# 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](#_Toc516820485)

[Table of Contents 2](#_Toc516820486)

[List of Tables 6](#_Toc516820487)

[List of Figures 6](#_Toc516820488)

[Definitions, Acronyms, and Abbreviations 6](#_Toc516820489)

[A. Introduction 7](#_Toc516820490)

[1. Project Information 7](#_Toc516820491)

[2. Introduction 7](#_Toc516820492)

[3. Current Situation 7](#_Toc516820493)

[4. Problem Definition 7](#_Toc516820494)

[5. Proposed Solution 8](#_Toc516820495)

[5.1 Feature functions 8](#_Toc516820496)

[5.2 Values and Challenges 8](#_Toc516820497)

[6. Functional Requirements 8](#_Toc516820498)

[7. Role and Responsibility 8](#_Toc516820499)

[B. Software Project Management Plan 9](#_Toc516820500)

[1. Problem Definition 9](#_Toc516820501)

[1.1 Name of this Capstone Project 9](#_Toc516820502)

[1.2 Problem Abstract 9](#_Toc516820503)

[1.3 Project Overview 9](#_Toc516820504)

[1.3.1 Current Situation 9](#_Toc516820505)

[1.3.2 The Proposed System 10](#_Toc516820506)

[The system will have four sub-systems: 10](#_Toc516820507)

[1.3.2.1 API application 10](#_Toc516820508)

[1.3.2.2 Mobile application 10](#_Toc516820509)

[1.3.2.3 Hotline server 10](#_Toc516820510)

[1.3.2.4 Web application 10](#_Toc516820511)

[1.3.3 Boundaries of the System 11](#_Toc516820512)

[Our system supports: 11](#_Toc516820513)

[o Providing appointment booking service 11](#_Toc516820514)

[o Patient can send SMS or call Clinic Hotline to make appointment. 11](#_Toc516820515)

[o Hotline with legacy call (call without internet) using Hotline server 11](#_Toc516820516)

[o Hotline with VoIP call (call via internet protocol) 11](#_Toc516820517)

[o When appointment has been booked successfully, Send SMS or send notification to patient 11](#_Toc516820518)

[Our system hasn’t supports: 11](#_Toc516820519)

[o Adjust appointment 11](#_Toc516820520)

[o Clinic phone auto pick up incoming call 11](#_Toc516820521)

[o Interactive with patient 11](#_Toc516820522)

[1.3.4 Future Plans 11](#_Toc516820523)

[1.3.5 Development Environment 11](#_Toc516820524)

[1.3.5.1 Hardware requirements 11](#_Toc516820525)

[1.3.5.2 Software requirements 12](#_Toc516820526)

[2. Project organization 12](#_Toc516820527)

[2.1 Software Process Model 12](#_Toc516820528)

[2.2 Roles and responsibilities 13](#_Toc516820529)

[2.3 Tools and Techniques 14](#_Toc516820530)

[3. Project Management Plan 14](#_Toc516820531)

[3.1 Product Backlog 14](#_Toc516820532)

[3.2 Sprint Backlog 15](#_Toc516820533)

[3.2.1 Sprint 1 (18.05.2018 – 25.05.2018): Project initiation 15](#_Toc516820534)

[3.2.1.1: Goal 15](#_Toc516820535)

[1.1 Project Information 15](#_Toc516820536)

[1.2 Introduction 15](#_Toc516820537)

[1.3 Current Situation 15](#_Toc516820538)

[1.4 Problem Definition 15](#_Toc516820539)

[1.5 Proposed Solution 15](#_Toc516820540)

[1.6 Role and Responsibility 15](#_Toc516820541)

[1.7 Functional Requirements 15](#_Toc516820542)

[2.1 Create Product Backlog 15](#_Toc516820543)

[3.1 Problem Definition 15](#_Toc516820544)

[3.2 Project Organization 15](#_Toc516820545)

[3.3 Project management plan 15](#_Toc516820546)

[3.4 Coding Convention 15](#_Toc516820547)

[3.2.1.2: Development 15](#_Toc516820548)

[3.3 All Meeting Minutes 15](#_Toc516820549)

[4. Coding Convention 15](#_Toc516820550)

[- 16](#_Toc516820551)

[- 16](#_Toc516820552)

[C. Software Requirement Specification 17](#_Toc516820553)

[1. User Requirement Specification 17](#_Toc516820554)

[1.1 Clinic Requirement 17](#_Toc516820555)

[1.2 Administrator Requirement 17](#_Toc516820556)

[1.3 System Requirement 17](#_Toc516820557)

[2. System Requirement Specification 17](#_Toc516820558)

[2.1 External Interface Requirement 17](#_Toc516820559)

[2.1.1 User Interface 17](#_Toc516820560)

[2.1.2 Hardware Interface 17](#_Toc516820561)

[2.1.3 Software Interface 17](#_Toc516820562)

[2.1.4 Communication Protocol 18](#_Toc516820563)

[2.2 System Overview Usecase 18](#_Toc516820564)

[2.3 List of Usecase 18](#_Toc516820565)

[2.3.1 Clinic Overview Usecase 18](#_Toc516820566)

[2.3.1.1 <Clinic> Get appointment list 19](#_Toc516820567)

[2.3.1.2 <Clinic> Search appointment Usecase 20](#_Toc516820568)

[2.3.1.3 <Clinic> Configure working hours Usecase 22](#_Toc516820569)

[2.3.1.4 <Clinic> Buy license Usecase 24](#_Toc516820570)

[2.3.2 Administrator Overview Usecase 25](#_Toc516820571)

[2.3.2.1 <Administrator> Create account 25](#_Toc516820572)

[2.3.2.2 <Administrator> Edit account 27](#_Toc516820573)

[2.3.3 Authenticated User Overview Usecase 29](#_Toc516820574)

[2.3.3.1 <Authenticated User> Logout account 29](#_Toc516820575)

[2.3.3.2 <Authenticated User> Change password 30](#_Toc516820576)

[2.3.3.3 <Authenticated User> Reset Password 32](#_Toc516820577)

[2.3.4 Guest Overview Usecase 34](#_Toc516820578)

[2.3.4.1 <Guest> Login account 34](#_Toc516820579)

[2.3.4.2 <Guest> Register 35](#_Toc516820580)

[2.3.4.2 <Guest, Clinic> Pay 37](#_Toc516820581)

[2.3.5 Third-Party Hotline Overview Usecase 39](#_Toc516820582)

[2.3.5.2 < Third-Party Hotline > Listen incoming Call, SMS 39](#_Toc516820583)

[2.3.6.3< Third-Party Hotline > Send SMS 40](#_Toc516820584)

[2.3.6 Hotline Overview Usecase 41](#_Toc516820585)

[2.3.6.1< Hotline Server > Analyze data from third-party hotline 42](#_Toc516820586)

[2.3.6.2< Hotline Server > Schedule Appointment 43](#_Toc516820587)

[2.3.6.3< Hotline Server > Request sending SMS 44](#_Toc516820588)

[2.3.7.4< Hotline Server > Request Notification to Clinic 46](#_Toc516820589)

[2.3.7 Firebase Overview Usecase 47](#_Toc516820590)

[2.3.7.1 < Firebase > Notify to Clinic 47](#_Toc516820591)

[3. Software System Attribute 48](#_Toc516820592)

[3.1 Usability 48](#_Toc516820593)

[3.2 Reliability 48](#_Toc516820594)

[3.3 Availability 48](#_Toc516820595)

[3.4 Security 48](#_Toc516820596)

[3.5 Maintainability 48](#_Toc516820597)

[3.6 Portability 49](#_Toc516820598)

[3.7 Performance 49](#_Toc516820599)

[4. Conceptual Diagram 49](#_Toc516820600)

[D. Software Design Description 50](#_Toc516820601)

[1. Design Overview 50](#_Toc516820602)

[2. System Architecture Design 50](#_Toc516820603)

[2.1 Web Server Architecture Design 50](#_Toc516820604)

[2.2 Client Architecture Design 51](#_Toc516820605)

[3. Component Diagram 52](#_Toc516820606)

[4. Detailed Description 53](#_Toc516820607)

[4.1 Class Diagram 53](#_Toc516820608)

[4.2 Class Diagram Explanation 54](#_Toc516820609)

[4.2.1 User 54](#_Toc516820610)

[4.2.2 Clinic 54](#_Toc516820611)

[4.2.3 License 55](#_Toc516820612)

[4.2.4 Administrator 55](#_Toc516820613)

[4.2.5 WorkingHours 55](#_Toc516820614)

[4.2.6 Appointment 55](#_Toc516820615)

[4.2.7 Patient 56](#_Toc516820616)

[4.2.8 Bill 56](#_Toc516820617)

[4.3. Interaction Diagram 57](#_Toc516820618)

[4.3.1. Sequence Diagram 57](#_Toc516820619)

[4.3.1.1. Patient call to book appointment 57](#_Toc516820620)

[4.3.2. Activity Diagram 58](#_Toc516820621)

[4.3.2.1 Register Account 58](#_Toc516820622)

[5. Interface 58](#_Toc516820623)

[6. Database Design 58](#_Toc516820624)

[7. Algorithms 60](#_Toc516820625)

[E. System Implementation & Test 60](#_Toc516820626)

[1. Introduction 60](#_Toc516820627)

[1.1 Overview 60](#_Toc516820628)

[1.2 Test Approach 60](#_Toc516820629)

[2. Database Relationship Diagram 60](#_Toc516820630)

[2.1 Physical Diagram 60](#_Toc516820631)

[2.2 Data Dictionary 61](#_Toc516820632)

[3. Performance Measures 63](#_Toc516820633)

[4. Test Plan 63](#_Toc516820634)

[4.1 Features to be tested 63](#_Toc516820635)

[4.2 Features not to be tested 63](#_Toc516820636)

[5. System Testing Test Case 63](#_Toc516820637)

# List of Tables

# List of Figures

Definitions, Acronyms, and Abbreviations

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

A. Introduction

## 1. 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 31th, 2018**

## 2. 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 switchboard operator. But, that solution has a few problems such as missed call or receive 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.

## 3. 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 sometime, 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 clinic and patients. At first, the clinic contacts to 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.

## 4. Problem Definition

**Clinic using Call-Center:**

**Advantages**:

* Provide professional service to take care patient

**Disadvantages**:

* 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 for wait a call from patient.
* Staff has difficulties to get information from patient for the appointment.
* Clinics are managing patient information, schedule appointment manually, that way makes them time-consuming and inconvenient.

## 5. Proposed Solution

### 5.1 Feature functions

Our solution is 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 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 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 clinic.
* Provide better information management system for the clinic.

**Challenges:**

* Make 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 patient.
* Receive and analyze SMS from patient.

## 6. Functional Requirements

Functional requirements of the system are listed as below:

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

## 7. 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

B. Software Project Management Plan

## 1. Problem Definition

### 1.1 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

### 1.2 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 smartphone as a switchboard to receive and answer 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 Android system and jailbreak IOS to take that privilege but it still not worked correctly. And event after many hours research we find the way to done the first step is 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 on 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.

### 1.3 Project Overview

1.3.1 Current Situation

Below are the problems encountered in this project:

* Difficulty to get permission 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, …

1.3.2 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 manage patient, schedule appointment.
* Hotline server handle incoming SMS and patient call.

1.3.2.1 API application

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

1.3.2.2 Mobile application

* Sign in
* View list schedule appointment.

1.3.2.3 Hotline server

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

1.3.2.4 Web application

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

1.3.3 Boundaries of the System

Our system supports:

* Providing appointment booking service
  + Patient can send SMS or call Clinic Hotline to make appointment.
  + Hotline with legacy call (call without internet) using Hotline server
  + Hotline with VoIP call (call via internet protocol)
  + When appointment has been booked successfully, Send SMS or send notification to patient

Our system hasn’t supports:

* + Adjust appointment
  + Clinic phone auto pick up incoming call
  + Interactive with patient
    1. Future Plans

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

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

1.3.5 Development Environment

1.3.5.1 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:

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

1.3.5.2 Software requirements

|  |  |  |
| --- | --- | --- |
| **Software** | **Name / Version** | **Description** |
| Environment | Node JS  Android | Specification for developing Hotline Server application  Specification for developing 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

## 2. Project organization

### 2.1 Software Process Model

This project is developed using Scrum model – part of an agile framework for Software development project. Our team choose 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 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 deadline.
* Product owner can change requirement or extend scope. The team will adapt to change better.

