ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

Công Nghệ Phần Mềm (CO3001)

Nhóm: L03.1 - Bài tập lớn 02 - Version 1.0

Software Design Document
Automotive Fixing Service

GVHD: Bùi Hoài Thắng
SV thực hiện: Trần Công Lực – 1511917
Nguyễn Minh Thám – 1613166
Nguyễn Nam Quân – 1512683
Nguyễn Xuân Nguyên – 1720037
Bùi Duy Hùng – 1511333
Tạ Văn Quang – 1512657

Tp. Hồ Chí Minh, Tháng 11/2017

Mục lục

1 Introduction

1.1 Purpose

Tài liệu này chứa đầy đủ mô tả thiết kế cho hệ thống "automotive fixing service". Bao gồm bản thiết kế chi tiết kiến trúc hệ thống, cách module code được thực hiện ra sao và cơ sở dữ liệu. Tài liệu cũng cung cấp cách thực hiện chi tiết các use cases trong tài liệu SRS.

1.2 Scope

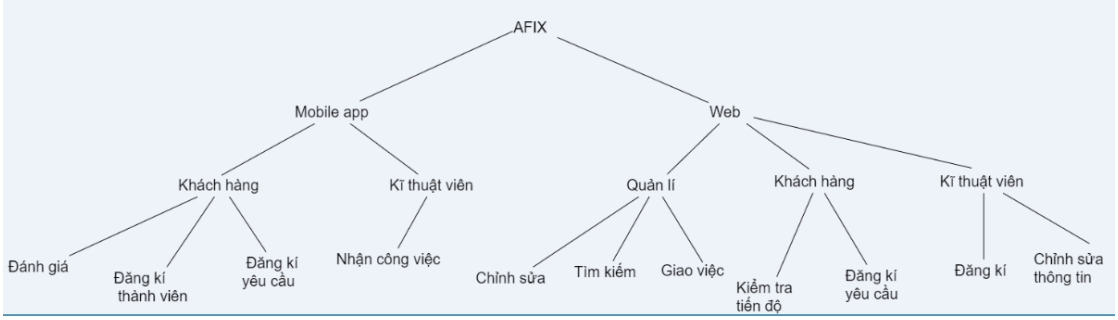
Hệ thống Automotive Fixing Service được thiết kế dựa trên đa nền tảng Web, IOS và Android. Các đối tượng chính sử dụng phần mềm bao gồm khách hàng (Customer), nhà quản lý (Manager) và kỹ thuật viên (Technician).
Đối với khách hàng, hệ thống cho phép người sử dụng có nhu cầu tìm kiếm kỹ thuật viên ở nhiều lĩnh vực khác nhau để giải quyết vấn đề mà mình đang gặp phải trong cuộc sống thường ngày. Yêu cầu của khách hàng được gửi thông qua nhiều cách khác nhau như điện thoại, email hoặc đăng ký trực tiếp trên hệ thống Website.
Đối với kỹ thuật viên,hệ thống lưu trữ tất cả các thông tin của họ để dễ dàng trong việc quản lý. Hệ thống được thiết kế cho phép kỹ thuật viên có thể tìm được những yêu cầu của khách hàng phù hợp với trình độ chuyên môn của mình.
Bằng cách tối đa hóa hiệu quả sắp xếp, lọc và quản lý tiến trình độ ưu tiên các danh sách yêu cầu của khách hàng. Qua đó người chịu công tác quản lý tác vụ có thể dễ sử dụng, dễ dàng chọn ra những kỹ thuật viên phù hợp với yêu cầu của từng khách hàng.

1.3 Overview of Document

Phần tiếp theo là phần mô tả một cách chi tiết kiến trúc và chức năng của hệ thống dựa vào các yêu cầu về Functional và Non-Functional và ràng buộc của hệ thống.

2 Architecture Design

2.1 Decomposition



Hình 1: *Decomposition*

2.2 Method Design

2.2.1 Khách hàng

Name: Web Service
Type: Web page
Node:
Description:
Attribute: none
Resources:
Operations:

Name: CheckRequest()
Arguments: không
Returns: List Request
Description: Khách hàng điền tên (full name) vào textbox "Họ và Tên" và Email hoặc số điện thoại vào textbox "Email or phone number" sau đó ấn button "check". List request sẽ hiển thị. Khách hàng nhấn chọn request mà mình muốn xem thông tin.
Pre-condition: khách hàng đang ở homepage.
Post-condition: bảng thông tin request được hiển thị cho khách hàng.
Exceptions: khách hàng điền sai thông tin tra cứu.
Flow and events:

1. form tra cứu request được nối tới.
2. form cung cấp searchbox để khách hàng điền thông tin tra cứu
3. form này gọi hàm SearchRequest() hàm này sẽ lọc ra các request trong database phù hợp với thông tin tra cứu và hiển thị kết quả.
4. Khách hàng chọn 1 request trong list để xem thông tin.

5. form này gọi hàm GetRequestInfo() để hiển thị thông tin request

6. Trong bảng thông tin Request khách hàng tìm đến thang đánh giá (chỉ xuất hiện khi Request đã hoàn thành) và chấm điểm vào đó rồi bấm "ok"

7. Form này gọi hàm GetRequestRating()

Name: CheckRequest

Type: Form

Node: WebServer

Description: Form này cho phép tìm kiếm Request bằng Tên (Họ và Tên) và Email (Hoặc số Điện thoại)

Attributes: tên Khách hàng , Email (Hoặc số điện thoại)

Resources: Không

Operation:

Name: SearchRequest()

Argument: Không

Return: List các Request

Description: lọc ra các Request được lưu trong database phù hợp với thông tin tìm kiếm.

Pre-condition: đang ở CheckRequest Form

Post-condition: khách hàng ấn button "check"

Exceptions: Có thể cancel

Flow of Events:

1. Form cung cấp các textbox để điền thông tin tìm kiếm và button "check"

2. khách hàng điền đúng thông tin để tìm kiếm

Name: SearchRequest()

Argument: tên khách hàng và Email (số Điện thoại)

Return: List các Request

Description: dùng từ khóa khách hàng nhập vào để tìm kiếm Request từ database.

Pre-condition: đang ở CheckRequest Form

Post-condition: List kết quả được hiển thị

Exceptions: Có thể cancel

Flow and Events:

1.Hiển thị kết quả tìm kiếm được ra form đã gọi.

2.Khách hàng chọn một Request

Name: GetRequestInfo()

Argument: Không

Return: Bảng thông tin request

Description: Hiển thị bảng thông tin chi tiết của Request.

Pre-condition: Đang ở CheckRequest Form

Post-condition: Bảng thông tin Request được hiển thị

Exceptions: Có thể cancel

Flow and Events:

1. Hiển thị bảng thông tin Request ra form CheckRequest

Name: GetRequestRating()

Argument: Không

Return: Mức đánh giá request

Description: Thu thập điểm đánh giá của Request.

Pre-condition: Đang ở CheckRequest Form

Post-condition: Khách hàng đã đánh giá request

Exceptions: Có thể cancel

Flow and Events:

1. Khách hàng đánh giá Request.
2. Khách hàng nhấn button "ok" để hoàn tất đánh giá.

Name: isSendRequest

Arguments: active(kiểu string).

Returns: không.

Description:

+ Khi bạn chọn nút Send Request thì acitve = "show" được truyền vào hàm. Cho phép hiện thị form yêu cầu

+ Khi bạn chọn nút Send thì acitve = "send" được truyền vào hàm. Cho phép gửi yêu cầu dịch vụcủa bạn

Pre-condition: Khi bạn chọn nút Send Request hoặc Send

Post-condition: active ="show" hoặc active = "send"

Mục tham khảo: III 2 2.2 Gửi yêu cầu

Pseudocode:

IF active == "show "

Gọi hàm displayRequest();

Return;

ELSE IF active =="send"

Gọi hàm checkRequest();

Return;

ELSE return;

ENDIF

Name: displayRequest

Arguments: không
Returns: không.
Description: Khi hàm được gọi sẽ hiện các thông tin cần điền như tên, mail, mật khẩu,số điện thoại,địa chỉ nhà,thời gian gửi yêu cầu, thời gian thực hiện dịch vụ để khách hàng có thể điền thông tin.
Pre-condition: Khi hàm isRequest được gọi
Post-condition: khi tham số hàm isRequest là active = "show"
Mục tham khảo: III 2 2.2 Gửi yêu cầu
Pseudocode:

Hiện thị tên, mail, mật khẩu,số điện thoại,địa chỉ nhà,thời gian gửi yêu cầu, thời gian thực hiện dịch vụ để khách hàng có thể điền thông tin.

Name: setCustomer{}
Arguments: _name,_email,_password,_address,_phone,_service, _timeStartService, _timeStartSend
Returns: không.
Description: Khi hàm được gọi sẽ lưu các thông tin người dùng nhập vào các biến thành viên của class SendRequest
Pre-condition: Khi truyền đầy đủ tham số
Post-condition: không
Mục tham khảo: III 2 2.2 Gửi yêu cầu
Pseudocode:

Name=_name;
Email=_email;
Password=_password;
Address=_address;
Phone=_phone;
timeStartService=_ timeStartService;
timeStartSend=_ timeStartSend;

Name: checkRequest
Arguments: không có
Returns: kiểu bool.
Description: Khi hàm được gọi sẽ kiểm tra các thông tin người dùng nhập đã đầy đủ chưa, có đúng định dạng hay không
Pre-condition: đã có giá trị trong các biến thành viên của class
Post-condition: không
Mục tham khảo: III 2 2.2 Gửi yêu cầu

Pseudocode:

IF name hoặc email hoặc password hoặc address hoặc phone hoặc timeStartSend hoặc timeStartService trống

return false;

ENDIF

IF email không đúng định dạng (@gmail.com,...)

