# MINISTRY OF EDUCATION AND TRAINING

­

­

**FPT UNIVERSITY**

Capstone Project Document

**Call-Center on Mobile for Clinics**

|  |  |
| --- | --- |
| **Group 1** | |
| **Group members** | Nguyễn Thế Phương – SE62087  Phan Thành Thuận - SE62063 Nguyễn Lương Tuấn Kiệt - SE61696  Nguyễn Cao Duy - SE61032 |
| **Supervisor** | Kiều Trọng Khánh |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** |  |

-Ho Chi Minh City, ***18/05/2018*-**

# Table of Contents

Table of Contents

[MINISTRY OF EDUCATION AND TRAINING 1](#_Toc520105415)

[Table of Contents 2](#_Toc520105416)

[List of Tables 6](#_Toc520105417)

[List of Figures 6](#_Toc520105418)

[Definitions, Acronyms, and Abbreviations 6](#_Toc520105419)

[A. Introduction 7](#_Toc520105420)

[1. Project Information 7](#_Toc520105421)

[2. Introduction 7](#_Toc520105422)

[3. Current Situation 7](#_Toc520105423)

[4. Problem Definition 7](#_Toc520105424)

[5. Proposed Solution 8](#_Toc520105425)

[5.1 Feature functions 8](#_Toc520105426)

[5.2 Values and Challenges 8](#_Toc520105427)

[6. Functional Requirements 8](#_Toc520105428)

[7. Role and Responsibility 8](#_Toc520105429)

[B. Software Project Management Plan 9](#_Toc520105430)

[1. Problem Definition 9](#_Toc520105431)

[1.1. Name of this Capstone Project 9](#_Toc520105432)

[1.2. Problem Abstract 9](#_Toc520105433)

[1.3. Project Overview 9](#_Toc520105434)

[1.3.1. Current Situation 9](#_Toc520105435)

[1.3.2. The Proposed System 10](#_Toc520105436)

[1.3.2.1. API application 10](#_Toc520105437)

[1.3.2.2. Mobile application 10](#_Toc520105438)

[1.3.2.3. Hotline server 10](#_Toc520105439)

[1.3.2.4. Web application 10](#_Toc520105440)

[1.3.3. Boundaries of the System 10](#_Toc520105441)

[1.3.4. Future Plans 10](#_Toc520105442)

[1.3.5. Development Environment 11](#_Toc520105443)

[2. Project Organization 11](#_Toc520105444)

[2.1. Software Process Model 11](#_Toc520105445)

[2.2. Roles and responsibilities 12](#_Toc520105446)

[2.3. Tools and Techniques 13](#_Toc520105447)

[3. Project Management Plan 13](#_Toc520105448)

[3.1. Product Backlog 13](#_Toc520105449)

[3.2. Sprint Backlog 13](#_Toc520105450)

[3.2.1. Sprint 1 (18.05.2018 – 25.05.2018): Project initiation 13](#_Toc520105451)

[3.2.1.1. Goal 13](#_Toc520105452)

[3.2.1.2. Development 14](#_Toc520105453)

[3.2.2. Sprint 2 14](#_Toc520105454)

[3.2.2.1. Goal 14](#_Toc520105455)

[3.2.2.2. Development 14](#_Toc520105456)

[3.3. All Meeting Minutes 14](#_Toc520105457)

[4. Coding Convention 14](#_Toc520105458)

[C. Software Requirement Specification 16](#_Toc520105459)

[1. User Requirement Specification 16](#_Toc520105460)

[1.1. Clinic Requirement 16](#_Toc520105461)

[1.2. Administrator Requirement 16](#_Toc520105462)

[1.3. System Requirement 16](#_Toc520105463)

[2. System Requirement Specification 16](#_Toc520105464)

[2.1. External Interface Requirement 16](#_Toc520105465)

[2.1.1. User Interface 16](#_Toc520105466)

[2.1.2. Hardware Interface 16](#_Toc520105467)

[2.1.3. Software Interface 16](#_Toc520105468)

[2.1.4. Communication Protocol 16](#_Toc520105469)

[2.2. System Overview Use case 17](#_Toc520105470)

[2.3. List of Use Case 17](#_Toc520105471)

[2.3.1. Third-party Hotline Overview User Case 17](#_Toc520105472)

[2.3.1.1. <Third-party Hotline> Listen to Incoming Call 18](#_Toc520105473)

[2.3.1.2. <Third-party Hotline> Send SMS 20](#_Toc520105474)

[2.3.1.3. <Third-party Hotline> Listen to Incoming SMS 22](#_Toc520105475)

[2.3.2. Hotline Overview Use Case 23](#_Toc520105476)

[2.3.2.1. <Hotline Server> Analyze SMS 23](#_Toc520105477)

[2.3.2.2. <Hotline Server> Analyze Call 25](#_Toc520105478)

[2.3.2.3. <Hotline Server> Schedule Appointment 26](#_Toc520105479)

[2.3.2.4. <Hotline Server> Request Sending SMS 28](#_Toc520105480)

[2.3.2.5. <Hotline Server> Request Notification to Clinic 29](#_Toc520105481)

[2.3.3. Firebase Overview Use Case 30](#_Toc520105482)

[2.3.3.1. < Firebase> Notify to Clinic 30](#_Toc520105483)

[2.3.4. Clinic Overview Use Case 32](#_Toc520105484)

[2.3.4.1. <Clinic> Get Appointment List 32](#_Toc520105485)

[2.3.4.2. <Clinic> Search Appointment 34](#_Toc520105486)

[2.3.4.3. <Clinic> Configure working hours 36](#_Toc520105487)

[2.3.4.4. <Clinic> Buy License 38](#_Toc520105488)

[2.3.4.5. <Clinic> Cancel Appointment 39](#_Toc520105489)

[2.3.4.6. <Clinic> Adjust Appointment Time 40](#_Toc520105490)

[2.3.4.7. <Clinic> Block Phone Number 42](#_Toc520105491)

[2.3.4.8. <Clinic> Create Medical Record 43](#_Toc520105492)

[2.3.4.9. <Clinic> View Detail Patient 45](#_Toc520105493)

[2.3.4.10. <Clinic> View Average Patient Statistic 47](#_Toc520105494)

[2.3.4.11. <Clinic> View Patient Statistic Through Years 48](#_Toc520105495)

[2.3.4.12. <Clinic> Pay 50](#_Toc520105496)

[2.3.5. Administrator Overview Use Case 51](#_Toc520105497)

[2.3.5.1. < Administrator> Create Account 52](#_Toc520105498)

[2.3.5.2. < Administrator> Edit Account 54](#_Toc520105499)

[2.3.5.3. < Administrator> Create License 56](#_Toc520105500)

[2.3.5.4. < Administrator> Edit License 57](#_Toc520105501)

[2.3.6. Authenticated User Overview Use Case 59](#_Toc520105502)

[2.3.6.1. < Authenticated User> Logout Account 59](#_Toc520105503)

[2.3.6.2. < Authenticated User> Change Password 60](#_Toc520105504)

[2.3.7. Guest Overview Use Case 62](#_Toc520105505)

[2.3.7.1. < Guest> Login Account 62](#_Toc520105506)

[2.3.7.2. < Guest> Register 65](#_Toc520105507)

[2.3.7.3. < Guest> Reset Password 67](#_Toc520105508)

[3. Software System Attribute 69](#_Toc520105509)

[3.1. Usability 69](#_Toc520105510)

[3.2. Reliability 69](#_Toc520105511)

[3.3. Availability 69](#_Toc520105512)

[3.4. Security 69](#_Toc520105513)

[3.5. Maintainability 69](#_Toc520105514)

[3.6. Portability 69](#_Toc520105515)

[3.7. Performance 70](#_Toc520105516)

[4. Conceptual Diagram 70](#_Toc520105517)

[D. Software Design Description 71](#_Toc520105518)

[1. Design Overview 71](#_Toc520105519)

[2. System Architecture Design 71](#_Toc520105520)

[2.1 Web Server Architecture Design 71](#_Toc520105521)

[2.2 Client Architecture Design 72](#_Toc520105522)

[3. Component Diagram 73](#_Toc520105523)

[4. Detailed Description 74](#_Toc520105524)

[4.1 Class Diagram 74](#_Toc520105525)

[4.2 Class Diagram Explanation 75](#_Toc520105526)

[4.2.1 User 75](#_Toc520105527)

[4.2.2 Clinic 76](#_Toc520105528)

[4.2.3 License 76](#_Toc520105529)

[4.2.4 Administrator 76](#_Toc520105530)

[4.2.5 WorkingHours 77](#_Toc520105531)

[4.2.6 Appointment 77](#_Toc520105532)

[4.2.7 Patient 77](#_Toc520105533)

[4.2.8 Bill 77](#_Toc520105534)

[4.3. Interaction Diagram 78](#_Toc520105535)

[4.3.1. Activity Diagram 78](#_Toc520105536)

[4.3.1.1. Patient call to book appointment 78](#_Toc520105537)

[4.3.1.2. Patient send SMS to book appointment 78](#_Toc520105538)

[4.3.1.3. Register Account 79](#_Toc520105539)

[5. Interface 80](#_Toc520105540)

[6. Database Design 94](#_Toc520105541)

[7. Algorithms 96](#_Toc520105542)

[7.1. Schedule appointment 96](#_Toc520105543)

[7.1.1. Definition 96](#_Toc520105544)

[7.1.2. Define problem 96](#_Toc520105545)

[7.1.3. Solution 96](#_Toc520105546)

[E. System Implementation & Test 96](#_Toc520105547)

[1. Introduction 96](#_Toc520105548)

[1.1 Overview 96](#_Toc520105549)

[1.2 Test Approach 96](#_Toc520105550)

[2. Database Relationship Diagram 97](#_Toc520105551)

[2.1 Physical Diagram 97](#_Toc520105552)

[2.3 Data Dictionary 97](#_Toc520105553)

[3. Performance Measures 102](#_Toc520105554)

[3.1 Book appointment performance 102](#_Toc520105555)

[3.2 Book multiple appointments on the same number at the same time 103](#_Toc520105556)

[4. Test Plan 103](#_Toc520105557)

[4.1 Features to be tested 103](#_Toc520105558)

[4.2 Features not to be tested 103](#_Toc520105559)

[5. System Testing Test Case 105](#_Toc520105560)

[5.1. Mobile Test Case 105](#_Toc520105561)

[5.1.1. Registration 105](#_Toc520105562)

[5.1.2. Login 105](#_Toc520105563)

[5.1.3. View appointment list 106](#_Toc520105564)

[5.1.4. Change working hour 106](#_Toc520105565)

[5.1.5. Change appointment duration 107](#_Toc520105566)

[5.1.5. Change greeting message 108](#_Toc520105567)

[5.1.6. Buy License 108](#_Toc520105568)

[5.1.7. Auto update appointment list 109](#_Toc520105569)

[5.2. Web Test Case (Role Clinic) 109](#_Toc520105570)

[5.2.1. View Appointment List 109](#_Toc520105571)

[5.2.2. Refresh appointment List 109](#_Toc520105572)

[5.2.3. Change patient attendant status 110](#_Toc520105573)

[5.2.4. Change phone number blocking status 110](#_Toc520105574)

[5.2.4. Unblock phone number 111](#_Toc520105575)

[5.3. System Test Case 111](#_Toc520105576)

[5.3.1. Book appointment via calling. 111](#_Toc520105577)

[5.3.2. Book appointment via SMS. 112](#_Toc520105578)

[6. System implement 113](#_Toc520105579)

[6.1 Web Server Application 113](#_Toc520105580)

[6.2 Web Client Application 113](#_Toc520105581)

[6.3 Mobile Application 114](#_Toc520105582)

[F. Software User’s Manual 114](#_Toc520105583)

[G. Appendix 114](#_Toc520105584)

# List of Tables

[Table 1 Roles and Responsibilities 9](#_Toc520036155)

[Table 2: Hardware Requirement for Server 11](#_Toc520036156)

[Table 3: Hardware Requirement for Client 11](#_Toc520036157)

[Table 4: Software requirements 11](#_Toc520036158)

[Table 5: Roles and Responsibilities Details 13](#_Toc520036159)

[Table 6: Hardware Requirement for Server 72](#_Toc520036160)

[Table 7: Hardware Requirement for Client 72](#_Toc520036161)

[Table 8: Software requirements 73](#_Toc520036162)

[Table 9: Roles and Responsibilities Details 74](#_Toc520036163)

# List of Figures

Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| PO | Product owner |

1. Introduction

## Project Information

* Project name: **Call-Center on Mobile for Clinic**
* Project Code: **CallClinic**
* Product Type: **Mobile Application**
* Start Date: **May 18th, 2018**
* End Date: **August 31st, 2018**

## Introduction

In this document, we introduce an automatically Call-Center system for clinics. At the present, the clinic usually uses traditional ways to receive the call, that is hiring a switchboard operator. But, that solution has a few problems such as missed call or receive the wrong information. So, the use of switchboard operator to received call do not high efficiency.

Another way to solve this problem for the clinic is Call-Center, the clinic will rent Call-Center to receive the call from patients. In this method, Call-Center will provide more professional service for the clinic, but the cost is quite expensive and it is hard to exchange appointment information for the clinic.

That is the reason why we decided to build an automatically Call-Center system to save time and cost for the clinic. When there is a call from the patient, the system will pick up the call automatically, receive information and schedule appointment for the patient. And then, the system will send SMS for the patient when the appointment is due. The clinic just accesses the system to view all schedule appointment.

## Current Situation

Currently, Clinic using two ways for make appointment:

* **Switchboard operator:**

When patients want to book an appointment in the afternoon, they must call the clinic that morning. An employee at the clinic will record this information, based on that information, the employee will give the patient a specific time. The patient will go to the clinic at the time given and have the examination. All this process must be done manually, this is time-consuming and sometimes, some information is missing or mistaken because employees usually do many things at the same time.

* **Call Center:**

Call center acts as a middleman between the clinic and patients. At first, the clinic contacts a call center and sign up for a service there. When patients want to make an appointment, they call to call center's number, the staff at the call center will receive information from patients and transfer back to the clinic.

## Problem Definition

* **Clinic using Call-Center:**
  + **Advantages**:
* Provide professional service to take care patient.
  + **Disadvantages**:
* The expensive cost that’s why small clinics do not have funds to hire Call-Center.
* **Switchboard operator:**
  + **Advantages**:
* Reduce costs and suitable for small clinics.
  + **Disadvantages**:
* Clinics need a staff to wait for a call from a patient.
* The staff has difficulties to get information from the patient for the appointment.
* Clinics are managing patient information, schedule appointment manually, that way makes them time-consuming and inconvenient.

## Proposed Solution

### 5.1 Feature functions

Our solution is to build an automatically Call-Center system for clinics to provide appointment booking service.

When there is a call from the patient, the system will pick up the call automatically, receive information and schedules the appointment for the patient. Appointments will automatically schedule base on doctor's hours and patient's free time. The end result is that the doctor will only need to follow those schedule, and the patient will receive the correct date and time without the need to pick up or wait for the appointment. Clinics free times will be optimized and no more overload.

### 5.2 Values and Challenges

**Values:**

* Save the time and the money for the clinic.
* Provide better information management system for the clinic.

**Challenges:**

* Make the clinic’s phone (the device that we put in the clinic) auto pick up an incoming call.
* Get patient voice and analyze that.
* Push reply (by voice) back to the patient.
* Receive and analyze SMS from the patient.

## Functional Requirements

Functional requirements of the system are listed as below:

* **Doctor component**
* View list appointment
* **Service component**
* Hotline call service for patient make an appointment
* SMS service for patient make an appointment
* Make schedule
* Generate list appointment
* Notification to doctor
* Notification to the patient when the appointment is due
* **Administrator component**
* Manage Clinic account

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Full Name** | **Role** | **Position** | **Contact** |
| 1 | Kiều Trọng Khánh | Project Manager | Supervisor | khanhkt@fpt.edu.vn |
| 2 | Nguyễn Thế Phương | Developer | Leader | phuongntse62087@fpt.edu.vn |
| 3 | Phan Thành Thuận | Developer | Member | thuanptse62063@fpt.edu.vn |
| 4 | Nguyễn Cao Duy | Developer | Member | duyncse61032@fpt.edu.vn |
| 5 | Nguyễn Lương Tuấn Kiệt | Developer | Member | kietnltse61696@fpt.edu.vn |

Table 1 Roles and Responsibilities

1. Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

* **Official name:**  Call-Center on Mobile for Clinic
* **Vietnamese name:** Hệ thống nhân cuôc gọi tại phòng mạch tư nhân
* **Abbreviation:** CallClinic

### Problem Abstract

This project is our exertion to help the clinic reduce costs, the patients save time and simplify the user experience. However, when we start to identify problems and find the way to resolve them, we found many difficult things. We decide to use a smartphone as a switchboard to receive and answer the call. We try on both Android and IOS but as we know that, take the privilege of system phone is really hard things. We try to root the Android system and jailbreak IOS to take that privilege but it still not work correctly. And even after many hours of research, we find the way to do the first step is an auto-pickup phone call (only working on Samsung’s device). We instantly face with another problem; we cannot send voice answer when we are receiving the call. So, we fail in that way.

After that, we research about third-party framework provides programmable voice (like Twilio, Nexmo), but the cost of doing research and demonstration voice-answering is quite expensive.

Moreover, we research more about VoIP technical (WebRTC) for the call over internet protocol and about a third-party framework to receive SMS from the patient. Finally, we find out solutions to remind patients when their appointment is due.

### Project Overview

#### Current Situation

Below are the problems encountered in this project:

* Difficulty to get permission for the privilege of system phone call for both IOS and Android.
* Limit in human resources and time: Team has only 4 members and time for all project is about 13 weeks for writing the document, implementing the products and testing
* New techniques: Some team members are new to the techniques used in the project. The team needs an amount of time to get familiar with those techniques.
* The cost for research and demonstration with third-party framework programmable voice is quite expensive.
* Lack of knowledge about manage the clinics.
* Lack of the amount of the necessary data: doctors, nurses, patients, …

#### The Proposed System

The system will have four sub-systems:

* An API application for handling data and response data for the mobile application.
* Mobile application for the doctor to view schedule appointment.
* Web application for clinic manages patient, schedule appointment.
* Hotline server handles incoming SMS and patient call.

##### API application

* The server system takes responsibility to respond to all the requests and also manages and processes data.
* Provide APIs for Mobile Application.

##### Mobile application

* Sign in.
* View list schedule appointment.

##### Hotline server

* Send SMS for mobile phone.
* Receive incoming SMS for make appointment.
* Receive incoming Call from patient.

##### Web application

* Base component:
  + Sign in
  + Sign out
* Clinic component:
  + View list schedule appointment
  + Manipulate patient

#### Boundaries of the System

* Our system supports:
  + Providing an appointment booking service.
  + The patient can send SMS or call Clinic Hotline to make **an appointment.**
  + Hotline with the legacy call (call without internet) using Hotline server.
  + Hotline with VoIP call (call via internet protocol).
  + When an appointment has been booked successfully, send SMS or send a notification to the patient.
* Our system hasn’t supported:
  + Adjust appointment.
  + Clinic phone auto pick up an incoming call.
  + Interactive with the patient.

#### Future Plans

The current system only supports Android, so we recommend some features for future plans:

* The mobile application will be available on IOS.
* Build clinic mobile application auto pick up the incoming call.
* Take the privilege of system phone.

#### Development Environment

* + - 1. **Hardware requirements**

**For server**

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Minimum Requirements** | **Recommended** |
| Internet Connection | Cable, Wi-Fi (7 Mbps) | Cable, Wi-Fi (20 Mbps) |
| Computer Processor | Intel® Core ® i7 2.4GHz | Intel® Core ® i7 2.4GHz |
| Computer Memory | 1GB RAM | 1GB RAM or more |

Table 2: Hardware Requirement for Server

**For smartphone**

|  |  |  |
| --- | --- | --- |
| **Hardware** | **Minimum Requirements** | **Recommended** |
| Internet Connection | Wi-Fi (7 Mbps) | Wi-Fi (14 Mbps) |
| Operating System | Android 5 | Android 8 |
| Memory | 2GB RAM | 4GB RAM or more |

Table 3: Hardware Requirement for Client

* + - 1. **Software requirements**

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| Environment | Node JS  Android | Specification for developing Hotline Server application  Specification for developing a mobile application |
| Modeling tool | Star UML | Used to design diagram |
| IDE | Visual Studio Code 1.23.1  Android Studio 3.1.2 | Programming tools |
| DBMS | MySQL 5.6.30 | Used to create & manage the database for system |
| Source control | SourceTree 2.7.3 | Used for source control |
| Web browser | Chrome 42 or later | Testing browser |
| Mobile OS | Android 5 or later | Testing mobile application |

Table 4: Software requirements

## Project Organization

### Software Process Model

This project is developed using the Scrum model – part of an agile framework for Software development project. Our team chooses the Scrum model because of the following reasons:

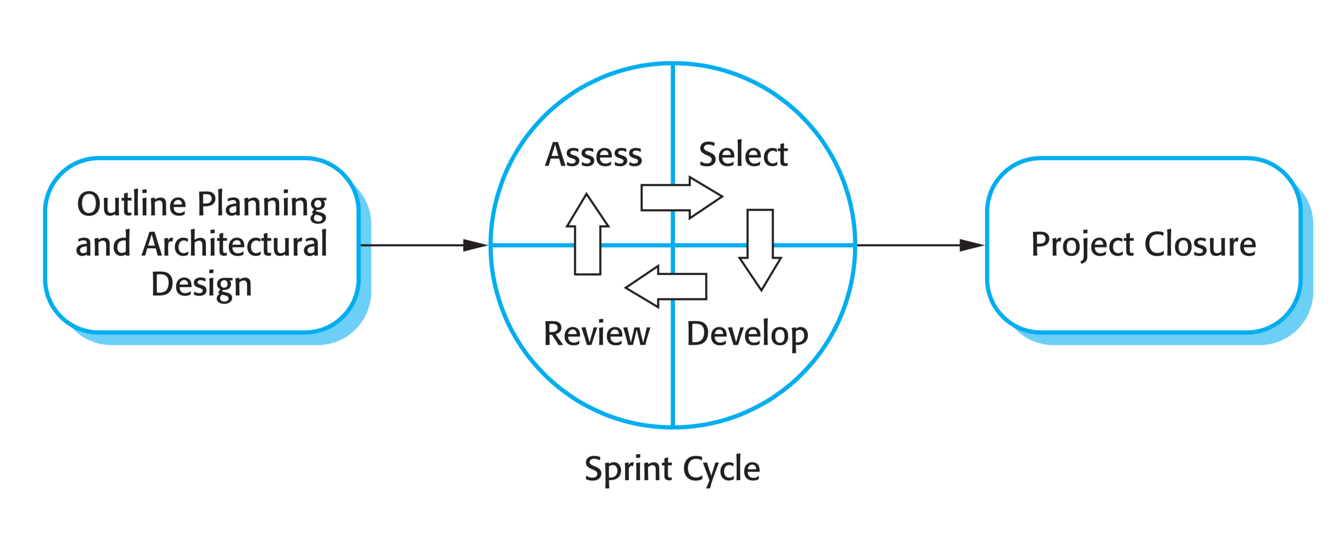
* Our team only has 4 members, and tasks are assigned vertically, do all steps from design, coding, testing, and implementation. Scrum is the most suitable model for the small and medium project.
* In the project, there are many new technologies that need to be learned. With the Scrum model, the team can learn and develop in parallel to meet the deadline.
* The product owner can change the requirement or extend scope. The team will adapt to change better.

