**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document Overview

**Office Rental Service**

|  |  |
| --- | --- |
| **Group 03** | |
| **Group members** | Lê Xuân Tiến – Team Leader – SE60897  Nguyễn Vũ Hoàng Quốc – Team Member – SE61112  Trương Tiến Thành – Team Member – SE61052  Trần Lê Tuấn – Team Member – 60350 |
| **Supervisor** | Mr. Nguyễn Trọng Tài |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | ORS |

-Ho Chi Minh City, May 2015 -

*This page is intentionally left blank*

**

**CAPSTONE PROJECT REGISTER**

Class: Duration time: from 11/5/2015…. To ../../2015…..

(\*) Profession: <Software Engineer> Specialty: <ES> <IS> ⌧

(\*) Kinds of person make registers: Lecturer ⌧ Students

1. Register information for supervisor (if have)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Full name** | **Phone** | **E-Mail** | **Title** |
| Supervisor 1 | Nguyễn Trọng Tài |  | taint@fpt.edu.vn | Mr. |

2. Register information for students (if have)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Full name** | **Student code** | **Phone** | **E-mail** | **Role in Group** |
| Student 1 |  |  |  |  |  |
| Student 2 |  |  |  |  |  |
| Student 3 |  |  |  |  |  |
| Student 4 |  |  |  |  |  |

3. Register content of Capstone Project

(\*) 3.1. Capstone Project name:

English: Office Rental Service.

Vietnamese: Dịch vụ cho thuê văn phòng.

Abbreviation: ORS

Building an online system allows to rent the office. The system also support to manage different types of office (apartment, house) and its amenities..., manage contract, manage request for repair and maintenance.

(\*) 3.2. Main proposal content (including result and product)

1. Theory and practice (document):

* Apply the software development process and the UML (OOA, OOD, OOP should be applied)
* N-tier/MVC and Object-relational mapping should be applied
* Web-based application

1. Program:

* Main features should be included:
  + Member management
  + Office Management (by category, by amenities...)
  + Rental management
  + Contract management
  + Request for repair and maintenance management
  + Searching, statistic

1. Other products:

N/A

4. Other comment (propose all relative thing if have)

N/A

|  |  |
| --- | --- |
| **Supervisor (If have)**  *(Sign and full name)* | HCM city, date 11/4/2015  **On behalf of Registers**  *(Sign and full name)* |

# Table of Contents

[Table of Contents 5](#_Toc427513204)

[List of Tables 6](#_Toc427513205)

[List of Figures 6](#_Toc427513206)

[Definitions, Acronyms, and Abbreviations 7](#_Toc427513207)

[A. Software Project Management Plan 8](#_Toc427513208)

[1. Problem Definition 8](#_Toc427513209)

[1.1 Name of this Capstone Project 8](#_Toc427513213)

[1.2 Problem Abstract 8](#_Toc427513214)

[1.3 Project Overview 8](#_Toc427513215)

[Table 3: Hardware Requirement for Web User 10](#_Toc427513216)

[Table 4: Requirement for mobile app 10](#_Toc427513217)

[2. Project Organization 11](#_Toc427513218)

[2.1 Software Process Model 11](#_Toc427513219)

[2.2 Tools and Techniques 11](#_Toc427513220)

[B. Software Requirement Specification 11](#_Toc427513221)

[1. User Requirement Specification 11](#_Toc427513222)

[1.1 Guest requirement 12](#_Toc427513223)

[1.2 Customer requirement 12](#_Toc427513224)

[1.3 Admin requirement 12](#_Toc427513225)

[1.4 Manager requirement 12](#_Toc427513226)

[1.5 Staff requirement 12](#_Toc427513227)

[1.6 System requirement 12](#_Toc427513228)

[2. System Overview Use Case 13](#_Toc427513229)

[3. Conceptual Diagram 14](#_Toc427513234)

[Table 5: Conceptual diagram data dictionary 14](#_Toc427513235)

[C. Software Design Description 15](#_Toc427513236)

[1. Design Overview 15](#_Toc427513237)

[2. System Architectural Design 15](#_Toc427513238)

[3. Component Diagram 16](#_Toc427513239)