Return false;

ENDIF

Return true;

Name: saveRequest
Arguments: check (kiểu bool)
Returns: không.
Description: Tham số Check là giá trị trả về của hàm checkRequest. Khi truyền giá trị true thì hàm sẽ lưu yêu cầu của khách hàng vào database. Còn nếu truyền giá trị false thì hàm không làm gì hết
Pre-condition: đã có giá trị trong các biến thành viên của class
Post-condition: không
Mục tham khảo: III 2 2.2 Gửi yêu cầu
Pseudocode:

IF check == true

lưu các thông tin name,mail,password,address, phone, timeStartService,timeStartSend vào database

ENDIF

Return;

Name: notice
Arguments: check (kiểu bool)
Returns: không.
Description: Tham số Check là giá trị trả về của hàm checkRequest. Khi truyền giá trị true thì hàm sẽ thông báo khách hàng gửi yêu cầu thành công. Còn nếu truyền giá trị false thì hàm sẽ thông báo khách hàng "Hãy điền đầy đủ thông tin"
Pre-condition: có tham số truyền vào
Post-condition: không
Mục tham khảo: III 2 2.2 Gửi yêu cầu
Pseudocode:

IF check == true

Thông báo "Khách hàng đã gửi yêu cầu thành công"

ELSE
Thông báo "Khách hàng hãy điền đầy đủ thông tin"
ENDIF

Name: loadInfomation
Arguments: không
Returns: không.
Description: Nếu khách hàng đang đăng nhập thì load các thông tin khách hàng vào form đăng ký
Pre-condition: đã đăng nhập khách hàng
Post-condition: không
Mục tham khảo: III 2 2.2 Gửi yêu cầu
Pseudocode:
setRequest(getCustomer());

2.2.2 Quản lí

Name: Manager
Description: Đây là class chứa các thuộc tính của người quản lí và hàm được thực hiện bởi người quản lí.
Attributes: tên Quản lí , Email, số điện thoại
Operation:

Name: AssignTaskToTech()
Arguments : none
Return : none
Description : button có chức năng giao công việc cho kỹ thuật viên.
Pre-condition : đã chọn 1 request trong request list.
Post-condition : none.
Exception : none.
Flow of Events:

1. Manager chọn 1 request
2. Manager chọn button "Giao công việc" (AssignTaskToTechButton())
3. AssignTaskToTech() mở ra bảng các kỹ thuật viên.

Name : sendRequestToTech()
Arguments : none
Return : gửi công việc đó cho Kỹ thuật viên.
Description :
Pre-condition : sau khi chọn các kỹ thuật viên và chọn "Phân công"

Post-condition : các kỹ thuật viên được phân công nhận được thông báo.
Exception : none.
Flow and events

1. Manager chọn 1 hoặc nhiều kỹ thuật viên.
2. Manager chọn "Phân công"
3. Gửi thông báo đến các kỹ thuật viên được chọn.

Name: Search_request()
Argument: Không
Return: Danh sách các Request
Description: Quản lí tìm kiếm 1 yêu cầu trong request database.
Pre-condition: hàm Search_request() phải được chọn và request database phải sẵn sàng
Post-condition: Danh sách các yêu cầu được chọn hiển thị trên màn hình.
Exceptions: Quản lí có thể hủy bỏ bất cứ lúc nào mà không cần hiển thị lên màn hình.
Flow and Events:

1. Hệ thống gọi display_text_box() để hiển thị 1 text box để gõ keyword vào.
2. Hệ thống gọi keyword_query() để hiện thị danh sách các request theo thứ tự ID có keyword xuất hiện trong những giá trị thuộc tính của request.
3. Quản lí chọn 1 request từ danh sách và có thể remove, add, update hoặc hủy bỏ hoạt động tìm kiếm.

Name: Remove_request()
Arguments: Không
Return: Không
Description: Quản lí cập nhật lại danh sách các yêu cầu bằng cách xóa một vài request trong request database
Pre-condition: Hàm remove_request() phải được chọn và request database phải sẵn sàng.
Pos-condition: Danh sách các yêu cầu được cập nhật mới
Exception: Quản lí có thể hủy bỏ cập nhật bất cứ lúc nào
Flow and Events:

1. Request database được truy cập tới và gọi display() để hiển thị lên màn hình dưới dạng pull-down menu.
2. Quản lí chọn 1 hoặc nhiều request từ danh sách
3. Hệ thống nhắc lại khẳng định xóa hay không.
4. Quản lí xác nhận xóa.
5. Hệ thống xóa các request trong request database

Name: Update_request()

Arguments: Không
Return: Không
Description: Quản lí cập nhật lại danh sách các yêu cầu bằng cách cập nhật một vài thuộc tính của request trong request database
Pre-condition: Hàm update_request() phải được chọn và request database phải sẵn sàng.
Pos-condition: Danh sách các yêu cầu được cập nhật mới
Exception: Quản lí có thể hủy bỏ cập nhật bất cứ lúc nào
Flow and Events:

1. Request database được truy cập tới và gọi display() để hiển thị lên màn hình dưới dạng pull-down menu.
2. Quản lí chọn 1 request từ danh sách
3. Hệ thống gọi display_form() để hiển thị form với tất cả thông tin về request.
4. Quản lí chỉnh sửa lại các thông tin đã chọn.
5. Hệ thống cập nhật thông tin mới vào request database.

Name: Add_request()
Arguments: Không
Return: Không
Description: Quản lí thêm 1 yêu cầu trong request database
Pre-condition: Form của yêu cầu là 1 cửa sổ động, hàm add_request() phải được chọn và request database phải sẵn sàng
Pos-condition: Danh sách các yêu cầu được cập nhật mới
Exception: Quản lí có thể hủy bỏ cập nhật bất cứ lúc nào
Flow and Events:

1. Hệ thống gọi display_form() để hiển thị form của 1 request .
2. Quản lí điền đầy đủ các thông tin vào form và nộp lại.
3. Hệ thống thêm 1 request vào trong request database .

Name: UserInterface
Description: Đây là class chứa các thành phần và phương thức để hiển thị những phẩn tử trên giao diện người dùng..
Attributes: btn: List<Button>, text: List<TextView>, img: List<Img>, menu: Menu, ...
Operation:

Name: Display()
Arguments: Không
Return: Không
Description: Hiển thị các các phần tử có trên giao diện
Pre-condition:

Pos-condition: Các phần tử được hiện lên trên màn hình giao diện
Exceptions: Không
Flow and Events:

1. Hệ thống gọi display_text_box() để hiển thị 1 text box để gõ keyword vào.
2. Hệ thống gọi keyword_query() để hiện thị danh sách các request theo thứ tự ID có keyword xuất hiện trong những giá trị thuộc tính của request.
3. Quản lí chọn 1 request từ danh sách và có thể remove, add, update hoặc hủy bỏ hoạt động tìm kiếm.

Name: Request
Description: Đây là class chứa các thuộc tính và phương thức của 1 request.
Attributes: name, ID, phone, time, catagory, time, customer ...
Operation:

Name: Display()
Arguments: Không
Return: Không
Description: Hiển thị đối tượng Request lên giao diện
Pre-condition: Hàm display() được gọi
Pos-condition: Đối tượng Request được hiện lên trên màn hình giao diện
Exceptions: Không
Flow and Events:

1. Hệ thống thực hiện hiển thị các thuộc tính của Request lên giao diện.

Name: Update()
Arguments: Không
Return: Không
Description: Thực hiện cập nhật mới các thuộc tính trên Request Database
Pre-condition: Hàm update() được gọi
Pos-condition: Thuộc tính tương ứng với đối tượng Request được cập nhật trên database
Exceptions: Không
Flow and Events:

1. Thực hiện tạo kết nối với Request Database
2. Thay đổi thuộc tính tương ứng trên Request Database.

Name: Remove()
Arguments: Không
Return: Không
Description: Xóa 1 entry ra khỏi Request Database

Pre-condition: hàm Remove() được gọi
Pos-condition: Cập nhật lại database khi có 1 entry bị xóa
Exceptions: Không
Flow and Events:

1. Thực hiện tạo kết nối với Request Database
2. Xóa 1 entry tương ứng trên Request Database.

Name: Add()
Arguments: Không
Return: Không
Description: thêm 1 entry vào Request Database
Pre-condition: hàm Add() được gọi
Pos-condition: Cập nhật lại database khi có 1 entry thêm vào
Exceptions: Không
Flow and Events:

1. Thực hiện tạo kết nối với Request Database
2. Thêm 1 entry tương ứng vào Request Database

Name: Request Form
Description: Dây là class chứa thuộc tính và phương thức của 1 request form, form để người dùng có thể nhập thông tin của 1 request vào.
Attributes: txt: List<TextBox>, txtv : List<TextView>, request: Reques
Operation:

Name: Send()
Arguments: Không
Return: Không
Description: thêm 1 entry vào trong RequestDatabase
Pre-condition: Hàm send() được gọi
Pos-condition: Đối tượng Request được hiện lên trên màn hình giao diện
Exceptions: Không
Flow and Events:

1. Tạo một đối tượng Request mới và truyền vào các thuộc tính tương ứng trên request form
2. Từ đối tượng mới gọi hàm Add() để thêm 1 entry vào Request Database.