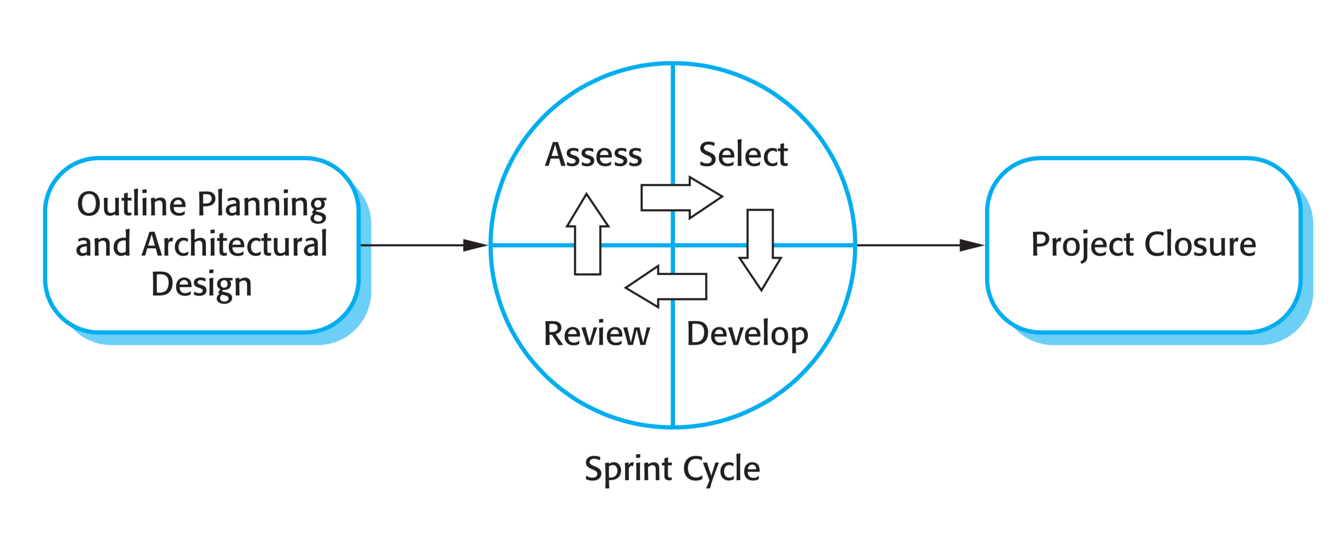


Figure 1 The Scrum Process

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

### 2.2 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

### 2.3 Tools and Techniques

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

Table 6: Tools

## 3. Project Management Plan

### 3.1 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 8: Sprint Backlog

### 3.2 Sprint Backlog

3.2.1 Sprint 1 (18.05.2018 – 25.05.2018): Project initiation

3.2.1.1: Goal

- Sprint 1 must complete the following tasks:

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

3.2.1.2: 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 |

### 3.3 All Meeting Minutes

All meeting minutes are saved at: [here](not%20provide%20yet)

## 4. 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 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 moving the top level component name to the beginning.

- Resource files in the values folder should be **plural**

* **Functions.**

- Don’t ignore exception and don’t catch 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 more terse.

* **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.ts, .directive.ts, .module.ts, .pipe.ts, 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

# C. Software Requirement Specification

## 1. User Requirement Specification

1.1 Clinic Requirement

Doctor can do the following functions*:*

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

1.2 Administrator Requirement

* Administrator is an employee in the system who has responsible for manage clinic account, Administrator can do the following functions:
* Manipulate clinic account

1.3 System Requirement

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

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

## 2. System Requirement Specification

### 2.1 External Interface Requirement

2.1.1 User Interface

* The user interface uses language is English for all web application and Vietnamese for mobile application.
  + 1. Hardware Interface
* **N/A**

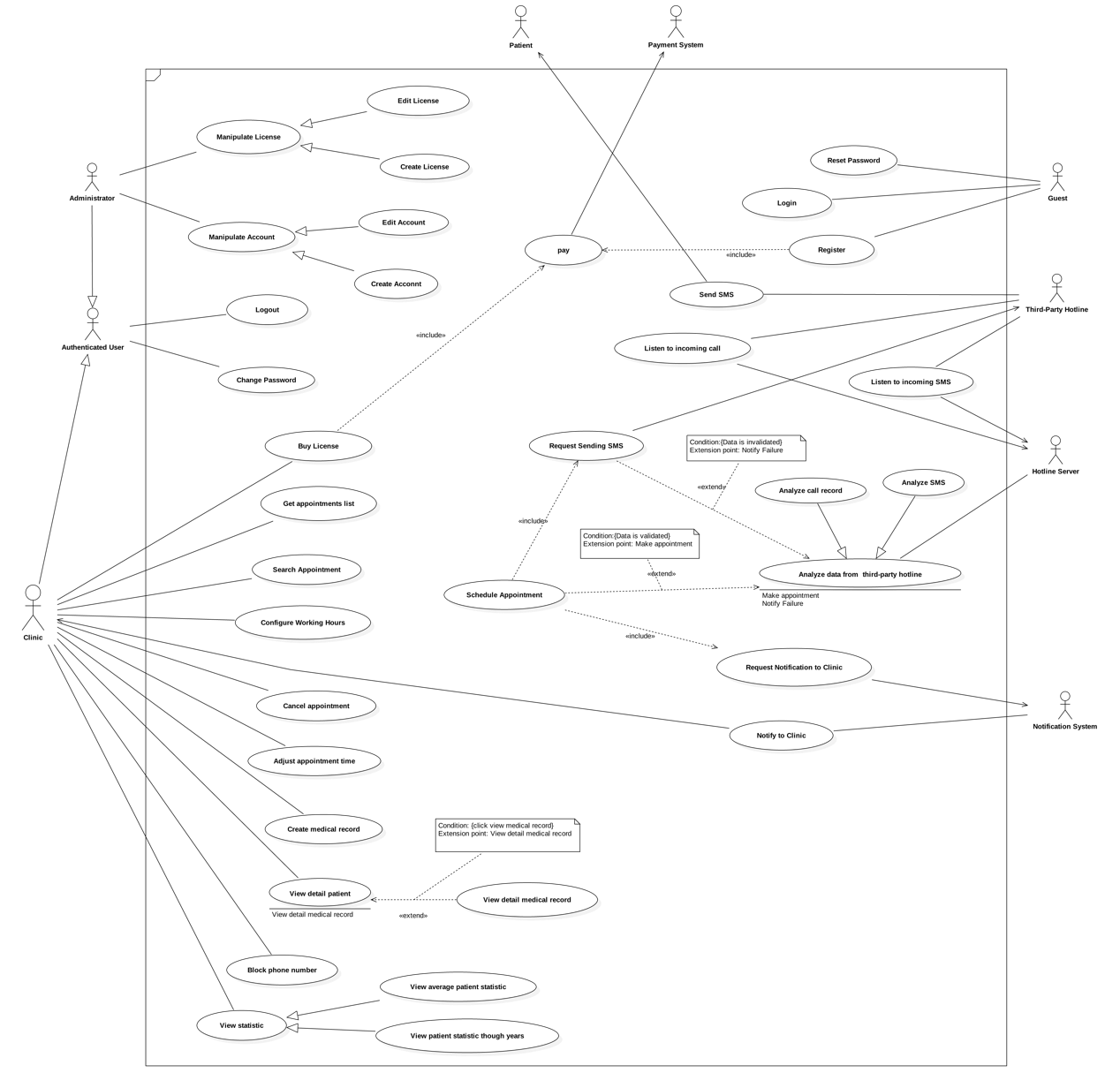
2.1.3 Software Interface

* Service 3rd party:
  + Cloud service
  + Hotline third-party framework

2.1.4 Communication Protocol

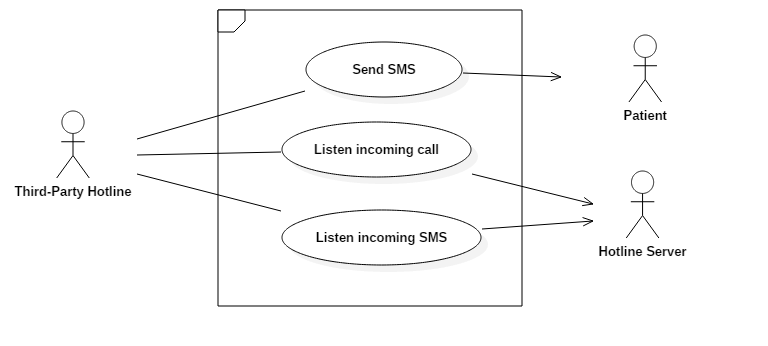
* Use HTTP protocol 1.1 for communication between:
* Web application and web server
* Mobile application and web server

### 2.2 System Overview Usecase

****

### List of Usecase

#### 2.3.1 Third-Party Hotline Overview Usecase

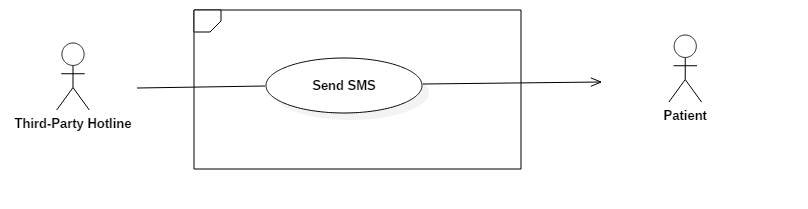


2.3.1.1 < Third-Party Hotline > Listen to incoming call

****

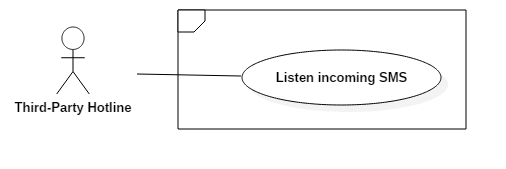
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_TPH.01 | | | |
| Use Case No. | UC\_TPH.01 | **Use Case Version** | 1.0 |
| Use Case Name | Listen to incoming call | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Patient   Summary:   * When the patient 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, 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 patient     Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Call to 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 system receive instruction with wrong format or not able to receive instruction. | Notify error to patient and write an error to log file |   Relationships:   * Analyze data from third-party hotline: At step 8 of Main Success Scenario, third party hotline will send call record information to hotline server and then hotline server receive and analyze data to make appointment by “Analyze data from third-party hotline” Usecase.   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 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 at 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), 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 * Third-party hotline will send request to hotline server when the record file is ready, the request contain information about a call and record file. | | | |

2.3.1.2< Third-Party Hotline > Send SMS



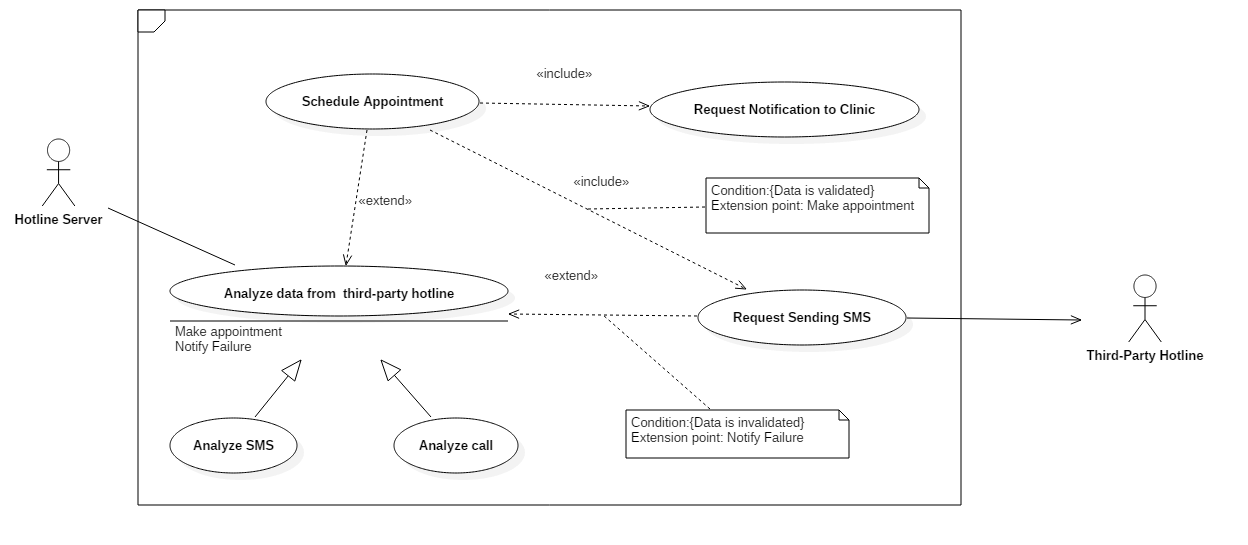
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_TPH.02 | | | |
| Use Case No. | UC\_TPH.02 | **Use Case Version** | 1.0 |
| Use Case Name | Send SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline server   Summary:   * This use case allows hotline 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 case 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. | | | | |

2.3.1.3< Third-Party Hotline > Listen incoming SMS

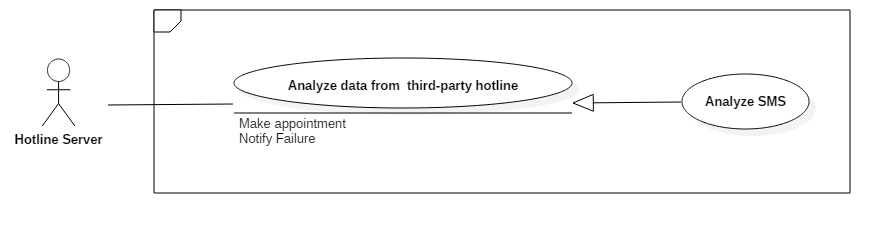


|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_TPH.03 | | | |
| Use Case No. | UC\_TPH.03 | **Use Case Version** | 1.0 |
| Use Case Name | Listen incoming SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Patient.   Summary:   * This use case allows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

