Test that doctor do not input allergy drug.	—	System displays error message “Allergy drug cannot be blank”. Result : 

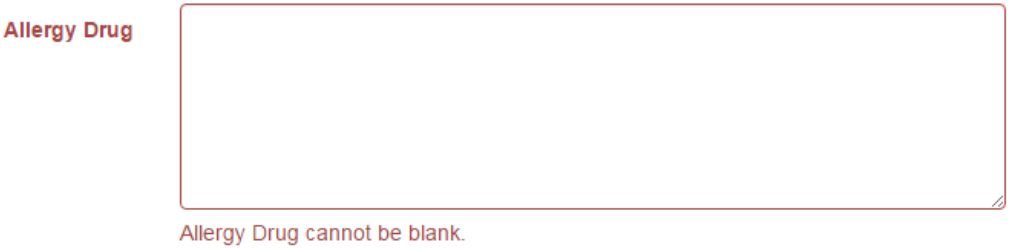
System Test Case (STC-18): Doctors can update an allergy report on the web application by inputting personal id, name, surname, and allergy drug.

Test Description	This test case is a system testing for URS-23. Doctors can update an allergy report on the web application.
Prerequisite	Doctor has create allergy report successfully before. The system shall display a doctor home page.
Prepared Data	The system prepare the old information and update form which includes personal id, name, surname, and allergy drug.
Test Steps	1. Doctor selects "Manage Allergy Reports" button on allergy option menu box.  2. Doctor selects "Update" button ( ) in the options column of table. 3. Input new information into update form for update allergy report. 4. Doctor selects "Update" button.

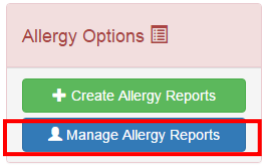
Test cases

ID	Description	Input	Expected Result
18.1	Test that input the new information to update the defult information. Input personal id are number 13 digits. Input name, surname in English language and input allergy drug.	personal_id = “1759900241047”, name = “Phithiwat”, surname = “Sitthitun”, allergy_drug = “Pyramidone”, “co-trimoxazole”, “aspirin”	System saves the new allergy report to the database and also displays the allergy report. Result : 
18.2	Test that doctor inputs personal id that already exist.	personal_id = “1600100485726”	System displays error message “Personal ID has already been exist. Please change.”. Result :

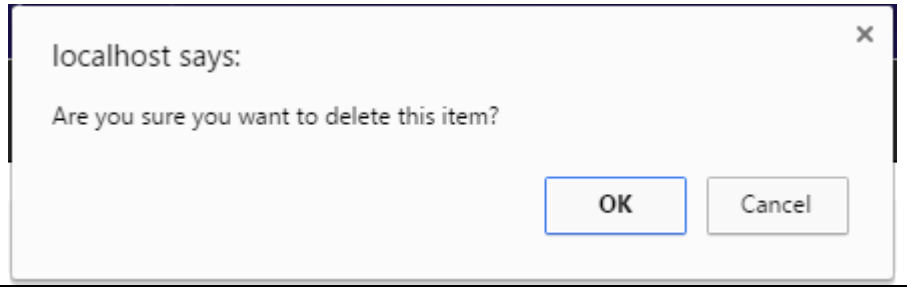
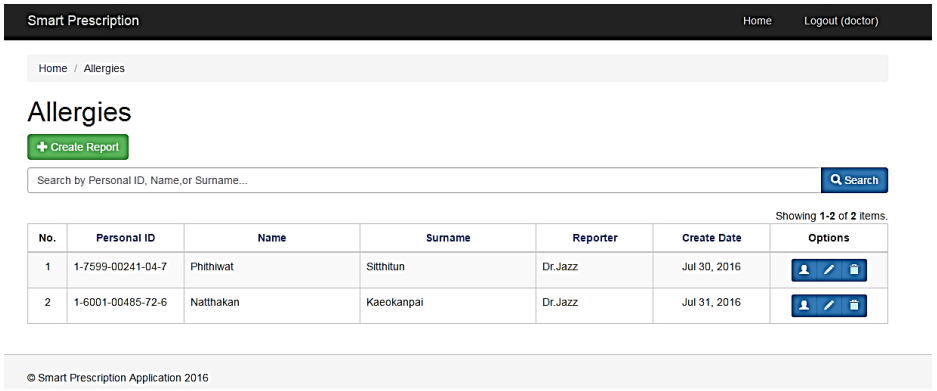
18.3	Test that doctor inputs personal id is incorrect.	personal_id = “1234567890987”	System displays error message “Personal ID is Invalid”. Result :
18.4	Test that doctor inputs personal id is number but less than 13 digits.	personal_id = “16100185”	System displays error message “Personal ID is Invalid”. Result :
18.5	Test that doctor do not input personal id.	—	System displays error message “Personal ID cannot be blank”. Result :
18.6	Test that doctor do not input name.	—	System displays error message “Name cannot be blank”. Result :
18.7	Test that doctor inputs name is not English letters.	name = “พิธิวัต”	System displays error message “Name is invalid Result :