Name: Display_form()
Arguments: Không
Return: Không

Description: Hiện thị 1 form rỗng cho người dùng có thể nhập liệu vào
Pre-condition: Hàm display_form() được gọi và form có thể chỉnh sửa được.
Pos-condition: Hiển thị 1 form rỗng lên trên màn hình giao diện
Exceptions: Không
Flow and Events:

1. Hệ thống hiện thị form request rỗng lên trên màn hình giao diện

2.2.3 Kĩ thuật viên

Name: searchNameTect()
Arguments: String
Returns: Danh sách các tác vụ
Description: Kỹ thuật viên chọn vào tên một kỹ sư nằm trong danh sách đưa ra của hệ thống. Một danh sách các tác vụ do người kỹ sư được chọn phụ trách sẽ hiện ra, thông qua đó kỹ thuật viên chọn một tác vụ mà mình cần, tác vụ đó sẽ xuất hiện các thông tin chi tiết của nó.
Pre-condition: Kỹ thuật viên phải truy cập vào hệ thống tìm kiếm và chọn tìm kiếm theo tên kỹ sư.
Post-condition: Di chuyển tới trang thông tin chi tiết của tác vụ được chọn.
Exceptions: Kỹ thuật viên có thể rời khõi quá trình tìm kiếm bất cứ lúc nào.
Stimulus: Kỹ thuật viên select vào "Tìm kiếm theo tên".
Flow of Events:

1. Form tìm kiếm theo tên được liên kết đến để lấy dữ liệu người dùng
2. Dữ liệu từ SearchView gửi đến Controller gọi hàm **searchNameTect(string)**
3. Controller gọi **queryTechnicians()** ở Model trả về danh sách các kỹ sư lấy ra trong database và trả về View hiển thị theo thứ tự bảng chữ cái từ trên xuống dưới
4. Kỹ thuật viên lựa chọn tên một kỹ sư từ danh sách trên.
5. Dữ liệu được chọn gửi đến Controller, gọi hàm **getListTask(int)**
6. Controller gọi **queryListTask(int)** ở Model để lấy những task thuộc quản lý của kỹ sư đã được chọn ở bước step 4 dựa vào id của kỹ sư đó trong database và trả về View hiển thị danh sách các tác vụ cần tìm.
7. Kỹ thuật viên lựa chọn tác vụ mà mình muốn tìm kiếm.
8. Dữ liệu được chọn gửi đến Controller, gọi hàm **getTask(int)**
9. Controller gọi hàm **queryTask(int)** ở Model để lấy chi tiết tác vụ dựa vào id của nó và trả về View hiển thị cho người dùng.

Name: searchCategory()
Arguments: String
Returns: Danh sách các tác vụ.

Description: Kỹ thuật viên chọn danh mục muốn tìm kiếm các task nằm trong danh mục đó.
Pre-condition: Kỹ thuật viên phải truy cập vào hệ thống tìm kiếm và chọn tìm kiếm theo danh mục cụ thể.
Post-condition: Di chuyển tới trang thông tin chi tiết của tác vụ được chọn.
Exceptions: Kỹ thuật viên có thể rời khỏi quá trình tìm kiếm bất cứ lúc nào.
Stimulus: Kỹ thuật viên select vào "Tìm kiếm theo danh mục".
Flow of Events:

1. Form tìm kiếm theo danh mục được liên kết đến, chọn danh mục cần tìm kiếm.
2. Dữ liệu từ lựa chọn người dùng gửi đến Controller gọi hàm **searchCategory(string)**.
3. Controller gọi **queryListTask(string)** ở Model để lấy những task thuộc cùng một danh mục đã chọn ở bước 2 trong database và trả về View hiển thị danh sách các tác vụ cần tìm.
4. Kỹ thuật viên lựa chọn tác vụ mà mình muốn tìm kiếm.
5. Dữ liệu được chọn gửi đến Controller, gọi hàm **getTask(int)**
6. Controller gọi hàm **queryTask(int)** ở Model để lấy chi tiết tác vụ dựa vào id của nó và trả về View hiển thị cho người dùng.

Name: searchKeyWord()
Arguments: String
Returns: Danh sách các tác vụ.
Description: Kỹ thuật viên nhập từ khóa tìm kiếm.
Pre-condition: Kỹ thuật viên phải truy cập vào hệ thống tìm kiếm và chọn tìm kiếm theo từ khóa.
Post-condition: Di chuyển tới trang thông tin chi tiết của tác vụ được chọn.
Exceptions: Kỹ thuật viên có thể rời khỏi quá trình tìm kiếm bất cứ lúc nào. Hoặc từ khóa tìm kiếm không tồn tại trong hệ thống hoặc nhập không hợp lệ.
Stimulus: Kỹ thuật viên select vào "Tìm kiếm theo từ khóa".
Flow of Events:

1. Form tìm kiếm theo danh mục được liên kết đến, nhập từ khóa cần tìm kiếm.
2. Dữ liệu từ lựa chọn người dùng gửi đến Controller gọi hàm **searchKeyWord (string)**.
3. Controller gọi **queryListTask(string)** ở Model để tìm kiếm những task có từ khóa gần giống hoặc chính xác ở bước 2 trong database và trả về View hiển thị danh sách các tác vụ cần tìm.
4. Kỹ thuật viên lựa chọn tác vụ mà mình muốn tìm kiếm.
5. Dữ liệu được chọn gửi đến Controller, gọi hàm **getTask(int)**
6. Controller gọi hàm **queryTask(int)** ở Model để lấy chi tiết tác vụ dựa vào id của nó và trả về View hiển thị cho người dùng.

Name: Xác nhận tác vụ

Arguments:
Returns:
Description: Kỹ thuật viên tiến hành xác nhận tác vụ được giao bởi Manager
Pre-condition: Truy cập vào hệ thống và xác nhận khi có thông báo đến
Post-condition: Tác vụ được gán cho kỹ thuật viên hoặc trả về Manager như một tác vụ mới.
Exceptions: Kỹ thuật viên có thể xác nhận thông báo, trong vòng 10 phút hệ thống phải xử lý
Stimulus:
Flow of Events:

Name: btnListenerTaskDetail(int id)
Arguments: int
Returns: Thông tin chi tiết của tác vụ có mã là id
Description: Lắng nghe sự kiện kỹ thuật viên nhất nút xem chi tiết tác vụ
Pre-condition: Tác vụ được giao mới hiện trong bảng thông báo
Post-condition: Chuyển qua trang Task Detail
Exceptions: Kỹ thuật viên có thể thoát bất cứ lúc nào.
Stimulus: Thực hiện click vào nút "Chi tiết"
Flow of Events:
1. Kỹ thuật viên click vào button, gọi hàm btnListenerTaskDetail(int) ở View
2. Dữ liệu theo phương thức Get chuyển đến Controller gọi hàm getTask(int)
3. Controller gọi hàm queryTask(int) ở Model để lấy chi tiết của tác vụ dựa vào id của nó và trả về View hiện thị cho kỹ thuật viên.

Name: btnListenerAcceptTask(int id_task, int id_tect)
Arguments: int, int
Returns:
Description: Lắng nghe sự kiện kỹ thuật viên nhất nút chấp nhận tác vụ được giao
Pre-condition: Tác vụ được giao mới hiện trong bảng thông báo hoặc kỹ thuật viên truy cập được vào trang chi tiết tác vụ.
Post-condition: Thông báo thành công
Exceptions: Có thể kỹ thuật viên không được giao task nhưng lại nhấn nút chấp nhận trong trường hợp tự ý truy cập vào trang chi tiết tác vụ bằng hình thức tìm kiếm.
Stimulus: Thực hiện click vào nut "Chấp nhận"
Flow of Events:
1. Kỹ thuật viên click vào button, gọi hàm btnListenerAcceptTask(int, int)
2. Dữ liệu chuyển đến Controller, gọi hàm handleAcceptTask(int, int)
3. handleAcceptTask(int, int) gọi hàm getTask(int) lấy đối tượng CTask cần chỉnh sửa
4. Controller gọi hàm queryTask(int) ở Model để thấy thông tin chi tết của đối tượng và

trả về Controller

5. Controller khi có được đối tượng tác vụ đó, gọi hàm queryUpdateTask(CTask) ở Modal để thêm thông tin id của kỹ thuật viên chịu trách nhiệm tác vụ này.

Name: btnListenerRefuse(int id)
Arguments: int
Return: boolean
Description: Lắng nghe sự kiện kỹ thuật viên nhấn nút từ chối tác vụ được giao
Pre-condition: Tác vụ được giao mới hiện trong bảng thông báo hoặc kỹ thuật viên truy cập được vào trang chi tiết tác vụ.
Post-condition: Thông báo từ chối thành công, trả về cho manager
Exceptions: Kỹ thuật viên không có quyền từ chối tác vụ đó.
Stimulus: Thực hiện click vào nút "Từ chối"
Flow of Events:
1. Kỹ thuật viên click vào button "Từ chối" gọi hàm **btnListenerRefuse(int)**
2. Dữ liệu được gửi đến Controller, gọi hàm **handleRefuseTask(int)**
3. Controller gọi hàm **queryUpdateStatus(int, int)** ở Model để thay đổi trạng thái cho tác vụ có dựa vào id của nó.
4. Trả về kết quả thành công hay thất bại cho Controller.