#### 2.3.2 Hotline Overview Usecase

****

2.3.2.1< Hotline Server > Analyze SMS



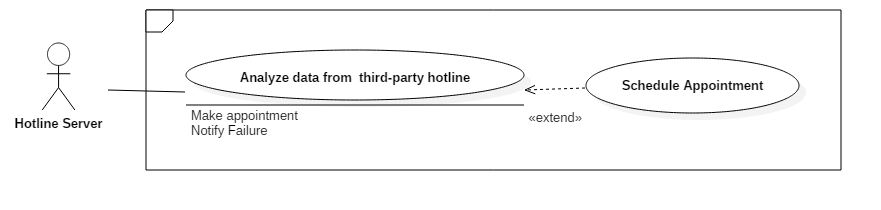
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.01 | | | |
| Use Case No. | UC\_HL.01 | **Use Case Version** | 1.0 |
| Use Case Name | Analyze SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline Server.   Summary:   * This use case allows hotline server to analyze data from third-party hotline. Whenever it’s receive a request from third-party hotline new SMS is received.   Goal:   * SMS’s information from third-party hotline is analyzed.   Triggers:   * Whenever it’s receive a request from third-party hotline new SMS is received   Preconditions: N/A  Post Conditions:   * Success: SMS from third-party hotline is analyzed. * Fail: Log error message to the system.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Send request contain information about new SMS | Get SMS data from 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 be less than 255 character. * SMS structure much follow is pattent: MP<space><patient’s name>   For example: MP Hoang Hoa | | | |

2.3.2.2< Hotline Server > Analyze Call



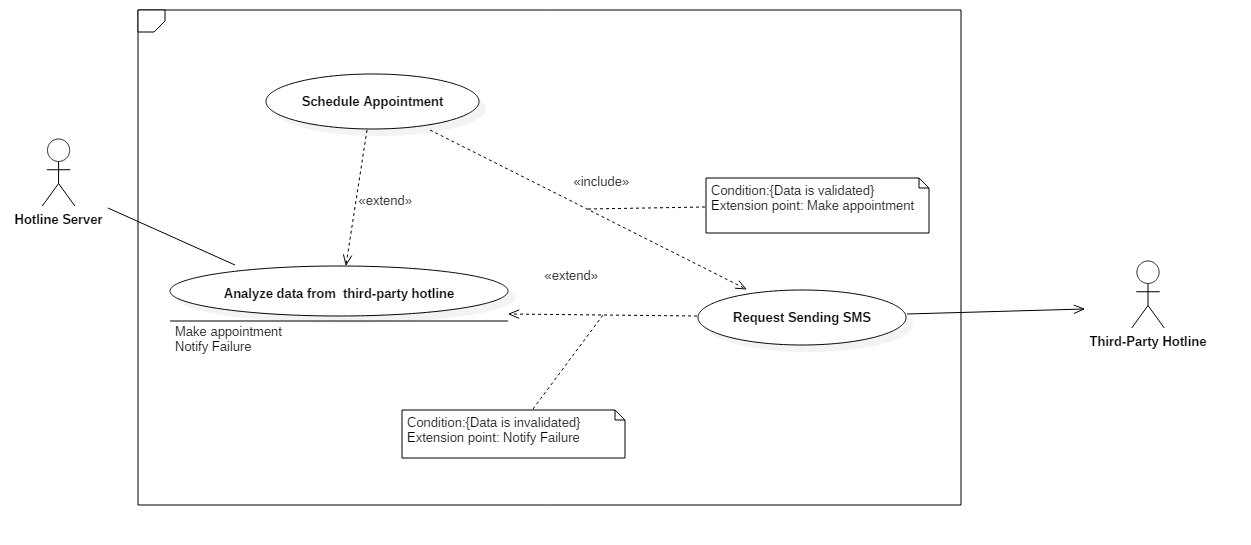
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.02 | | | |
| Use Case No. | UC\_HL.01 | **Use Case Version** | 1.0 |
| Use Case Name | Analyze Call | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline Server.   Summary:   * This use case allows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

2.3.2.3< Hotline Server > Schedule Appointment



|  |  |  |  |
| --- | --- | --- | --- |
| 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 third-party hotline.   Goal:   * 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 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 new appointment (not have enough time left, off day v…v…) | Request sending failure SMS to third-party hotline and log the event into the system |   Exceptions: N/A  Relationships:   * Request sending SMS * Analyze data from third-party hotline * Request notification to clinic   Business Rules:   * If the clinic is 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 calculate base on the formular: Base time + Examinate 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 exceed clinic’s end working hours. Then the appointment is canceled.   Ex: New appointment time is 13:45 and the examination duration is 30 but clinic is not working after 14:00, then the appointment is canceled   * No appointments is made in off day. | | | |

2.3.2.4< Hotline Server > Request sending SMS



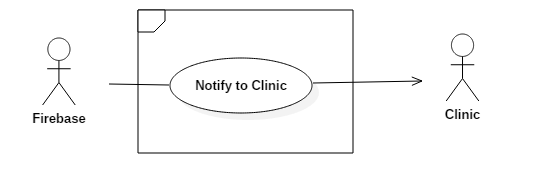
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.03 | | | |
| Use Case No. | UC\_HL.03 | **Use Case Version** | 1.0 |
| Use Case Name | Request sending SMS | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Hotline Server   Summary:   * This usecase allow Hotline server request sending a SMS to third-party hotline when ever it need to informe patient some information. The information can be various such as notify patient about their appointment’s information, or their appointment cannot be booked.   Goal:   * SMS is send to patient   Triggers:   * When actor request sending SMS to 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 third-party hotline |   Alternative Scenario: N/A  Exceptions: N/A  Relationships:   * Schedule appointment * Analyze data from third-party hotline   Business Rules:   * The send SMS request only made when   + New appointment is made (send SMS contain appointment’s information)   + An appointment is cancel (send SMS to inform patient that their appointment cannot be booked at the moment)   + An appointment is about on time, the SMS is send before appointment’s time 30 minutes | | | |

2.3.2.5< Hotline Server > Request Notification to Clinic

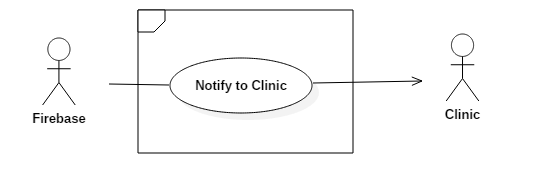


|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_HL.04 | | | |
| Use Case No. | UC\_HL.04 | **Use Case Version** | 1.0 |
| Use Case Name | Request Notification to Clinic | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Low |
| Actor:   * Firebase   Summary:   * This use case allows Firebase send notification to clinic in case that clinic have 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:   * Only request send notification when a new appointment is successful created * Notify clinic that associate with the previous appointment only ( notify the right clinic, not notify all) | | | |

#### 2.3.3 Firebase Overview Usecase

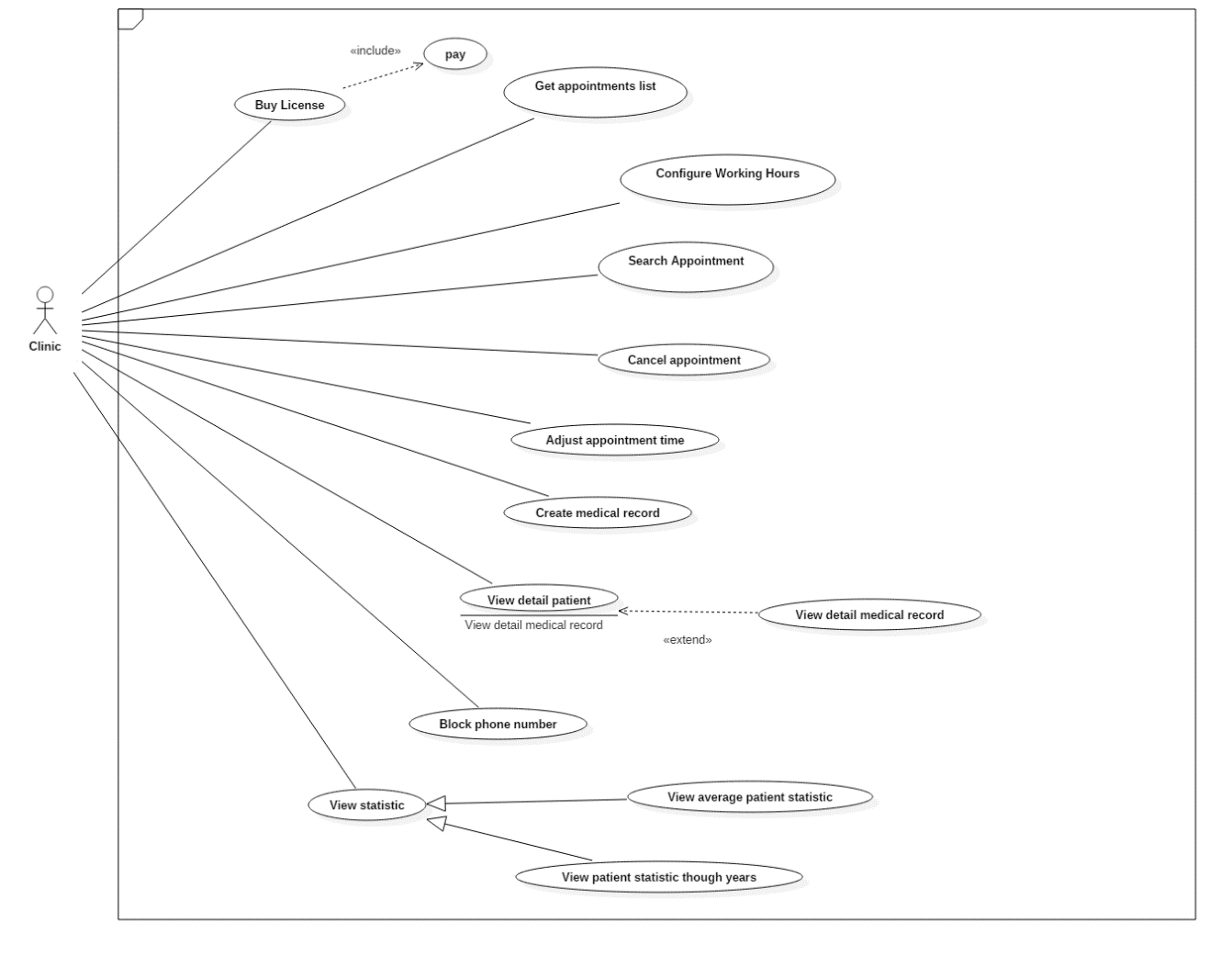


2.3.3.1 < Firebase > Notify to Clinic

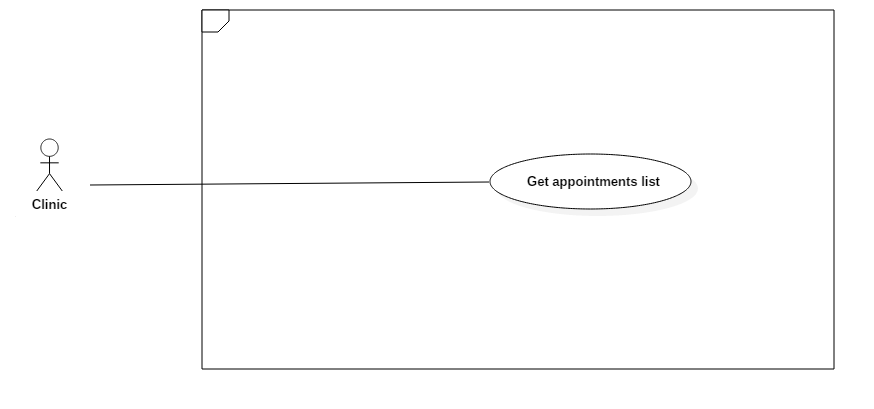


|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_FB.01 | | | |
| Use Case No. | UC\_FB.01 | **Use Case Version** | 1.0 |
| Use Case Name | Notify to Clinic | | |
| Author | KietNLT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Firebase   Summary:   * This use case allows Firebase send notification to Clinic   Goal:   * Send a notification to 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 notification to clinic | Push notification to clinic |   Alternative Scenario: N/A  Exceptions: N/A  Relationships: N/A  Business Rules:   * Only request send notification when a new appointment is successful created * Notification content: Bạn có một cuộc hẹn mới! | | | |