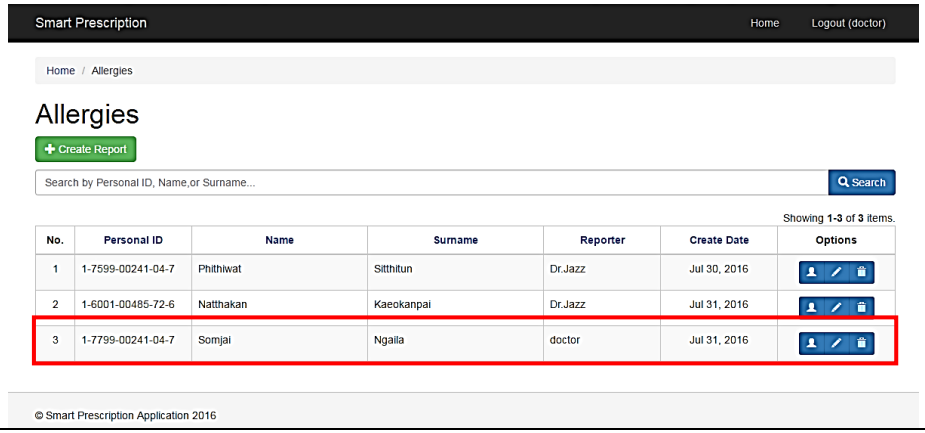
18.8	Test that doctor do not inputs surname.	—	System displays error message “Surname cannot be blank”. Result :
18.9	Test that doctor inputs surname is not English letters.	surname = “Sitthitun123”	System displays error message “Surname is invalid”. Result :
18.10	Test that doctor do not input allergy drug.	—	System displays error message “Allergy drug cannot be blank”. Result : 

System Test Case (STC-19): Doctors can delete an allergy report on the web application.

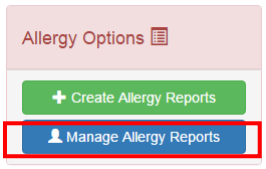
Test Description	This test case is a system testing for URS-24. Doctors can delete an allergy report on the web application.
Prerequisite	Doctor has create allergy report successfully before. The system shall display a doctor home page.
Prepared Data	The system prepare the allergy reports.
Test Steps	1. Doctor selects "Manage Allergy Reports" button on allergy option menu box.  2. Doctor selects "Delete" button ( ) in the options column of table. 3. Doctor select 'OK' button on the confirm box.

Test cases

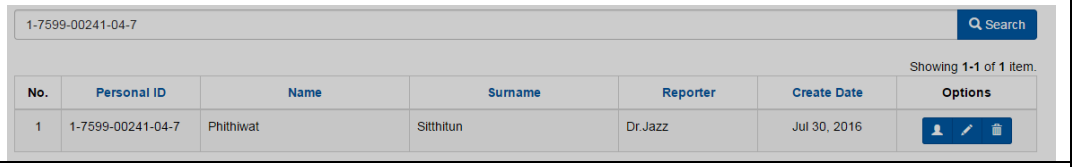
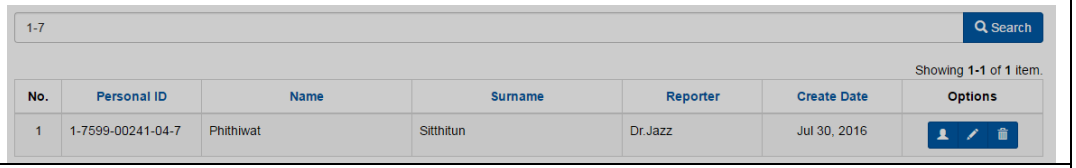
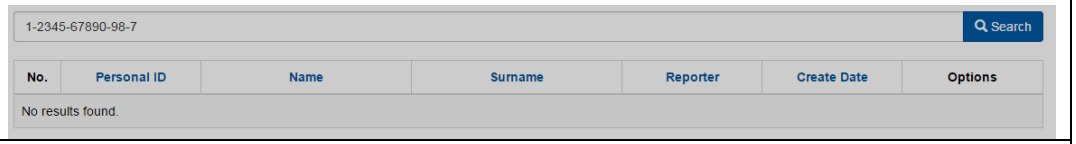
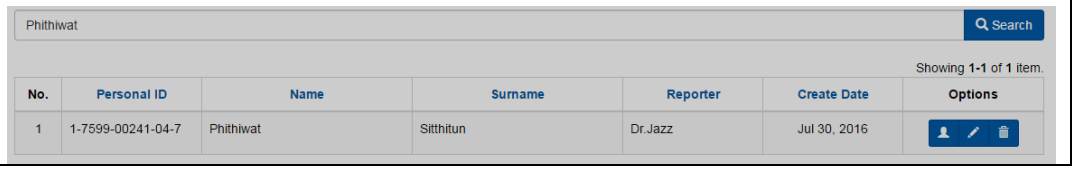
ID	Description	Input	Expected Result
19.1	Test that doctor clicks on ‘Delete’ button for delete allergy report from the database.	Click on ‘Delete’ button.	The system interface to show the confirm box “Are you sure you want to delete this item?” and delete from database after click button ‘OK’. Result: 
3.2	Test that doctor click on ‘OK’ button for confirm the deleteion.	Click on ‘Ok’ button.	The system display list of allergy reports without report that deleted. Result: 