2.2.4 Khách hàng

Name: isSignUp{}
Arguments: active(kiểu string).
Returns: không.
Description:
+ Khi bạn chọn nút Registration thì acitve = "show" được truyền vào hàm. Cho phép hiện thị form đăng ký
+ Khi bạn chọn nút Sign Up thì acitve = "send" được truyền vào hàm. Cho phép gửi yêu cầu đăng ký của bạn
Pre-condition: Khi bạn chọn nút Registration hoặc Sign Up
Post-condition: active ="show" hoặc active = "send"
Mục tham khảo: III 2 2.1 Đăng ký thành viên
Pseudocode:

```
IF active == "show "
    Gọi hàm displaySignUp(); Return;
ELSE IF active =="send"
    Gọi hàm checkInfomatioon(); Return;
ELSE return; ENDIF
```

Name: displaySignUp{}
Arguments: không
Returns: không.
Description: Khi hàm được gọi sẽ hiện các thông tin cần điền như tên, mail, mật khẩu,số điện thoại,địa chỉ nhà để khách hàng có thể điền thông tin.
Pre-condition: Khi hàm isSignUp được gọi
Post-condition: khi tham số hàm isSignUp là active = "show"
Mục tham khảo: III 2 2.1 Đăng ký thành viên
Pseudocode:

Hiện thị tên,mail, mật khẩu, số điện thoại, địa chỉ nhà để khách hàng có thể điền thông tin.

Name: setCustomer
Arguments: _name,_email,_password,_address,_phone
Returns: không.
Description: Khi hàm được gọi sẽ lưu các thông tin người dùng nhập vào các biến thành viên của class SignUpCustomer
Pre-condition: Khi truyền đầy đủ tham số
Post-condition: không
Mục tham khảo: III 2 2.1 Đăng ký thành viên
Pseudocode:

Name=_name;
Email=_email;
Password=_password;
Address=_address;
Phone=_phone;

Name: checkInfomation
Arguments: không có
Returns: kiểu bool.
Description: Khi hàm được gọi sẽ kiểm tra các thông tin người dùng nhập đã đầy đủ chưa, có đúng định dạng hay không
Pre-condition: đã có giá trị trong các biến thành viên của class
Post-condition: không
Mục tham khảo: III 2 2.1 Đăng ký thành viên
Pseudocode:

IF name hoặc email hoặc password hoặc address hoặc phone trống
return false;

ENDIF
IF email không đúng định dạng (@gmail.com,...)
Return false;
ENDIF
Return true;

Name: checkInfomation
Arguments: check (kiểu bool)
Returns: không.
Description: Tham số Check là giá trị trả về của hàm checkInfomation. Khi truyền giá trị true thì hàm sẽ lưu thông tin của khách hàng vào database. Còn nếu truyền giá trị false thì hàm không làm gì hết
Pre-condition: đã có giá trị trong các biến thành viên của class
Post-condition: không
Mục tham khảo: III 2 2.1 Đăng ký thành viên
Pseudocode:
IF check == true
lưu các thông tin name,mail,password,address,phone và database
ENDIF
Return;

Name: notice
Arguments: check (kiểu bool)
Returns: không.
Description: Tham số Check là giá trị trả về của hàm checkInfomation. Khi truyền giá trị true thì hàm sẽ thông báo khách hàng đăng ký thành công. Còn nếu truyền giá trị false thì hàm sẽ thông báo khách hàng "Hãy điền đầy đủ thông tin"
Pre-condition: có tham số truyền vào
Post-condition: không
Mục tham khảo: III 2 2.1 Đăng ký thành viên
Pseudocode:
IF check == true
Thông báo "Khách hàng đã đăng ký thành công"
ELSE
Thông báo "Khách hàng hãy điền đầy đủ thông tin"
ENDIF

3 Data Structure Design

3.1 Request Table

Field	Type	Description
Họ và Tên khách hàng	Text	Tên khách hàng
Email khách hàng	Text	Email của khách hàng
Số Điện thoại Khách hàng	Integer	Số Điện thoại của khách hàng
Địa chỉ khách hàng	Text	Địa chỉ của khách hàng
Tên kỹ thuật viên	Text	Tên của kỹ thuật viên nhận task
ID của kỹ thuật viên	Integer	ID của kỹ thuật viên
ID của Request	Integer	Id của Request
Loại dịch vụ	Text	Tên của dịch vụ mà khách hàng yêu cầu
Mức độ	Text	Mức độ cần thiết của công việc
Ngày yêu cầu	Date	Thời gian gửi yêu cầu của request
Status	Text	Tình trạng thực hiện
Ngày hoàn thành	Date	Thời gian hoàn thành task của kỹ thuật viên (bỏ trống nếu status không phải là "đã hoàn thành")
Rating	Float	Điểm đánh giá của khách hàng về mức độ hài lòng với dịch vụ của công ty (bỏ trống nếu status không phải là "đã hoàn thành")

3.2 Customer Table

Field	Type	Description
Họ và Tên	Text	Tên khách hàng
Email khách hàng	Text	Email của khách hàng
Mật khẩu	Text	Mật khẩu dùng để đăng nhập của khách hàng
Số Điện thoại Khách hàng	Integer	Số Điện thoại của khách hàng
Địa chỉ khách hàng	Text	Địa chỉ của khách hàng
Yêu cầu	Pointer	Những yêu cầu mà khách hàng đã gửi

3.3 Technician Table

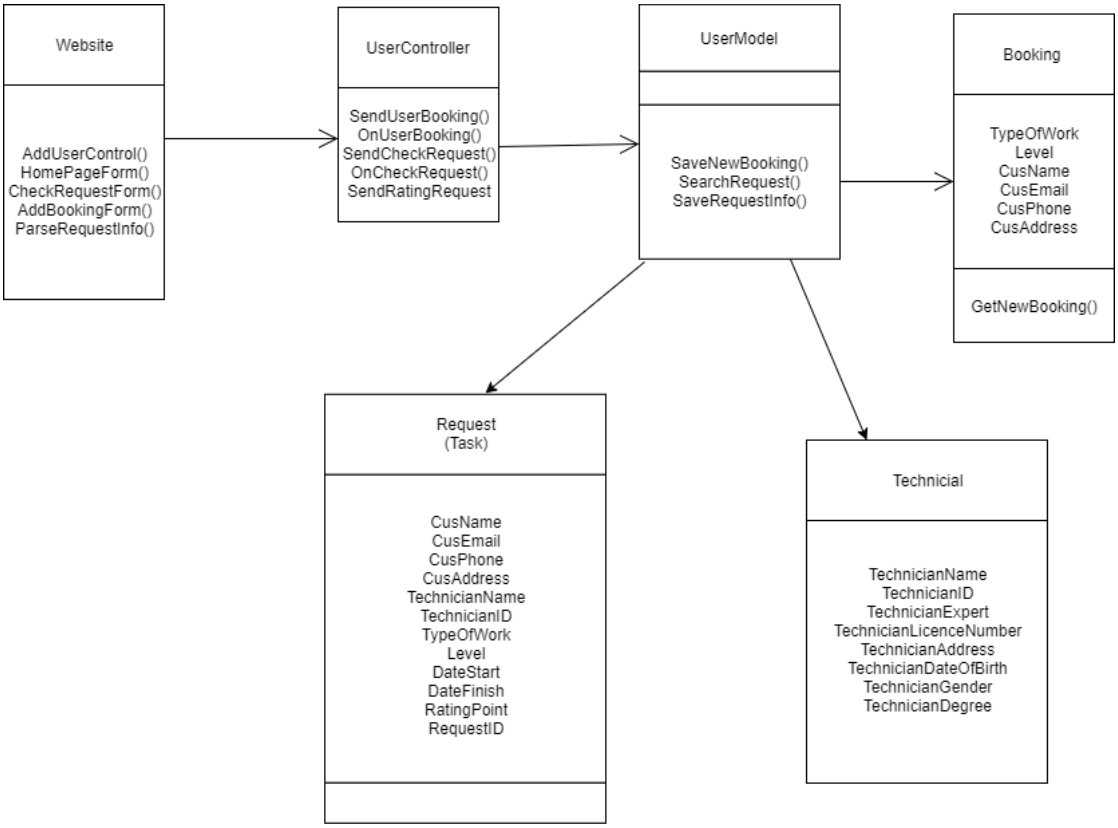
Field	Type	Description
Name	Text	Tên kỹ thuật viên
Id	Integer	ID của kỹ thuật viên
DOB	Date	Ngày tháng năm sinh
Address	String	Địa chỉ của kỹ thuật viên
Degrees	List<CDegree>	Kiểu tự định nghĩa, lưu trữ thông tin các loại bằng cấp của kỹ thuật viên
Work_exp	List<CWorkExp>	Kiểu tự định nghĩa, lưu trữ thông tin về kinh nghiệm làm việc của kỹ thuật viên
report_tasks	List<CReportTask>	Kiểu tự định nghĩa, lưu trữ thông tin về báo cáo tác vụ của kỹ thuật viên