2.3.4 Clinic Overview Usecase

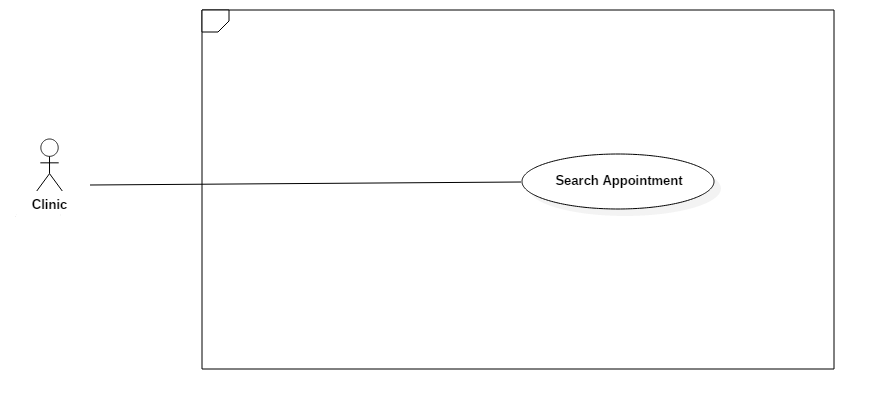
****

2.3.4.1 <Clinic> Get appointment list



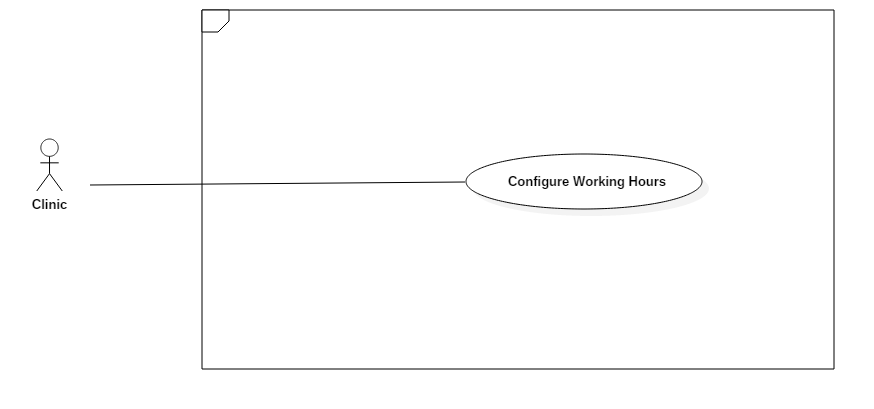
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C01 | | | |
| Use Case No. | UC\_C01 | **Use Case Version** | 1.0 |
| Use Case Name | Get appointment list | | |
| Author | ThuanPT | | |
| Date | 27/05/2018 | **Priority** | High |
| Actor:   * Clinic.   Summary:   * This use case allows the clinic 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 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 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 command to get appointment list to system. | Appointment list is display not information. |   Exceptions: N/A  Relationships: N/A.  Business Rules:   * The clinic will get appointment list, it be register by patient. * Show appointment list belong base on 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 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. | | | |

2.3.4.2 <Clinic> Search appointment Usecase



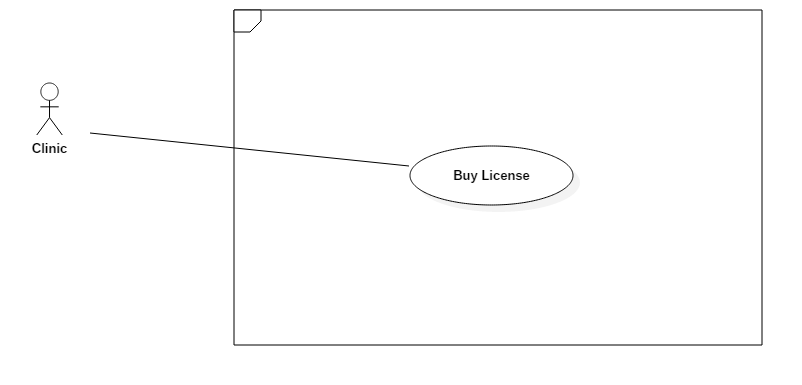
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C02 | | | |
| Use Case No. | UC\_C02 | **Use Case Version** | 1.0 |
| Use Case Name | Search appointment | | |
| Author | ThuanPT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows clinic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

2.3.4.3 <Clinic> Configure working hours Usecase

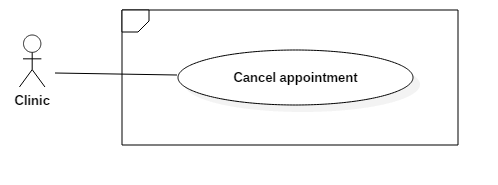


|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C03 | | | |
| Use Case No. | UC\_C03 | **Use Case Version** | 1.0 |
| Use Case Name | Configure working hours | | |
| Author | ThuanPT | | |
| Date | 27/05/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows clinic to choose start hour, end hour and days of week and then setup working hours of clinic their self.   Goal:   * Clinic have 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 | Clinic goes to working hours view | System requires identify information from clinic:   * Start hours: time input, required. * End hours: time input, required. * Days of the week: number input, required | | 2 | Clinic input start hour and end hour.  Clinic input option the days of week.  [Exception 1] |  | | 3 | Clinic send command to setup working hours to system | Clinic will be setup working hours |   Alternative Scenario: N/A  Exceptions:   |  |  |  | | --- | --- | --- | | Step | Cause | System Response | | 1 | Clinic input start hour later than end hour | System shows 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, system will display working hours information of every day follow card format:   * Thứ. * Giờ bắt đầu. * Giờ kết thúc. | | | |

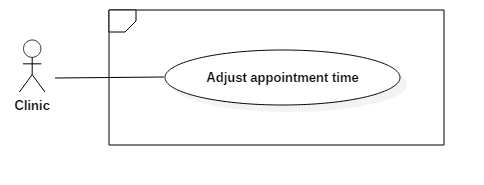
2.3.4.4 <Clinic> Buy license Usecase



|  |  |  |  |
| --- | --- | --- | --- |
| 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 payment system.   Goal:   * Clinic have 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 | Clinic goes to buy license view | System requires identify information from clinic:   * Tên * Giá * Thời lượng * Mô tả | | 2 | Clinic select license. |  | | 3 | Clinic send command to buy license. | 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, clinic will send buy license to payment system and then payment system receive and pay to license by “Pay” Usecase..   Business Rules:  - License list will be display in the following information:   * Tên * Giá * Thời lượng * Mô tả   - Description will help clinic understand about that license.  - Clinic will select license before payment.  - After clinic choose license, clinic will pay through payment system.  - The clinic can use functions be provided by the system after payment license is successfully. | | | |

2.3.4.5 <Clinic> Cancel Appointment

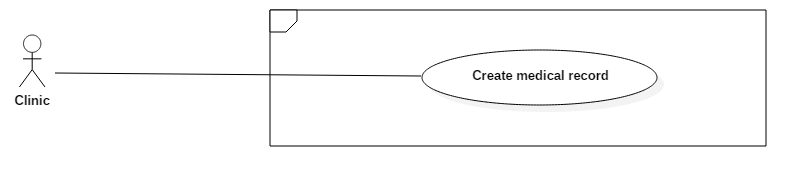
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C05 | | | |
| Use Case No. | UC\_C05 | **Use Case Version** | 1.0 |
| Use Case Name | Cancel Appointment | | |
| Author | KietNLT | | |
| Date | 17/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * Sometimes doctors are busy or have an urgent, they need to take a whole day off. The remaining appointments need to be cancel, this use case is made to support clinics in those situation. Clinic can send command cancel appointment and all the appointment remaining in that day will be cancel.   Goal:   * All appointment remaining in the day is cancel.   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 | | | | |

2.3.4.6 <Clinic> Adjust Appointment Time

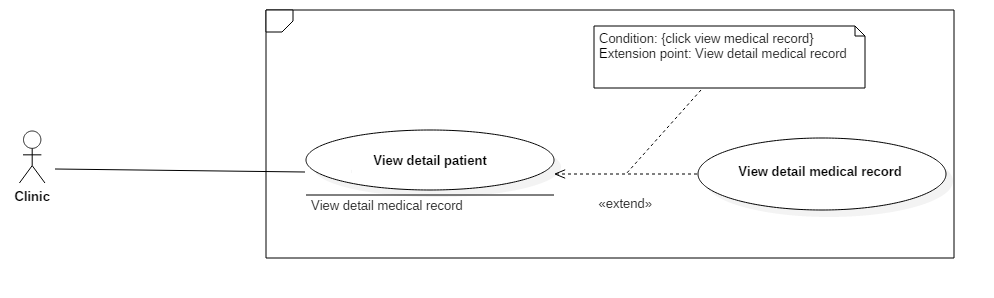
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C06 | | | |
| Use Case No. | UC\_C06 | **Use Case Version** | 1.0 |
| Use Case Name | Adjust Appointment Time | | |
| Author | KietNLT | | |
| Date | 17/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allow clinic adjust appointment time in case doctors is late for the appointment. The adjustment period can be various as long as after adjust the time is not bound to the next day.   Goal:   * Adjust the appointment time.   Triggers:   * Clinic send 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 adjust, 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 | | | | |

2.3.4.7 <Clinic> Block Phone Number

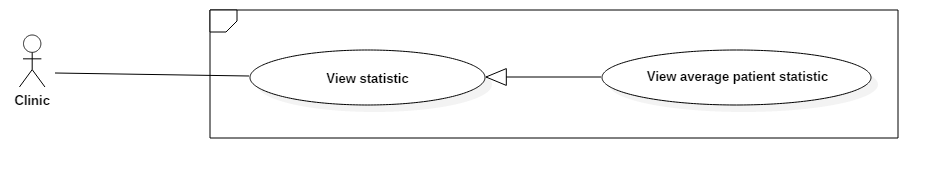
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C07 | | | |
| Use Case No. | UC\_C07 | **Use Case Version** | 1.0 |
| Use Case Name | Block Phone Number | | |
| Author | KietNLT | | |
| Date | 17/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * This use case allows clinic 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 send 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. | | | | |

2.3.4.8 <Clinic> Create Medical Record

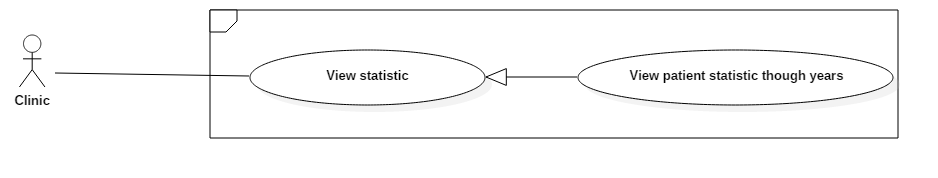
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_C08 | | | |
| Use Case No. | UC\_08 | **Use Case Version** | 1.0 |
| Use Case Name | Create medical record | | |
| Author | DuyNC | | |
| Date | 18/07/2018 | **Priority** | Normal |
| Actor:   * Clinic   Summary:   * When the patient visits the clinic, the clinic creates a patient record to store the patient's information including 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 command to create the patient’s medical record   Preconditions:   * The patient must be visit 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 command to create 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 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 command to save  [Alternative 3] | New medical record has been adding to the system  [Exception 1,2]  Show message created successfully |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor does not sent command to add medicine. | The system will not display medicine information. | | 2 | Actor sends command to remove. | Medicine will be removed out of the list medicines. | | 3 | Actor sends 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 multiply 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”. | | | |

2.3.4.9 <Clinic> View detail patient

|  |  |  |  |
| --- | --- | --- | --- |
| 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 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 command to view 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 command to view 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 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 | Actor does not input new information. | Data does not change. | | 2 | Actor does not sent command to view medical record. | The system will not display view medical record information. | | 3 | Actor sends 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 validating by date format: dd/mm/yyyy * When the clinic input patient’s information and send command to update:   + If successfully, the patient’s information will be 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. | | | |

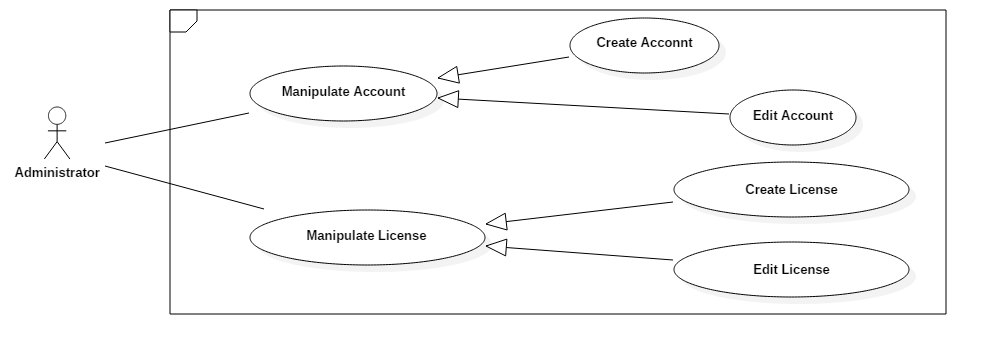
2.3.4.10 <Clinic> View average patient statistic

|  |  |  |  |
| --- | --- | --- | --- |
| 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 command to view 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 | Actor goes to 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 | 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”. | | | |

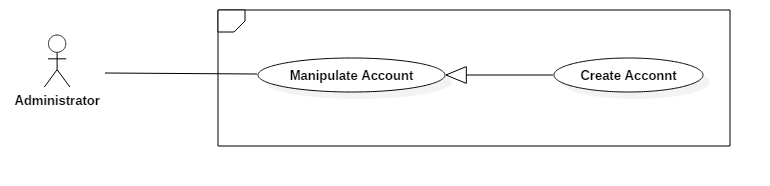
2.3.4.11 <Clinic> View patient statistic through years

|  |  |  |  |
| --- | --- | --- | --- |
| 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 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 | 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 | 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”. | | | |