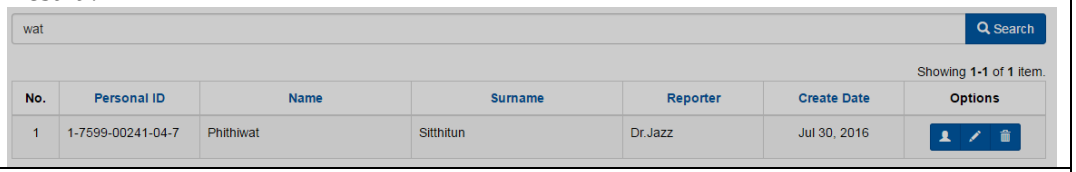
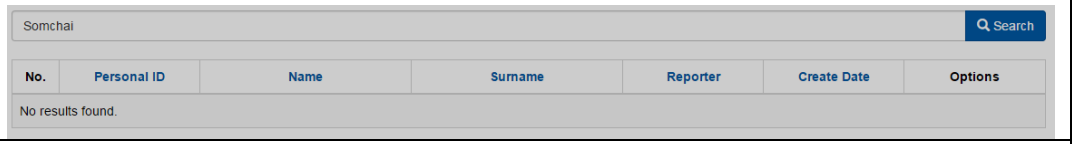
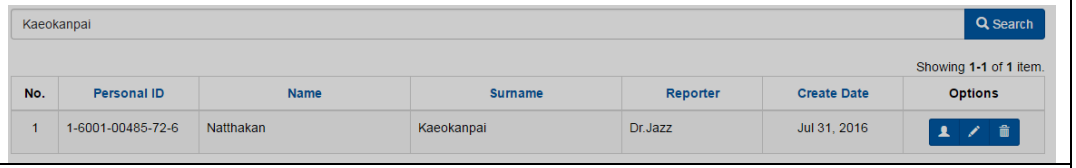
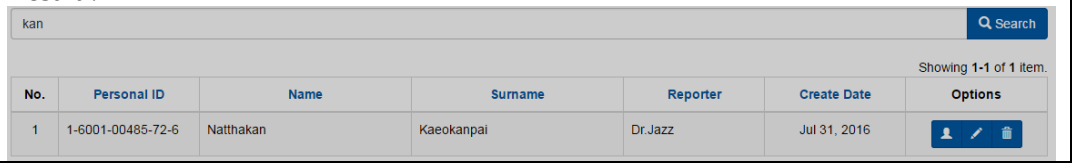
3.3	Test that doctor click on ‘Cancel’ button for cancel the deletetion.	Click on ‘Cancel’ button.	The system display list of allergy reports that any allergy report not to deleted. Result: 

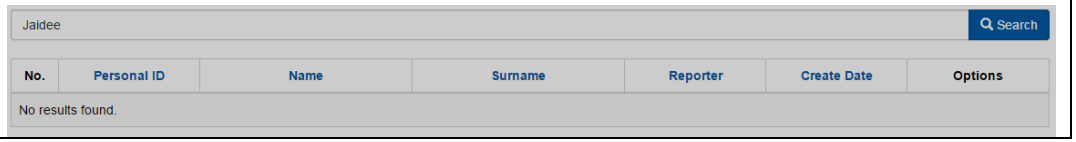
System Test Case (STC-20): Doctors can search allergy reports on the web application by using personal id, name or surname.

Test Description	This test case is a system testing for URS-25. Doctors can search allergy report on the web application by using personal id, name or surname.
Prerequisite	Doctor has create an allergy report successfully. The system shall display a doctor home page.
Prepared Data	The system prepare the information from the database to searching about personal id, name and surname.
Test Steps	1. Doctor selects “Manage Allergy Reports” button on allergy options menu box.  2. Input the data about personal ID, name, or surname for searching. 3. Doctor selects “Seach” button.