Figure 1: The Scrum Process

Reference: *Software Engineering 9th by Somerville, page 73*

### Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Full name** | **Role in group** | **Responsibilities** |
| 1 | Kiều Trọng Khánh | Project Owner | ·         Specify scope and user requirement  ·         Give out technique and business analysis support  ·          Control the development process |
| 2 | Nguyễn Thế Phương | Scrum master | ·         Create Sprint Backlog and Product  Backlog  ·         Make sure the Scrum teams understand and follow the process.  ·         Help the team master scrum artifacts such as Sprint Backlog, Product Backlog, ...  ·         Writing report  ·         Always be present to answer questions and give advice when product owner or scrum member needs. |
| 3 | Nguyễn Thế Phương  Phan Thành Thuận  Nguyễn Cao Duy  Nguyễn Lương Tuấn Kiệt | Scrum team members | ·         Clarifying requirements  ·         Prepare documents  ·         Designing database  ·         GUI Design  ·         Coding  ·         Testing |

Table 5: Roles and Responsibilities Details

### Tools and Techniques

|  |  |
| --- | --- |
| **Tool/Technique** | **Name and version** |
| Back-end | NodeJS |
| IDE | Android Studio 3.1.2, Visual Studio Code 1.23.1 |
| Database | MySQL |
| Modeling Tool | Star UML |

Table 6: Tools and Techniques

## Project Management Plan

### Product Backlog

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sprint** | **Story ID** | **Story** | **Task ID** | **Task** |
| **1** | 1 | Introduction document | 1.1 | Project Information |
|  |  |  | 1.2 | Introduction |
|  |  |  | 1.3 | Current Situation |
|  |  |  | 1.4 | Problem Definition |
|  |  |  | 1.5 | Proposed Solution |
|  |  |  | 1.6 | Role and Responsibility |
|  |  |  | 1.7 | Functional Requirements |
|  | 2 | Product Backlog | 2.1 | Create Product Backlog |
|  | 3 | Project management plan | 3.1 | Problem Definition |
|  |  |  | 3.2 | Project Organization |
|  |  |  | 3.3 | Project management plan |
|  |  |  | 3.4 | Coding Convention |
| **2** |  |  |  |  |

Table 7: Sprint Backlog

### Sprint Backlog

#### Sprint 1 (18.05.2018 – 25.05.2018): Project initiation

##### Goal

1.1 Project Information

1.2 Introduction

1.3 Current Situation

1.4 Problem Definition

1.5 Proposed Solution

1.6 Role and Responsibility

1.7 Functional Requirements

2.1 Create Product Backlog

3.1 Problem Definition

3.2 Project Organization

3.3 Project management plan

3.4 Coding Convention

##### Development

|  |  |  |  |
| --- | --- | --- | --- |
| **Task ID** | **Task** | **Responsible** | **Review** |
| 1.1 | Project Information | PhuongNT | DuyNC |
| 1.2 | Introduction | DuyNC | ThuanPT, KietNLT |
| 1.3 | Current Situation | KietNLT | ThuanPT |
| 1.4 | Problem Definition | DuyNC | KietNLT |
| 1.5 | Proposed Solution | KietNLT | PhuongNT, KietNLT |
| 1.6 | Role and Responsibility | DuyNC | ThuanPT |
| 1.7 | Functional Requirements | ThuanPT | PhuongNT |
| 2.1 | Create Product Backlog | PhuongNT | DuyNC, ThuanPT |
| 3.1 | Problem Definition | PhuongNT | KietNLT |
| 3.2 | Project Organization | ThuanPT | KietNLT |
| 3.3 | Project management plan | PhuongNT | DuyNC, ThuanPT |
| 3.4 | Coding Convention | ThuanPT | PhuongNT |

**Table 8: Development**

#### Sprint 2

##### Goal

##### Development

### All Meeting Minutes

All meeting minutes are saved at: [here](https://d.docs.live.net/b38cb93f382240da/not%20provide%20yet)

## Coding Convention

* **NodeJS**
* **Naming convention:**
* Variables, properties and function names should use **lowerCamelCase**. They should also be descriptive. Single character variables and uncommon abbreviations should generally be avoided.
* Constants should be declared as a regular variables or static class properties, using all uppercase letters.
* **Functions.**
* Feel free to give your closures a name. It shows that you care about them, and will produce better stack traces, heap and CPU profiles.
* Use closures, but don't nest them. Otherwise, your code will become a mess.
* One method per line should be used if you want to chain methods. You should also indent these methods so it's easier to tell they are part of the same chain.
* **Comment:**

Use slashes for both single line and multiline comments. Try to write comments that explain higher-level mechanisms or clarify difficult segments of your code. Don't use comments to restate trivial things.

* **Android**
* **Naming convention:**
* Class names are written in **UpperCamelCase**. Ex: SignInActivity.
* Resources file names are written in **lowercase\_underscore**.
* Layout files should match the name of the Android components that they are intended for but move the top level component name to the beginning.
* Resource files in the values folder should be **plural.**
* **Functions**

Don’t ignore the exception and don’t catch the generic exception.

* **Comment:**

Use TODO comments for code that is temporary, a short-term solution, or good-enough but not perfect. TODOs should include the string TODO in all caps.

* **Others:**

Fully quality imports.

* **Angular**
* **File structure conventions**
* Some code examples display a file that has one or more similarly named companion files. For example, hero.component.ts and hero.component.html.
* The guideline uses the shortcut hero.component.ts|html|css|spec to represent those various files. Using this shortcut makes this guide's file structures easier to read and terser.
* **Rule of One**
* Style 01-01:

**Do** define one thing, such as a service or component, per file. **Consider** limiting files to 400 lines of code.

* Style 02-02:

**Do** define small functions. **Consider** limiting to no more than 75 lines.

* **Naming**
* Style 02-01:

**Do** use consistent names for all symbols. **Do** follow a pattern that describes the symbol's feature then its type. The recommended pattern is feature.type.ts.

* Style 02-02:
* **Do** use dashes to separate words in the descriptive name.
* **Do** use dots to separate the descriptive name from the type.
* **Do** use consistent type names for all components following a pattern that describes the component's feature then its type. A recommended pattern is feature.type.ts.
* **Do** use conventional type names including. service, component, pipe, module, and. directive. Invent additional type names if you must but take care not to create too many.
* Style 02-03**:**
* **Do** use consistent names for all assets named after what they represent.
* **Do** use upper camel case for class names.
* **Do** match the name of the symbol to the name of the file.
* **Do** append the symbol name with the conventional suffix (such as [Component](https://angular.io/api/core/Component), [Directive](https://angular.io/api/core/Directive), Module, [Pipe](https://angular.io/api/core/Pipe), or Service) for a thing of that type.
* **Do** give the filename the conventional suffix (such as: component’s, directive’s, module’s, pipettes, or. service.ts) for a file of that type.

Using Android coding convention from [https://source.android.com/setup/contribute/code-style#dont-use-finalizers](https://source.android.com/setup/contribute/code-style)

Using NodeJS coding convention from <https://google.github.io/styleguide/jsguide.html>

Using Angular coding convention from

[https://angular.io/guide/styleguide#single-responsibility](https://angular.io/guide/styleguide%23single-responsibility)

1. Software Requirement Specification

## User Requirement Specification

### Clinic Requirement

The doctor can do the following functions*:*

* Configuration clinic setting
* Get list appointment
* Update appointment status
* Manipulate patient information

### Administrator Requirement

The administrator is an employee in the system who has responsibility for manage clinic account, Administrator can do the following functions:

* Manipulate clinic account

### System Requirement

The system performs functions such as make an appointment and connects to external systems. The system does the following functions:

* Automatic appointment scheduling
* Analyze call recording
* Send notification when booking an appointment is successfully

## System Requirement Specification

### External Interface Requirement

#### User Interface

The user interface uses language is English for all web application and Vietnamese for mobile application.

#### Hardware Interface

**N/A**

#### Software Interface

Service 3rd party:

* Cloud service.
* Hotline third-party framework.

#### Communication Protocol

Use HTTP protocol 1.1 for communication between:

* Web application and web server.
* Mobile application and web server.

### System Overview Use case

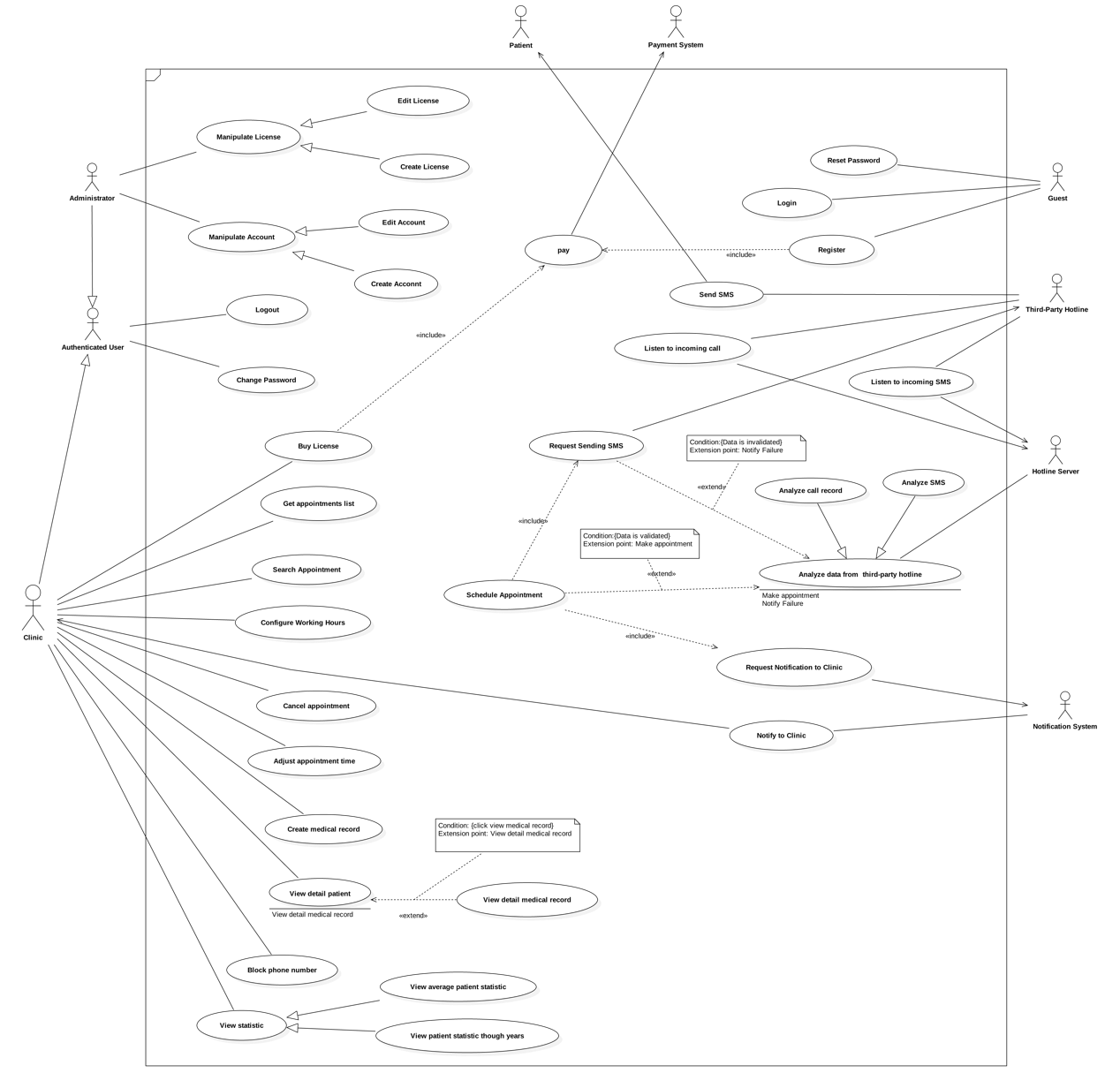
****

Figure 2: The system overview use case

### List of Use Case

#### Third-party Hotline Overview User Case

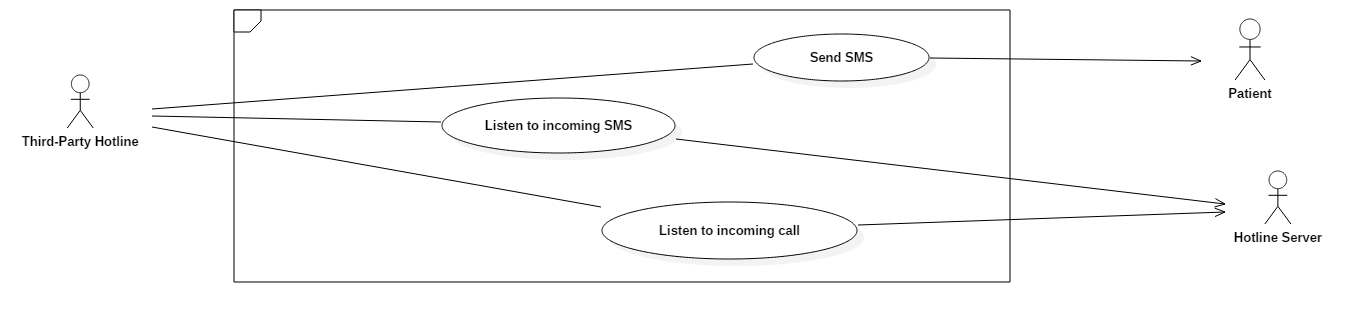


Figure 3:

##### <Third-party Hotline> Listen to Incoming Call



Figure 4:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_TPH.01 | | | |
| Use Case No. | UC\_TPH.01 | **Use Case Version** | 1.0 |
| Use Case Name | Listen to incoming call | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Patient   Summary:   * When the patient calls to third-party hotline server, it will request and receive an instruction document from hotline server. * Base on the information in the instruction document, a third-party hotline server will play the corresponding greeting message to the patient * After the beep sound (finish playing greeting message) third-party hotline will record the call content, save it into the system and then inform the information to hotline server.   Goal:   * Third-party hotline able to records the call information saves that information to the system and informs to hotline server.   Triggers:   * Patient call command to the third-party hotline   Preconditions:   * The clinic phone number must be registered at the third-party hotline   Post Conditions:   * Success: All information about the call is recorded * Fail: Notify an error to the patient     Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Call a hotline number. | Receive incoming call. | | 2 |  | Send request for an instruction to hotline server. | | 3 |  | Receive instruction.  (Hotline server response the request)  [Exception 1] | | 4 |  | Play greeting message to the caller. | | 5 |  | Start record the content. | | 6 | Speak up their name. | Recording the call. | | 7 | Hang-up. | Finish recording and save recorded file into the system. | | 8 |  | Send call record information to hotline server. |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | If the system receives instruction with the wrong format or not able to receive instruction. | Notify error to the patient and write an error to log file |   Relationships:   * Analyze data from third-party hotline: At step 8 of Main Success Scenario, the third-party hotline will send call record information to hotline server and then hotline server receives and analyze data to make an appointment by “Analyze data from third-party hotline” Use case.   Business Rules:   * When third-party hotline request for instruction, hotline server will return a formatted file that contains the instruction step by step, tell third-party hotline server how to handle the call. An example formatted instruction file below, the instruction tells third-party hotline to play the hello.mp3 to the caller first, then record the call. After finish record, the third-party hotline will send a request to http://example.com/callback to notify hotline server that record file is ready Example of formatted instruction file:  |  | | --- | | <?xml version="1.0" encoding="UTF-8"?>  <Response>  <Play>https://example.com/hello.mp3</Play> <Record recordingStatusCallback="http://example.com/callback" method="POST" />  </Response> |  * The greeting message file is in the MP3 format and provided by the clinic. * The record file store in third-party hotline is saved as WAV format, this is the same file that third-party hotline sends to hotline server after finish recording. * After the greeting message is finished (finish by the “beep” sound), the third-party hotline will start to record the content of the call. * The content that user speak via their phone during the call is a patient name in the appointment * The third-party hotline will send a request to the hotline server when the record file is ready, the request contains information about the call and the recording file. | | | |

Table 9 Use Case:

##### <Third-party Hotline> Send SMS

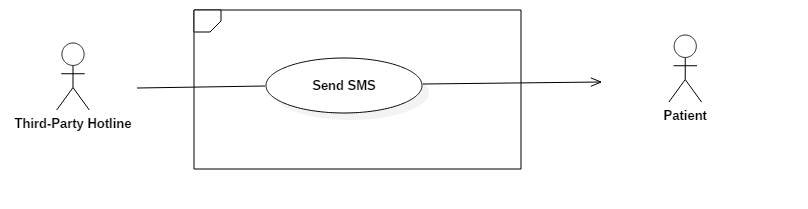


Figure 5:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_TPH.02 | | | |
| Use Case No. | UC\_TPH.02 | **Use Case Version** | 1.0 |
| Use Case Name | Send SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline server   Summary:   * This use case allows hotline server send an SMS to the patient through third-party Hotline. * The purposes of sending SMS to the patient are:   + When an appointment is booked: let the patient know about their appointment’s detail (also inform that they have successfully booked the appointment)   + When an appointment cannot be booked: let the patient know that they cannot book the appointment.   Goal:   * Third-party hotline sent SMS to the patient.   Triggers:   * Hotline server requests sending SMS to the third-party hotline.   Preconditions:   * The clinic phone number much be registered at the third-party hotline (third-party hotline use this number to send SMS to the patient)   Post Conditions:   * Success: Third-party hotline send SMS to the patient. * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Request sending SMS to | Send SMS to patient  [Exception 1] |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * In the case of SMS sent to inform the patient about their appointment information, the SMS structure is:  |  | | --- | | \_\_\_\_(1) mã số \_\_\_\_(2) đã đặt lịch khám tại phòng khám \_\_\_\_(3) ngày \_\_\_\_(4) lúc \_\_\_\_(5) |   (1): Patient’s name  (2): Appointment’s number  (3): Clinic’s name  (4): Appointment’s date  (5): Appointment’s time  **Example**: Phương Lan mã số 8 đã đặt lịch khám tại phòng khám Hoàng Hoa ngày 17-06-2018 lúc 03:52:00   * In case SMS sent to let the patient know that the clinic cannot take any appointments on that day anymore (the clinic is full). The SMS content is:  |  | | --- | | Hôm nay phòng khám đã nhận đủ lịch khám, xin quý khách quay lại vào ngày hôm sau. |  * In case SMS sent to inform the patient that they have already booked the appointment (patient book appointments multiple time on the same day). The SMS’ structure is:  |  | | --- | | Hôm nay quý khách đã đặt lịch khám cho bệnh nhân \_\_\_\_\_(1) rồi. Xin quý khách vui lòng quay lại vào hôm sau. |   (1): Patient’s name  **Example:** Hôm nay quý khách đã đặt lịch khám cho bệnh nhân Phương Lan rồi. Xin quý khách vui lòng quay lại vào hôm sau.   * In case the appointment cannot be booked due to system error, the SMS content is  |  | | --- | | Đã có lỗi xảy ra khi đặt lịch hẹn, xin quý khách vui lòng thử lại sau. | | | | |

Table 10 Use Case:

##### <Third-party Hotline> Listen to Incoming SMS



Figure 6:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_TPH.03 | | | |
| Use Case No. | UC\_TPH.03 | **Use Case Version** | 1.0 |
| Use Case Name | Listen incoming SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Patient.   Summary:   * This use case allows third-party hotline listen to incoming SMS from patient. Third-party will save SMS information and send a request to hotline server, notify hotline server that a new SMS has received.   Goal:   * Third-party hotline able to listen to incoming SMS.   Triggers:   * When the patient sends SMS to a hotline number.   Preconditions:  Post Conditions:   * Success: SMS’s information is recorded. * Fail: N/A.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Send SMS to the hotline number | Receive incoming SMS | | 2 |  | Save SMS’s information | | 3 |  | Send request to hotline server, notify that a new SMS is received |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * SMS content much be less than 255 characters * SMS structure much follow is patient: MP<space><patient’s name>   For example: MP Hoang Hoa | | | |

Table 11 Use Case:

#### Hotline Overview Use Case

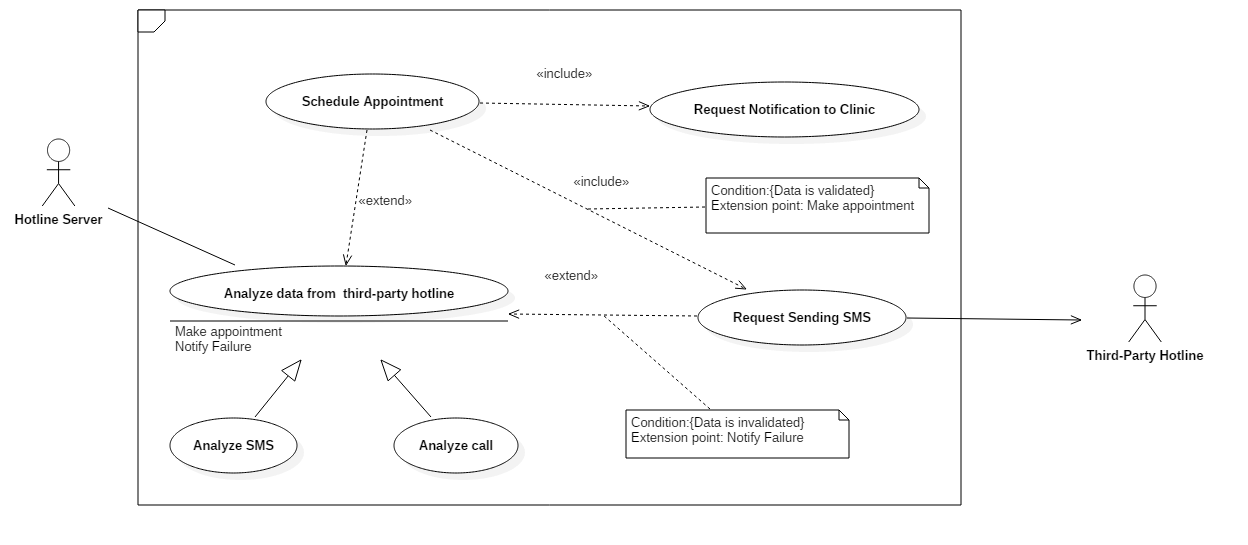


Figure 7:

##### <Hotline Server> Analyze SMS

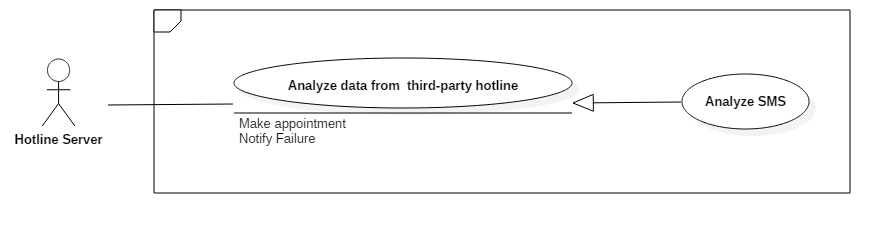


Figure 8:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.01 | | | |
| Use Case No. | UC\_HL.01 | **Use Case Version** | 1.0 |
| Use Case Name | Analyze SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline Server.   Summary:   * This use case allows the hotline server to analyze data from the third-party hotline. Whenever it receives a request from third-party hotline new SMS is received.   Goal:   * SMS’s information from a third-party hotline is analyzed.   Triggers:   * Whenever it receives a request from third-party hotline new SMS is received   Preconditions: N/A  Post Conditions:   * Success: SMS from a third-party hotline is analyzed. * Fail: Log error message to the system.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Send request to contain information about new SMS | Get SMS data from the third-party hotline. | | 2 |  | Analyze SMS data  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | SMS content’s length is over 255 character or SMS is not structured properly | Log error message |   Relationships:   * Schedule Appointment * Request Sending SMS   Business Rules:   * SMS’s content much is less than 255 characters. * SMS structure much follow is pattent: MP<space><patient’s name>   For example: MP Hoang Hoa | | | |