#### 2.3.5 Administrator Overview Usecase

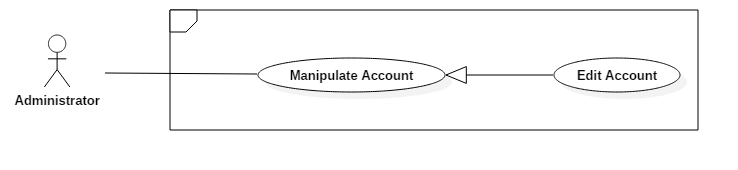
****

2.3.5.1 <Administrator> Create account

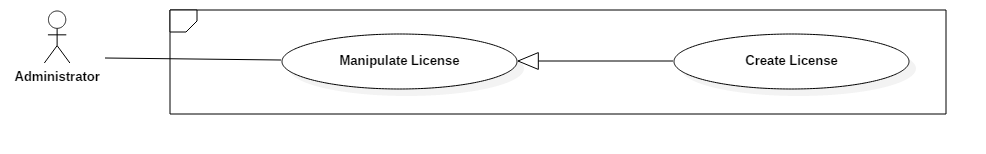


|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_A01 | | | |
| Use Case No. | UC\_A01 | **Use Case Version** | 2.0 |
| Use Case Name | Create Account | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Administrator.   Summary:   * This use case allows the administrator to create a new account in the system. * The administrator can use the new account to login into the system.   Goal:   * New account will be adding into the system.   Triggers:   * The administrator sends command to create a new account.   Preconditions:   * Actor has been login with administrator role.   Post Conditions:   * Success: New account is created. * Fail: Account is not created and show error message.   Main Success Scenario:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Actor goes to create account view. | 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 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 have been existed”. | | 2 | Actor inputs email already exist. | System show warning message “Email have been existed”. | | 3 | Actor inputs phone number already exist | System show warning message “Phone number have been existed”. | | 4 | Actor does not input required field. | System notices that actor need to input all those field. | | 5 | Actor inputs wrong some fields with requirement. | System notices that actor need to re-input all those fields. |   Relationships: N/A  Business Rules:   * Administrator must be input require information to create, input information include:   + 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 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. Administrator can be access in the system.   + If create fail, account is not added in the system. And system will show error message for administrator. | | | |

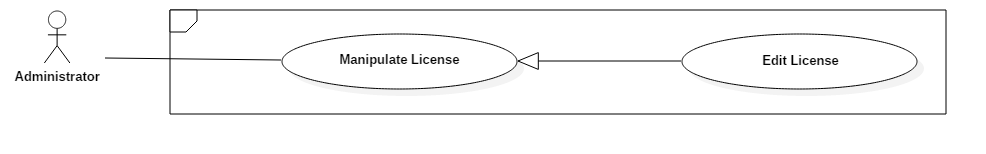
2.3.5.2 <Administrator> Edit account



|  |  |  |  |
| --- | --- | --- | --- |
| 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 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. | System shows list all accounts | | 2 | Actor sends command to edit account information | 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 command to save.  [Alternative 1] | Account information is updated.  [Exception 1]  [Exception 2]  [Exception 3] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor sends 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 been existed”. | | 2 | Actor inputs phone number already exist | System show warning message “Phone number have been existed”. | | 3 | Actor does not input required field. | System notices that actor need to input all those field. |   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 system will show error message for administrator. * Reload account information and display new information has been updated. | | | |

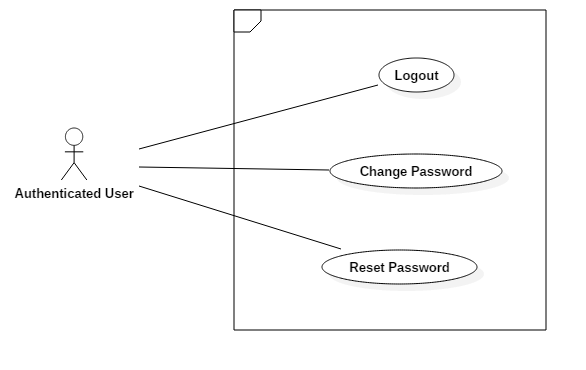
2.3.5.3 <Administrator> Create license

|  |  |  |  |
| --- | --- | --- | --- |
| 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 command to create a new license.   Preconditions:   * 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. | 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 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 command to reset. | The system will reset all field to blank. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor does not input 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 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. | | | |

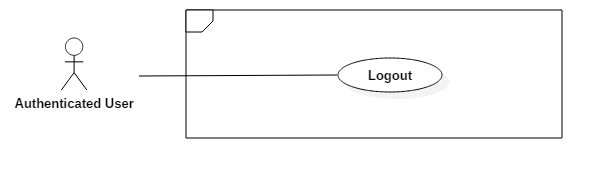
2.3.5.4 <Administrator> Edit license

|  |  |  |  |
| --- | --- | --- | --- |
| 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 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. | System shows list all licenses | | 2 | Actor sends command to edit license information | 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 command to save.  [Alternative 1] | License information is updated.  [Exception 1]  [Exception 2] |   Alternative Scenario:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor sends command to cancel. | The system will close the edit license view. |   Exceptions:   |  |  |  | | --- | --- | --- | | No | Cause | System Response | | 1 | Actor does not input required field. | System notices that actor need to input all those field. | | 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 integer. * 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 system will show error message for administrator. * Reload account information and display new information has been updated. | | | |

#### 2.3.6 Authenticated User Overview Usecase

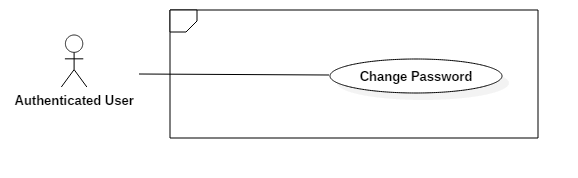


2.3.6.1 <Authenticated User> Logout account



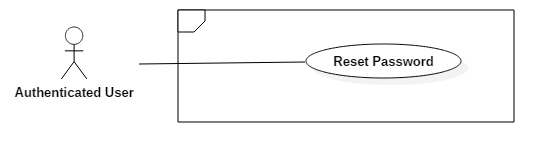
|  |  |  |  |
| --- | --- | --- | --- |
| USE CASE – UC\_AU01 | | | |
| Use Case No. | UC\_AU01 | **Use Case Version** | 1.0 |
| Use Case Name | Logout Account | | |
| Author | DuyNC | | |
| Date | 28/05/2018 | **Priority** | Low |
| Actor:   * Authenticated User.   Summary:   * This use case allows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

2.3.6.2 <Authenticated User> Change password



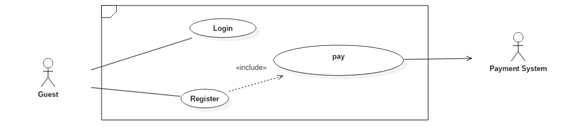
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| USE CASE – UC\_A02 | | | | | | |
| Use Case No. | | UC\_A02 | **Use Case Version** | | | 1.0 |
| Use Case Name | | Change password | | | | |
| Author | | DuyNC | | | | |
| Date | | 28/05/2018 | **Priority** | | | Low |
| Actor:   * Authenticated User   Summary:   * This use case allows actor change password.   Goal:   * Update new password in the system and actor can login by new password.   Triggers:   * Authenticated user sends change password command.   Preconditions:   * Authenticated user has been login with Authenticated user role.   Post Conditions:   * Success: Updated new password in the system. * Fail: New password cannot update and show error message   Main Success Scenario: | | | | | | |
| Step | Actor Action | | | System Response | | |
| 1 | Authenticated user goes to change password view. | | | System requires information:   * Current password: password, require, max length 72 * New password: password, require, max length 72 * Confirm new password: password, require, max length 72 | | |
| 2 | Authenticated user inputs information. | | |  | | |
| 3 | Authenticated user sends command to change password. | | | Show message to notify that new password have been updated successfully.  [Exception 1] | | |
| Alternative Scenario: | | | | | | |
| No | Actor Action | | | System Response | | |
| 1 | Authenticated user enter wrong current password. | | | System shows error message “Mật khẩu hiện tại không đúng”. | | |
| Exceptions: | | | | | | |
| No | Cause | | | | System Response | |
| 1 | Authenticated user inputs new password and confirm 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:   * In case of success scenarios, password would be updated in the system. * New password and confirm new password are matched each other. * New password would be hash before save in the system. * Display successful message when new password has been updated. | | | | | | |

2.3.6.3 <Authenticated User> Reset Password

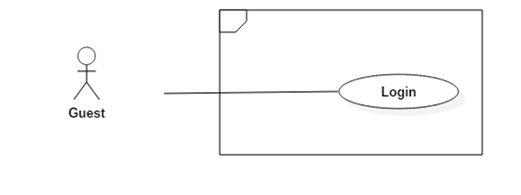


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| USE CASE – UC\_AU03 | | | | | | |
| Use Case No. | | UC\_AU03 | **Use Case Version** | | | 1.0 |
| Use Case Name | | Reset password | | | | |
| Author | | DuyNC | | | | |
| Date | | 28/05/2018 | **Priority** | | | Low |
| Actor:   * Authenticated User   Summary:   * This use case allows actor reset password.   Goal:   * Authenticated user gets back forgotten account.   Triggers:   * Authenticated user sends reset password command.   Preconditions: N/A  Post Conditions:   * Success: System will send new password via email. * Fail: Password cannot reset and show error message.   Main Success Scenario: | | | | | | |
| Step | Actor Action | | | System Response | | |
| 1 | Authenticated user goes to reset password view | | | System requires identity information:  Email: free text input, email format, unique | | |
| 2 | Authenticated user inputs information | | |  | | |
| 3 | Authenticated user sends command to reset password | | | System send confirm code via email.  System redirect to confirm code view.  [Exception 1]  [Exception 2] | | |
| 4 | Authenticated user input confirm code | | | System redirect to input new password view. | | |
| 5 | Authenticated user input new password and send command to save | | | New password will update in the system.  System view redirect to login view. | | |
| Alternative Scenario: N/A | | | | | | |
| Exceptions: | | | | | | |
| No | Cause | | | | System Response | |
| 1 | Authenticated user inputs email not exist in the system | | | | Show message notify entered email not existed | |
| 2 | Authenticated user does not input required field. | | | | System notices that actor need to input all those field. | |
| Relationships: N/A.  Business Rules:   * In case of success scenarios, password would be updated to the system. * Authenticated user input email and receive new password via email. * Input email must be existing in the system. * System sends confirm code via email:   + If successful, confirm code will send for authenticated user via email.   + If fail, confirm code not send for authenticated user via email. * Authenticated user input confirm code to set up new password. * New password would be hash before save in the system * Display successful message when new password has been updated. | | | | | | |

#### 2.3.7 Guest Overview Usecase

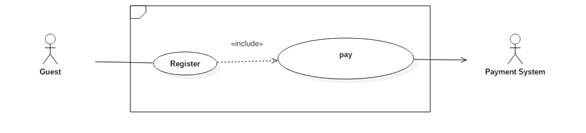


2.3.7.1 <Guest> Login account



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| USE CASE – UC\_G01 | | | | | | |
| Use Case No. | | UC\_G01 | **Use Case Version** | | | 1.0 |
| 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 proper role.   Triggers:  Guest sends the login command to the system.  Preconditions:   * Guest must be register and authenticated before access the system   Post Conditions:   * Success: Guest accesses to the system successfully * Fail: Guest cannot login into the system and show error message   Main Success Scenario: | | | | | | |
| Step | Actor Action | | | System Response | | |
| 1 | Guest goes to login view | | | System requires identity information:   * Username: free text input, required, max length 30 * Password: password, require, max length 30 | | |
| 2 | Guest inputs information | | |  | | |
| 3 | Guest sends command to login. | | | Guest will login system with their specific role.  [Alternative 1]  [Exception 1] | | |
| Alternative Scenario: | | | | | | |
| No | Actor Action | | | System Response | | |
| 1 | Guest inputs wrong identity information | | | System show error message “Tên đăng nhập hoặc mật khẩu không đúng”. | | |
| Exceptions: | | | | | | |
| No | Cause | | | | System Response | |
| 1 | Guest does not input required field. | | | | System notices that actor need to input all those field. | |
| Relationships: N/A.  Business Rules:   * In case of success scenarios, guest will be redirected to specific view based on their role on the system:   + If role is “Administrator”, the system will display to Administrator Dashboard view.   + If role is “Clinic”, the system will display to Clinic Dashboard view. * If role is “Administrator”, must be login at the Website. * If role is “Clinic”, must be login at the android mobile. | | | | | | |

2.3.7.2 <Guest> Register



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| USE CASE – UC\_ G02 | | | | | | |
| Use Case No. | | UC\_ G02 | **Use Case Version** | | | 1.0 |
| Use Case Name | | Register | | | | |
| Author | | DuyNC | | | | |
| Date | | 28/05/2018 | **Priority** | | | Normal |
| Actor:   * Guest   Summary:   * This use case allows guest to register new account.   Goal:   * Guest can login into the system by using this account.   Triggers:  Guest sends register command to the system.  Preconditions: N/A.  Post Conditions:   * Success: New account is added to the system. * Fail: Account is not added and show error message.   Main Success Scenario: | | | | | | |
| Step | Actor Action | | | System Response | | |
| 1 | Guest goes to register view | | | System requires information:  **User information:**   * Username: free text input, required, length (9-20), unique * Password: password, length 6 - 72, required * Confirm password: password, length 6 - 72, required * Email: free text input, email format, unique * Full Name: free text input, required, max length 255   **Clinic information:**   * Clinic Name: free text input, required, max length 255 * Address: free text input, max length 255 * Examination Duration: time, required | | |
| 2 | Guest inputs information | | |  | | |
| 3 | Guest sends command to register.  [Alternative 1]  [Alternative 2] | | | New account has added to the system  [Exception 1]  [Exception 2]  [Exception 3]  [Exception 4]  Display new view to show license information:   * License name: text * License price: number * License duration: number * License description: text | | |
| 4 | Guest select license | | | System display new view to select payment method | | |
| Alternative Scenario: | | | | | | |
| No | Actor Action | | | System Response | | |
| 1 | Guest sends command to reset | | | System reset all field to blank | | |
| 2 | Guest sends command to cancel | | | System will navigate to login page | | |
| Exceptions: | | | | | | |
| No | Cause | | | | System Response | |
| 1 | Guest inputs username already exist. | | | | System show warning message “Tên đăng nhập này đã có người sử dụng”. | |
| 2 | Guest inputs email already exist | | | | System show warning message “Email này đã tồn tại”. | |
| 3 | Guest inputs new password and confirm 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 field. | |
| Relationships: Pay.  Business Rules:   * In case of success scenarios, a new account would be added to the system. * Guest must be input require information to create, input information include:   + 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 hash before save in the system. * A new account would be creating with inputted information.   + If create success, a new account will be added with role.   + If create fail, account is not added in the system. And system will show error message for guest * The initial status of account will be set to “Inactive”. * New account has been adding into the system with clinic role. * Guest must be select one license. | | | | | | |