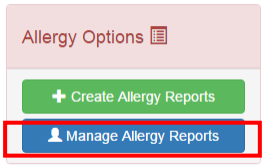
Test cases

ID	Description	Input	Expected Result
20.1	Test that input personal ID is exist in the database.	personal id = “1-7599-00241-04-7”	System displays report’s informations. Result : 
20.2	Test that input some digits of personal ID are exist in the database.	personal id = “1-7”	System displays report’s informations. Result : 
20.3	Test that input personal ID is not exist in the database.	personal id = “1-2345-67890-98-7”	System displays message “No results found”. Result : 
20.4	Test that input name is exist in the database.	name = “Phithiwat”	System displays report’s information. Result : 

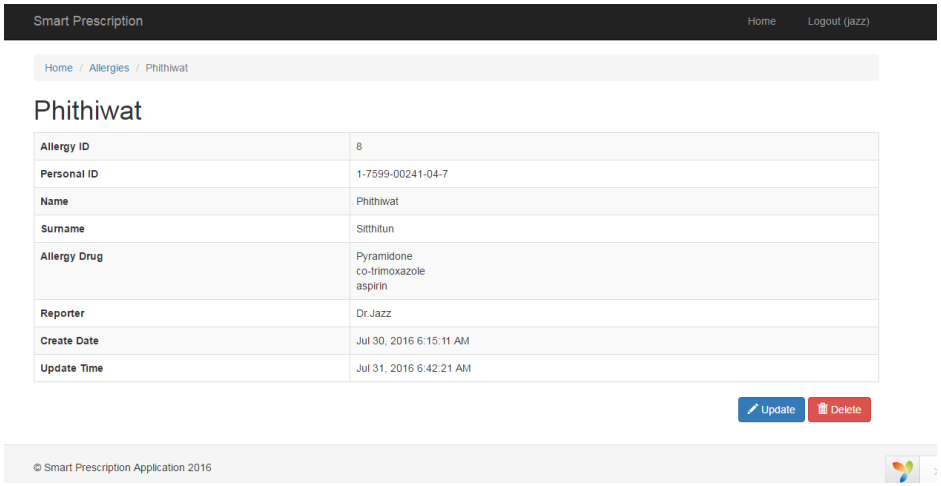
20.5	Test that input some letters of name are exist in the database.	name = “wat”	System displays report’s information. Result : 
20.6	Test that input name is not exist in the database.	name = “Somchai”	System displays message “No results found.” Result : 
20.7	Test that input surname is exist in the database.	surname = “Kaeokanpai”	System displays report’s information. Result : 
20.8	Test that input some letters of surname is exist in the database.	surname = “kan”	System displays report’s information. Result : 

20.9	Test that input surname is not exist in the database.	surname = “Jaidee”	System displays message “No results found” Result : 

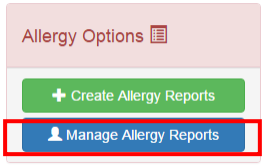
System Test Case (STC-21): Doctors can view an allergy report on the web application.

Test Description	This test case is a system testing for URS-26. Doctors can view an allergy report on the web application.
Prerequisite	Doctor has create allergy report successful. The system shall display a doctor home page.
Prepared Data	The system prepare the information from the database which includes allergy id, personal id, name, surname, allergy drug, reporter, create date time and update date time.
Test Steps	1. Doctor selects “Manage Allergy Reports” button on allergy options menu box.  2. Doctor selects “View” button ( ) in the options column of table.

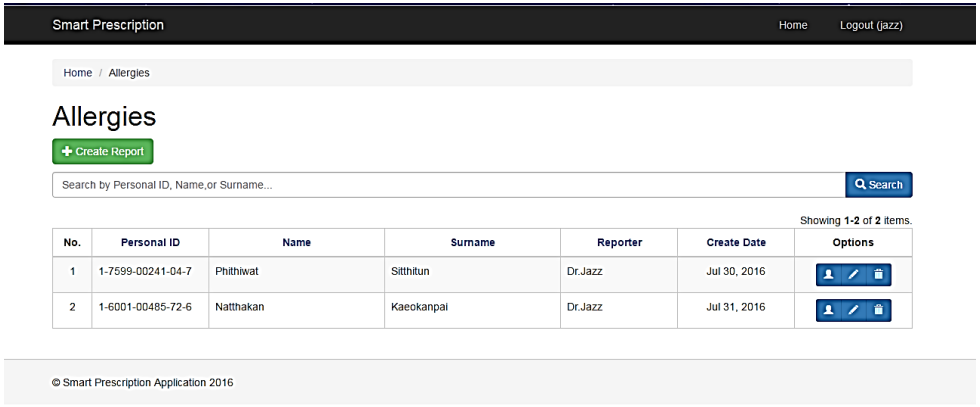
Test cases

ID	Description	Input	Expected Result
21.1	Test that doctor click on ‘View’ button to view the allergy report.	Click on ‘View’ button	System displays the allergy report, which includes allergy Id, personal Id, name, surname, allergy drug, reporter, create date and update date of allergy report, update button and delete button also. Result : 

System Test Case (STC-22): Doctors can view list of allergy reports on the web application.

Test Description	This test case is a system testing for URS-27. Doctors can view list of allergy reports on the web application.
Prerequisite	Doctor must login and has create allergy report successful. The system shall display a doctor home page.
Prepared Data	The system prepare the allergy reports.
Test Steps	1. Doctor selects "Manage Allergy Reports" button on allergy options menu box. 