Table 12 Use Case:

##### <Hotline Server> Analyze Call



Figure 9:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.02 | | | |
| Use Case No. | UC\_HL.01 | **Use Case Version** | 1.0 |
| Use Case Name | Analyze Call | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline Server.   Summary:   * This use case allows the hotline server to analyze data from the third-party hotline. Whenever it receives a request from third-party hotline new Call is received.   Goal:   * Call’s information from third-party hotline is analyzed.   Triggers:   * Whenever it receives a request from third-party hotline new SMS is received   Preconditions: N/A  Post Conditions:   * Success: SMS from a third-party hotline is analyzed. * Fail: Log error message to the system.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Send request to contain information about new SMS | Get Call data from the third-party hotline | | 2 |  | Send call’s record URL (record file) to Speech To Text Service | | 3 |  | Analyze result from Speech to Text Service  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Text content is over 255 characters of empty | Log error message |   Relationships:   * Schedule Appointment * Request Sending SMS   Business Rules:   * Text content (extract from the call ) much be less than 255 characters and could not be empty | | | |

Table 13 Use Case:

##### <Hotline Server> Schedule Appointment

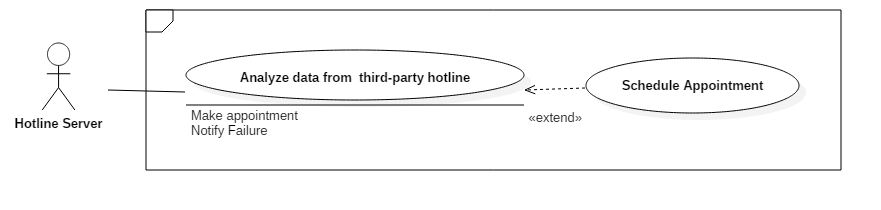


Figure 10:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.02 | | | |
| Use Case No. | UC\_HL.02 | **Use Case Version** | 1.0 |
| Use Case Name | Schedule Appointment | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | High |
| Actor:   * Hotline Server   Summary:   * This use case allows hotline server to schedule an appointment base on clinic’s configuration whenever a valid data is analyzed from the third-party hotline.   Goal:   * The new appointment is scheduled   Triggers:   * When data from third-party hotline is analyzed and valid   Preconditions:   * Data from third-party is valid   Post Conditions:   * Success: Appointment scheduled successful * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Send analyzed data | Receive analyzed data from the system | | 2 |  | Check user’s configuration  [Alternative Scenario 1] | | 3 |  | Schedule an appointment |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | If user’s configuration not suitable for a new appointment (not have enough time left, off day v…v…) | Request sending failure SMS to the third-party hotline and log the event into the system |   Exceptions: N/A  Relationships:   * Request sending SMS * Analyze data from the third-party hotline * Request notification to the clinic   Business Rules:   * If the clinic does not have enough time left to schedule an appointment, the system will do the procedure to send SMS error to the patient * The appointment’s time is calculated base on the formula: Base time + Examination duration The base time is the last appointment time + examination duration. If not appointment is made on that day, base time is the start working time of clinic   Ex: Last appointment time is 13:30 and the examination duration is 30 minutes then the new appointment time is 14:00   * If the value of new appointment time + examination duration is exceeding clinic’s end working hours. Then the appointment is canceled.   Ex: New appointment time is 13:45 and the examination duration is 30 but the clinic is not working after 14:00, then the appointment is canceled   * No appointments are made in off day. | | | |

Table 14 Use Case:

##### <Hotline Server> Request Sending SMS

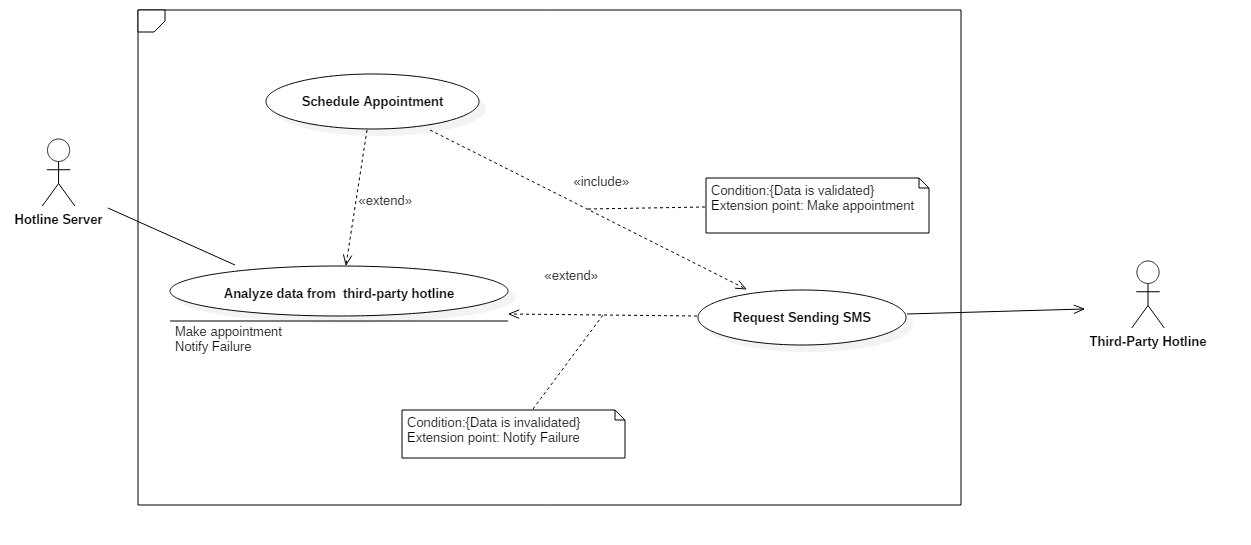


Figure 11:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.03 | | | |
| Use Case No. | UC\_HL.03 | **Use Case Version** | 1.0 |
| Use Case Name | Request sending SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline Server   Summary:   * This use case allows Hotline server request sending an SMS to third-party hotline whenever it needs to inform patient some information. The information can be various such as notify the patient about their appointment’s information, or their appointment cannot be booked.   Goal:   * SMS is sent to the patient   Triggers:   * When actor request sending SMS to the Third-party hotline   Preconditions: N/A  Post Conditions:   * Success: Request sending SMS successful requested * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 |  | Request sending SMS to the third-party hotline |   Alternative Scenario: N/A  Exceptions: N/A  Relationships:   * Schedule appointment * Analyze data from the third-party hotline   Business Rules:   * The send SMS request only made when   + The new appointment is made (send SMS contain appointment’s information)   + An appointment is canceled (send SMS to inform the patient that their appointment cannot be booked at the moment)   + Clinic adjust appointment time (send SMS to contain appointment’s information) | | | |

Table 15 Use Case:

##### <Hotline Server> Request Notification to Clinic



Figure 12:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.04 | | | |
| Use Case No. | UC\_HL.04 | **Use Case Version** | 1.0 |
| Use Case Name | Request Notification to Clinic | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Low |
| Actor:   * Firebase   Summary:   * This use case allows Firebase to send a notification to the clinic in case that clinic has a new appointment   Goal:   * Notify clinic that they have a new appointment   Triggers:   * When hotline server request send notification   Preconditions:   * A new appointment is made   Post Conditions:   * Success: Clinic receive notification. * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Request send notification to clinic | Push notification to clinic |   Alternative Scenario: N/A  Exceptions: N/A  Relationships:   * Schedule appointment   Business Rules:   * The only request sends a notification when a new appointment is successfully created * Notify clinic that associate with the previous appointment only ( notify the right clinic, not notify all) | | | |

Table 16 Use Case:

#### Firebase Overview Use Case

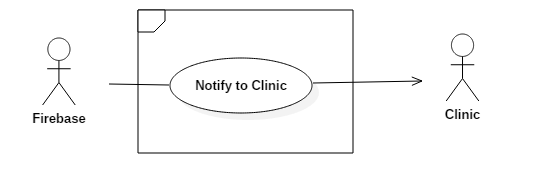


Figure 13:

##### < Firebase> Notify to Clinic

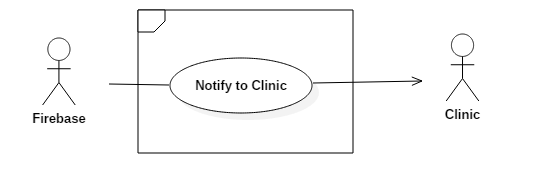


Figure 14:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_FB.01 | | | |
| Use Case No. | UC\_FB.01 | **Use Case Version** | 1.0 |
| Use Case Name | Notify to Clinic | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Firebase   Summary:   * This use case allows Firebase to send a notification to Clinic   Goal:   * Send a notification to the clinic when we want to notify clinic that they have a new appointment   Triggers:   * When hotline server request sending notification   Preconditions: N/A  Post Conditions:   * Success: Clinic receive notification * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Send request sending a notification to the clinic | Push notification to the clinic |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * The only request sends a notification when a new appointment is successfully created * Notification content: Bạn có một cuộc hẹn mới! | | | |

Table 17 Use Case:

#### Clinic Overview Use Case

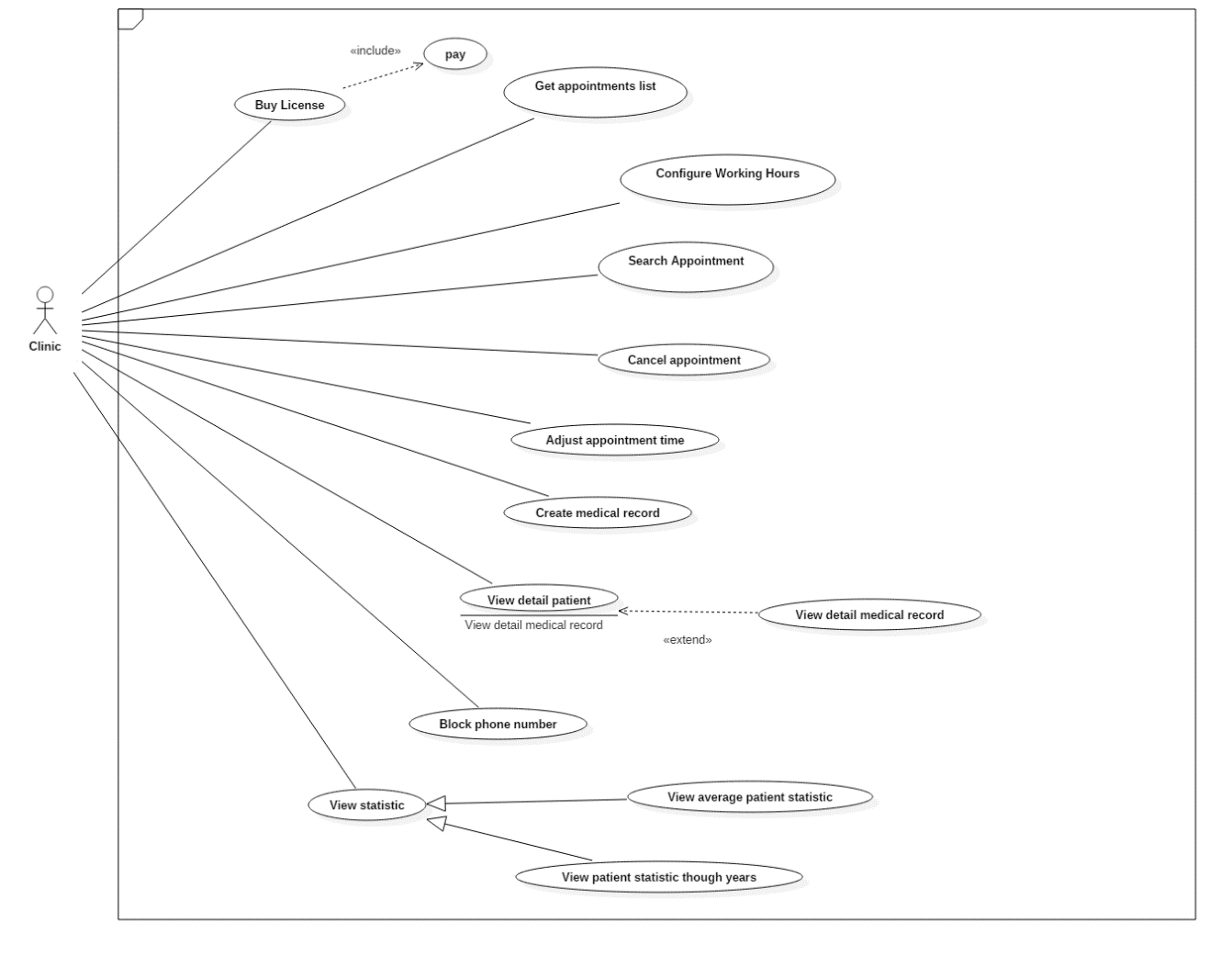


Figure 15:

##### <Clinic> Get Appointment List

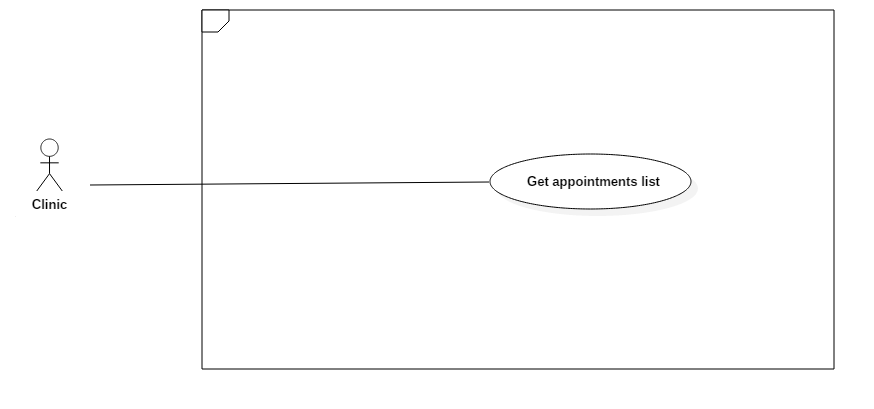


Figure 16:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C01 | | | |
| Use Case No. | UC\_C01 | **Use Case Version** | 1.0 |
| Use Case Name | Get appointment list | | |
| Author | ThuanPT | | |
| Date | 27/05/2018 | **Priority** | High |
| Actor:   * Clinic.   Summary:   * This use case allows the clinic to choose a date and then get appointment list base on that date. Default appointment list will get the current date.   Goal:   * The clinic can view appointment’s information.   Triggers:   * Clinic sends the get appointment list command.   Preconditions:   * Login the system with clinic role.   Post Conditions:   * Success: Appointment list will display appointments in UI or will be displayed on the label “Không có cuộc hẹn”. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Clinic select date, which from active license date to current date. |  | | 2 | Clinic sends a command to get the appointment list to the system. | Appointment list is displayed with information:   * STT * Tên * Số Điện Thoại * Giờ khám   [Alternative 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Clinic sends a command to get the appointment list to the system. | Appointment list is displayed not information. |   Exceptions: N/A  Relationships: N/A.  Business Rules:   * The clinic will get the appointment list, it is registered by the patient. * Show appointment list belongs base on the selected date. * The appointment list is organized in the following by information: * STT * Tên * Số Điện Thoại * Giờ khám   - Time will be format HH:MM:SS. Ex: 10:30 hours  - Length maximum of patient’s name is 30 characters.  - Length maximum of the phone number is 15 numbers.   * The format will be sort by time increasing. It’s convenient for reading and printing. * Appointment’s color display:   + Gray - The time of appointment has passed.   + White - The time of appointment has not yet arrived. | | | |

Table 18 Use Case:

##### <Clinic> Search Appointment

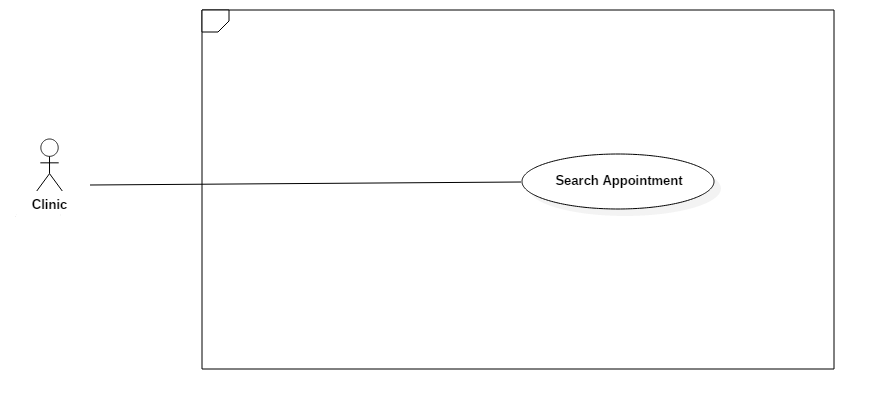


Figure 17:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C02 | | | |
| Use Case No. | UC\_C02 | **Use Case Version** | 1.0 |
| Use Case Name | Search appointment | | |
| Author | ThuanPT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows the clinic to input username or phone number and then search appointment.   Goal:   * The clinic would be found an appointment.   Triggers:   * Clinic sends search command.   Preconditions:   * Login the system with clinic role.   Post Conditions:   * Success: Appointment will be found or appointment is not found. * Fail: N/A.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Clinic input name or phone number in the search field |  | | 2 | Clinic sends a command to get an appointment to follow name or phone number in the search field. | Show Appointment has been searching on appointment list.   * STT * Tên * Số điện thoại * Giờ   [Alternative 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Clinic sends a command to get an appointment to follow name or phone number in the search field. | Show result not found on the appointment list. |   Exceptions: N/A.  Relationships: N/A.  Business Rules:  - The clinic will input an appointment’s information. Just input name or phone number, cannot input both of them.  - The result after searching will be shown with an approximate name or phone number and belong to the date selected.  - The result of searching will display on appointment list.   * If the appointment exists: show appointment in the list. * If the appointment is not found: show “không tìm thấy kết quả” label in the list. | | | |

Table 19 Use Case:

##### <Clinic> Configure working hours

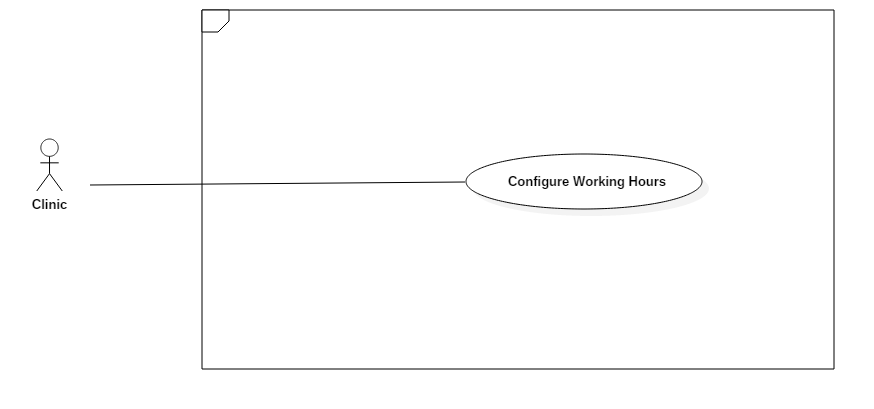


Figure 18:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C03 | | | |
| Use Case No. | UC\_C03 | **Use Case Version** | 1.0 |
| Use Case Name | Configure working hours | | |
| Author | ThuanPT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows the clinic to choose start hour, end hour and days of the week and then setup working hours of clinic their self.   Goal:   * The clinic has working hours.   Triggers:   * Clinic sends working hours’ command.   Preconditions:   * Login the system with clinic role.   Post Conditions:   * Success: Clinic setup working hours the system. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | The clinic goes to working hours view | The system requires to identify information from the clinic:   * Start hours: time input, required. * End hours: time input, required. * Days of the week: number input, required | | 2 | Clinic inputs start hour and end hour.  Clinic input option the days of the week.  [Exception 1] |  | | 3 | Clinic send a command to set up working hours to the system | The clinic will be set up working hours |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Clinic input start an hour later than end hour | The system shows an error message to ask clinic enters be conflict. |   Relationships: N/A.  Business Rules:  - To set working hours, the clinic will select a start time and an end time of one day and then mark the days of the week.  **-** “Giờ bắt đầu” field must to earlier than “Giờ kết thúc”.  - After clinic setup working hours, the system will display working hours’ information of every day follow card format:   * Thứ. * Giờ bắt đầu. * Giờ kết thúc. | | | |

Table 20 Use Case:

##### <Clinic> Buy License

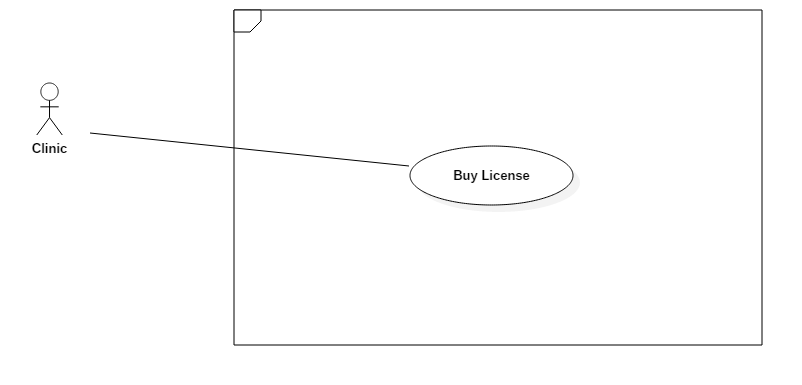


Figure 19:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C04 | | | |
| Use Case No. | UC\_C04 | **Use Case Version** | 1.0 |
| Use Case Name | Buy license | | |
| Author | ThuanPT | | |
| Date | 05/06/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows clinic select license and then send request pay command to the payment system.   Goal:   * The clinic has a license.   Triggers:   * Clinic sends buy license command.   Preconditions:   * Login the system with clinic role.   Post Conditions:   * Success: Clinic select license for checkout. * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | The clinic goes to buy license view | The system requires to identify information from the clinic:   * Tên * Giá * Thời lượng * Mô tả | | 2 | Clinic select license. |  | | 3 | Clinic sends a command to buy a license. | The clinic will send buy license command to payment methods. |   Alternative Scenario: N/A  Exceptions: N/A   * Relationships: Pay: At step 3 of Main Success Scenario, the clinic will send buy a license to the payment system and then payment system receive and pay to license by “Pay” Use case.   Business Rules:  - License list will be displayed in the following information:   * Tên * Giá * Thời lượng * Mô tả   - Description will help the clinic understand about that license.  - The clinic will select license before payment.  - After the clinic chooses a license, the clinic will pay through the payment system.  - The clinic can use functions be provided by the system after payment license is successful. | | | |

Table 21 Use Case:

##### <Clinic> Cancel Appointment

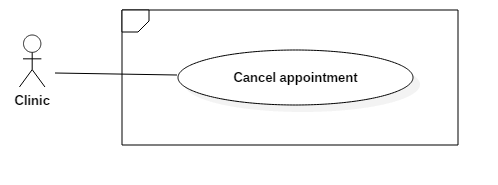


Figure 20:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C05 | | | |
| Use Case No. | UC\_C05 | **Use Case Version** | 1.0 |
| Use Case Name | Cancel Appointment | | |
| Author | KietNLT | | |
| Date | 17/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * Sometimes doctors are busy or have an urgent, they need to take a whole day off. The remaining appointments need to be canceled; this use case is made to support clinics in that situation. The clinic can send command cancel appointment and all the appointment remaining in that day will be canceled.   Goal:   * All appointment remaining in the day is canceled.   Triggers:   * Clinic send cancel appointment command.   Preconditions:   * Clinic much have at least one appointment left to cancel.   Post Conditions:   * Success: All appointment remaining are canceled * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Request cancel appointment command | Cancel all appointment remaining in that day. |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * Success message:  |  | | --- | | Thay đổi giờ khám thành công | | | | |

Table 22 Use Case:

##### <Clinic> Adjust Appointment Time

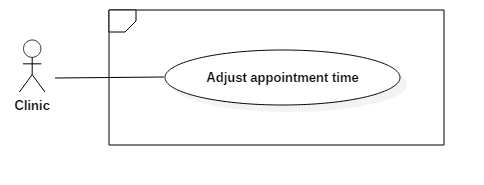


Figure 21:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C06 | | | |
| Use Case No. | UC\_C06 | **Use Case Version** | 1.0 |
| Use Case Name | Adjust Appointment Time | | |
| Author | KietNLT | | |
| Date | 17/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows clinic adjust appointment time in case doctors are late for the appointment. The adjustment period can be varied as long as after adjusting the time is not bound to the next day.   Goal:   * Adjust the appointment time.   Triggers:   * Clinic sends adjust appointment command.   Preconditions:   * Clinic much have at least one appointment left to adjust.   Post Conditions:   * Success: All appointment remaining are adjusted. * Fail: Error message is shown to Clinic.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Make adjust appointment time request | Show the time picker | | 2 | Input time and confirm | Adjust appointments time  [Exception 1]  Show success message |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | After adjusting, the appointment time is bound to another day | Show error message to Clinic |   Relationships: N/A  Business Rules:   * Error message in [Exception 1] is  |  | | --- | | Thay đổi thời gian khám bệnh không thành công. Nguyên nhân: sau khi thay đổi, giờ khám vượt quá sang ngày hôm khác. |  * Success message:  |  | | --- | | Thay đổi giờ khám thành công | | | | |

Table 23 Use Case:

##### <Clinic> Block Phone Number

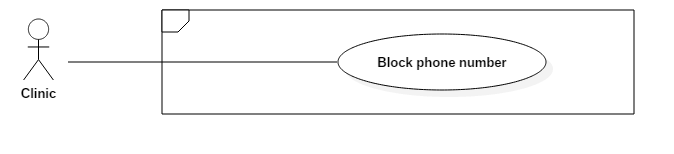


Figure 22:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C07 | | | |
| Use Case No. | UC\_C07 | **Use Case Version** | 1.0 |
| Use Case Name | Block Phone Number | | |
| Author | KietNLT | | |
| Date | 17/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows clinic block a phone number, stop that phone number from making an appointment again. This can be useful when some person continues making an appointment without actually coming.   Goal:   * Phone number being blocked in the system   Triggers:   * Clinic sends block phone number command.   Preconditions:   * N/A   Post Conditions:   * Success: Show success message * Fail: N/A   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Send block phone number command | Block phone number |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * Success message:  |  | | --- | | Thay đổi trạng thái chặn thành công. | | | | |

Table 24 Use Case:

##### <Clinic> Create Medical Record

Figure 23:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C08 | | | |
| Use Case No. | UC\_08 | **Use Case Version** | 1.0 |
| Use Case Name | Create medical record | | |
| Author | DuyNC | | |
| Date | 18/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * When the patient visits the clinic, the clinic creates a patient record to store the patient's information including the patient’s information, symptom, medicine, and counseling.   Goal:   * The clinics can be creating new patient’s medical record and the clinic based on the patient's medical records to make treatment easier and faster when the patient return.   Triggers:   * The clinic sends a command to create the patient’s medical record   Preconditions:   * The patient must be visited and complete the examination.   Post Conditions:   * Success: The patient’s medical record history will be saved. * Fail: Show error message.     Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor sends a command to create a medical record. | The system requires medical record information of the patient:   * Reminding: free text input, max length 255 * Disease: free text input, max length 255 * Symptoms: free text input, max length 255 | | 2 | Actor inputs information |  | | 3 | Actor sends a command to add medicine  [Alternative 1] | The system requires medicine information of the patient:   * Name: drop-down list, require, unique * Description: free text input, max length 255 * Quaintly: number, max length 11 | | 4 | Actor inputs information  [Alternative 2] |  | | 5 | Actor sends a command to save  [Alternative 3] | The new medical record has been adding to the system  [Exception 1,2]  Show message created successfully |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The actor does not send a command to add medicine. | The system will not display medicine information. | | 2 | The actor sends a command to remove. | Medicine will be removed out of the list medicines. | | 3 | The actor sends a command to cancel. | The system will close the create medical record view. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Patient’s medical record has existed. | Show message to notify the actor that medical record has existed in the system. | | 2 | The actor does not choose medicine name. | Show message to notify the actor that medicine name is required. |   Relationships: N/A  Business Rules:   * After the examination is completed, the clinic stores the patient’s medical record. * The medical record can contain multiple disease, symptom, and medicines. * If the actor adds medicine, medicine name is required and unique. * When the actor chooses medicine, the system will suggest description and quantity for this medicine. * When the actor creates medical record complete:   + If successfully, the patient’s medical record will be stored in the system and show the message “Tạo bệnh án thành công”.   + If fail, patient’s medical record will not be stored in the system and the system will show error message “Tạo bệnh án thất bại”. * If patient’s medical record has existed, the system show message “Bệnh án đã tồn tại”. | | | |

Table 25 Use Case:

##### <Clinic> View Detail Patient

Figure 24:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C09 | | | |
| Use Case No. | UC\_C09 | **Use Case Version** | 1.0 |
| Use Case Name | View detail patient | | |
| Author | DuyNC | | |
| Date | 18/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * When the patient comes to the clinic, the clinic will see patient’s information such as personal info, medical record history.   Goal:   * The clinic can see the details of the patient's information.   Triggers:   * The clinic sends a command to view the patient’s information.   Preconditions: N/A  Post Conditions:   * Success: All patient information shows up. * Fail: Show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor sends a command to view the patient’s information. | The system shows the patient’s information detail:   * Full name: free text input, max length 30 * Phone Number: text * Address: free text input, max length 255 * Date of birth: number text input, date format * Gender: drop-down list * Disease name: text * Symptom: text * Time: date | | 2 | Actor inputs information  [Alternative 1] |  | | 3 | Actor send a command to view the medical record  [Alternative 2] | The system shows the medical record detail:   * Medicine Name: text * Medicine Description: text * Medicine Quaintly: number * Reminding: text * Symptom: text | | 4 | Actor sends a command to save  [Alternative 3] | New patient’s information has been updating to the system.  Show message update successfully |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The actor does not input new information. | Data does not change. | | 2 | The actor does not send a command to view the medical record. | The system will not display view medical record information. | | 3 | Actor sends a command to cancel. | The system will close the medical record view. |   Exceptions: N/A  Relationships: View detail medical record.  Business Rules:   * When the patient comes to the clinic, the clinic can see the patient’s information. * Date of birth must be validated by date format: dd/mm/yyyy * When the clinic input patient’s information and sends a command to update:   + If successfully, the patient’s information will be a change in the system and show the message “Cập nhật thông tin bệnh nhân thành công”.   + If fail, the patient’s information will not be updated in the system and the system will show error message “Cập nhật thông tin bệnh nhân thất bại”. * When the clinic requests to view medical record detail, a pop up which contains the patient’s medical record information will display. | | | |

Table 26 Use Case:

##### <Clinic> View Average Patient Statistic

Figure 25:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C10 | | | |
| Use Case No. | UC\_C10 | **Use Case Version** | 1.0 |
| Use Case Name | View average patient statistic | | |
| Author | DuyNC | | |
| Date | 19/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * The clinic will see a line, which presents the average number of patients coming to the clinic across the years.   Goal:   * The clinic can know the number of patients who come to the clinic across the years.   Triggers:   * The clinic sends a command to view the average patient statistic chart.   Preconditions:   * The patient must book an appointment by call center system.   Post Conditions:   * Success: Display average patient statistics chart. * Fail: N/A.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | The actor goes to the average patient statistics chart view. | The system shows an average number of patients coming to the clinic across the years | | 2 | Actor chooses “from year” and “to year”  [Alternative 1] | The system shows an average number of patients coming to the clinic across the years have been chosen  [Exception 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The actor does not choose from year and to year. | The system shows an average number of the patient coming to the clinic in the last three years. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor chooses “from date” larger than “to date”. | Show message to notify the actor that “from date” must be smaller “to date”. |   Relationships: N/A  Business Rules:   * When the clinic goes to the average patient statistics chart view, the system will display an average number of the patient coming to the clinic in the last three years by a line. * When the clinic chooses “from date” and “to date”, the system will display an average number of patients coming to the clinic across the years have been chosen by a line. * If the clinic chooses “from date” larger than “to date”, the system will show error message “Năm bắt đầu phải nhỏ hơn năm kết thúc”. | | | |

Table 27 Use Case:

##### <Clinic> View Patient Statistic Through Years

Figure 26:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_ | | | |
| Use Case No. | UC\_ | **Use Case Version** | 1.0 |
| Use Case Name | View patient statistic through years | | |
| Author | DuyNC | | |
| Date | 19/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * The clinic will see lines, which presents the number of patients coming to the clinic each year.   Goal:   * The clinic can know the number of patients who come to the clinic each year.   Triggers:   * The clinic sends a command to view patient statistic through year chart.   Preconditions:   * The patient must book an appointment by call center system.   Post Conditions:   * Success: Display patient statistic through year chart. * Fail: N/A.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | The actor goes to patient statistic through years chart view. | The system shows a number of patients coming to the clinic each year | | 2 | Actor chooses “from year” and “to year”  [Alternative 1] | The system shows a number of patients coming to the clinic each year have been chosen  [Exception 1] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The actor does not choose from year and to year. | The system shows a number of the patient coming to the clinic in the last three years. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor chooses “from date” larger than “to date”. | Show message to notify the actor that “from date” must be smaller “to date”. |   Relationships: N/A  Business Rules:   * When the clinic goes to patient statistic through year chart view, the system will display a number of the patient coming to the clinic in the last three years by the lines. * When the clinic chooses “from date” and “to date”, the system will display a number of patients coming to the clinic each year have been chosen by the lines. * If the clinic chooses “from date” larger than “to date”, the system will show error message “Năm bắt đầu phải nhỏ hơn năm kết thúc”. | | | |

Table 28 Use Case:

##### <Clinic> Pay

Figure 27:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_ | | | |
| Use Case No. | UC\_ | **Use Case Version** | 1.0 |
| Use Case Name | Pay | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows actor pay for a license to use the call center system.   Goal:   * The actor can use the service of the call center system.   Triggers:  Actor sends pay command to the system.  Preconditions:   * This username has existed in the system with clinic role. * The actor must be choosing one license.   Post Conditions:   * Success: The account will be added used date. * Fail: The account will not be added used date and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor goes to payment method view | System displays license information:   * Name: text * Price: text * Duration: text * Description: text | | 2 | Actor chooses a license | System display payment method | | 3 | Actor sends a command to select a payment method | Forward to payment method view to process  the payment | | 4 | Actor process the payment method | Show message pay successful  [Exception 1] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | If payment failed | Show message to notify the user that payment failed and the license request has been aborted |   Relationships: Buy license.  Business Rules:   * The actor chooses a license to payment, license information includes:   + Name: name of the license   + Price: the price of license when paying   + Duration: time to use when buy license   + Description: description information of the license * When payment process is over:   + If the payment successfully, the expiration date of the license will be added.   + If the payment fails, the expiration date of the license does not change. | | | |

Table 29 Use Case:

#### Administrator Overview Use Case

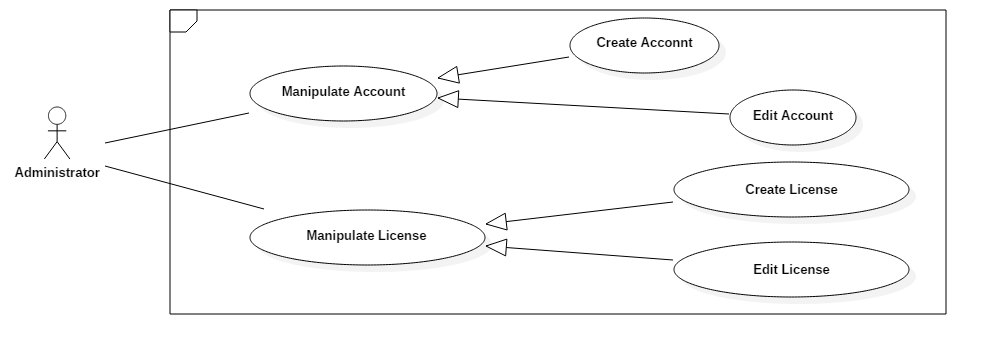


Figure 28:

##### < Administrator> Create Account

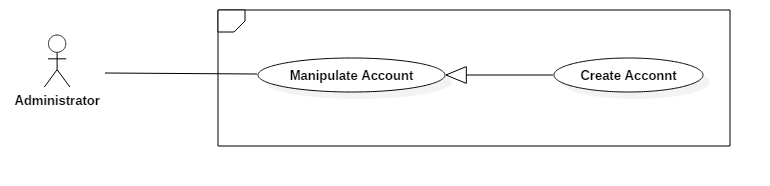


Figure 29:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_A01 | | | |
| Use Case No. | UC\_A01 | **Use Case Version** | 2.0 |
| Use Case Name | Create Account | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Administrator.   Summary:   * This use case allows the administrator to create a new account in the system. * The administrator can use the new account to login into the system.   Goal:   * The new account will be adding into the system.   Triggers:   * The administrator sends a command to create a new account.   Preconditions:   * The actor has been login with administrator role.   Post Conditions:   * Success: The new account is created. * Fail: Account is not created and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor goes to create the account view. | The system requires information:   * Username: free text input, required, max length 30, unique * Email: free text input, email format, unique, require * Phone Number: free text input, unique, max length 12, require * Full Name: free text input, max length 255 | | 2 | Actor inputs information. |  | | 3 | Actor sends create new account command.  [Alternative 1] | Account created and show message successfully.  [Exception 1]  [Exception 2]  [Exception 3]  [Exception 4]  [Exception 5] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor sends a command to reset. | The system will reset all field to blank. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor inputs username already exist. | System show warning message “Username has existed”. | | 2 | Actor inputs email already exist. | System show warning message “Email have existed”. | | 3 | Actor inputs phone number already exist | System show warning message “Phone number has existed”. | | 4 | The actor does not input the required field. | System notices that actor need to input all those fields. | | 5 | Actor inputs wrong some fields with the requirement. | System notices that actor need to re-input all those fields. |   Relationships: N/A  Business Rules:   * The administrator must be input require information to create, input information includes:   + Username, email, and phone number must not be duplicate.   + An email address must be validated by this regular expression:   /^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$/   * Password default is “123456”. * Password would be encrypting before save in the system and administrator can change the password. * A new account would be creating with inputted information:   + If create success, a new account will be added with role admin and active in the system. The administrator can be accessed in the system.   + If create fail, the account is not added to the system. And the system will show an error message for an administrator. | | | |

Table 30 Use Case:

##### < Administrator> Edit Account

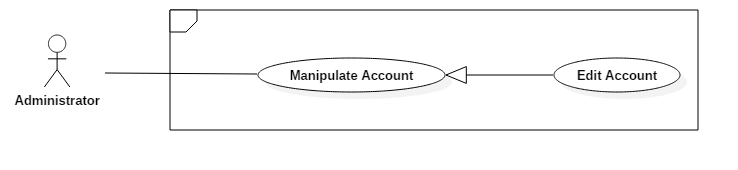


Figure 30:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_A02 | | | |
| Use Case No. | UC\_A02 | **Use Case Version** | 2.0 |
| Use Case Name | Edit Account | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Administrator.   Summary:   * This use case allows the administrator to update account information such as full name, phone number, and email.   Goal:   * Account information will be changed in the system.   Triggers:   * The administrator sends a command to edit account information.   Preconditions:   * The actor has been login and accessed in the system with the proper role.   Post Conditions:   * Success: The account information is updated. * Fail: The account information is not updated and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor goes to list account view. | The system shows list all accounts | | 2 | Actor sends a command to edit account information | The system requires information:  **User information:**   * Email: free text input, email format, unique * Full Name: free text input, required, max length 255 * Phone Number: number text input, unique, max length 12   **Clinic information:**   * Clinic Name: free text input, max length 255 * Address: free text input, max length 255 * Account Sid: free text input, max length 255 * Authentication Token: free text input, max length 255 | | 3 | Actor inputs information. |  | | 4 | Actor sends a command to save.  [Alternative 1] | Account information is updated.  [Exception 1]  [Exception 2]  [Exception 3] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor sends a command to cancel. | The system will close the edit account view. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor inputs email already exist | System show warning message “Email have existed”. | | 2 | Actor inputs phone number already exist | System show warning message “Phone number has existed”. | | 3 | Actor does not input the required field. | System notices that actor need to input all those fields. |   Relationships: N/A  Business Rules:   * The administrator must be input information to update, input information includes:   + Username, email, and phone number must not be duplicate.   + An email address must be validated by this regular expression:   /^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$/   * A new account would be updated with inputted information:   + If update success, the account information would be change.   + If update fail, the account information wouldn’t be change. And the system will show an error message for an administrator.   Reload account information and display new information has been updated. | | | |

Table 31 Use Case:

##### < Administrator> Create License

Figure 31:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_A03 | | | |
| Use Case No. | UC\_A03 | **Use Case Version** | 1.0 |
| Use Case Name | Create License | | |
| Author | DuyNC | | |
| Date | 19/07/2018 | **Priority** | Low |
| Actor:   * Administrator.   Summary:   * This use case allows the administrator to create a new license in the system to register using call center service.   Goal:   * A new license will be adding into the system.   Triggers:   * The administrator sends a command to create a new license.   Preconditions:   * The actor has been login with administrator role.   Post Conditions:   * Success: The new license is created. * Fail: License is not created and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor goes to create license view. | The system requires information:   * Name: free text input, required, max length 255 * Price: number text input, max length 11, required * Duration: number text input, max length 11, required * Description: free text input, max length 255 | | 2 | Actor inputs information. |  | | 3 | Actor sends a command to create new a license.  [Alternative 1] | License created and show message successfully.  [Exception 1]  [Exception 2] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor sends a command to reset. | The system will reset all field to blank. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The actor does not input the required field. | System notices that actor need to input all those fields. | | 2 | Actor inputs wrong some fields with the requirement. | System notices that actor need to re-input all those fields. |   Relationships: N/A  Business Rules:   * The administrator must be input require information to create a new license, input information includes price, and duration must be an integer. * A new license would be creating with inputted information:   + If create success, a new license will be added in the system. And the system will show a message “Create license successfully”   + If create fail, the license is not added to the system. And the system will show an error message for the administrator. | | | |

Table 32 Use Case:

##### < Administrator> Edit License

Figure 32:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_A04 | | | |
| Use Case No. | UC\_A04 | **Use Case Version** | 1.0 |
| Use Case Name | Edit License | | |
| Author | DuyNC | | |
| Date | 19/07/2018 | **Priority** | Low |
| Actor:   * Administrator.   Summary:   * This use case allows the administrator to update license information such as name, price, duration, and description.   Goal:   * License information will be changed in the system.   Triggers:   * The administrator sends a command to edit license information.   Preconditions:   * The actor has been login and accessed in the system with the proper role.   Post Conditions:   * Success: The license information is updated. * Fail: The license information is not updated and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor goes to list license view. | The system shows list all licenses | | 2 | Actor sends a command to edit license information | The system requires information:   * Name: free text input, required, max length 255 * Price: number text input, max length 11 * Duration: number text input, max length 11 * Description: free text input, max length 255 | | 3 | Actor inputs information. |  | | 4 | Actor sends a command to save.  [Alternative 1] | License information is updated.  [Exception 1]  [Exception 2] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor sends a command to cancel. | The system will close the edit license view. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The actor does not input the required field. | System notices that actor need to input all those fields. | | 2 | Actor inputs wrong some fields with the requirement. | System notices that actor need to re-input all those fields. |   Relationships: N/A  Business Rules:   * The administrator must be input information to update, input information includes price and duration must be an integer. * A new account would be updated with inputted information:   + If the update success, the account information would be change.   + If the update fails, the account information wouldn’t be change. And the system will show an error message for the administrator. * Reload account information and display new information has been updated. | | | |

Table 33 Use Case:

#### Authenticated User Overview Use Case

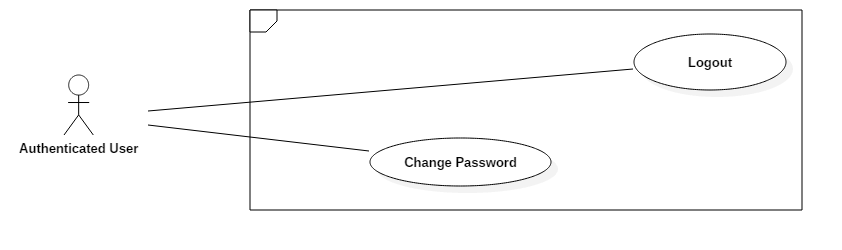


Figure 33:

##### < Authenticated User> Logout Account

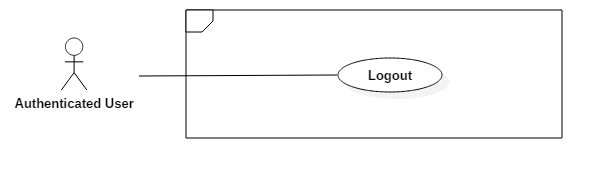


Figure 34:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_AU01 | | | |
| Use Case No. | UC\_AU01 | **Use Case Version** | 1.0 |
| Use Case Name | Logout Account | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Authenticated User.   Summary:   * This use case allows the actor to log out of the system.   Goal:   * The authenticated user stops accessing the system.   Triggers:   * The authenticated user sends the logout command.   Preconditions:   * Authenticated user has been login and accessed in the system.   Post Conditions:   * Success: Account is log out successfully. * Fail: N/A.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor sends a command to log out | The system requires to confirm logout account | | 2 | Authenticated user sends a command to confirm | * The system will log the user out of the system * The user will be navigated to the login page |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * After log out process, the role “Authenticated User” will become “Guest”. * User’s session is removed out of the system. * The system will display to Login view. | | | |

Table 34 Use Case:

##### < Authenticated User> Change Password

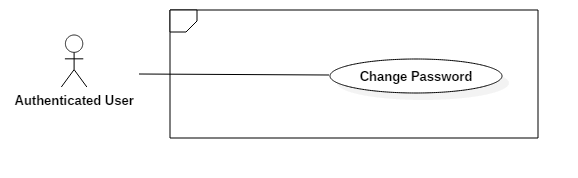


Figure 35:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_AU02 | | | |
| Use Case No. | UC\_AU02 | **Use Case Version** | 2.0 |
| Use Case Name | Change Password | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Authenticated User.   Summary:   * This use case allows actor change current password to new password.   Goal:   * The new password has been updated in the system and actor can log in by the new password.   Triggers:   * The authenticated user sends the change password command.   Preconditions:   * Authenticated user has been login with their property role.   Post Conditions:   * Success: The new password has been updated. * Fail: The new password cannot update and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Authenticated user goes to change password view. | The system requires information:   * Current password: password text input, require, max length 72 * New password: password text input, require, max length 72 * Confirm new password: password text input, require, max length 72 | | 2 | Authenticated user inputs information. |  | | 3 | Authenticated user sends a command to change the password. | Show message to notify that new password has been updated successfully.  [Exception 1,2] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The authenticated user enters a wrong current password. | Show message to notify that the current is not correct. | | 2 | The authenticated user inputs a new password and confirms new password not match. | Show message to notify that the confirm new password is not matched with the new password. |   Relationships: N/A  Business Rules:   * If the authenticated user enters a wrong current password, the system shows an error message “Current password is not correct”. * New password and confirm new password are matched each other if it does not match, the system shows an error message “Confirm password is not correct”. * New password would be encrypted before save in the system. * When the new password has been changed, the system shows the message “Change password successfully”. | | | |

Table 35 Use Case:

#### Guest Overview Use Case

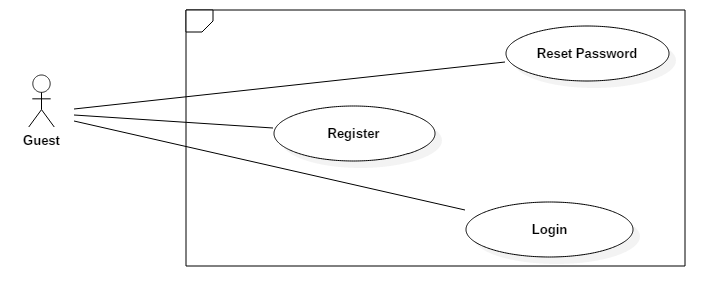


Figure 36:

##### < Guest> Login Account

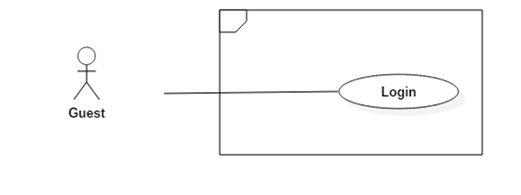


Figure 37:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_G01 | | | |
| Use Case No. | UC\_G01 | **Use Case Version** | 2.0 |
| Use Case Name | Login | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Guest.   Summary:   * This use case allows guest to login into the system.   Goal:   * Guest can access in the system with the property role to use functions of the system.   Triggers:   * Guest sends a command to log in.   Preconditions:   * The account uses to log in must be registered and authentication before access the system.   Post Conditions:   * Success: The guest accesses to the system successfully. * Fail: The guest cannot access the system and shows an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest goes to login view | The system requires identity information:   * Username: free text input, required, max length 30 * Password: password text input, required, max length 72 | | 2 | Guest inputs information. |  | | 3 | Guest sends a command to log in. | Guest will log in system with their specific role.  [Alternative 1]  [Exception 1,2] |   Alternative Scenario 1:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest goes to configuration view | The system requires configuration information:   * Examination Duration: time format, required * Start working Hour: time format, required * End working Hour: time format, required * Apply Dates: checkbox | | 2 | Guest sends a command to save | The system requires greeting message | | 3 | Guest sends a command to record a greeting message  [Alternative 2] | The system redirects to record view | | 4 | Guest sends a command to record | The system starts to record the content | | 5 | Guest speak up greeting message | The system records the greeting message | | 6 | Guest sends a command to finish | Finish recording and save record file into the system  System redirect to configure greeting message | | 7 | Guest send a command to save | System redirect to dashboard view |   Alternative Scenario 2:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest sends a command to choose greeting message file | System redirect to choose greeting message file | | 2 | Guest chooses a greeting message file | System redirect to configure greeting message |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Guest does not input required field. | System notices that actor need to input all those fields. | | 2 | Guest inputs wrong identity information. | Show message to notify that the identity information is not correct. |   Relationships: N/A  Business Rules:   * The guest will be redirected to a specific view based on their role in the system:   + If the role is “Administrator”, the system will display to Administrator dashboard view.   + If the role is “Staff”, the system will display to Staff dashboard view.   + If the role is “Clinic”, the system will display to Clinic dashboard view. * With the role is “Clinic”, for the first time to log in, the actor must be set up required information to use the system:   + Examination duration: Time for an examination with “hh:mm:ss” format.   + Start working hour: Time start working with “HH:mm:ss” format.   + End working hour: Time end working with “HH:mm:ss” format.   + Apply dates: Dates working in the week from Monday to Sunday.   + Greeting message: greeting of the clinic with “.mp3” format. * When guest login, if the license is expired, the system will redirect to payment view. | | | |

Table 36 Use Case:

##### < Guest> Register

Figure 38:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_G02 | | | |
| Use Case No. | UC\_G02 | **Use Case Version** | 2.0 |
| Use Case Name | Register | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Normal |
| Actor:   * Guest   Summary:   * This use case allows guest to register a new account to use the call center system.   Goal:   * New account added into the system.   Triggers:  Guest sends register command to the system.  Preconditions: N/A.  Post Conditions:   * Success: The new account is added to the system. * Fail: Account is not added and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest goes to register view | The system requires information:  **User information:**   * Username: free text input, required, max length 30, unique * Password: password text input, max length 72, required * Confirm password: password, max length 72, required * Email: email text input, unique, required   **Clinic information:**   * Clinic Name: free text input, required, max length 255 * Address: free text input, max length 255, required * Phone number: free text input, max length 12, required | | 2 | Guest inputs information |  | | 3 | Guest sends a command to register.  [Alternative 1]  [Alternative 2] | The new account has added to the system  [Exception 1]  [Exception 2]  [Exception 3]  [Exception 4]  The system sends an active link via email | | 4 | Guest send commands to active | System display active successfully view |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Guest sends a command to reset | System reset all field to blank | | 2 | Guest sends a command to cancel | The system will navigate to the login page |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Guest inputs username already exist. | Show message to notify that the username has existed. | | 2 | Guest inputs email already exist. | Show message to notify that the email has existed. | | 3 | Guest inputs a new password and confirms password not match. | Show message to notify that the confirm password is not matched with the new password. | | 4 | Guest does not input required field. | System notices that actor need to input all those fields. |   Relationships: N/A  Business Rules:   * Guest must be input require information to register, input information includes:   + Username, email, and phone number must not be duplicate.   + An email address must be validated by this regular expression:   /^\w+([\.-]?\w+)\*@\w+([\.-]?\w+)\*(\.\w{2,3})+$/   * Password would be encrypting before save in the system. * A new account would be creating with inputted information.   + If create success, a new account will be added with the role.   + If create fail, the account is not added to the system. And the system will show error message “Đăng ký tài khoản không thành công” * The initial status of account will be set to “Inactive”. * The new account has been adding into the system with clinic role. * The system sends an active link via email, after guest active account, the system will change the status of account from “Inactive” to “Active”. | | | |

Table 37 Use Case:

##### < Guest> Reset Password

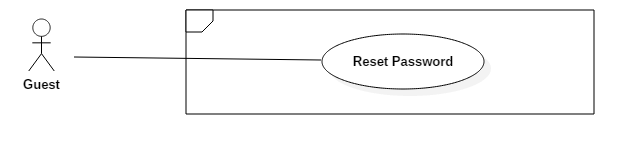


Figure 39:

|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_AU03 | | | |
| Use Case No. | UC\_G03 | **Use Case Version** | 2.0 |
| Use Case Name | Reset Password | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Authenticated User.   Summary:   * This use case allows actor change forgot password to new password. * The system will send a verification code to set up a new password via email.   Goal:   * The new password has been updated in the system and actor can log in by the new password.   Triggers:   * The authenticated user sends the reset password command.   Preconditions: N/A  Post Conditions:   * Success: The new password has been updated. * Fail: The new password cannot update and show an error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Authenticated user goes to request reset password view | The system requires information:   * Username: free text input, required, max length 30 * Email: free text input, required, email format, unique | | 2 | Authenticated user inputs information |  | | 3 | Authenticated user sends a command to reset | The system sends verify code via email  [Exception 1,2]  The system redirects to verify code view  The system requires information:   * Verify code: number text input, required, max length 11 | |  | Authenticated user inputs information |  | |  | Authenticated user sends a command to verify code | The system redirects to change password view  [Exception 3]  The system requires information:   * New password: password text input, require, max length 72 * Confirm new password: password text input, require, max length 72 | | 5 | Authenticated user inputs information. |  | | 6 | Authenticated user sends a command to change the password. | Show message to notify that new password has been updated successfully.  [Exception 4] |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | The authenticated user enters a wrong email | Show message to notify that the email is not correct | | 2 | The authenticated user enters email and username are not match in the database | Show message to notify that the email does not exist with any username | | 3 | The authenticated user enters a wrong verify code | Show message to notify that the verification code is not correct | | 4 | The authenticated user inputs a new password and confirms new password not match | Show message to notify that the confirm new password is not matched with the new password |   Relationships: N/A  Business Rules:   * When the authenticated user enters an email:   + If the email and username are not matching in the database, the system shows error message “Email chưa được đăng ký với tài khoản nào”.   + If the email and username are matched in the database, the system sends verify code via email. * If the authenticated user does not input verify code, the authenticated user uses username and password normally. * When authenticated user inputs verify code:   + If verify code is not correct, the system shows error message “Mã xác nhận không chính xác”.   + If verify code is correct, the system redirects to set up new password view. * New password and confirm new password are matched each other if it does not match, the system shows an error message “Confirm password is not correct”. * New password would be encrypted before save in the system. * When a new password has been changed, the system shows the message “Change password successfully”. | | | |

Table 38 Use Case:

## Software System Attribute

### Usability

* UI website is fit for each browser in each device.
* Mobile and web application use Vietnamese including layouts, dialogs, and messages.
* Staff should need less than 1 hour of training to use the system.

### Reliability

* Appointment always correct and never be duplicate.
* Using cloud computing that’s make the system more secure and prevent losing data.

### Availability

* System replies in maximum 20 seconds.
* The server has a backup electronic source.
* Hotline always pick up the incoming call.

### Security

* Each role of the user has a specific permission to interact with the system.
* The user should be authenticated and authorized when accessing to the system
* Input data is validated before saving to the database.

### Maintainability

The system is divided into separated modules such as: Server API, Server Hotline, Mobile App, …

### Portability

* The web application can be run on Chrome browser version 42 or later.
* The web application can run on Chrome, Firefox.
* The user can use the mobile application on devices running Android 5 or later.
* The user can book appointment send SMS or Call to Hotline.
* The user can call with or without internet.

### Performance

* System return for patients in 5 seconds or less when patients call to book an appointment.

## Conceptual Diagram

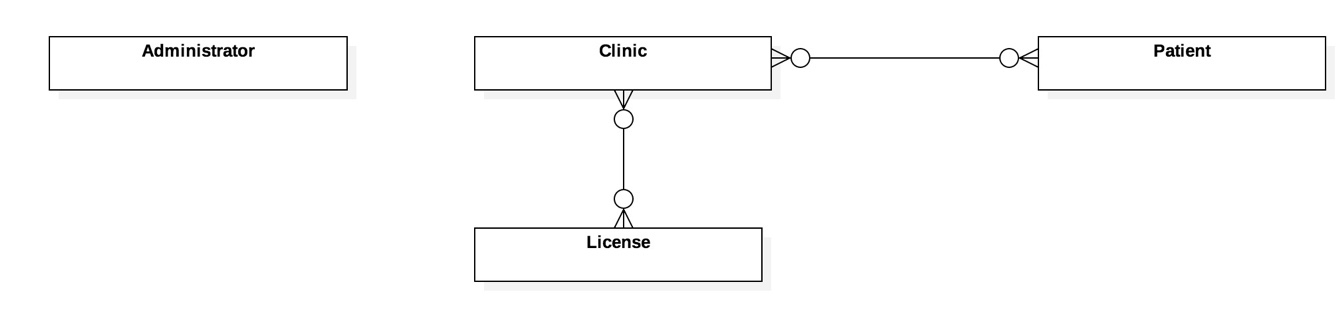


Figure 40: Conceptual diagram

|  |  |
| --- | --- |
| **Entity Data dictionary: describe all content of all entities** | |
| **Entity Name** | **Description** |
| Administrator | Contain the information's administrator.   * username: username of the administrator. It uses to log in to the system. * password: each user would be having a password. * phoneNumber: A telephone number of each user. * fullName: real name of the user |
| Clinic | Contain the information's clinic.   * Composite attribute Working hours: contain working hours of the clinic from Monday to Sunday. * address: location of the clinic. * clinicName: name of the clinic. * examinationDuration: the examination time of one slot * expiredLicense: Expiry date of the license   Relationship entity clinic:   * Each clinic can buy a license and each license has a deadline. So, when the license is expired, the clinic must buy a new license. * Each patient can book many appointments by the time at the clinic and each clinic has many appointments from patients, |
| License | Contain the information's license.   * price: A price of each license. * duration: duration of each license. * name: name of each license. * description: description of each license. |
| Patient | Contain the information's patient.   * phoneNumber: telephone number of the patient. * fullname: A real name of the patient. * address: address of the patient. |

Table 39: Conceptual diagram data dictionary

1. Software Design Description

## 1. Design Overview

- This document describes the technical and user interface Call-center. It includes the architectural design, the detailed design of common functions and business functions and the design of the database model.

- The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

- The detailed design describes a static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.

- The database design describes the relationships between entities and details of each entity.

- Document overview

* Section 2: gives an overall description of the system architecture design.
* Section 3: gives component diagrams that describe the connection and integration of the system.
* Section 4: gives the detail design description which includes a class diagram, class explanation, activity diagram and sequence diagram to details the application functions.
* Section 5: describe screen design.
* Section 6: describe fully attribute ERD.
* Section 7: describe algorithms.

## 2. System Architecture Design

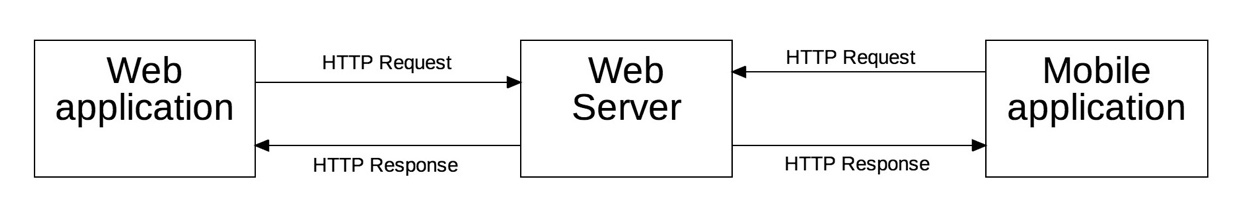


Figure 9. System Architecture

### 2.1 Web Server Architecture Design

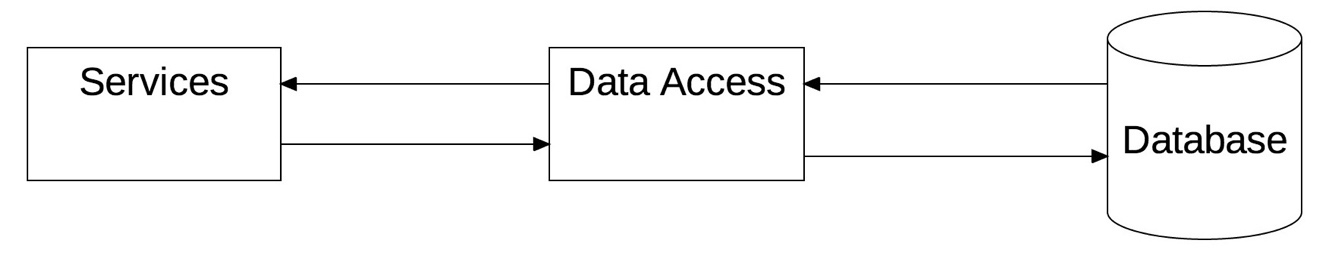


Figure 9. Web Server Architecture

In Web Server, we choose this architecture for development with the following components:

* Services: Provide API Services to communicate with the client
* Data Access: Refer to Services to access or retrieve data stored within a Database
* Database: Stored data of the system

### 2.2 Client Architecture Design

In Mobile Application and Web Application, the system is developed under MVC architecture. We choose this architecture because of the following advantage:

* The Model-View-Controller pattern highly supports the separation of concerns. This advantage not only increases the testability of the code but it also makes it easier to extend, allowing a fairly easy implementation of new features.
* If the Views respect the **single responsibility principle** then their role is just to update the Controller for every user event and just display data from the Model, without implementing any business logic. In this case, UI tests should be enough to cover the functionalities of the View.

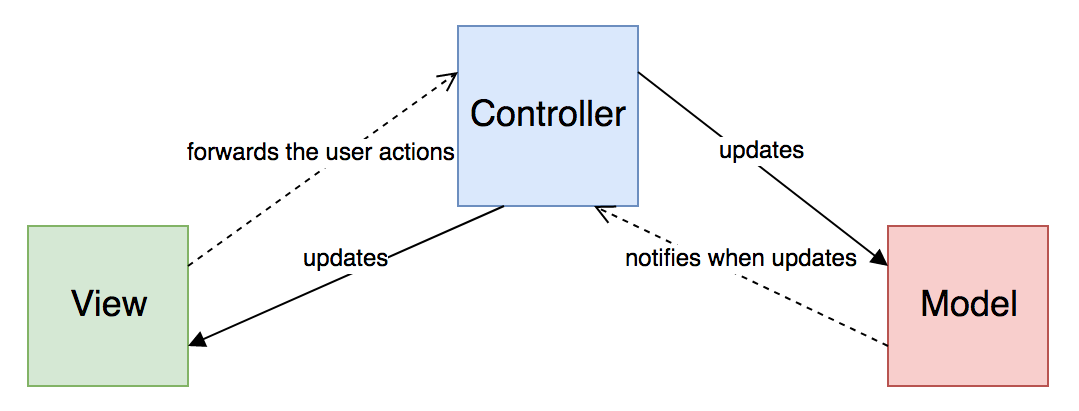


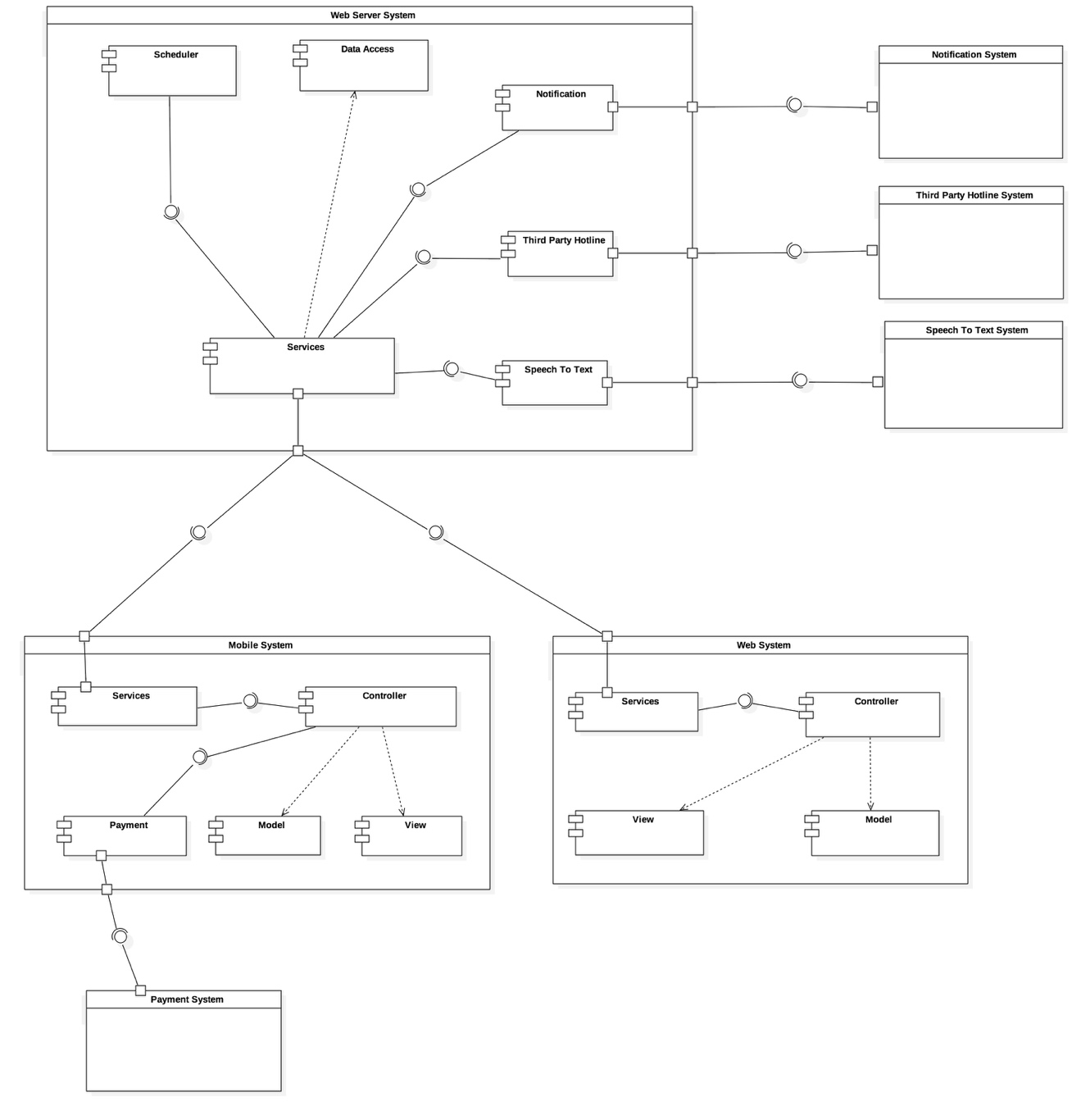
Figure 9. Mobile MVC Architecture

(Reference: https://medium.com/swlh/ios-design-patterns-a9bd07818129)

MVC architecture has the following components:

* **Model:** is where the application’s data objects are stored. The model doesn’t know anything about View and Controller
* **View:** The view is the Representation of the Model. The view has a responsibility to render the User Interface (UI) and communicate with the controller when the user interacts with the application
* **Controller:** The controller is Glue that ties the app together. It’s the master controller for what happens in the application. When the View tells the controller that a user clicked a button, the controller decides how to interact with the model accordingly. Based on data changing in the model, the controller may decide to update the state of the view as appropriate.