3.4 Task Table

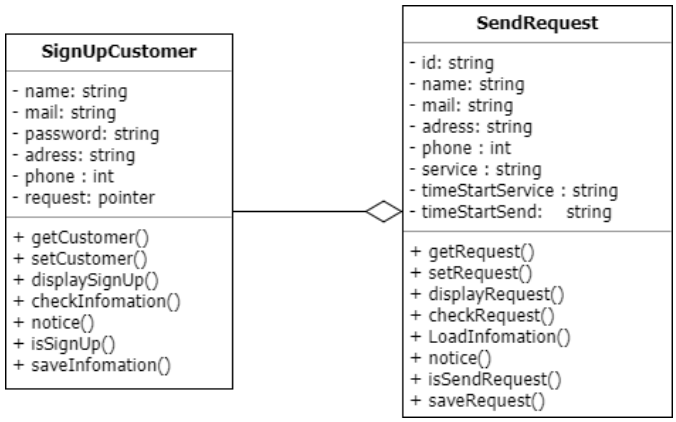
Field	Type	Description
ID	Integer	ID của tác vụ
Service	String	Mô tả tác vụ thuộc loại dịch vụ nào
pre_predition	Date	Điều kiện thông báo trước về tác vụ (Rất quan trọng, quan trọng, khẩn cấp, bình thường, ...)
Location	String	Địa chỉ thực hiện tác vụ
Phone	String	Số điện thoại của người dùng
Email	String	Email của người dùng
technician_name	String	Tên của kỹ thuật viên phụ trách
technician_id	Integer	ID của ky thuật viên
time_free	Date	Thời gian rãnh có thể bắt đầu thực hiện tác vụ
time_request	Date	Thời gian yêu cầu
category	String	Phân loại tác vụ

4 Class Diagram

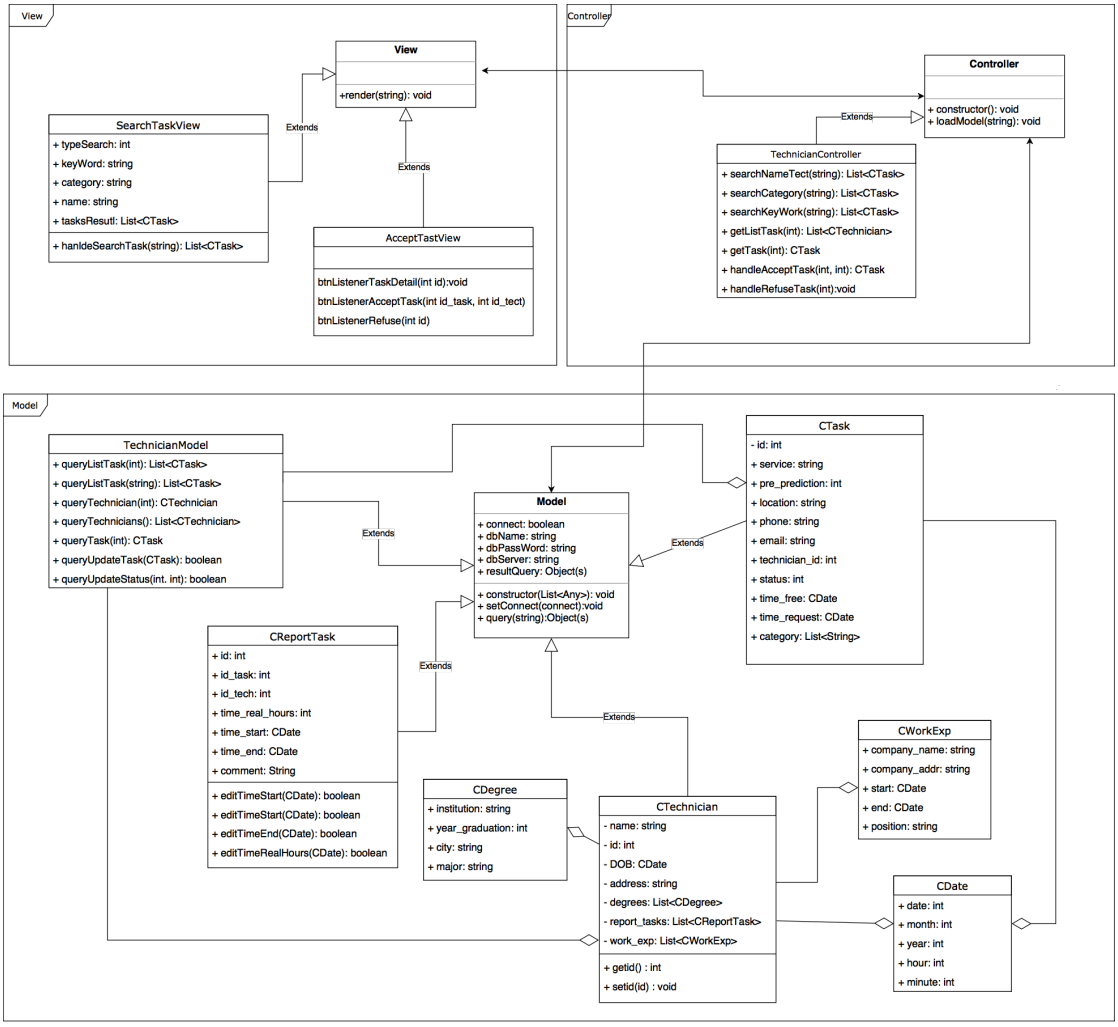
Website Class: được dùng để hiển thị thông tin cho người dùng, nhận các yêu cầu của người dùng để gửi đến Controller
UserController: dùng để gửi yêu cầu đến Model và xử lý hiển thị ra lại Class Website
Booking: Tạo Booking mới
Request: Lưu trữ dữ liệu Request
Technician: Lưu trữ dữ liệu của Kỹ thuật viên



Hình 2: *Class Diagram Xử lý yêu cầu người dùng*



Hình 3: *Class Diagram Đăng ký và gửi yêu cầu*



Hình 4: *Class Diagram Kỹ thuật viên và Tác vụ*

5 User Interface Design

- Phần giao diện trên web có logo của hệ thống như là hình nền. Có 1 thông điệp chào mừng ở trên top, bên dưới thông điệp là 1 tab với các nút có chức năng như: Về sản phẩm, Liên hệ, Trang chủ. Có phần đăng nhập ở phía trên bên phải. Sau khi đăng nhập thì đối với từng đối tượng sẽ có các giao diện khác nhau:

    1. Quản lí: Sẽ có 2 nút để chọn đó là Kiểm tra hoặc là giao việc. Nếu chọn Kiểm tra thì sau đó trang web sẽ hiển thị danh sách các yêu cầu đang cần được xử và 2 nút cập nhật hoặc xóa. Nếu chọn cập nhật thì trang web sẽ hiện thị tiếp 2 nút là cập nhật và thêm cho quản lí lựa chọn; nếu cập nhật thì hiển thị danh sách các yêu cầu và có nút cập nhật còn lại thì hiển thị form để điền vào và nút submit. Nếu chọn xóa thì trang

web sẽ hiện thị danh sách yêu cầu và nút xóa.

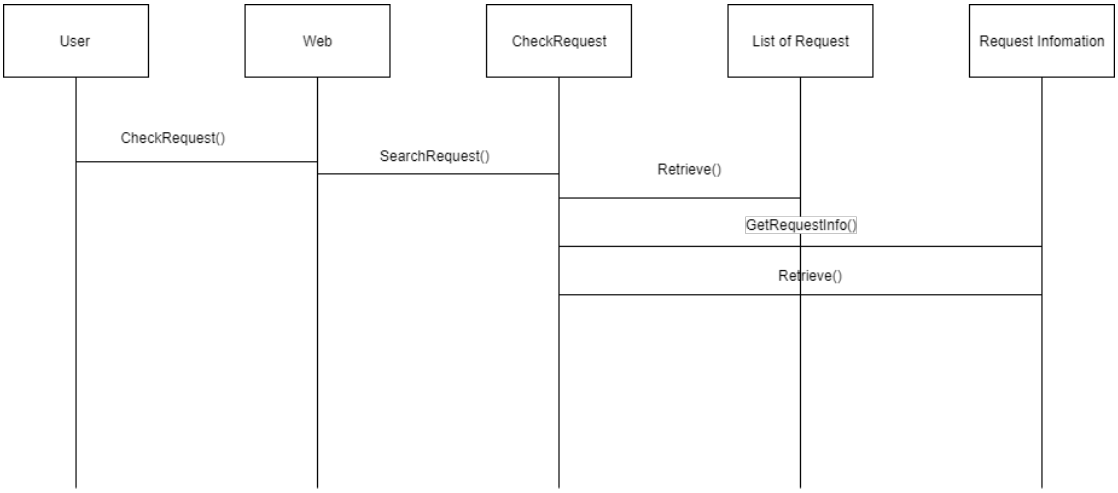
2. Khách hàng: Sẽ có 2 nút để chọn đó là Đăng kí yêu cầu mới và Kiểm tra tiến độ. Nếu chọn Đăng ký thì trang web hiển thị 1 form rỗng để điền vào và 1 nút submit. Còn nếu chọn Kiểm tra tiến độ thì sẽ hiện ra danh sách các yêu cầu đang được thực cho khách hàng và kèm theo đó là tiến độ đang hoạt động như thế nào.
3. Kĩ thuật viên: Sẽ có 2 nút để chọn là Đăng kí hoặc chỉnh sửa thông tin. Nếu chọn Đăng kí thì trang web sẽ hiện thị 1 form trống và 1 nút submit. Nếu chọn chỉnh sửa thì sẽ hiển thị profile của kĩ thuật viên dưới dạng có thể chỉnh sửa được và 1 nút submit.