2.3.7.2 <Guest, Clinic> Pay

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| USE CASE – UC\_G03 | | | | | | |
| Use Case No. | | UC\_G03 | **Use Case Version** | | | 1.0 |
| Use Case Name | | Pay | | | | |
| Author | | DuyNC | | | | |
| Date | | 28/05/2018 | **Priority** | | | Normal |
| Actor:   * Guest, Clinic.   Summary:   * This use case allows actor pay for license to use the call center system.   Goal:   * Actor can use 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. * 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 error message.   Main Success Scenario: | | | | | | |
| Step | Actor Action | | | System Response | | |
| 1 | Actors goes to payment method view | | | Display new view let actors select one of following payment gateways | | |
|  | Actors sends command to select payment method  [Alternative 1] | | | Forward to payment method view to process  the payment | | |
| 2 | Actors process the payment method | | | Show message pay successful  [Exception 1] | | |
| Alternative Scenario: | | | | | | |
| No | Actor Action | | | System Response | | |
| 1 | If actors are not choose payment method | | | System will display view to show license information | | |
| Exceptions: | | | | | | |
| No | Cause | | | | System Response | |
| 1 | If payment failed | | | | Show message to notify user that payment failed and the license request has been aborted | |
| Relationships: Register, buy license.  Business Rules:   * In case of success scenarios, the account will be added used date. * When actor completed payment process:   + When register:     - If the payment successfully, the account’s status would change from “Inactive” to “Active” and the expiration date of the license will be added.     - If the payment fails, the account’s status is not changed.   + When buy license:     - If the payment successfully, the expiration date of the license will be added.     - If the payment fails, the expiration date of the license is not change. | | | | | | |

## 3. Software System Attribute

### 3.1 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.

### 3.2 Reliability

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

### 3.3 Availability

* System replies in maximum 20 seconds.
* Server have a back-up electronic source.
* Hotline always pick up incoming call.

### 3.4 Security

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

### 3.5 Maintainability

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

### 3.6 Portability

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

### 3.7 Performance

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

## 4. Conceptual Diagram

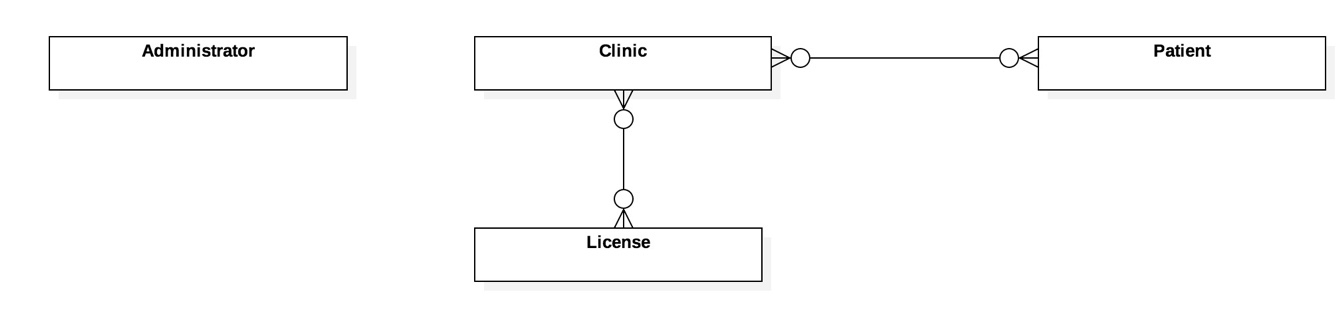


Figure 8 Conceptual diagram

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

Table 7 Conceptual diagram data dictionary

# D. 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 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 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 class diagram, class explanation, activity diagram and sequence diagram to details the application functions.
* Section 5: describe screen design.
* Section 6: describe a fully attribute ERD.
* Section 7: describe algorithms.

## 2. System Architecture Design

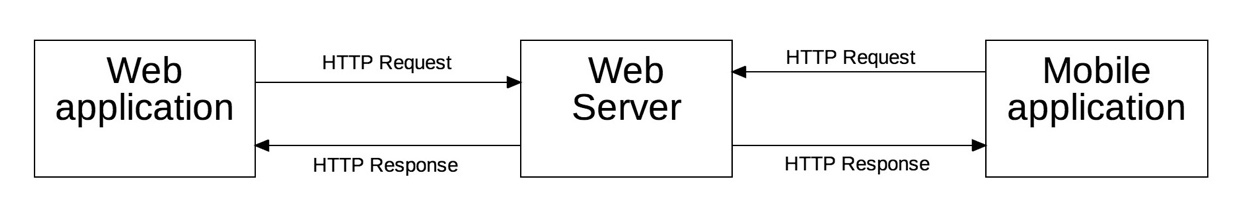


Figure 9. System Architecture

### 2.1 Web Server Architecture Design

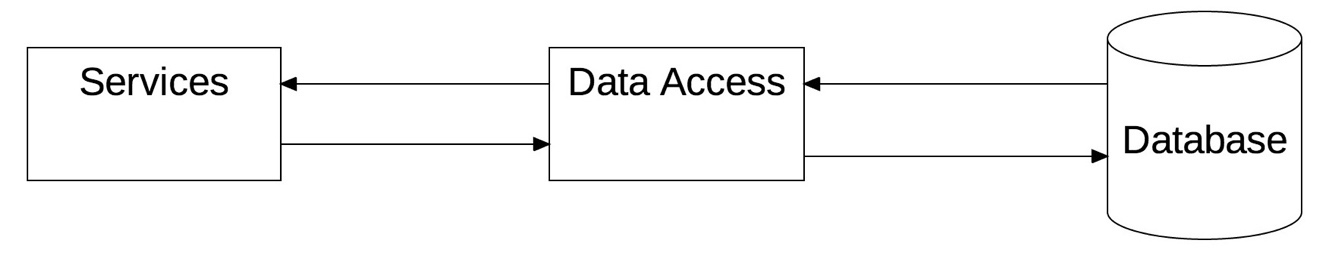


Figure 9. Web Server Architecture

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

* Services: Provide API Services to communication with 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 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.

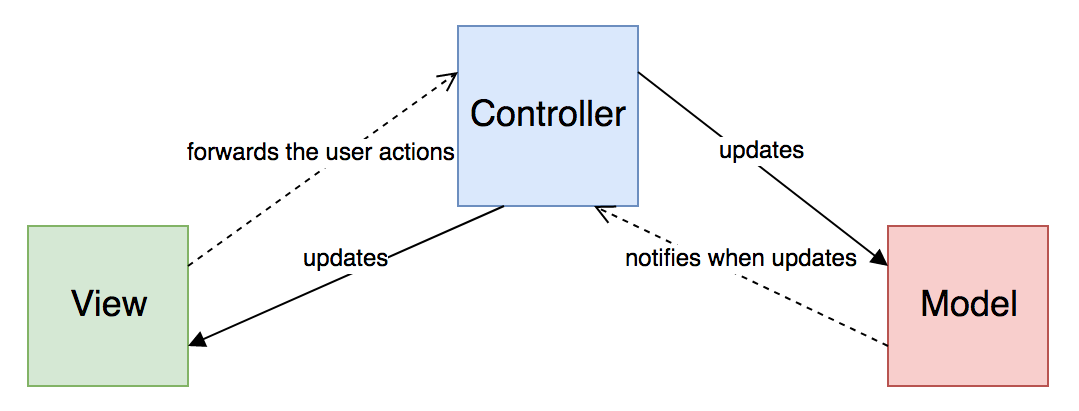


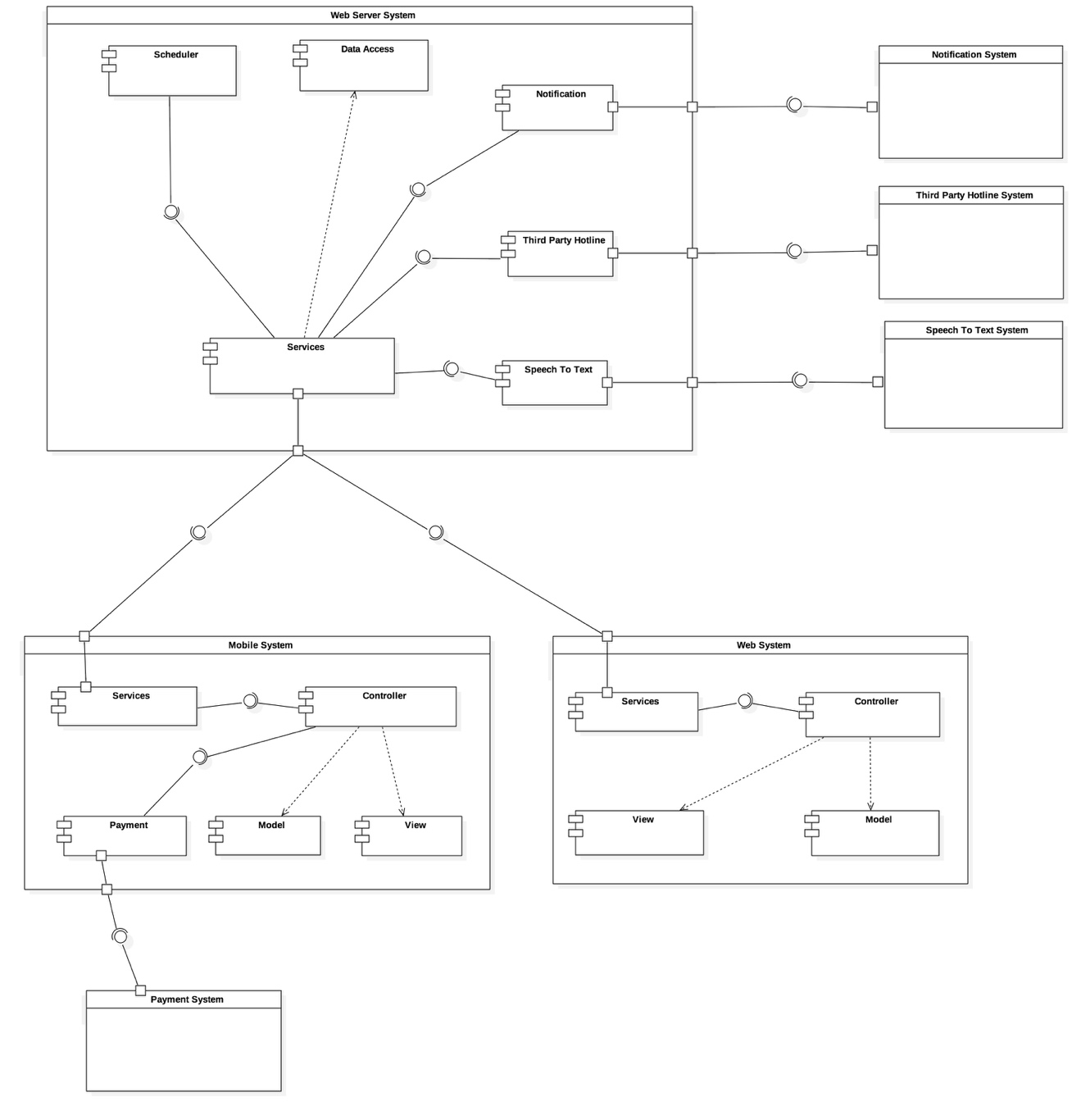
Figure 9. Mobile MVC Architecture

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

MVC architecture has 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 to 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.