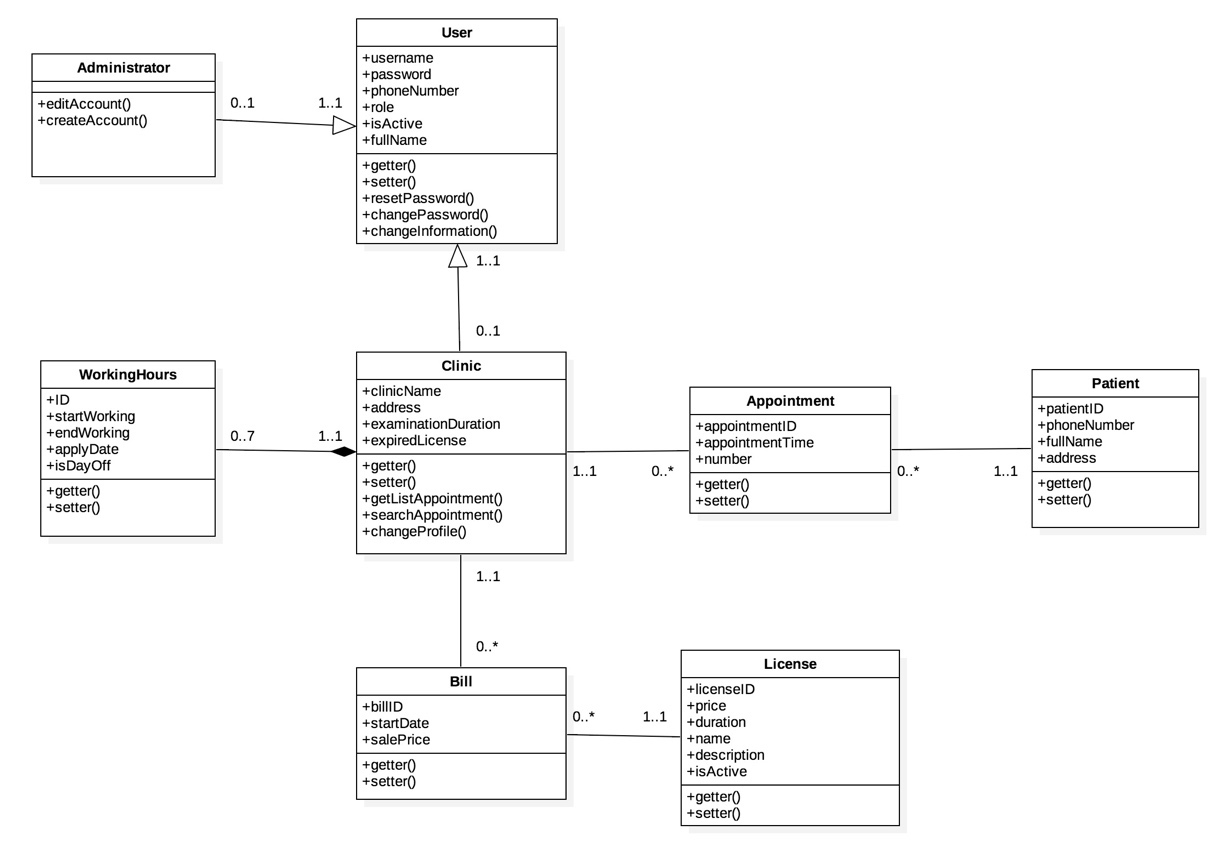
## 3. Component Diagram



|  |  |
| --- | --- |
| **Component dictionary: describe component** | |
| **Component Name** | **Description** |
| Scheduler | The component supports our system to schedule an appointment. |
| Data Access | Supporting component to retrieve data stored in database. |
| Services of the web server | Providing all of the services for mobile application and web application. |
| Services of web | Providing all of the services for web server system. |
| Services of mobile | Providing all of the services for web server system. |
| Payment component | Support our system for working with Payment System. |
| Payment System | Providing payment method for the clinic. |
| Third-party Hotline System | Supporting our system of working with SMS and Call. |
| Third-party Hotline Component | Handle communicate hotline between Web Server and third-party Hotline System. |
| Third-party Speech to Text System | Providing service to translate voice to text. |
| Third-party Speech to Text Component | Handle communicate speech to text between Web Server and Speech to Text System. |
| Notifications System | Providing service to notify. |
| Notifications Component | Handle communicate notify between Web Server and Notifications System. |
| Controller of Mobile System | Handle event and binding data from the model. |
| View of Mobile System | Content all views of the Mobile system. |
| Model of Mobile System | Content all data of the Mobile system. |
| Controller of Web System | Handle event and binding data from the model. |
| View of Web System | Content all views of the Web system. |
| Model of Web System | Content all data of Web system. |

## 4. Detailed Description

### 4.1 Class Diagram



|  |  |
| --- | --- |
| **Class dictionary: Describe Class** | |
| **Class Name** | **Description** |
| User | Contain the user’s account information:   * username: Name of the user. * password: password for each user. * phoneNumber: The phone number of each user. * role: the role of the user. It maybe is administrator or clinic. * isActive: status of the user. * fullName: full name of the user. |
| Administrator | Contain the information the relationship between user whom role is Administrator.  The administrator can create and edit information for the user. |
| Clinic | Contain the information the relationship between user whom role is Clinic and clinic information.  The clinic can view and edit information. |
| WorkingHours | Contain working hours’ information:   * startWorking: opening hours in one day. * endWorking: closing hours in one day. * applyDate: a day in the week. * isDayOff: status opening/closing. |
| License | Contain license information:   * price: A price of each license. * duration: duration of each license. * name: name of each license. * description: description of each license. |
| Appointment | Contain appointment information:   * appointmentTime: the time when appointment be created * number: order number of the patient |
| Patient | Contain patient information:   * phoneNumber: telephone number of the patient. * fullname: A real name of the patient. * address: address of the patient. |
| Bill | Contain payment information:   * startDate: day of beginning. * salePrice: price to pay |

### 4.2 Class Diagram Explanation

#### 4.2.1 User

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| username | String | Private | Unique username |
| password | String | Private | Hash password of the user |
| phoneNumber | String | Private | Phone number of the user |
| roleID | byte | Private | User’s role |
| isActive | boolean | Private | Determine the user is active or not |
| fullName | String | Private | Full name of the user |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | void | Public | Set value of attribute |
| resetPassword | boolean | Public | Reset user’s password |
| changePassword | boolean | Public | Change user’s password |

#### 4.2.2 Clinic

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| clinicName | String | Private | Name of clinic |
| address | String | Private | Clinic’s address |
| examinationDuration | Time | Private | Time for one examination |
| expiredLicense | Datetime | Private | Expiration day of the license |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | void | Public | Set value of attribute |
| getListAppointment() | List<Appointment> | Public | Return appointments of current day |
| searchAppointment() | List<Appointment> | Public | Search for appoinments |
| changeProfile() | boolean | Public | Change profile infomation |

#### 4.2.3 License

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| licenseID | long | Private | Unique identifier of license. |
| price | double | Private | Price of license |
| duration | Time | Private | Duration of price |
| name | String | Private | Name of license |
| description | String | Private | Description of license |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### 4.2.4 Administrator

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| createAccount() | boolean | Public | Create account |
| editAccount() | boolean | Public | Change account information |

#### 4.2.5 WorkingHours

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ID | long | Private | Unique identifier of working hours. |
| startWorking | Time | Private | Start working time |
| endWorking | Time | Private | End working time |
| applyDate | int | Private | The apply day in the week |
| isDayOff | boolean | Private | Determine that day is day-off or not |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | void | Public | Set value of attribute |

#### 4.2.6 Appointment

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| appointmentID | long | Private | Appointment’s identifier |
| appointmentTime | Datetime | Private | Appointment’s time |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### 4.2.7 Patient

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| patientID | long | Private | Patient’s identifier |
| phoneNumber | long | Private | Phone number of patient |
| fullName | String | Private | Patient’s full name |
| address | String | Private | The address of the patient |

**Method**

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | void | Public | Set value of attribute |

#### 4.2.8 Bill

**Attribute**

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| billID | int | Private | Bill’s identifier |
| startDate | Datetime | Private | Bill’s start date |
| salePrice | double | Private | Bill’s savle price |

**Method**

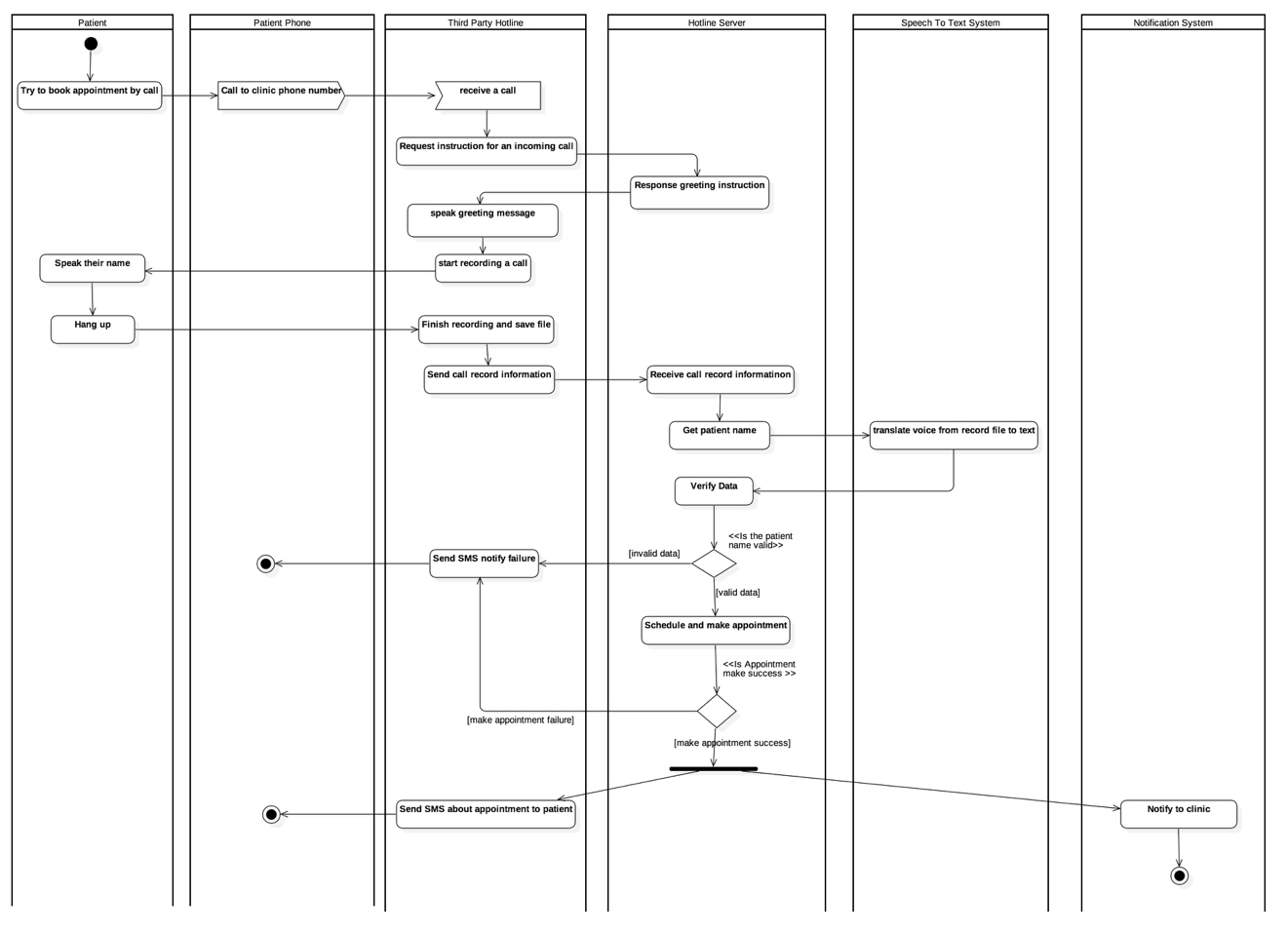
|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | void | Public | Set value of attribute |

### 4.3. Interaction Diagram

#### 4.3.1. Activity Diagram

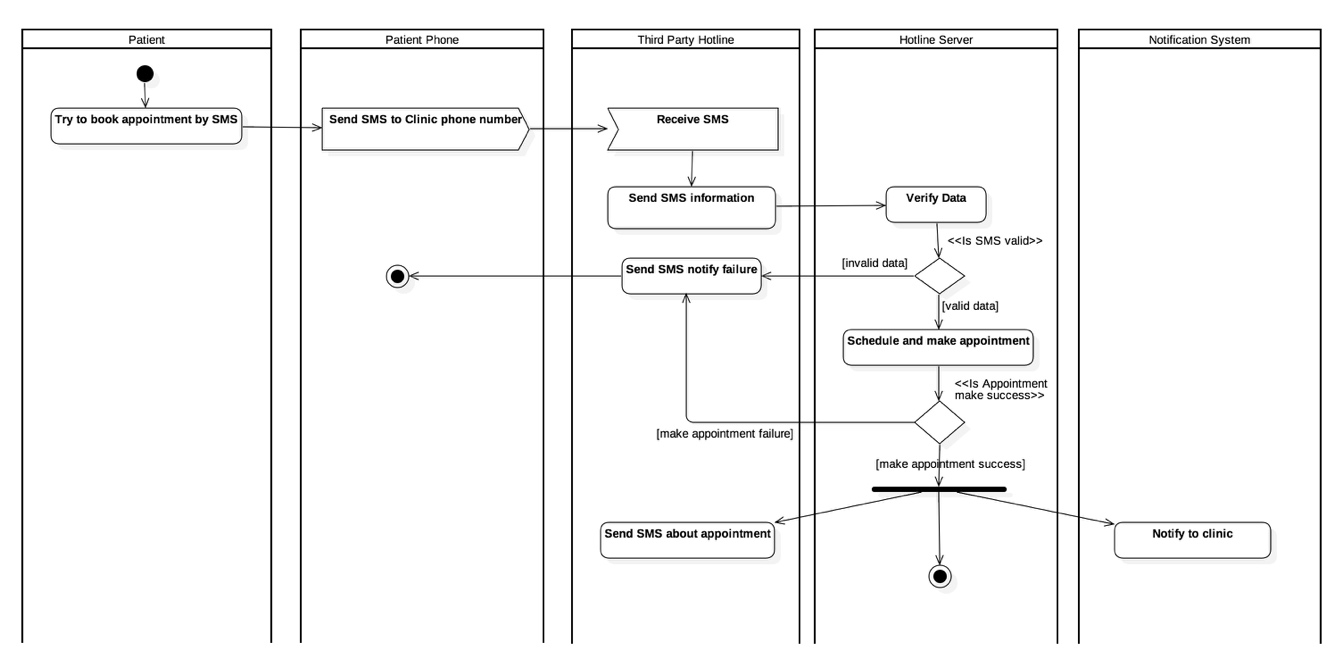
#### 4.3.1.1. Patient call to book appointment

**Summary:** This diagram shows how to patient to book an appointment by cellphone.



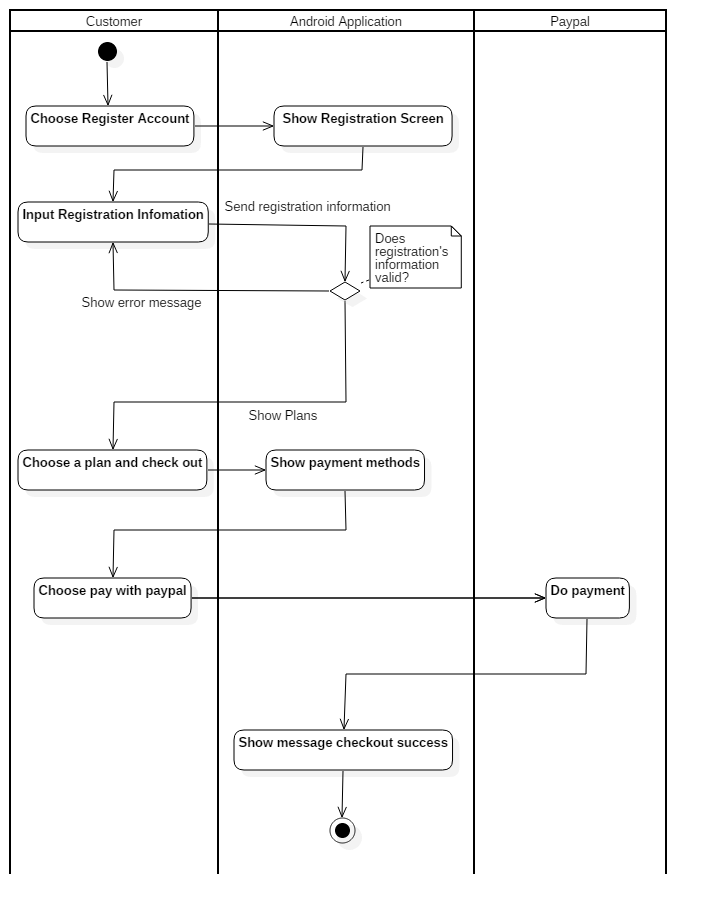
#### 4.3.1.2. Patient send SMS to book appointment

**Summary:** This diagram shows how to patient to book an appointment by SMS.



#### 4.3.1.3. Register Account

**Summary:** This diagram shows how to guest register clinic account in the system.



## 5. Interface

**5.1 Component interface**

**5.1.1 Mobile Web Service Interface**

- Standard output format:

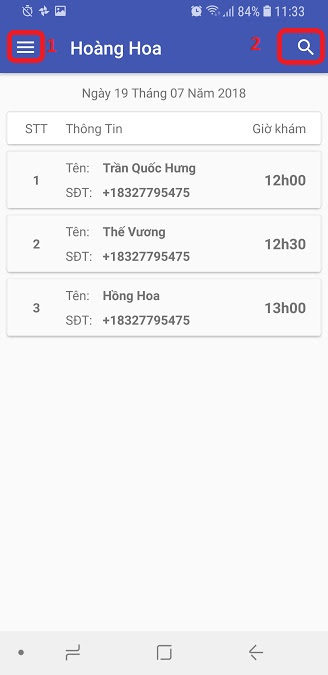
|  |
| --- |
| {  "status": boolean,  "value": T,  "error": string  } |

|  |  |
| --- | --- |
| **Name** | **Description** |
| status | value can be true or false to determine the request success processed or not |
| value | Json value corresponding to the request |
| error | contain error message content in case the request is not successes |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signature** | **Description** | **Input** | **Output** | **Output Format** | **Exception** |
| ResponseObject login(R r) | Login a user | Request object r | Json | [1] | N/A |
| ResponseObject getAppointmentList(R r) | Get appointment list of current day | Request object r | Json | [1] | N/A |
| ResponseObject cancelWorking(R r) | Cancel all appointment remaining in the current day | Request object r | Json | [1] | N/A |
| ResponseObject shiftWorking(R r) | Adjust all appointment’s time in current day | Request object r | Json | [1] | N/A |
| ResponseObject changePassword(R r) | Change password | Request object r | Json | [1] | N/A |
| ResponseObject changeInfo(R r) | Change user’s information (address, full name…) | Request object r | Json | [1] | N/A |
| ResponseObject register(R r) | Register a new user | Request object r | Json | [1] | N/A |
| ResponseObject checkDuplicate(R r) | Check whether a username already exists in the system | Request object r | Json | [1] | N/A |
| ResponseObject getToken(R r) | Get payment token to perform payment process | Request object r | Json | [1] | N/A |
| ResponseObject checkout(R r) | Checkout a license item | Request object r | Json | [1] | N/A |
| ResponseObject getAllLs(R r) | Get all available license items | Request object r | Json | [1] | N/A |
| ResponseObject updateWorkingHour(R r) | Change the working hour of a specific day | Request object r | Json | [1] | N/A |
| ResponseObject updateWorkingHours(R r) | Change many working hours at the same time | Request object r | Json | [1] | N/A |
| ResponseObject updateProfile(R r) | Change clinic’s information including a profile photo and greeting message | Request object r | Json | [1] | N/A |
| ResponseObject getWorkingHours(R r) | Get working hours information of current clinic | Request object r | Json | [1] | N/A |
| ResponseObject sendMail(R r) | Resend confirm email to user | Request object r | Json | [1] | N/A |

**5.2. User Interface Design**

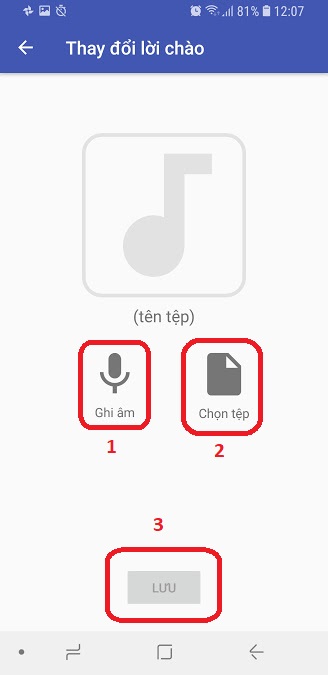
**5.2.1. Mobile Interface Design**

** 5.2.1.1. View Appointment List**

**Buttons**

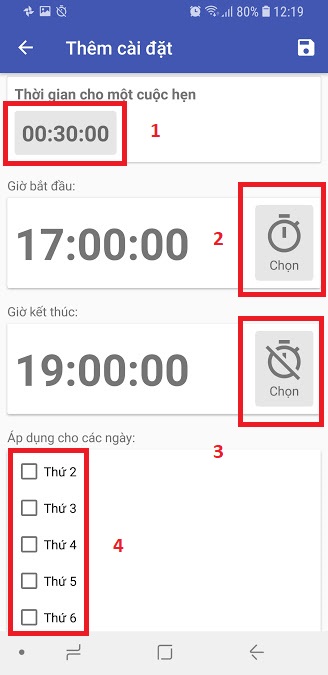
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Show menu panel | Show left menu panel | N/A | Left menu panel is shown |
| 2 | Search appointment | Search in the current appointment list | N/A | Show search field |

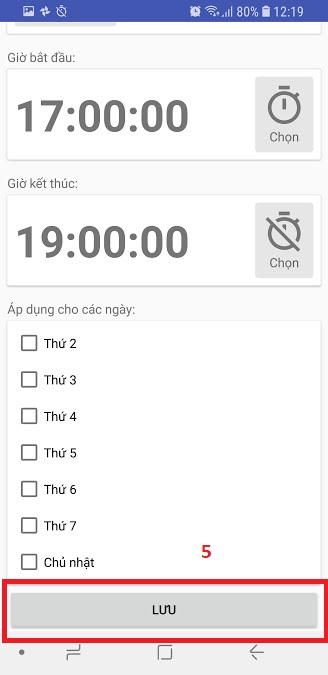
**5.2.1.2. Change greeting message**

****

**Buttons**

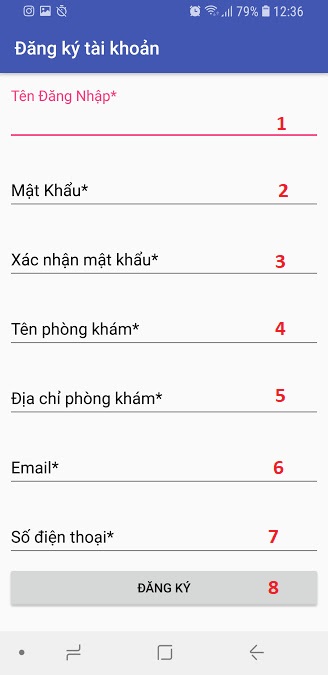
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Record greeting message | Allow the user to record the greeting message | Record file size’s much less than 3Mb | Launch the voice record screen |
| 2 | Pick greeting message file | Allow user pick an audio file as a greeting message | The picked file much be in MP3 format and the size is less than 3Mb | Launch file picker screen |
| 3 | Save greeting message | Save the information to the server | A file much be chosen ( record file or file from file picker) | Perform the save process |