- Phần giao diện trên App mobile: Màn hình chính sẽ có logo của hệ thống và các ô đăng nhập. Sau khi đăng nhập, tùy vào tài khoản là khách hàng hay kĩ thuật viên thì hệ thống sẽ chuyển sang màn hình tương ứng:

  1. Khách hàng: Hiển thị tiến độ của những yêu cầu đang được thực hiện cho khách hàng và kèm theo đó sẽ có 1 tab bar gồm 5 ngôi sao cho khách hàng có thể đánh giá cho dịch vụ. Trên cùng màn hình chính sẽ có 2 nút, 1 cho phần đăng kí thành thành viên, 1 cho phần đăng kí yêu cầu mới. Cả 2 để sẽ hiện thị form để điền vào trong màn hình mới, nhưng nội dung sẽ khác nhau.
  2. Kĩ thuật viên: Hiển thị danh sách các yêu cầu chưa có người thực hiện ở chính giữa màn hình, còn bên trái trên có 1 nút cho thông báo. Nếu nhấn vào nút đó, 1 màn hình mới sẽ đè lên màn hình hiện tại, trong màn hình đó có danh sách yêu cầu được gửi cho kĩ thuật viên giải quyết và 2 nút Từ chối/Đồng ý.

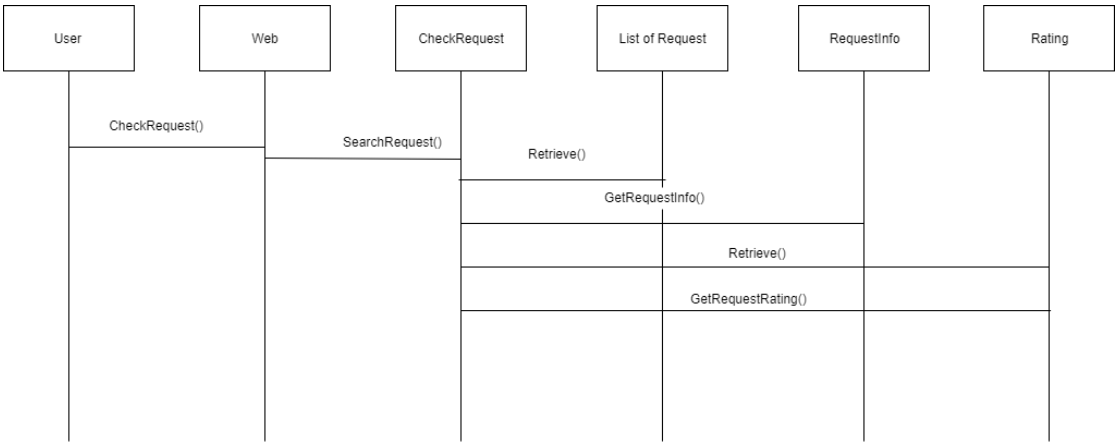
6 Use Case Realizations

6.1 Check Request



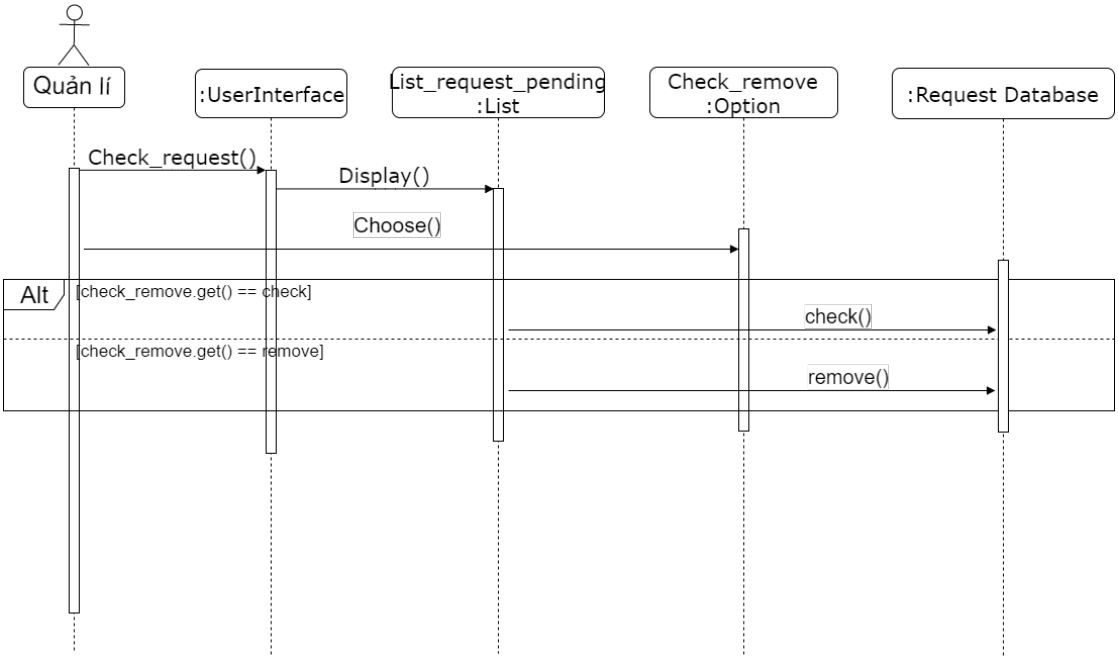
Hình 5: *Use case Check Request - Xref: SRS 2.4*

6.2 Rating



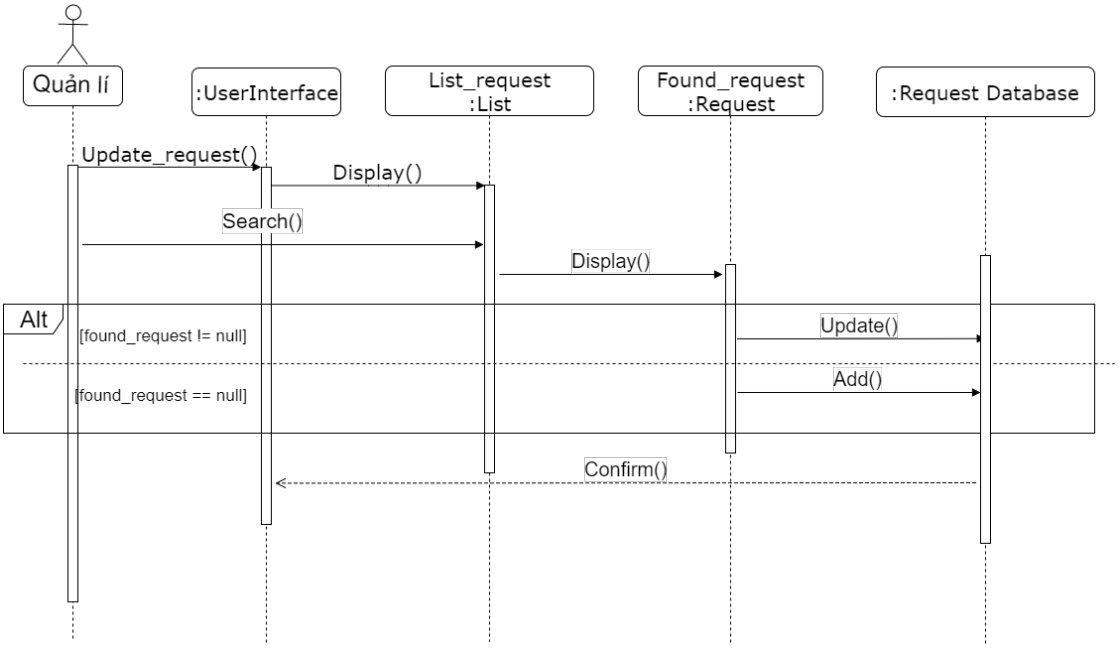
Hình 6: *Use case Rating - Xref: SRS 2.3*