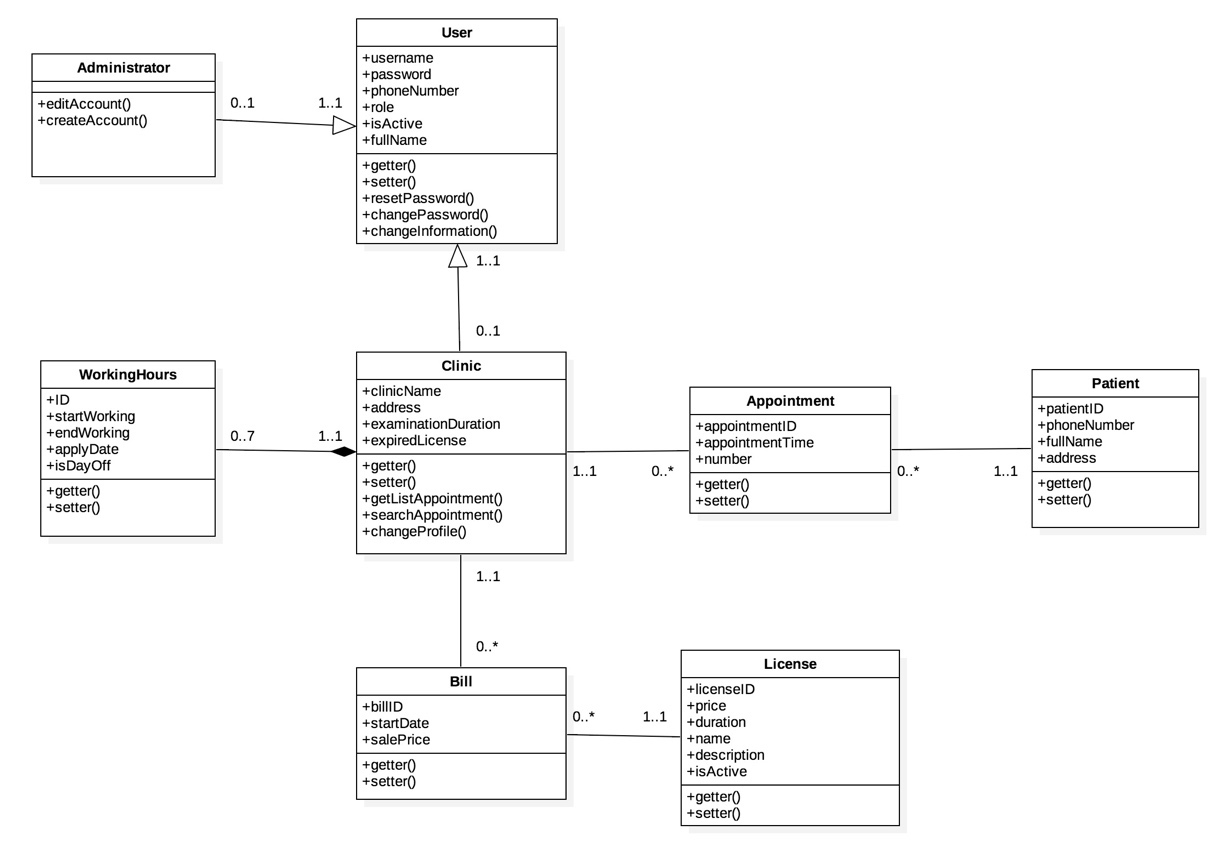
## 3. Component Diagram



|  |  |
| --- | --- |
| **Component dictionary: describe component** | |
| **Component Name** | **Description** |
| Scheduler | Component support our system to schedule appointment. |
| Data Access | Supporting component to retrieve data stored in database. |
| Services of 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 to working with Payment System. |
| Payment System | Providing payment method for clinic. |
| Third-party Hotline System | Supporting our system to 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 model. |
| View of Mobile System | Content all views of the Mobile system. |
| Model of Mobile System | Content all data of Mobile system. |
| Controller of Web System | Handle event and binding data from 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 user. * password: password for each user. * phoneNumber: The phone number of each user. * role: role of user. It maybe is administrator or clinic. * isActive: status of user. * fullName: full name of user. |
| Administrator | Contain the information the relationship between user whom role is Administrator.  Administrator can create and edit information of user. |
| Clinic | Contain the information the relationship between user whom role is Clinic and clinic information.  Clinic can view and edit information. |
| WorkingHours | Contain working hours’ information:   * startWorking: opening hours in one day. * endWorking: closing hours in one day. * applyDate: day in 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 patient. * fullname: A real name of patient. * address: address of 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 user |
| phoneNumber | String | Private | Phone number of user |
| roleID | byte | Private | User’s role |
| isActive | boolean | Private | Determine user is active or not |
| fullName | String | Private | Full name of 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 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 fullname |
| address | String | Private | The address of 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**

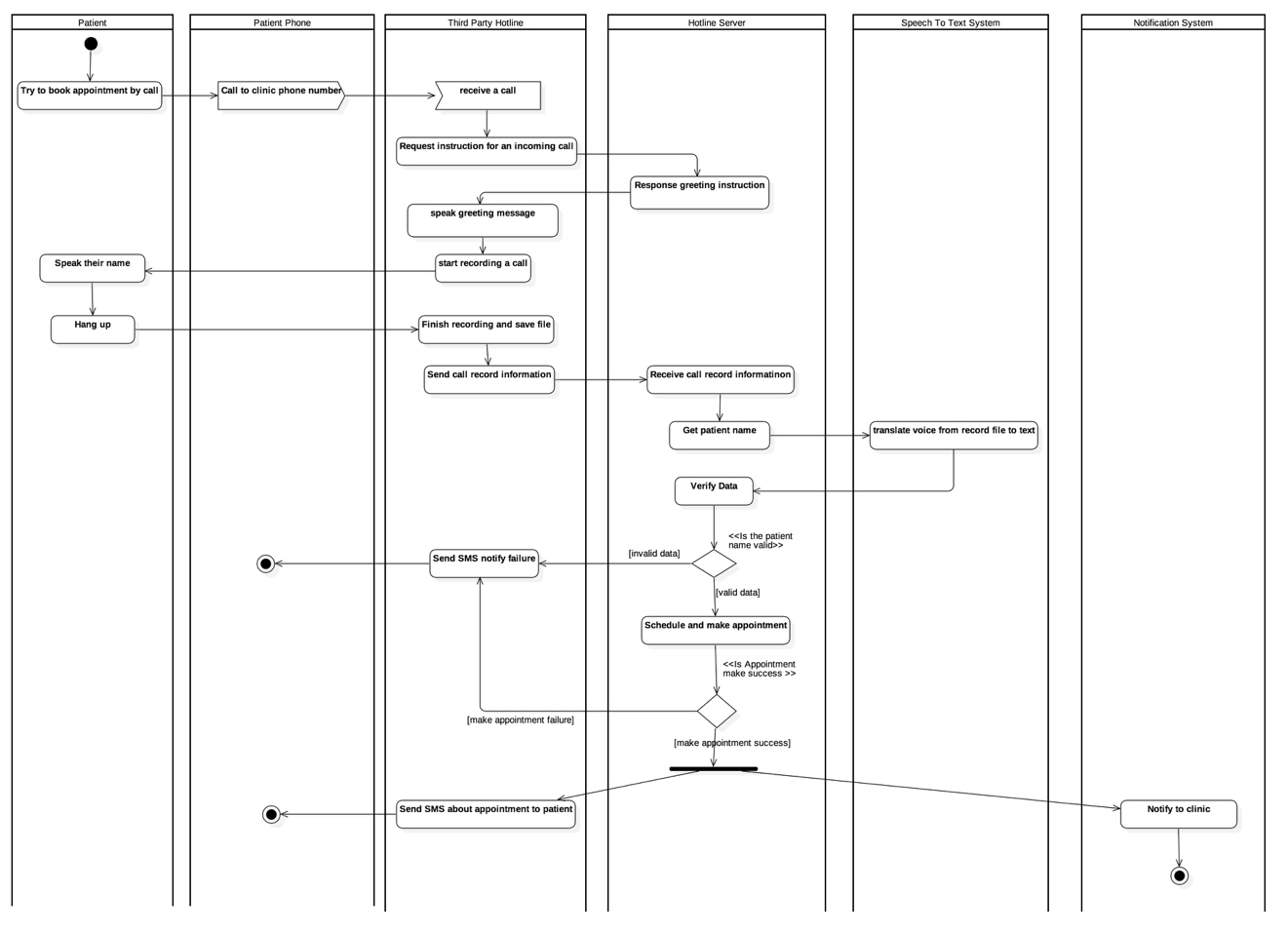
|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | void | Public | Set value of attribute |

### 4.3. Interaction Diagram

#### 4.3.1. Activity Diagram

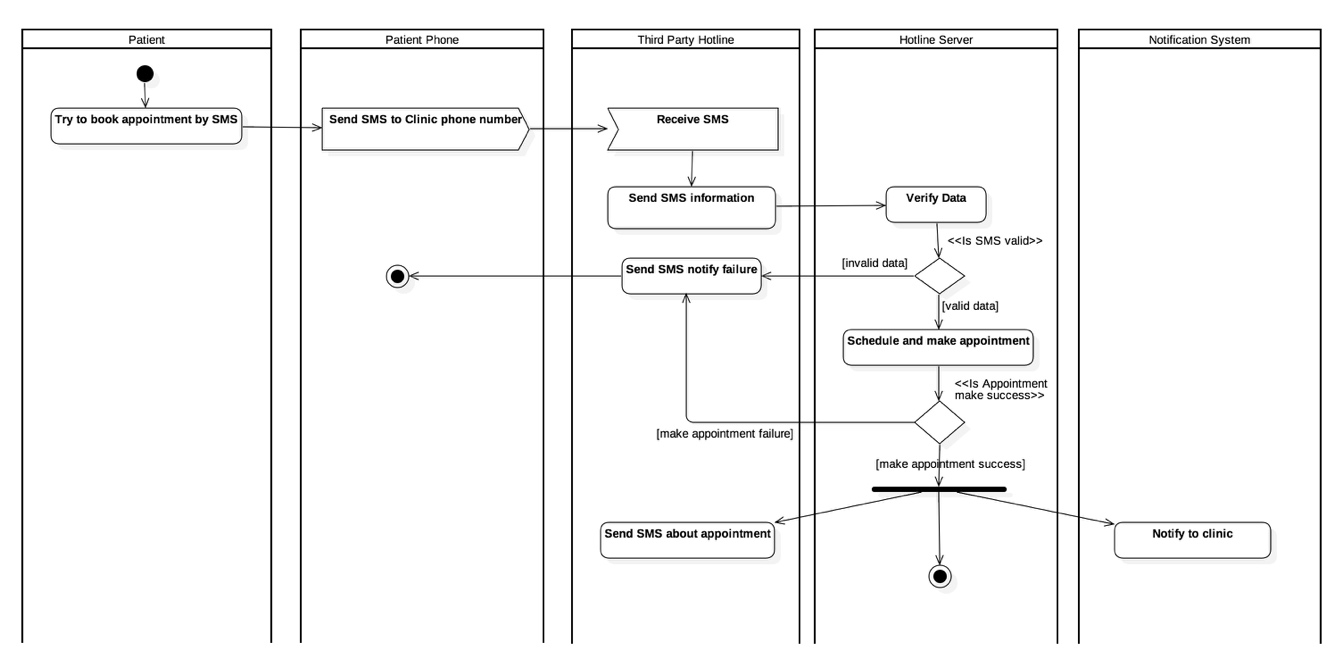
#### 4.3.1.1. Patient call to book appointment

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



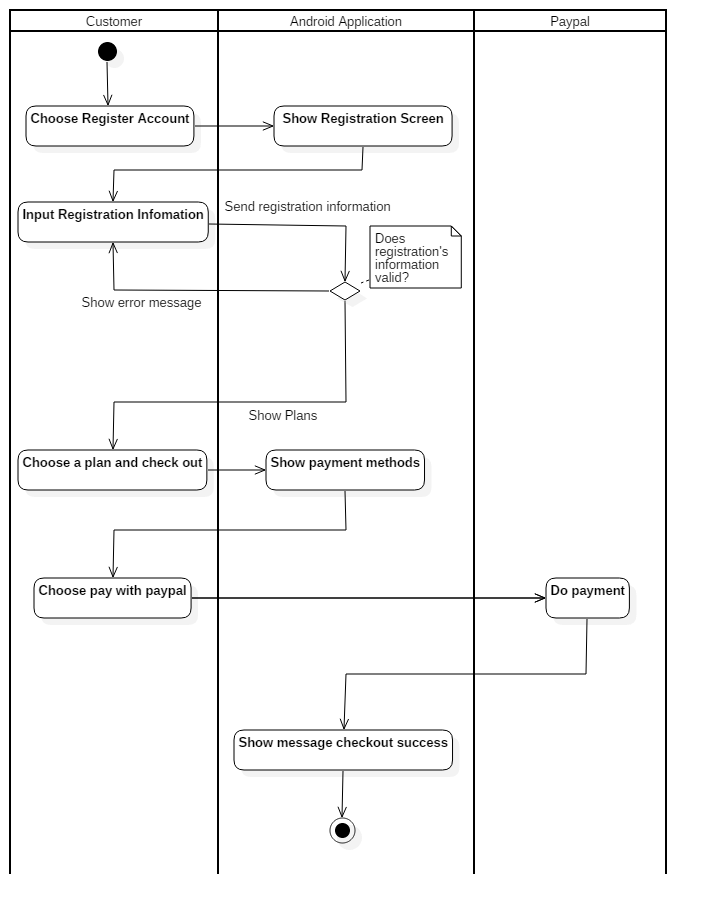
#### 4.3.1.2. Patient send SMS to book appointment

**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  } |

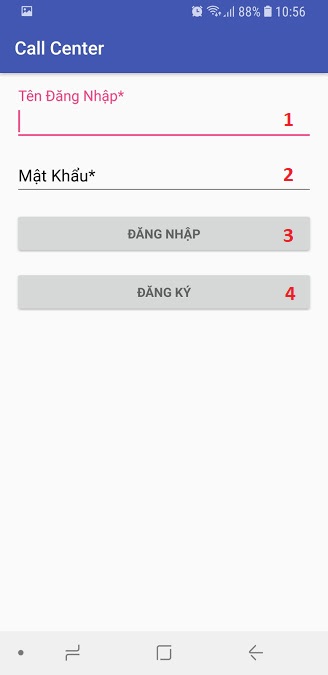
|  |  |
| --- | --- |
| **Name** | **Description** |
| status | value can be true or false to determine the request success processed or not |
| value | Json value corresponding to the request |
| error | contain error message content in case the request is not successes |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **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 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 is already exit 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 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. Login**