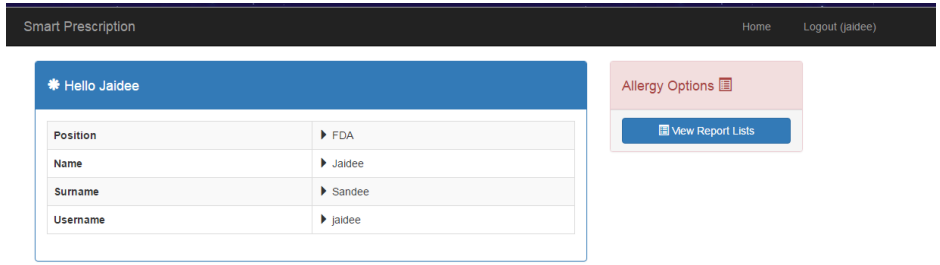
Test cases

ID	Description	Input	Expected Result
22.1	Test that doctor click ‘Manage Allergy Reports’ in the allergy options menu box to view the list of allergy report.	Click ‘Manage Allergy Reports’ on allergy options menu box.	System displays the lists of allergy report on the management page.. Result : 

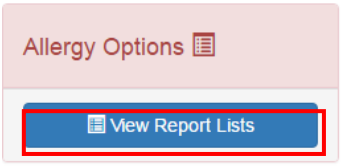
System Test Case (STC-23): FDAs can view a FDA home page on the web application.

Test Description	This test case is a system testing for URS-28. FDAs view a FDA home page on the web application.
Prerequisite	FDA must login.
Prepared Data	The system prepare the information from database which includes position, name, surname and username.
Test Steps	1. FDA selects to “Home” on menu bar.

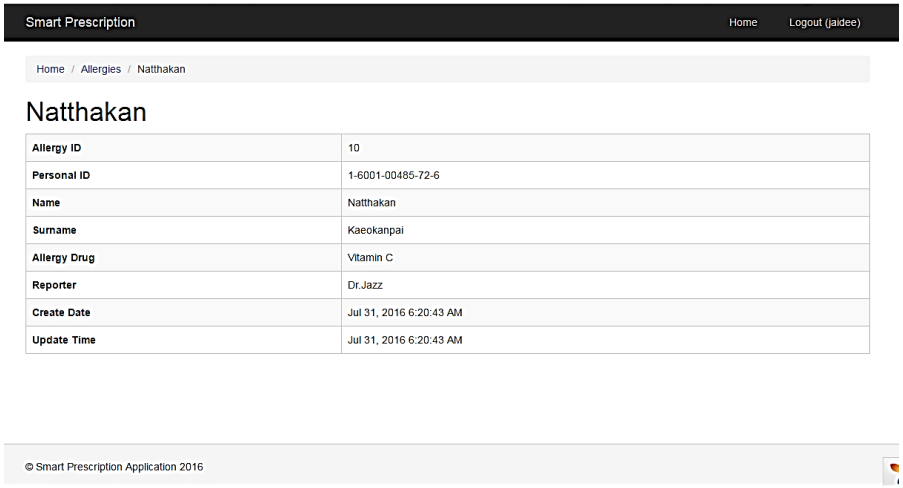
Test cases

ID	Description	Input	Expected Result
23.1	Test that FDA login to the web application successful.	-	The system shall display the FDA home page which includes information of current FDA, allergy options menu box. Result : 

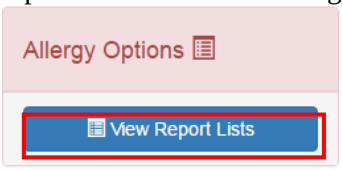
System Test Case (STC-24): FDAs can view an allergy report on the web application.

Test Description	This test case is a system testing for URS-29. FDAs can view an allergy report on the web application.
Prerequisite	Doctor has create allergy report successful. The system shall display a FDA home page.
Prepared Data	The system prepare the information from the database which includes allergy id, personal id, name, surname, allergy drug, reporter, create date time and update date time.
Test Steps	1. FDA selects “View List Reports” button on allergy options menu box.  2. FDA selects “View” button ( ) in the options column of table.