**5.2.1.3. Change working hours**

****

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Choose working duration | Choose the duration for each appointment, the default value is 00:30:00 | The working duration much be greater than 0 minutes | Launch time picker |
| 2 | Choose to start working hour | Choose the time when clinic begin to work, the default value is 17:00:00 | The start of working time much before 24:00:00 | Launch time picker |
| 3 | Choose end working hour | Choose the time when the clinic is off work, the default value is 19:00:00 | The end of working time much be after the start of working time and greater than 00:00:00 | Launch time picker |
| 4 | Choose apply dates | The dates that this setting will apply on | Much specify as less 1 apply date | Status of the selected checkbox is switched |
| 5 | Save | Save the setting | N/A | Perform the saving process |

**5.2.1.4. Register**

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Username | Fill username | No | Yes | EditText | String | 6 – 30 characters |
| 2 | Password | Fill password | No | Yes | EditText | String | 6 – 12  characters |
| 3 | ComfirmPassword | Fill password again | No | Yes | EditText | String | 6 – 12  characters |
| 4 | ClinicName | Fill clinic’s name | No | Yes | EditText | String |  |
| 5 | ClinisAddress | Fill clinic’s address | No | Yes | EditText | String | 6 – 80 characters |
| 6 | Email | Fill clinic’s email | No | Yes | EditText | String | 6 – 80 characters |
| 7 | PhoneNumber | Fill clinic’s hotline number | No | Yes | EditText | String | 10 – 13 characters |

**Buttons**

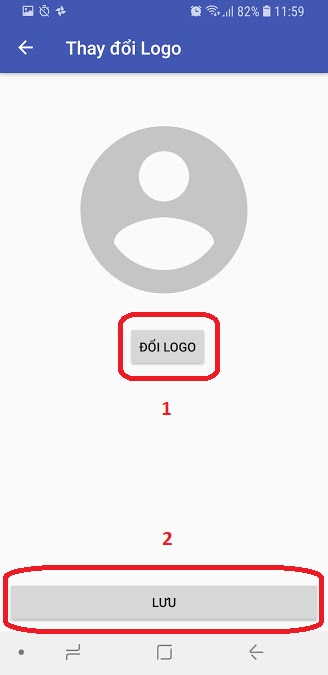
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Register | Validate all field and perform register | Validate all field | Send register request to server |

**5.2.1.5. Buy License**

****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Select a plan | User selects a license plan to buy. | N/A | Transfer to payment screen |

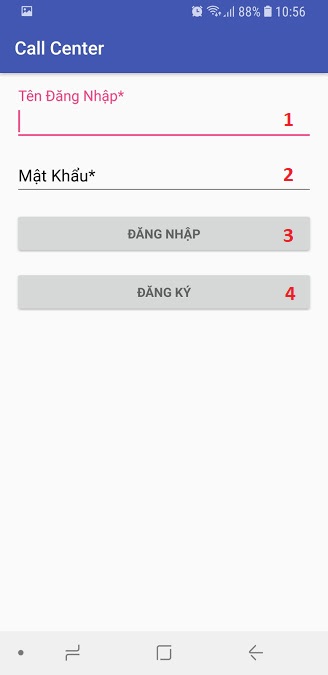
**5.2.1.6. Change profile Logo**

****

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Choose image | Choose an image file that will be the user’s profile logo | Image file’s size is less than 2Mb | Launch file picker |
| 2 | Save | Save the chosen image | An image much is chosen to save. | Validate and save the image |

**5.2.1.7. Login**

****

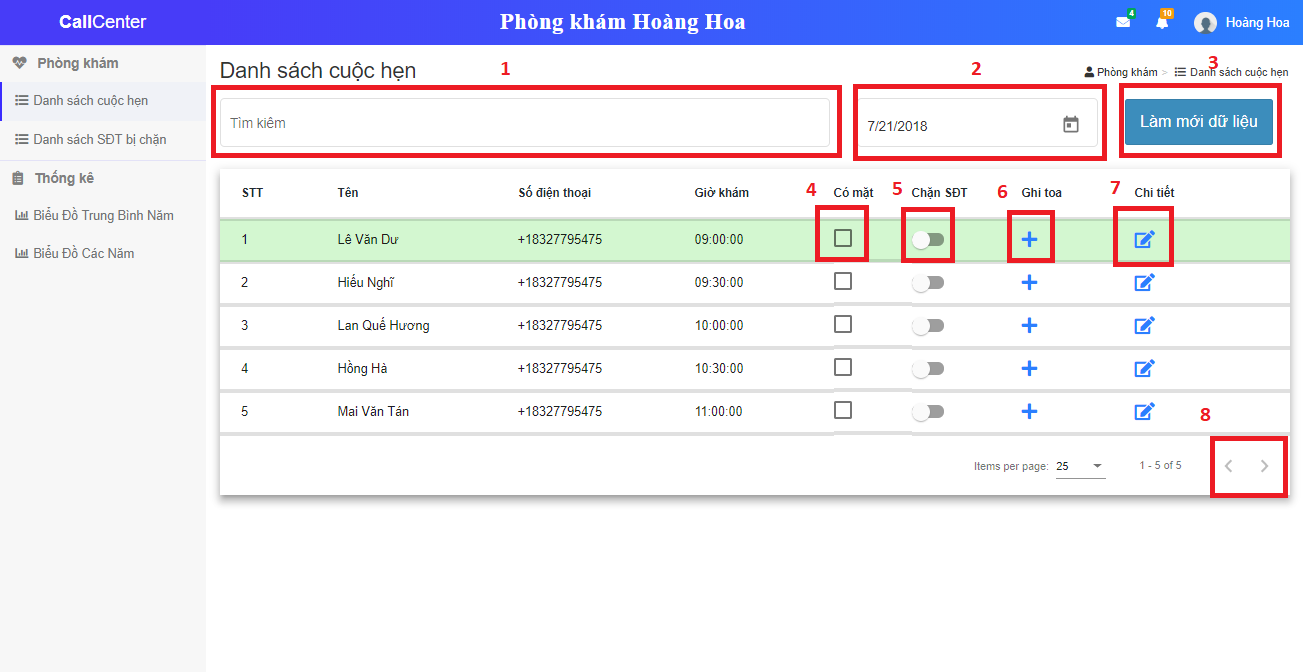
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Username | Fill user name | No | Yes | EditText | String | 6 – 30 characters |
| 2 | Password | Fill password | No | Yes | EditText | String | 6 – 12  characters |

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Log In | Log In user to the system | All fields are meet the required length range | Perform login process |
| 4 | Sign up | Transfer to sign up screen | N/A | Transfer to sign up screen |

**5.2.2. Clinic Web Interface Design**

**5.2.2.1 View Appointment List**

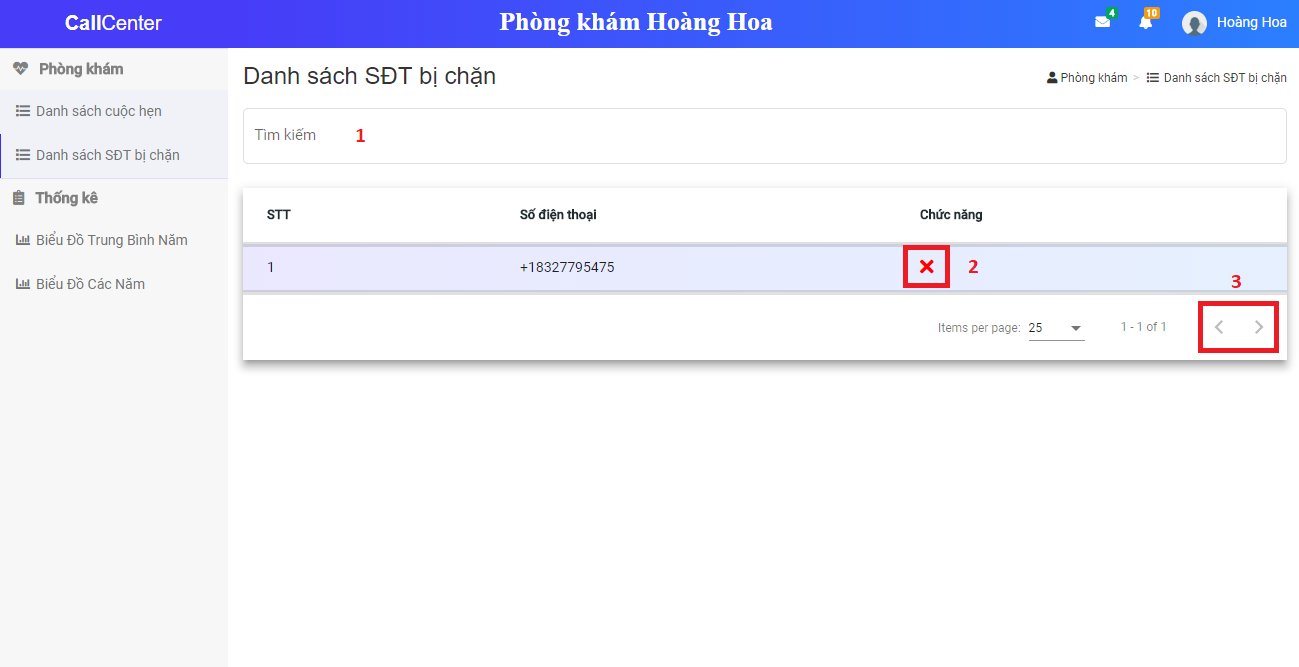
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | SearchValue | Fill in patient name to filter patient list | No | No | Text | String | 0 – 30 characters |

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | ChooseDate | Choose the day to display appointments | The picked day is from 01-01-2018 to current day | The list is refreshed with appointments in a picked day |
| 3 | Refresh | Refresh the current appointment list to see new data | N/A | The appointment list is refreshed |
| 4 | ChangeAppointmentStatus | Check/uncheck the appointment status to determine the patient has attended or not | N/A | The appointment status is switched |
| 5 | BlockPhoneNumber | Change the phone number blocking stage | N/A | The phone number blocking stage is switched |
| 6 | AddPrescription | Add a prescription to the patient | N/A | The Add prescription modal is shown |
| 7 | ShowPatientDetail | Display patient detail including basic information and medical history | N/A | The Patient detail modal is shown |
| 8 | SelectPage | Select the next/previous page of the list | N/A | Next/Previous page is shown |

**5.2.2.2 View Blocked Phone Number List**

****

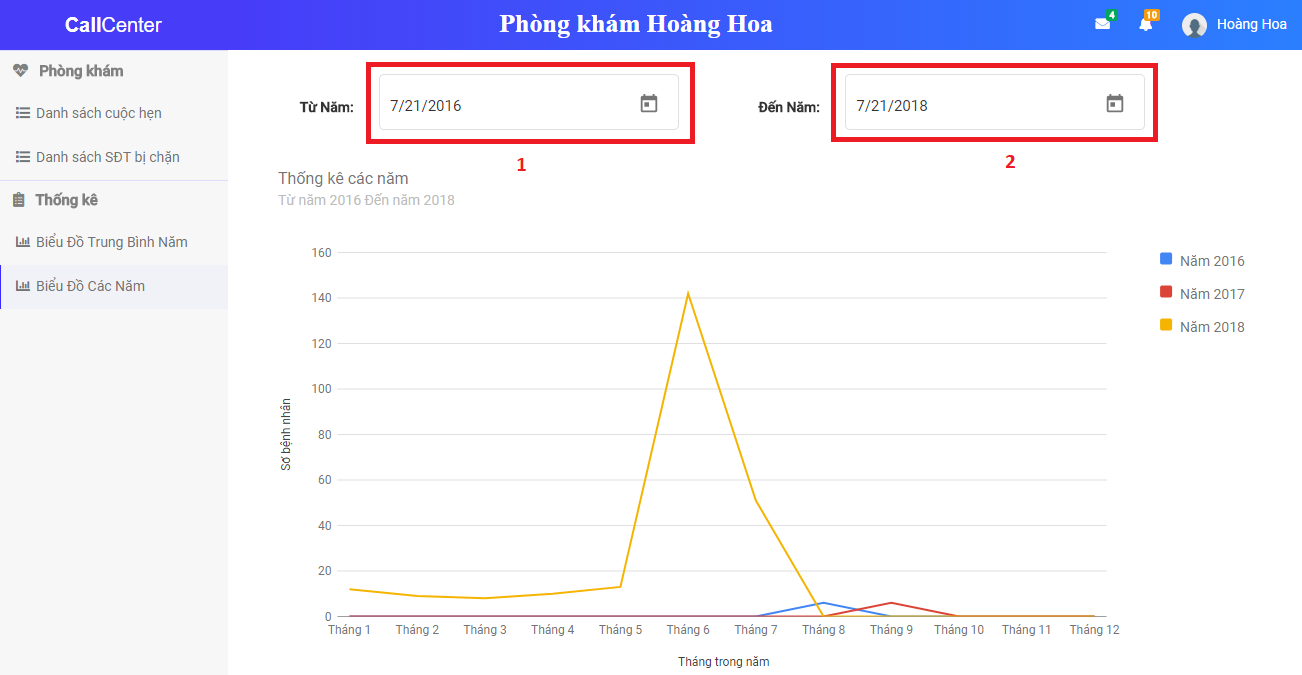
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read-only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | SearchValue | Fill in a phone number to filter phone number list | No | No | Text | String | 0 – 30 characters |

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | CancelBlocking | Unblock phone number | N/A | The phone number is unblocked and the list is refresh |
| 3 | SelectPage | Select the next/previous page of the list | N/A | Next/Previous page is shown |

**5.2.2.3 View Blocked Phone Number List**

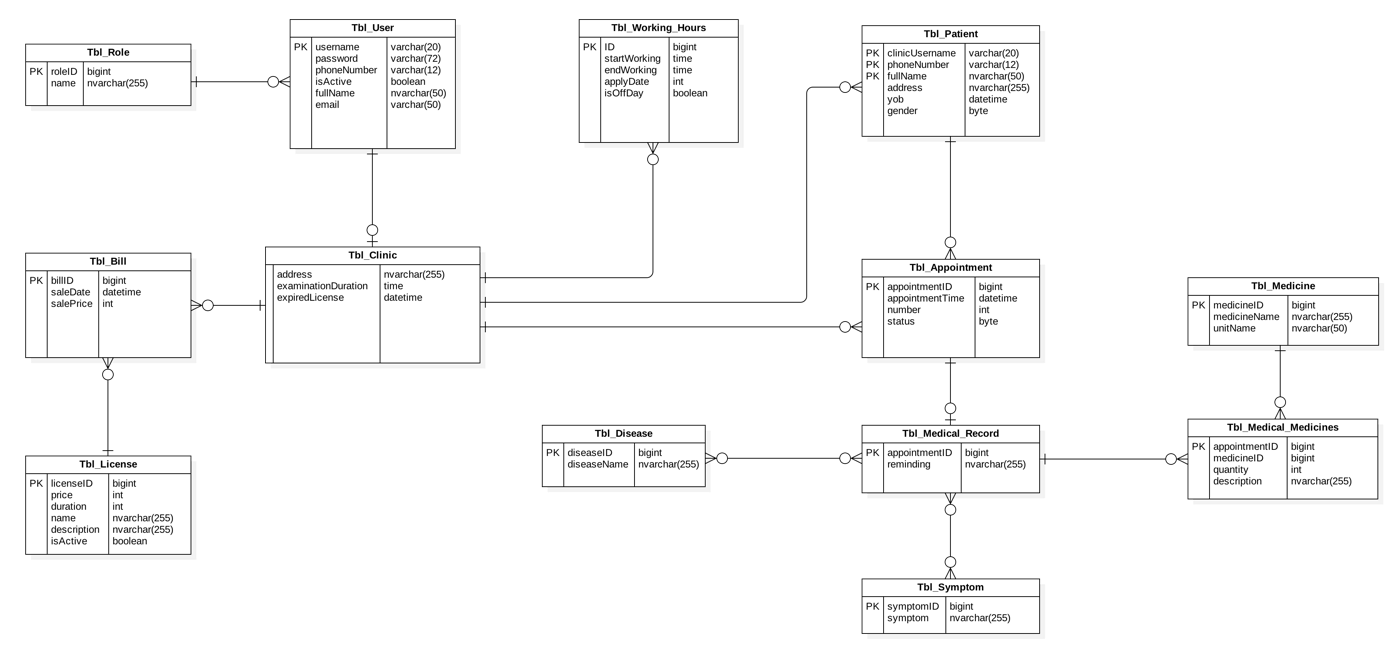
****

**Buttons**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | ChooseStartDate | Select the start day of the report | Picked value much greater than 01-01-2000 and smaller than end day | The graph is updated |
| 2 | ChromeEndDate | Select the end day of the report | Picked value much smaller than current day | The graph is updated |

## 6. Database Design

**6.1 Entity relationship diagram (ERD)**



**6.2 Data Dictionary**

|  |  |
| --- | --- |
|  | Entity Data Dictionary |
| **Entity name** | **Description** |
| Tbl\_Working\_Hours | Contain the working hour's information:   * startWorking: opening hours in one day. * endWorking: closing hours in one day. * applyDate: the dates of the week. * isDayOff: status opening/closing. |
| Tbl\_User | Contain the user information:   * username: Name of the user. User maybe is administrator or clinic. * password: each user would be having a password. * phoneNumber: A telephone number of each user. * role: the role of the user. * isActive: status of the user. * fullName: real name of the user. |
| Tbl\_Clinic | Contain the clinic information:   * address: address of clinic. * clinicName: name of the clinic. * examinationDuration: the examination time of one slot * expiredLicense: Expiry date of the license |
| Tbl\_License | Contain the license information:   * price: A price of each license. * duration: duration of each license. * name: name of each license. * description: description of each license. |
| Tbl\_Appointment | Contain the appointment information:   * appointmentTime: the time when the appointment is created. |
| Tbl\_Patient | Contain the patient information's:   * phoneNumber: telephone number of the patient. * fullname: A real name of the patient. * address: address of the patient. |
| Tbl\_Bill | Contain the bill information:   * startDate: day of beginning. * salePrice: price to pay. |
| Tbl\_Role | Contain the role information:   * Name: kind of role in the system. |

## 7. Algorithms

### 7.1. Schedule appointment

#### 7.1.1. Definition

#### 7.1.2. Define problem

#### 7.1.3. Solution

1. System Implementation & Test

## 1. Introduction

### 1.1 Overview

This chapter describes the testing and implementation Call Center system. It includes test plans, test cases, test result and risks estimations and some modification to the previous design phase and system testing to minimize the programming and system error.

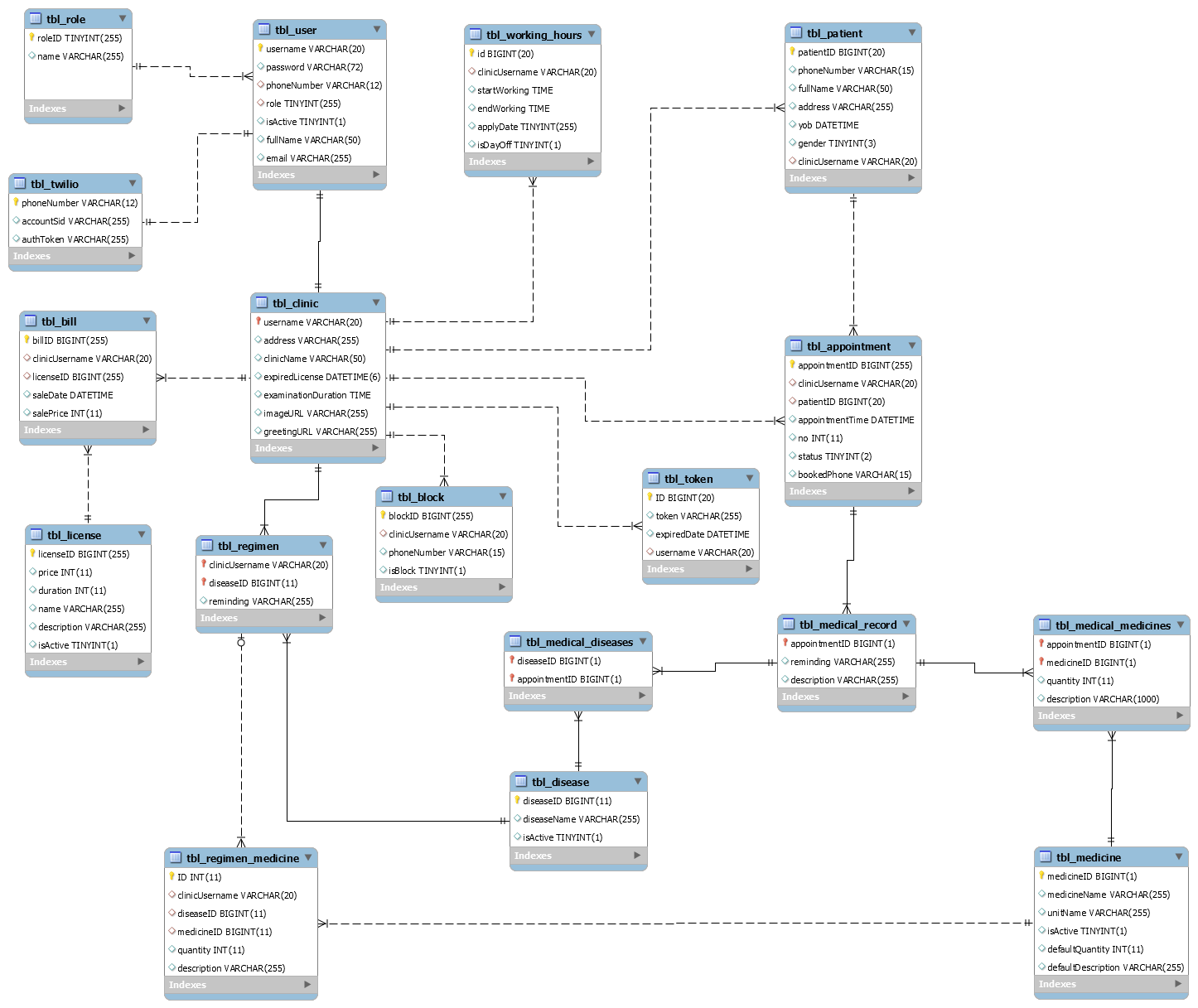
### 1.2 Test Approach

* Goal: Test all features in the whole Call Center system based on the core flow.
* Method: black-box testing

Testing based on the functionality of the software, customer requirements, and implementations. So testers input data and only see results on the screen.