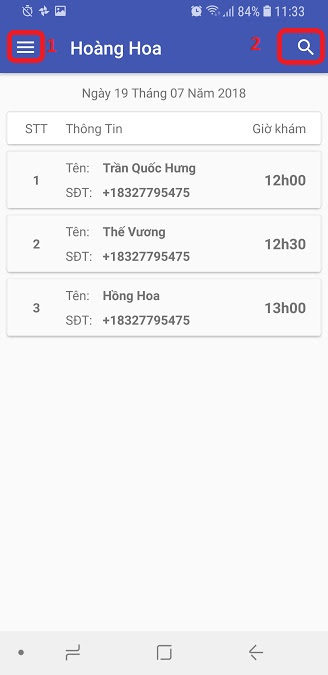
****

**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**

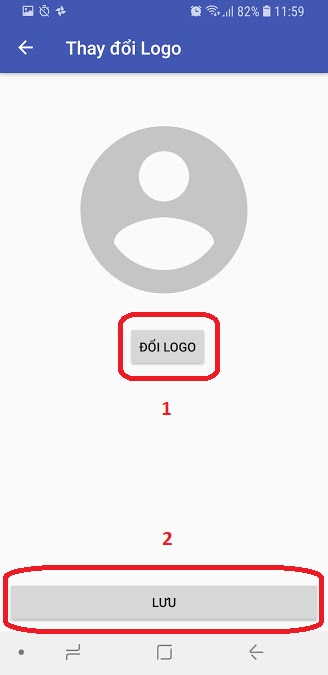
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 3 | Log In | Log In user to the system | All fields is meet the required length range | Perform login process |
| 4 | Sign up | Transfer to sign up screen | N/A | Transfer to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

** 5.2.1.2. 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 |

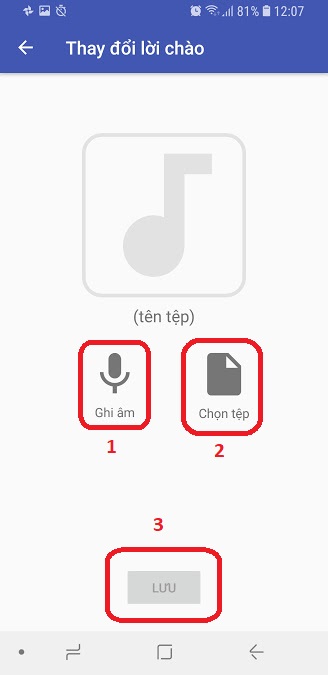
**5.2.1.3. Change profile Logo**

****

**Buttons**

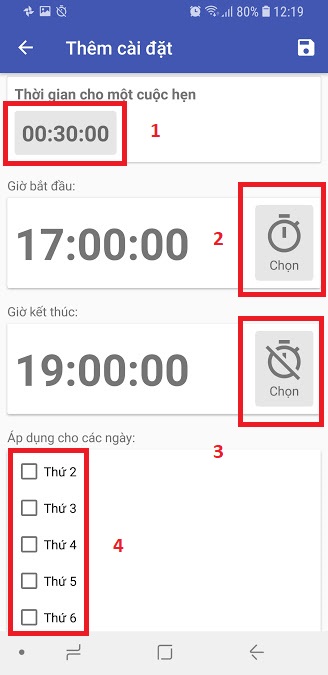
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Choose image | Choose an image file that will be the user’s profile logo | Image file’s size is less than 2Mb | Launch file picker |
| 2 | Save | Save the chosen image | An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

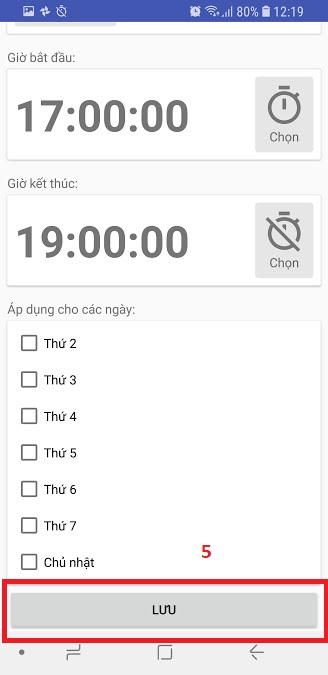
**5.2.1.4. Change greeting message**

****

**Buttons**

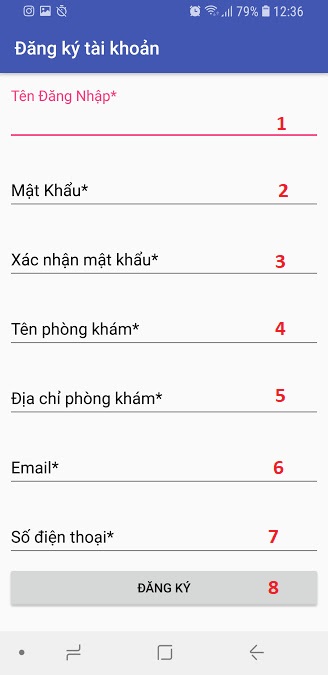
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Record greeting message | Allow user to record the greeting message | Record file size’s much less than 3Mb | Launch the voice record screen |
| 2 | Pick greeting message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

**5.2.1.4. 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 start working hour | Choose the time when clinic begin to work, the default value is 17:00:00 | The start working time much before 24:00:00 | Launch time picker |
| 3 | Choose end working hour | Choose the time when clinic is off work, the default value is 19:00:00 | The end working time much be after the start 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 saving process |

**5.2.1.5. Register**

**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 |
| 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 |

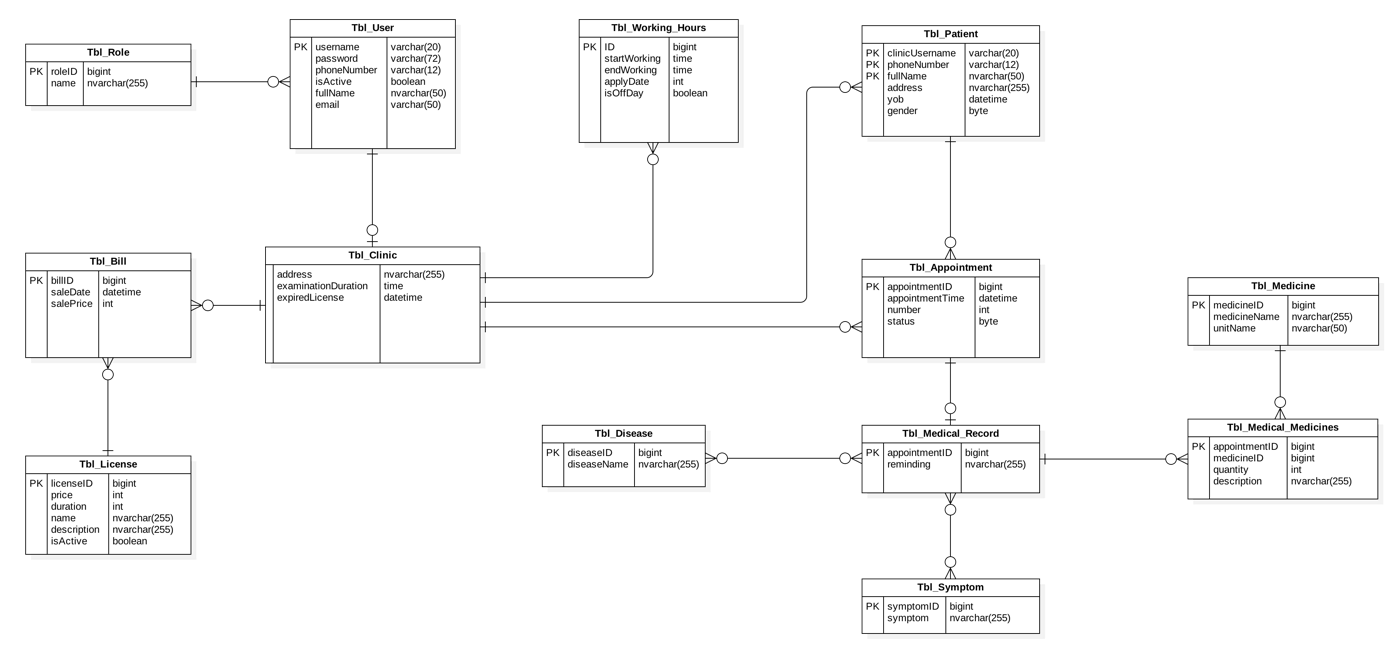
**5.2.1.6. Buy License**

****

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

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

## 7. Algorithms

### 7.1. Schedule appointment

#### 7.1.1. Definition

#### 7.1.2. Define problem

#### 7.1.3. Solution

# E. System Implementation & Test

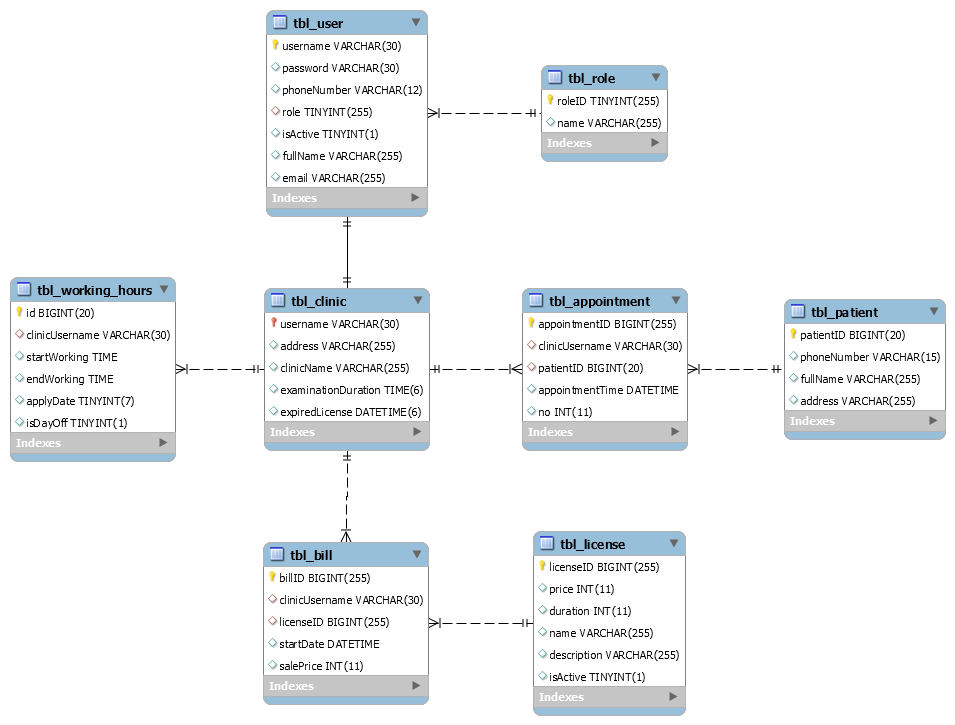
## 1. Introduction

### 1.1 Overview

### 1.2 Test Approach

## 2. Database Relationship Diagram

### 2.1 Physical Diagram



### Data Dictionary

|  |  |
| --- | --- |
|  | |
| **Table Name** | **Description** |
| tbl\_working\_hours | Table which contains the working hours information |
| tbl\_user | Table which contains the user information. |
| tbl\_clinic | Table which contains the clinic information. |
| tbl\_role | Table which contains the role information. |
| tbl\_appointment | Table which contains the appointment information |
| tbl\_patient | Table which contains the patient information |
| tbl\_bill | Table which contains the bill information |
| tbl\_license | Table which contains the license 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 clinic in one day of week. | TIME | No |
| EndWorking | The time closing of clinic in one day of week. | TIME | No |
| applyDate | The day of week. | TINYINT(7) | No |
| isDayOff | Status of that day of week, it allow know clinic opening or closing in that day. | TINYINT(1) | No |
| tbl\_user | Username{PK} | Each user will have one username , it use login to system. User maybe is administrator or clinic. | VARCHAR(30) | No |
| password | Each user would be having password. | VARCHAR(30) | No |
| phoneNumber | A telephone number of each user. | VARCHAR(12) | No |
| role{FK} | Role of user. Foreign key, role identifier. | TINYINT(255) | No |
| isActive | Status of user. | TINYINT(1) | No |
| fullName | Real name of user. | VARCHAR(255) | No |
| Email | Email of user. | VARCHAR(255) | No |
| tbl\_clinic | Username{PK}  {FK} | Unique identifier of member. Foreign key, user identifier. | VARCHAR(30) | No |
| address | Address of clinic. | VARCHAR(255) | No |
| clinicName | Name of clinic. | VARCHAR(255) | No |
| examinationDuration | The duration of one slot | TIME(6) | No |
| expiredLicense | Expiry date of 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 patient. Foreign key, patient identifier. | BIGINT(20) | No |
| appointmentTime | The time when appointment be created. | DATETIME | No |
| no | Number of order. | INT(11) | No |
| tbl\_patient | patientID{PK} | Unique identifier of patient, auto increment. | BIGINT(20) | No |
| phoneNumber | Phone number of patient. | VARCHAR(15) | No |
| fullName | A real name of 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 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 |

## 3. Performance Measures

## 4. Test Plan

### 4.1 Features to be tested

### 4.2 Features not to be tested

## 5. System Testing Test Case