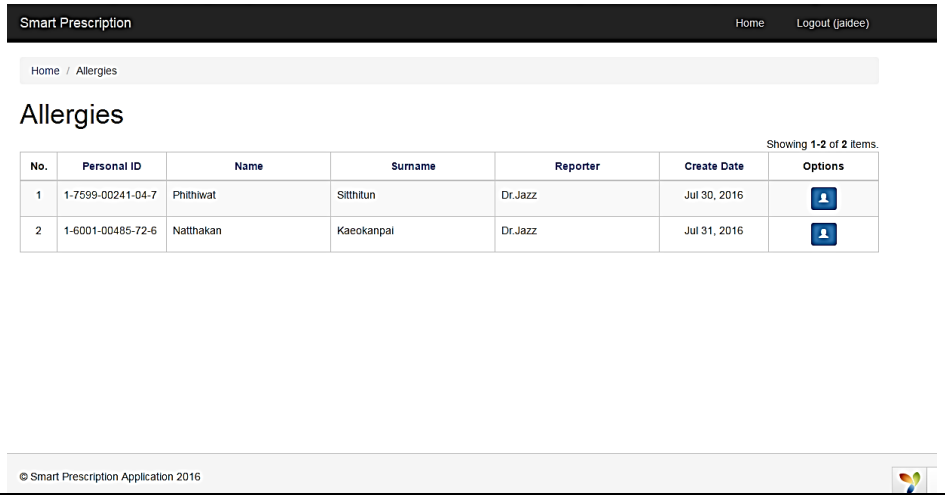
Test cases

ID	Description	Input	Expected Result
24.1	Test that FDA click on ‘View’ button to view the allergy report.	Click on ‘View’ button	System displays the allergy report, which includes allergy Id, personal Id, name, surname, allergy drug, reporter, create date and update date of allergy report. Result : 

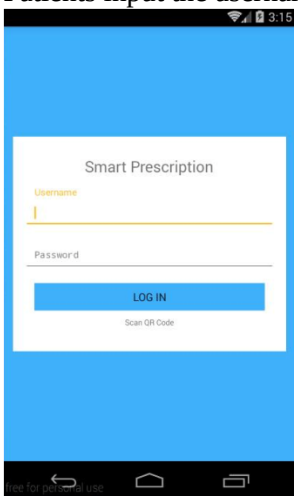
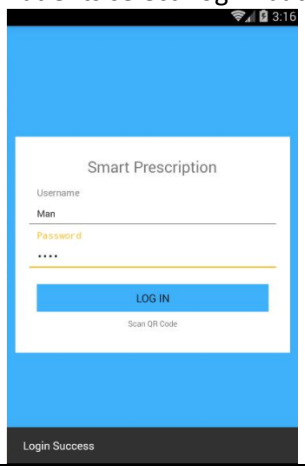
System Test Case (STC-25): FDAs can view list of allergy reports on the web application.

Test Description	This test case is a system testing for URS-30. FDAs can view list of allergy reports on the web application.
Prerequisite	Doctor must login and has create allergy report successful. The system shall display a FDA home page.
Prepared Data	The system prepare the allergy reports.
Test Steps	1. FDA selects “View Report Lists” button on allergy options menu box. 

Test cases

ID	Description	Input	Expected Result
25.1	Test that doctor click ‘View Report Lists’ in the allergy options menu box to view the list of allergy report.	Click ‘View Report Lists’ on allergy options menu box.	System displays the lists of allergy report. Result : 

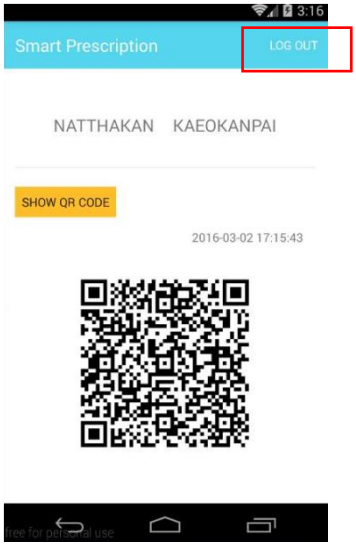
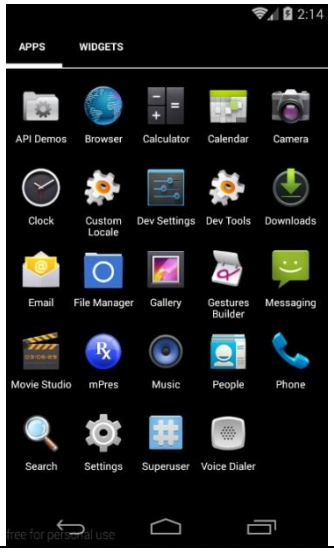
System Test Case (STC-26): Patients can login to the mobile application by using username and password.

Test Description	This test case is a system testing for URS-17. Patients can login to the mobile application by using username and surname. The system shall provide the UI for login, to receive username and password.
Prerequisite	Patients access to the mobile application.
Test Steps	1. Patients input the username and password to the login form.  2. Patients select 'login' button on the login form. 

Test cases

ID	Description	Input	Expected Result
17.1	Test that user can input the username into the username input box.	Username = Test Password = “ ”	System display error message “Invalid Username or Password”
17.2	Test that user can input the password into the username input box.	Username = “ ” Password = 1234	System display error message “Invalid Username or Password”
17.3	Test that user can login to the application with the wrong password	Username = Test Password = 1222 Click “Login” button	System displays the error message. “Invalid Username or Password”.
17.4	Test that user can login to the application with the wrong username	Username = Test22 Password = 1234 Click “Login” button	System displays the error message. “Invalid Username or Password”.
17.5	Test that user can login to the application with the correct username and password with connect the internet	Username = Test Password = 1234 Click “Login” button	System displays message “Login success” and redirect user to the main page.