6.3 Kiểm tra yêu cầu



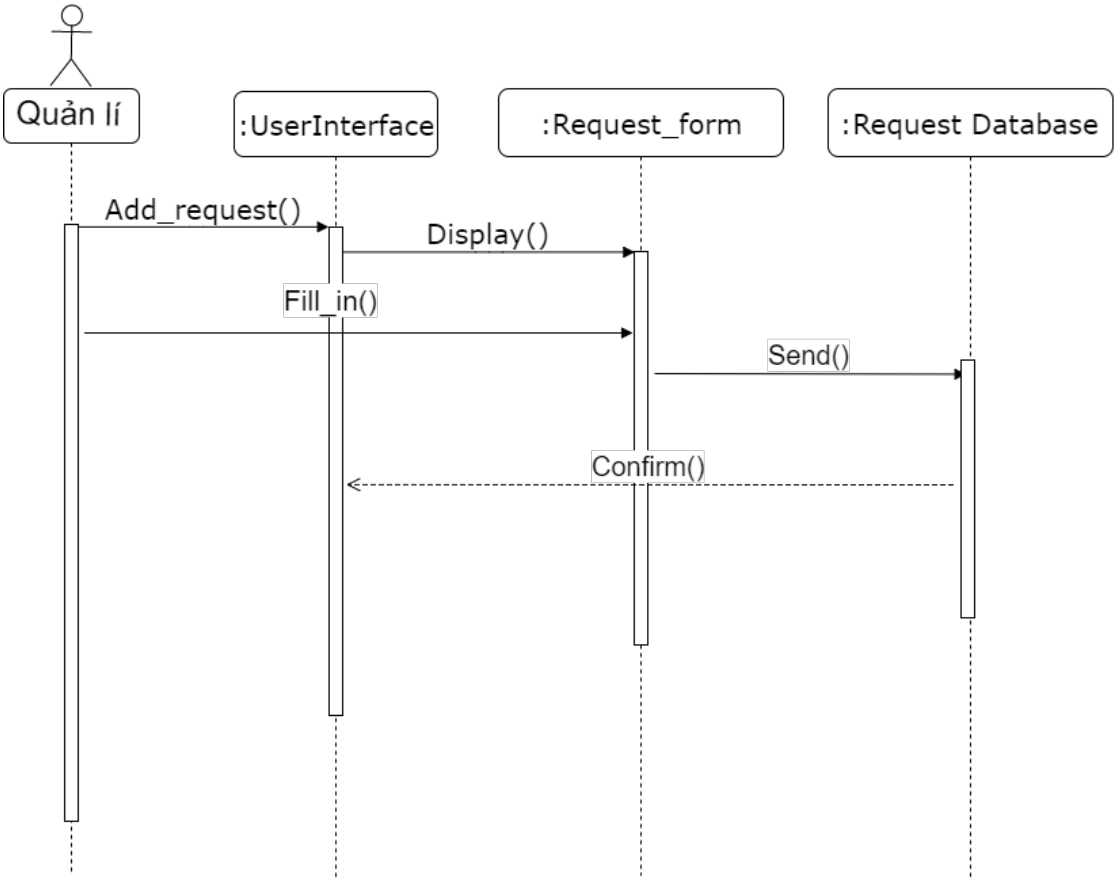
Hình 7: *Use case Kiểm tra yêu cầu - Xref: SRS 2.6*

6.4 Cập nhật yêu cầu



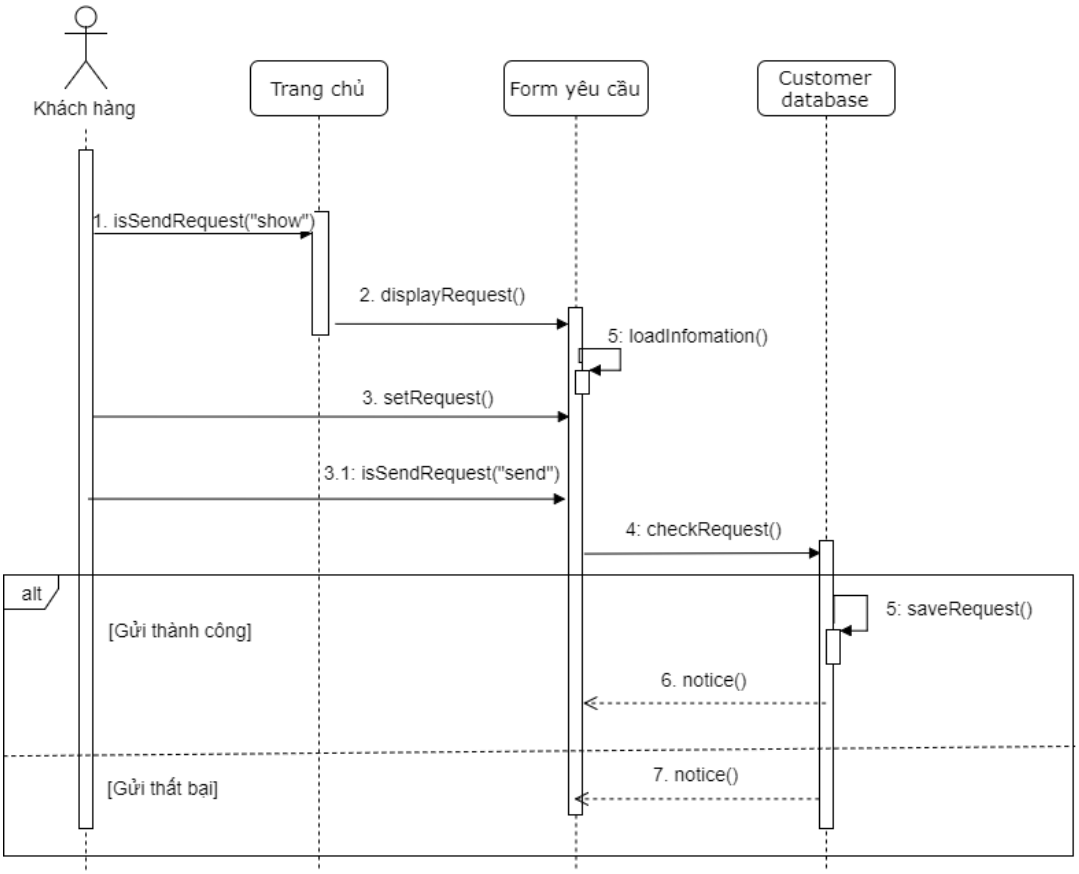
Hình 8: *Use case Cập nhật yêu cầu - Xref: SRS 2.7*

6.5 Thêm yêu cầu



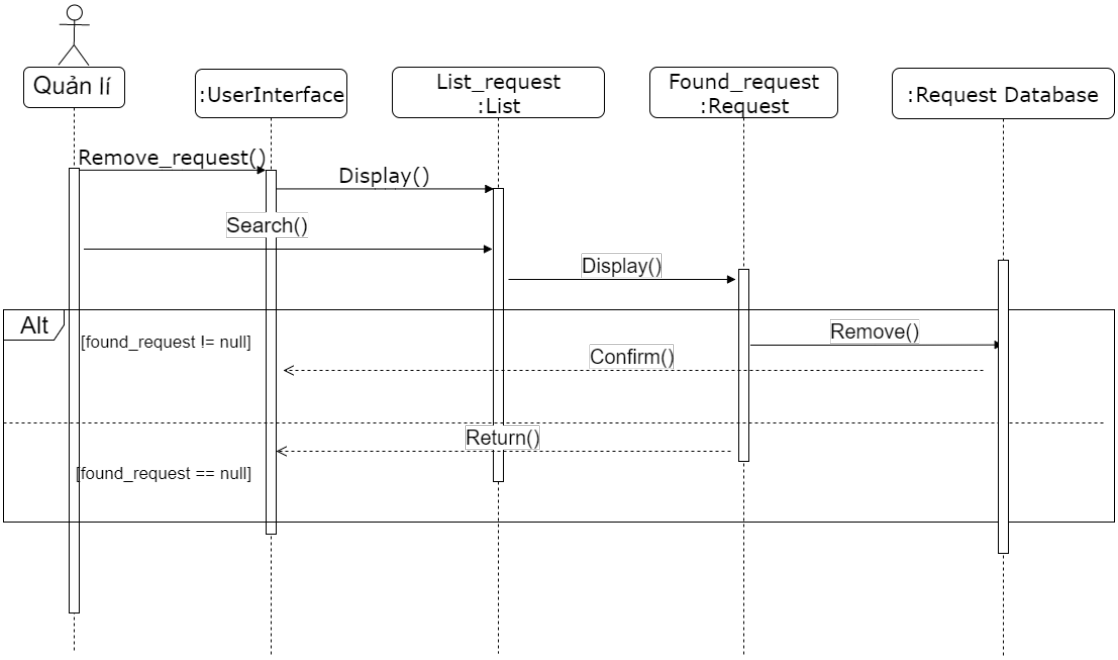
Hình 9: *Use case Thêm yêu cầu - Xref: SRS 2.8*

6.6 Gửi yêu cầu



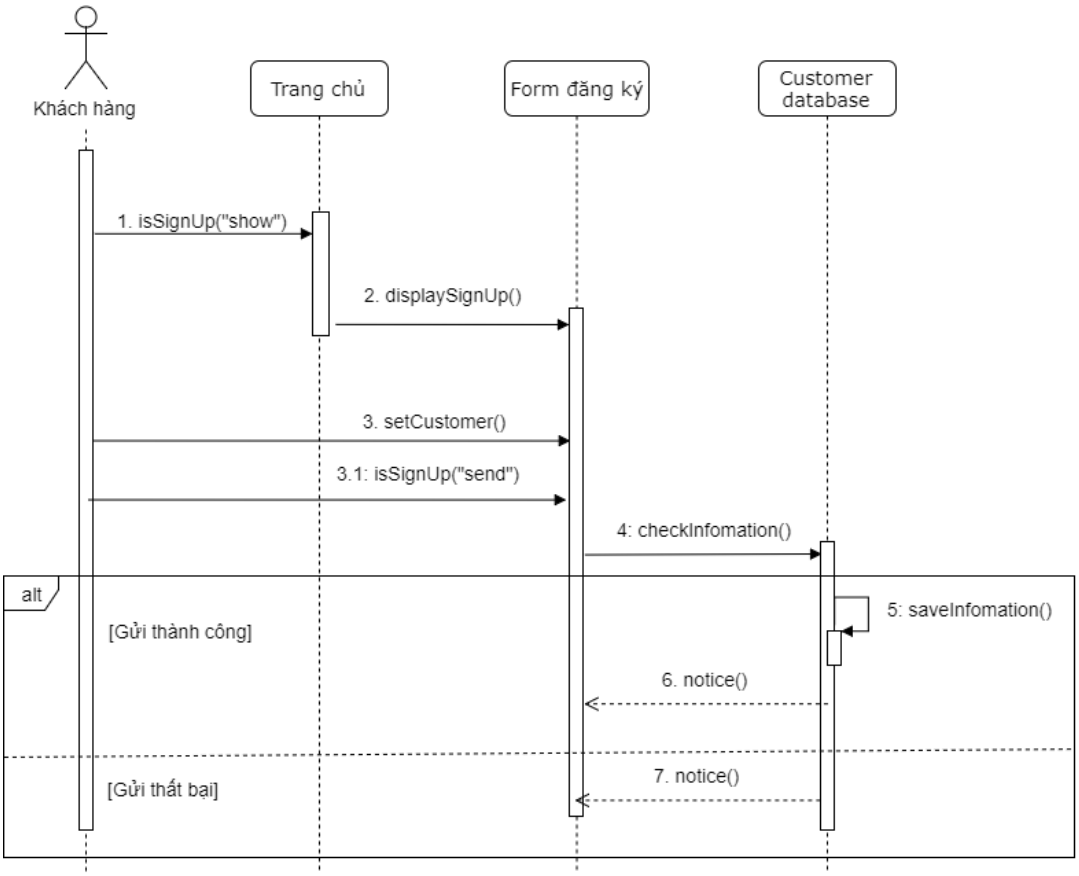
Hình 10: *Use case Gửi yêu cầu - Xref: SRS 2.2*