## 2. Database Relationship Diagram

### 2.1 Physical Diagram



### Data Dictionary

|  |  |
| --- | --- |
|  | |
| **Table Name** | **Description** |
| tbl\_working\_hours | The table which contains the working hours information |
| tbl\_user | The table which contains the user information. |
| tbl\_clinic | The table which contains the clinic information. |
| tbl\_role | The table which contains the role information. |
| tbl\_appointment | The table which contains the appointment information. |
| tbl\_patient | The table which contains the patient information. |
| tbl\_bill | The table which contains the bill information. |
| tbl\_license | The table which contains the license information. |
| tbl\_block | The table which contains the block information. |
| tbl\_disease | The table which contains the disease information. |
| tbl\_medical\_disease | The table which contains the relationship between tbl\_medical\_record and tbl\_disease information. |
| tbl\_medical\_medicines | The table which contains the relationship between tbl\_medical\_record and tbl\_medicine information. |
| tbl\_medical\_record | The table which contains the medical record. |
| tbl\_medicine | The table which contains the medicine information. |
| tbl\_regimen | The table which contains the regimen information. |
| tbl\_regimen\_medicine | The table which contains the relationship between tbl\_regimen and tbl\_medicine information. |
| tbl\_token | The table which contains the token information. |
| tbl\_twilio | The table which contains the twilio information. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity name** | **Attributes** | **Description** | **Domain** | **Null** |
| tbl\_working\_hours | Id{PK} | Unique identifier of working hours, auto increment. | BIGINT(20) | No |
| clinicUsername  {FK} | Name of each clinic. Foreign key, clinic identifier. | VARCHAR(30) | No |
| StartWorking | The time opening of a clinic in one day of the week. | TIME | No |
| EndWorking | The time closing of a clinic in one day of the week. | TIME | No |
| applyDate | The day of the week. | TINYINT(7) | No |
| isDayOff | Status of that day of the week, it allows knowing clinic opening or closing in that day. | TINYINT(1) | No |
| tbl\_user | Username{PK} | Each user will have one username, it uses the login to the system. User maybe is administrator or clinic. | VARCHAR(30) | No |
| password | Each user would be having a password. | VARCHAR(30) | No |
| phoneNumber | A telephone number of each user. | VARCHAR(12) | No |
| role{FK} | Role of the user. Foreign key, role identifier. | TINYINT(255) | No |
| isActive | Status of the user. | TINYINT(1) | No |
| fullName | The real name of the user. | VARCHAR(255) | No |
| Email | Email of the user. | VARCHAR(255) | No |
| tbl\_clinic | Username{PK}  {FK} | Unique identifier of the member. Foreign key, user identifier. | VARCHAR(30) | No |
| address | Address of the clinic. | VARCHAR(255) | No |
| clinicName | Name of the clinic. | VARCHAR(255) | No |
| examinationDuration | The duration of one slot | TIME(6) | No |
| expiredLicense | The expiry date of the license | DATETIME(6) | No |
| tbl\_role | roleID{PK} | Unique identifier of role, auto increment. | TINYINT(255) | No |
| name | Role of name | VARCHAR(255) | No |
| tbl\_appointment | AppointmentID{PK} | Unique identifier of appointment, auto increment. | BIGINT(255) | No |
| clinicUsername{FK} | Name of each clinic. Foreign key, clinic identifier. | VARCHAR(30) | No |
| patientID{FK} | Id of the patient. Foreign key, patient identifier. | BIGINT(20) | No |
| appointmentTime | The time when an appointment is created. | DATETIME | No |
| no | A number of order. | INT(11) | No |
| tbl\_patient | patientID{PK} | Unique identifier of patient, auto increment. | BIGINT(20) | No |
| phoneNumber | Phone number of the patient. | VARCHAR(15) | No |
| fullName | A real name of the patient. | VARCHAR(255) | No |
| address | Address of patient. | VARCHAR(255) | Yes |
| tbl\_bill | billID{PK} | Unique identifier of bull, auto increment. | BIGINT(255) | No |
| clinicName{FK} | Name of each clinic. Foreign key, clinic identifier. | VARCHAR(30) | No |
| licenseID{FK} | Id of license. Foreign key, license identifier. | BIGINT(255) | No |
| startDate | Day of beginning | DATETIME | No |
| salePrice | Price to pay. | INT(11) | No |
| tbl\_license | licenseID{PK} | Unique identifier of license, auto increment. | BIGINT(255) | No |
| price | A price of the license. | INT(11) | No |
| duration | Duration of each license. | INT(11) | No |
| name | Name of license. | VARCHAR(255) | No |
| description | Description of license. | VARCHAR(255) | Yes |
| isActive | Status of license. | TINYINT(1) | No |
| tbl\_block | blockID{PK} | Unique identifier of block, auto increment. | BIGINT(255) | No |
| clinicUsername{FK} | Username of clinic which is block phone number. | VARCHAR(20) | No |
| phoneNumber | Phone number is blocked. | VARCHAR(15) | No |
| isBlock | Status of | TINYINT(1) | No |
| tbl\_disease | diseaseID{PK} | Unique identifier of disease, auto increment. | BIGINT(11) | No |
| diseaseName | Name of disease | VARCHAR(255) | No |
| tbl\_medical\_disease | diseaseID{PK} {FK} | Unique identifier of diseaseID. | BIGINT(1) | No |
| appointmentID{PK} {FK} | Unique identifier of appointmentID. | BIGINT(1) | No |
| tbl\_medical\_medicines | appointmentID{PK} {FK} | Unique identifier of appointmentID. | BIGINT(1) | No |
| medicineID{PK} {FK} | Unique identifier of medicineID. | BIGINT(1) | No |
| quantity | Quantity of medicine in a record. | INT(11) | No |
| description | Description of each medicine. | VARCHAR(1000) | Yes |
| Tbl\_medical\_record | appointmentID{PK} {FK} | Unique identifier of appointment. | BIGINT(1) | No |
| reminding | Remind of medical record | VARCHAR(255) | Yes |
| description | Description of medical record | VARCHAR(255) | Yes |
| tbl\_medicine | medicineID {PK} | Unique identifier of medicine, auto increment. | BIGINT(1) | No |
| medicineName | Name of medicine. | VARCHAR(255) | No |
| unitName | Unit of each medicine. | VARCHAR(255) | No |
| isActive | Exist status of medicine. | TINYINT(1) | No |
| defaultQuantity | Quantity default of each medicine. | INT(11) | Yes |
| defaultDescription | Description default of each medicine. | VARCHAR(255) | Yes |
| tbl\_regimen | clinicUsername{PK} {FK} | Unique identifier of username of clinic. | VARCHAR(20) | No |
| diseaseID{PK}{FK} | Unique identifier of disease. | BIGINT(11) | No |
| reminding | Remind of regimen. | VARCHAR(255) | Yes |
| tbl\_regimen\_medicine | ID{PK} | Unique identifier, auto increment. | INT(11) | No |
| clinicUsername{PK} {FK} | Username of clinic. | VARCHAR(20) | No |
| diseaseID{PK}{FK} | Unique identifier of disease. | BIGINT(11) | No |
| medicineID{PK}{FK} | Unique identifier of medicine. | BIGINT(11) | No |
| quantity | Quantity of medicine in regimen treatment. | INT(11) | No |
| description | Description of medicine in regimen treatment. | VARCHAR(255) | Yes |
| tbl\_token | ID{PK} | Unique identifier of token, auto increment. | BIGINT(20) | No |
| token | Name of token. | VARCHAR(255) | No |
| expiredDate | Expired of token | DATETIME | No |
| Username {FK} | Username use that token | VARCHAR(20) | No |
| tbl\_twilio | phoneNumber{PK} | Phone number is provided by twilio third-party. | VARCHAR(12) | No |
| accountSid | ID of twilio third-party account. | VARCHAR(255) | Yes |
| authToken | Token is provided by twilio third-party. | VARCHAR(255) | Yes |

## 3. Performance Measures

### 3.1 Book appointment performance

* When an appointment is booked, whether via SMS or phone call, the system should finish process the request in at maximum 5 seconds.

### 3.2 Book multiple appointments on the same number at the same time

* When many appointments being booked on the same number at the same time, the clinic should receive all appointment within at maximum 15 second since the patient has finished the call.

**3.3 Response time when interacting with the user interface**

* When user interacts, whether via mobile application or web application, the application should respond immediately, the maximum delay time is 5 second.

## 4. Test Plan

The purpose of this document is to describe the overall test plan for testing the Call Center System. Testing strategies include conditions to know when the test is complete. So, it can verify and ensure that the Call Center meets its design specification and other requirements from the user.

### 4.1 Features to be tested

* Mobile
  + Registration
  + Log In
  + View appointment list
  + Change working hour
  + Change appointment duration
  + Change greeting message
  + Buy License
  + Auto update appointment list
* Web (Role Clinic)
  + View appointment list
  + Refresh appointment list
  + Change patient attendant status
  + Change phone number blocking status
  + Unblock phone number
* System
  + Book appointment via calling
  + Book appointment via SMS

### 4.2 Features not to be tested

* Mobile
  + Appointment notification
  + View appointment list by date
  + Cancel all appointment in a current day
  + Adjust appointments time
  + Change multiple working hours
  + Change account information
  + Change password
  + Export Excel file
  + Search appointment
  + Change clinic’s logo
  + Log Out
* Web (Role Clinic)
  + Search patient
  + Add prescription
  + Change patient information
  + View medical histories of the patient
  + View medical record detail
  + Search blocked a phone number
  + View report (Average in years)
  + View report
  + Change password
* Web (Role Admin)
  + Change password
  + Create administrator
  + View administrator list
  + Search administrator list
  + Delete administrator
  + Update administrator information
  + View clinic list
  + Search clinic
  + Delete clinic
  + Reset clinic phone number
  + Create license
  + View License list
  + Search license
  + Remove license
  + Edit license
  + Create staff
  + View staff list
  + Delete staff
  + Edit staff
* Web (role staff)
  + Change password
  + View clinic list
  + Search clinic
  + Delete clinic
  + Reset clinic phone number
  + Edit clinic information
  + Create a new Twilio account
  + View Twilio account list
  + Search Twilio account
  + View month chart
  + View single year chart
  + View multiple year chart

## 5. System Testing Test Case

### 5.1. Mobile Test Case

#### 5.1.1. Registration

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **RGT\_1** | Test when user input all valid information (perfect case) | N/A | 1. Open app  2. Press [Đăng ký] button  3. Fill all required field  4. Press [Đăng ký] button  5. Confirm email  6. Login with previous username password  7. Buy license  8. Change greeting message | Appointment list screen is shown. | - Change working hour  - Change greeting message  - Buy License  - Login | from 21/7/2018  to  4/8/2018 | 5 | 4/1 |

#### 5.1.2. Login

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **LG\_1** | Test when user input all valid information | The user has confirmed email, the license is still valid, greeting message and working hours have been set up | 1. Open app  2. Fill in username and password  3. Press [Đăng Nhập] | Appointment list screen is shown. | N/A | from 21/7/2018  to  4/8/2018 | 8 | 5/3 |
| **LG\_2** | Test when user input correct username and password but not confirm email yet. | N/A | 1. User open app  2. Fill in username and password  3. User press [Đăng Nhập] | Error message dialog is shown | N/A | from 21/7/2018  to  4/8/2018 | 7 | 6/1 |
| **LG\_3** | Test when user confirmed email but not buy a license yet | The user has confirmed the email | 1. User open app  2. Fill in username and password  3. User press [Đăng Nhập] | Transfer to buy license screen | N/A | from 21/7/2018  to  4/8/2018 | 5 | 4/2 |

#### 5.1.3. View appointment list

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **VAL\_1** | When there is no appointment to display, the list should be empty | No appointment has been booked | 1. Login | Appointment list screen is shown with an empty list | **LG\_1** | from 21/7/2018 to 4/8/2018 | 3 | 3/0 |
| **VAL\_2** | Test the visualization of the appointment | 2 appointments has been booked in that day, 1 is in the past and 1 in the future | 1. Login  2. Book 2 appointment with the previous condition | 2 The appointment should be shown in the list, the one in the past have the grey background the one in the future have a white background | N/A | from 21/7/2018 to  4/8/2018 | 4 | 3/1 |

#### 5.1.4. Change working hour

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **CWH\_1** | Test when the user has logged in and want to change the working hour | User has a valid account in the system | 1. Login  2. Open the navigation menu  3. Choose [Cài đặt giờ làm việc]  4. Change the start working of Sunday to 05:00 AM | A success message is shown | N/A | from  21/7/2018  to  4/8/2018 | 4 | 3/1 |

#### 5.1.5. Change appointment duration

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **CAD\_1** | Test change the appointment duration when user input valid time. | User have a valid account in the system | 1. Login  2. Open the navigation menu  3. Choose [Cài đặt giờ làm việc]  4. Press the button below [Thời gian cho một cuộc hẹn]  5. Choose 15 minutes  6. Press OK | A success message is shown | N/A | from  21/7/2018  to  4/8/2018 | 3 | 3/0 |
| **CAD\_2** | Test change the appointment duration when user input invalid time. | User have a valid account in the system | 1. Login  2. Open the navigation menu  3. Choose [Cài đặt giờ làm việc]  4. Press the button below [Thời gian cho một cuộc hẹn]  5. Choose **00 minutes**  6. Press OK | The error message is shown, the working duration is not updated | N/A | from  21/7/2018  to  4/8/2018 | 5 | 4/1 |

#### 5.1.5. Change greeting message

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **CGM\_1** | Change the greeting message when user input directly from the recording | User has a valid account in the system | 1. Login  2. Open the navigation menu  3. Choose [Thay đổi lời chào]  4. Press the [Ghi âm] button  5. Record the message  6. Press [Lưu] button | Success message is shown, and the greeting message is changed | N/A | from  21/7/2018  to  4/8/2018 | 3 | 2/1 |

#### 5.1.6. Buy License

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **BL\_1** | User have to buy a license when their account has no license or the license has expired | User has an account in the system and has confirmed email | 1. Login  2. In the license list screen, choose any license  3. Process payment | Success message is shown and transfer to appointment list screen | N/A | from  21/7/2018  to  4/8/2018 | 3 | 3/0 |

#### 5.1.7. Auto update appointment list

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **AU\_1** | The appointment list should refresh automatically when a new appointment is booked | N/A | 1. Login  2. Book new appointment | A notify message a shown, and the appointment list is refresh | N/A | from  21/7/2018  to  4/8/2018 | 4 | 4/0 |

### 5.2. Web Test Case (Role Clinic)

* + View appointment list
  + Refresh appointment list
  + Change patient attendant status
  + Change phone number blocking status
  + Unblock phone number

#### 5.2.1. View Appointment List

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **W\_VAL\_1** | The appointment list screen should be shown after the user logged in | N/A | 1. Book an appointment  2. Login | An appointment list is shown with 1 appointment | N/A | from  21/7/2018  to  4/8/2018 | 2 | 2/0 |

#### 5.2.2. Refresh appointment List

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **W\_RAL\_1** | When the [Làm mới dữ liệu] button is pressed, the list should refresh with the newest data. | Do all the step in **W\_VAL\_1** | 1. Book an appointment  2. Press [Làm mới dữ liệu] button | 2 appointment list is shown in the list | **W\_VAL\_1** | from  21/7/2018  to  4/8/2018 | 3 | 3/0 |

#### 5.2.3. Change patient attendant status

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **W\_CPAS\_1** | After a new appointment is created, the user can able to change the attendant status of the appointment | Do all the step in **W\_VAL\_1** | 1. Check on the [Có mặt] checkbox | The attendant status is switched and a success message is shown | **W\_VAL\_1** | from  21/7/2018  to  4/8/2018 | 5 | 4/1 |

#### 5.2.4. Change phone number blocking status

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **W\_CPNBS\_1** | After an appointment is made, the user can block a phone number. | Do all the step in **W\_VAL\_1** | 1. Click on the [Chặn SĐT] toggle | The blocking status is switched from **off** to **on** | **W\_VAL\_1** | from  21/7/2018  to  4/8/2018 | 3 | 3/0 |

#### 5.2.4. Unblock phone number

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **W\_UPN\_1** | The user can unblock a phone number in the block phone numbers list | Do all the step in **W\_CPNBS\_1** | 1. On the left side, menu click [Danh sách SĐT bị chặn]  2. Click on the [x] button | The list is empty out and a success message is shown | **W\_CPNBS\_1** | from  21/7/2018  to  4/8/2018 | 3 | 2/1 |

### 5.3. System Test Case

#### 5.3.1. Book appointment via calling.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **W\_VAL\_1** | The user can book appointment via calling | N/A | 1. Call to clinic phone number  2. speak name when greeting message is done | - The appointment will be created successfully  - Patient receive SMS  - Clinic receive notification if appointment created success | N/A | from  21/7/2018  to  4/8/2018 | 100 | 93/7 |

#### ­5.3.2. Book appointment via SMS.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | **Test Case Description** | **Precondition** | **Test case procedure** | **Expected output** | **Inter-test case dependence** | **From date to date** | **Number of tests** | **Success/Fail** |
| **W\_VAL\_1** | The user can book appointment via SMS | N/A | Send SMS to clinic phone number | - The appointment will be created successfully  - Patient receive SMS  - Clinic receive notification if appointment created success | N/A | from  21/7/2018  to  4/8/2018 | 70 | 66/4 |

## 6. System implement

#### 6.1 Web Server Application

We using NodeJS to implement Web server. We choose this because of the following advantage:

* Node.js works on a non-blocking I/O model that makes it clean and usable, ideal for the data-intensive real-time applications that have to perform in varied environments
* The event-driven architecture caters to both the client-side and the server-side that are written in JavaScript and thus the synchronization process is fast and orderly. The event loop through web socket protocol which works on TCP handles the multi-user function and prevents the overhead of HTTP for web development.

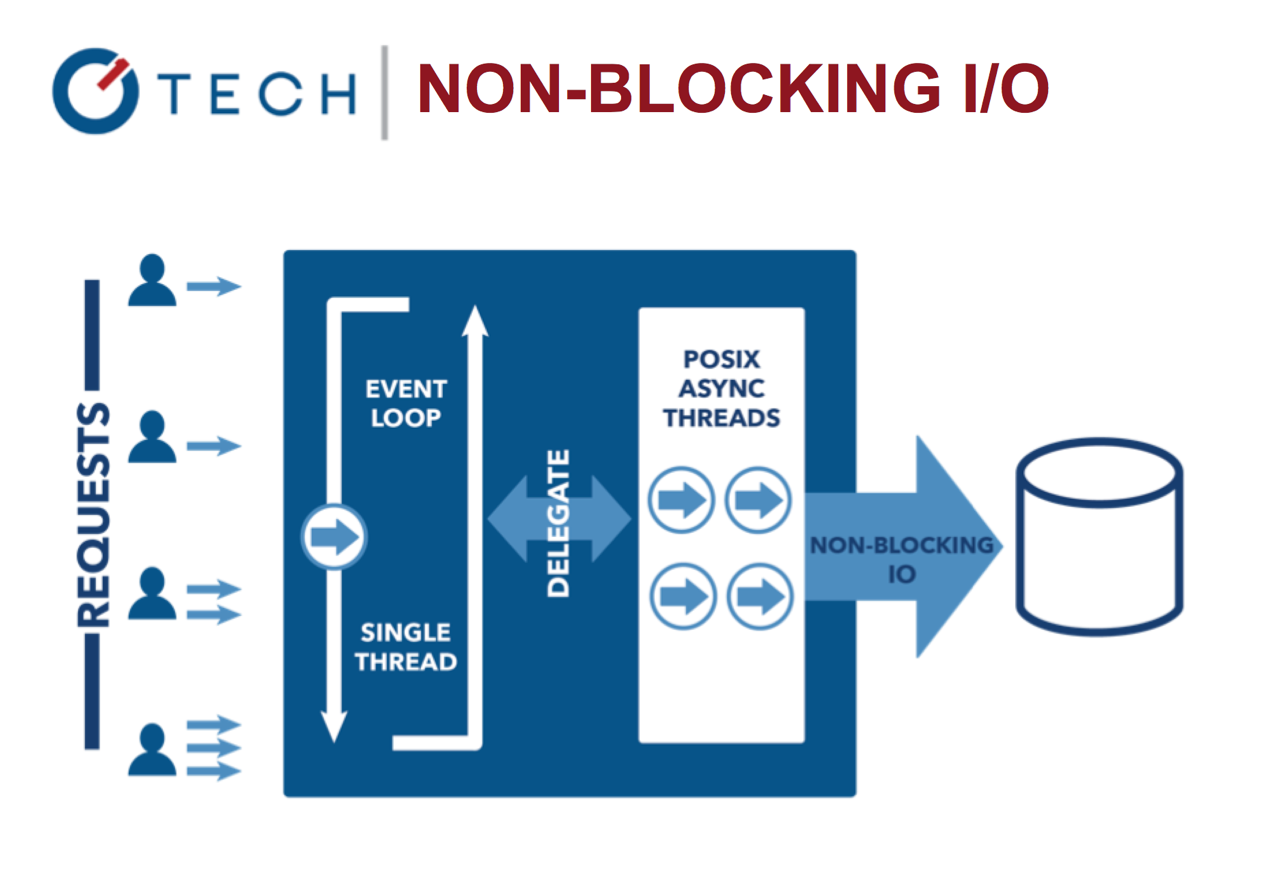


Figure 2 Non-Blocking I/O Model

Reference: “*https://think360studio.com/12-benefits-of-using-node-js-for-web-application/”*

#### 6.2 Web Client Application

In Web client application, we are using Angular under maybe like MVC architecture. We choose this Angular because of the following advantage:

* + Maintainability: Angular code can be built using TypeScript which provides a host of benefits, especially in the enterprise. TypeScript code can be debugged directly in the browser.
  + Using dependency injection: lets you keep your component classes lean and efficient.
  + Consistency: code is consistency, which is an important goal to strive for in any code base. The overall framework is based on components and services that you can think of as Lego blocks. All components and service start out the same way.
  + An Angular application is built from the following 8 elements: Module, Component, Template, Metadata, Data Binding, Directive, Service, Dependency Injection.
    - Module: Each Angular application is called a module and Angular itself has its own module for managing other modules.
    - Component: The component controls the display, control View, so you can imagine the Component as a controller in the MVC model.
    - Template: A template is an HTML code for the component based on which it is displayed on the screen.
    - Metadata: It is information that helps Angular handle classes.
    - Data Binding: It fills the data from the model/controller into the view. Data binding in Angular is two-dimensional, we can import data from view into model/controller.
    - The directive: It is a class and has a metadata declaration that is @Directive. Usually, the directive will be in an element - or an HTML tag as a normal attribute. The two types of directive are structural and attribute
      * The structural directive has the function of assigning data according to a certain rule.
      * The attribute directive has the ability to display data directly.
    - Service classes are capable of performing some commonly used functions. Some common services are: logging service, data service, message bus,...
    - Dependency injection: It is the more extensible / module / service class, Dependency injection is the ability to allow the creation of class objects with all the additional modules / modules / services.



Figure 3 Angular Component Example

Reference: [*https://angular.io/guide/architecture*](https://angular.io/guide/architecture)

#### 6.3 Mobile Application

In Mobile Application, we are using Android under MVC architecture. We choose this architecture because of the following advantage

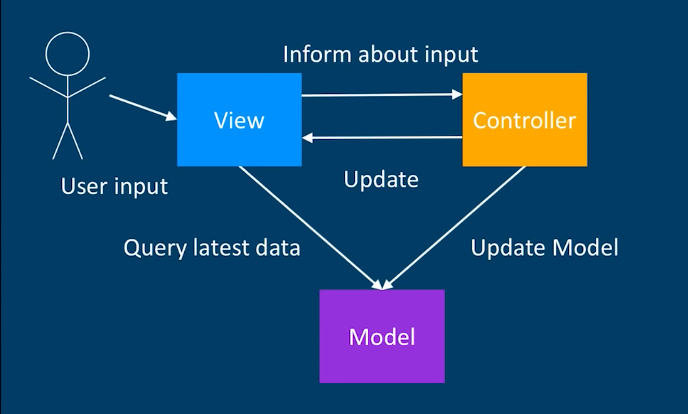


Figure 4 The Scrum Process

Reference: “*https://medium.com/@vicky7230/android-architecture-patterns-mv-c-p-vm-4594574eeaa1”*

1. Software User’s Manual
2. Appendix