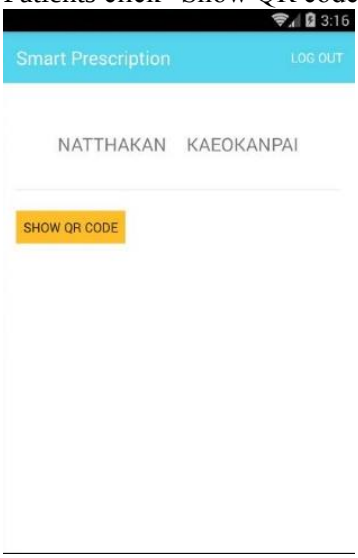
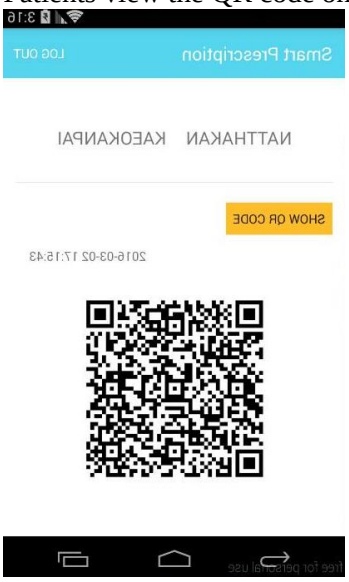
System Test Case (STC-27): Patients can logout from the mobile application.

Test Description	This test case is a system testing for URS-18. Patients can logout from the mobile application. The system shall provide "logout" button on the main page for logout from the application.
Prerequisite	Patients already login to the application.
Test Steps	1. Patients select "logout" button.  2. Patient exists from the application 

Test cases

ID	Description	Input	Expected Result
18.1	Test that user can click “logout” button.	Click “logout” button.	Patient can click “Logout” button.
18.2	Test that user exits from the application	Click “logout” button.	System exits from the application

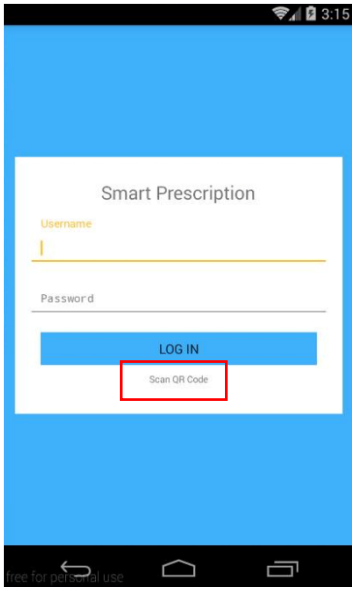
System Test Case (STC-28): Patients can view the QR code on the mobile application.

Test Description	This test case is a system testing for URS-19. Patients can view the QR code on the mobile application. The system shall display QR code to patient.
Prerequisite	Patients already login to the application.
Test Steps	1. Patients click “Show QR code” button.  2. Patients view the QR code on the screen. 

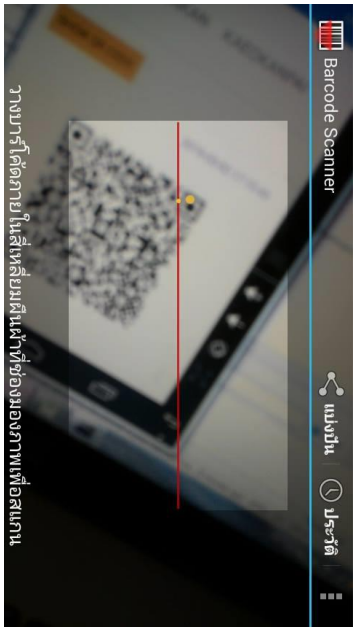
Test cases

ID	Description	Input	Expected Result
18.1	Test for user can click “Show QR code” button.	Click “Show QR code” button.	Patient can click “Show QR code” button.
18.2	Test for user can view QR code after select “Show QR code” button.	Click “Show QR code” button.	Patient can view the QR code on the screen.

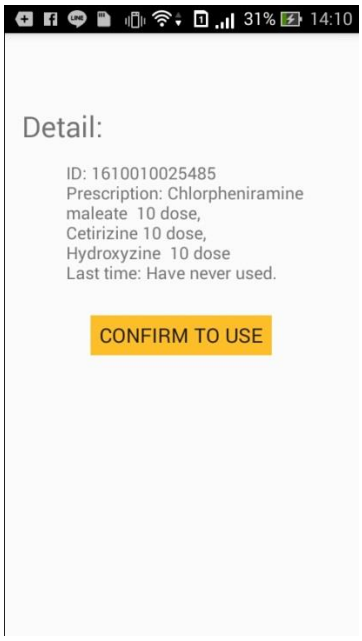
System Test Case (STC-29): **Pharmacists can scan the QR code on the mobile application.**

Test Description	This test case is a system testing for URS-20. Pharmacists can scan the QR code on the mobile application. The system shall display error message if the picture that user scanning are not the QR code.
Prerequisite	Pharmacist already access to the application.
Test Steps	1. Pharmacist select “Scan QR code” button. 