6.7 Xóa yêu cầu



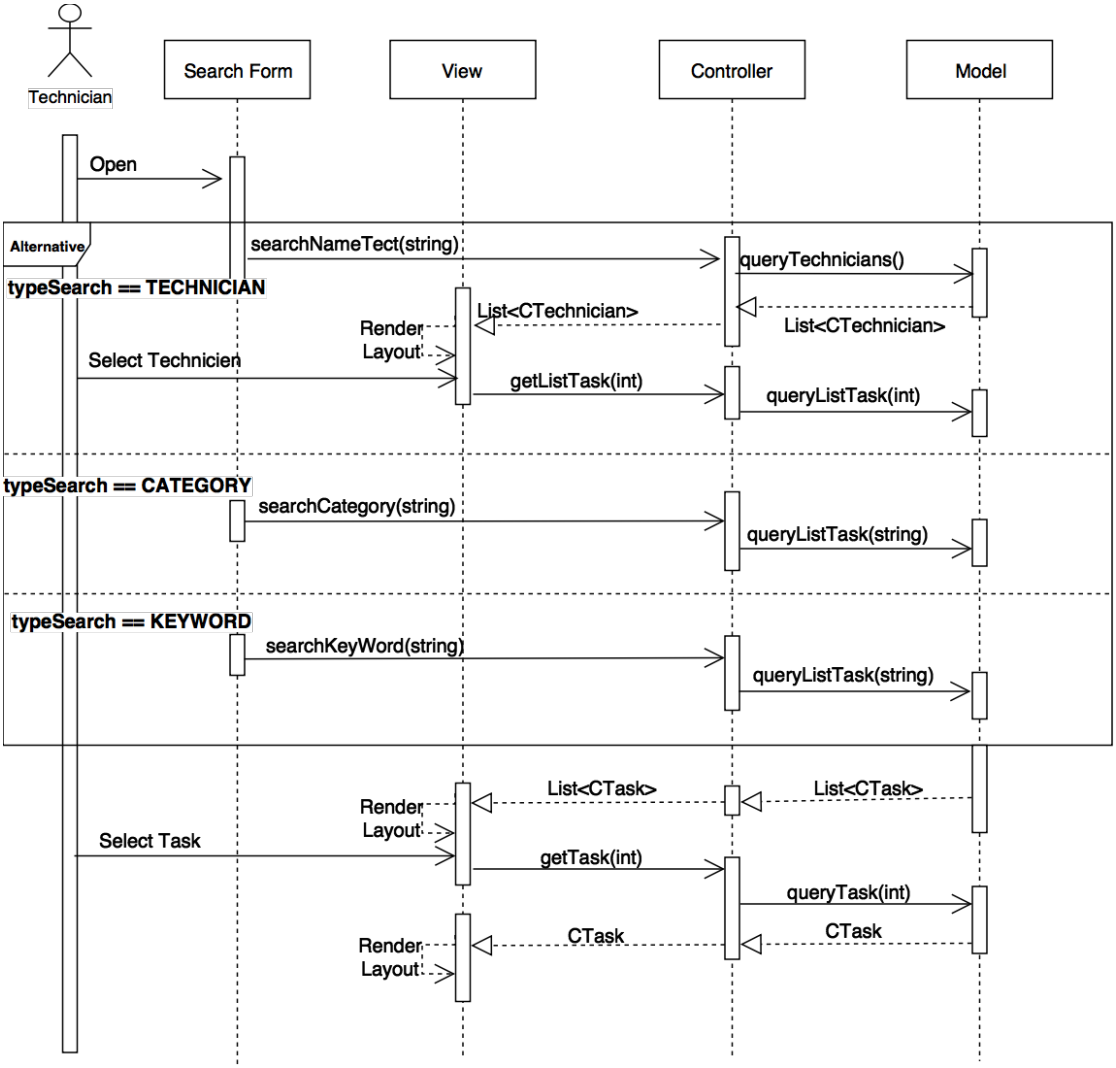
Hình 11: *Use case Xóa yêu cầu - Xref: SRS 2.9*

6.8 Đăng ký



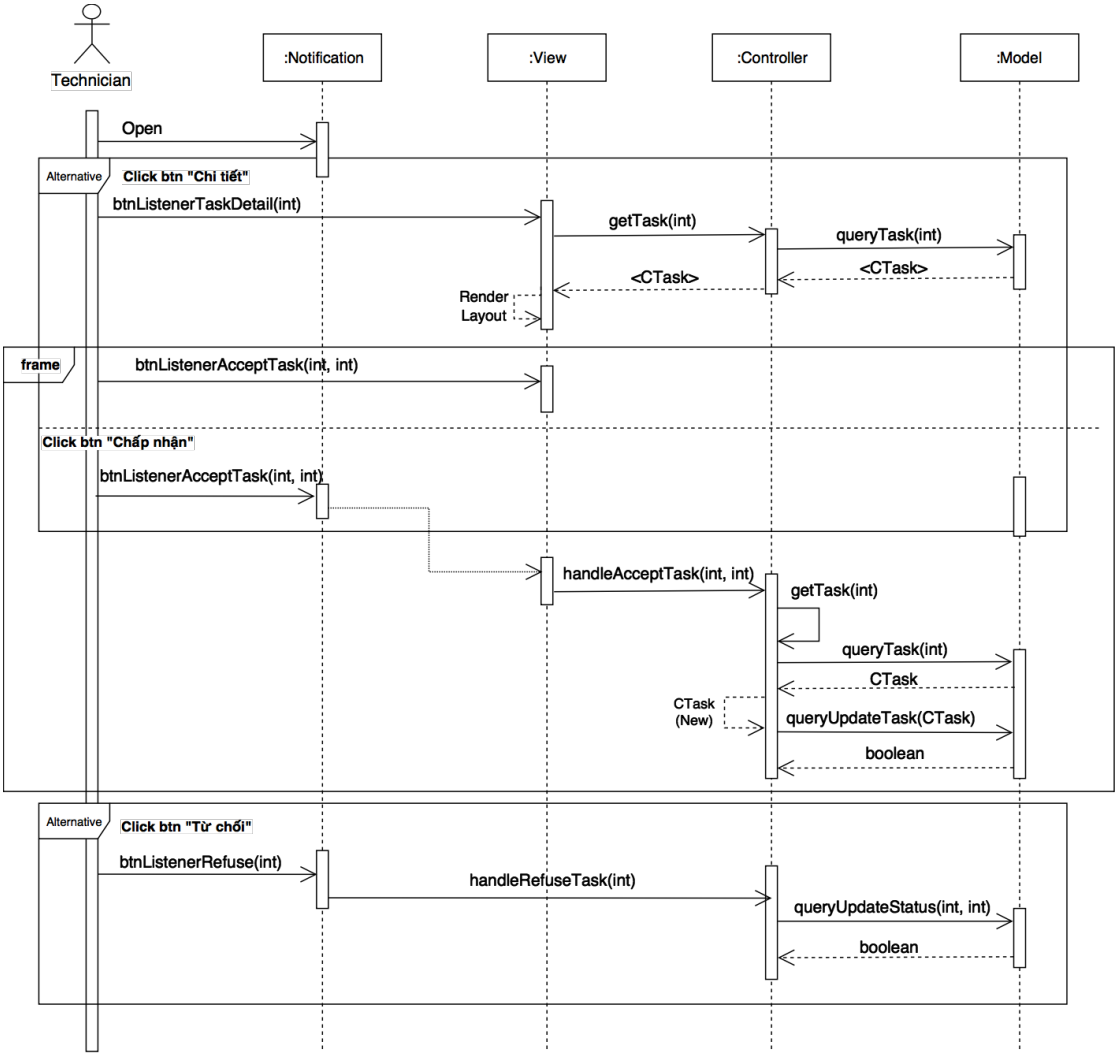
Hình 12: *Use case Đăng ký - Xref: SRS 2.1*

6.9 Tìm kiếm tác vụ



Hình 13: *Use case Tìm kiếm tác vụ - Xref: SRS 2.11*

6.10 Chấp nhận tác vụ



Hình 14: *Use case Tìm kiếm tác vụ - Xref: SRS 2.12*