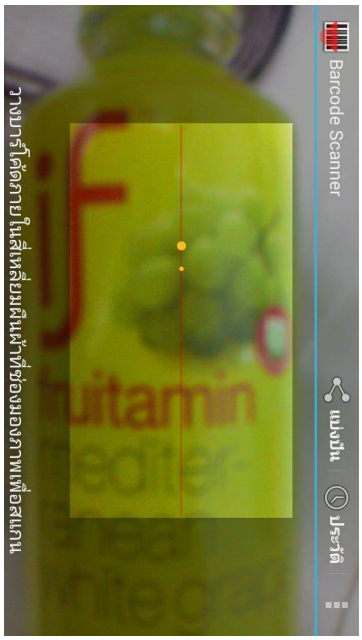
2. Pharmacist scans QR code.



3. Pharmacist views the information of patient's profile after scan QR code.



4. The system cannot scan the other picture if the picture is not the QR code.



5. The system can scan the other barcode but it is not information of patient's profile.

Test cases

ID	Description	Input	Expected Result
18.1	Test for user can click “Scan” button.	Click “Scan” button.	Pharmacist can click “Show QR code” button.
18.2	Test for user can use the scan QR code function after select “Show QR code” button.	Scan QR code	Pharmacist can view the information of patient’s profile after scan QR code.
18.3	Test for pharmacist cannot use the scan QR code function for scan other picture after select “Scan” button.	Click “Scan” button.	Pharmacist cannot view the information of patient’s profile after scan QR code.
18.4	Test for pharmacist can use the scan QR code function for scan other barcode after select “Scan” button.	Click “Scan” button.	Pharmacist can view the other information of barcode after scan QR code.

System Test Case (STC-30): **Pharmacists can add the time of dispensation to patient’s profile on the mobile application.**

Test Description	This test case is a system testing for URS-21. Pharmacists can add the time of dispensation to patient’s profile on the mobile application. The system shall record the current time to the patient’s profile.
Prerequisite	Pharmacist already scans QR code.
Test Steps	1. Pharmacist select “Confirm” button. 2. Pharmacist views the message “successful” after select “confirm” button.

Test cases

ID	Description	Input	Expected Result
18.1	Test for user can click “Confirm” button.	Click “Confirm” button.	Pharmacist can click “Confirm” button.
18.2	Test for user can view the message ”Successful” after select “confirm” button.	Click “Confirm” button.	Pharmacist can view the message”Successful” after select “confirm” button.

Chapter Five | Appendix

Appendix A

Database for unit testing part of web application.

User

id	2
name	Natthakan
surname	Kaeokanpai
username	natthakan
password	man
password2	man
roleId	2

id	3
name	Tony
surname	Stark
username	tony
password	tony
password2	tony
roleId	3

id	15
name	Phithiwat
surname	Sitthitun
username	phithi
password	1234
password2	1234
roleId	1

id	29
name	Steve
surname	Rogers
username	steve
password	1234
password2	1234
roleId	1

id	31
name	Dr.Hussen
surname	Hamonica
username	hussen
password	hussen
password2	hussen
roleId	2

id	32
name	Dr.Jazz
surname	Hamonica
username	jazz
password	jazz
password2	jazz
roleId	2

Patient

personal_id	1719900241047
name	Somkiat
surname	Warahala
prescription	Paracetamol 500 ml.
create_by	31
username	somkiat
password	54139
create_at	2016-07-31 18:36:28
update_at	2016-07-31 18:36:28

personal_id	1769900241047
name	Dangdee
surname	Rakna
prescription	Vitamins 90 ml.
create_by	32
username	dang
password	67045
create_at	2016-07-30 23:55:17
update_at	2016-07-30 23:59:23

personal_id	1660100485726
name	Vikom
surname	Chomjai
prescription	Dextrometrophan
create_by	32
username	vikom
password	58706
create_at	2016-08-01 03:39:11
update_at	2016-08-01 03:39:11

Allergy

allergy_id	8
personal_id	1759900241047
name	Phithiwat
surname	Sitthitun
allergy_drug	Pyramidone, Co-Trimoxazole, Aspirin
reporter	32
create_at	2016-07-30 04:15:11
update_at	2016-07-31 04:42:21

allergy_id	10
personal_id	1600100485726
name	Natthakan
surname	Kaeokanpai
allergy_drug	Vitamin C.
reporter	32
create_at	2016-07-31 04:20:43
update_at	2016-07-31 04:20:43

allergy_id	11
personal_id	1779900241047
name	Somjai
surname	Ngaila
allergy_drug	Multivitamins.
reporter	31
create_at	2016-07-31 18:47:11
update_at	2016-07-31 18:47:11

Chapter Six | Reference

[1] Definition of Unit testing
http://artofunittesting.com/definition-of-a-unit-test

[2] Definition of System testing
http://www.softwaretestingclass.com/system-testing-what-why-how/