[Table 6: Component dictionary 16](#_Toc427513240)

[4. Detailed Description Explanation 17](#_Toc427513241)

[4.1 Class Diagram 17](#_Toc427513242)

[4.2 Sequence Diagram 17](#_Toc427513243)

[5 Database Design 19](#_Toc427513244)

[5.2 Logical Diagram 19](#_Toc427513245)

[5.3 Data Dictionary 19](#_Toc427513246)

[Table 7: Data dictionary 20](#_Toc427513247)

[6 Algorithms 20](#_Toc427513248)

[6.2 Clustering data 20](#_Toc427513249)

[6.3 Matching data 22](#_Toc427513250)

[D. System Implementation & Test 23](#_Toc427513251)

[Database Relationship Diagram 23](#_Toc427513252)

[Physical Diagram 23](#_Toc427513253)

[E. Demonstrations Workflow 25](#_Toc427513254)

[1. <Manager> Create contract 25](#_Toc427513255)

[2. <Customer> Request repair 25](#_Toc427513256)

[3. <Customer> Request appointment 27](#_Toc427513257)

[4. <System> Check request office 27](#_Toc427513258)

# List of Tables

[Table 1: Definitions 7](#_Toc427513259)

[Table 2: Hardware Requirement for Server 10](#_Toc427513260)

[Table 3: Hardware Requirement for Web User 10](#_Toc427513261)

[Table 4: Requirement for mobile app 10](#_Toc427513262)

[Table 5: Conceptual diagram data dictionary 14](#_Toc427513263)

[Table 6: Component dictionary 16](#_Toc427513264)

[Table 7: Data dictionary 20](#_Toc427513265)

# List of Figures

[Figure 1: Proposed System 9](#_Toc427532358)

[Figure 2: Software Process Model 11](#_Toc427532359)

[Figure 3: System Overview Use Case 13](#_Toc427532360)

[Figure 4: Conceptual diagram 14](#_Toc427532361)

[Figure 5: ORS System Architectural 15](#_Toc427532362)

[Figure 6: Component Diagram 16](#_Toc427532363)

[Figure 7: Class diagram 17](#_Toc427532364)

[Figure 8: View Repair Request Sequence Diagram 18](#_Toc427532365)

[Figure 9: Create Appointment Request Sequence Diagram 18](#_Toc427532366)

[Figure 10: View Repair Request List Sequence Diagram 19](#_Toc427532367)

[Figure 11: Logical Diagram 19](#_Toc427532368)

[Figure 12: Physical Database Diagram 24](#_Toc427532369)

[Figure 13: <Manager> Create contract 25](#_Toc427532370)

[Figure 14: <Customer> Request repair 26](#_Toc427532371)

[Figure 15: <Customer> Request appointment 27](#_Toc427532372)

[Figure 16: <System> Check request office 28](#_Toc427532373)

# Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| ORS | Office Rental Service |
| SRS | Software Requirement Specification |
| Admin | Administrator |
| SMS | Short message service |
| Info | Information |

Table 1: Definitions

# Software Project Management Plan

## Problem Definition



### Name of this Capstone Project

* **Official name**: Office Rental Service
* **Vietnamese name:** Dịch vụ cho thuê văn phòng
* **Abbreviation:** ORS

### Problem Abstract

Nowadays, the way companies find and rent an office, is complex and hard to find a suitable office. It means they need to do many steps before they can rent a suitable office. About the office owner, they have to contact with customer, manage contract, manage repair request,… on the paper. It means there is human mistake.

To solve those problems already mentioned above, we need to make the way customer find an office and the way customer manage their request effectively. About the office owner, their contracts, requests for their office,… are easy to manage.

### Project Overview

#### Current Situation

There are some current websites such as vanphongthue.com.vn, rongbay.com, timvanphong.vn, etc. All of them have some advantages and disadvantages.

* Advantages:
  + Friendly and specialized interface.
  + Providing search engines criteria.
  + Direct consultation with the manager.
* Disadvantages:
  + Only supply direct meeting, not arrange for a meeting.
  + Not support map.
  + Not provide additional equipment and maintenance.

#### The Proposed System

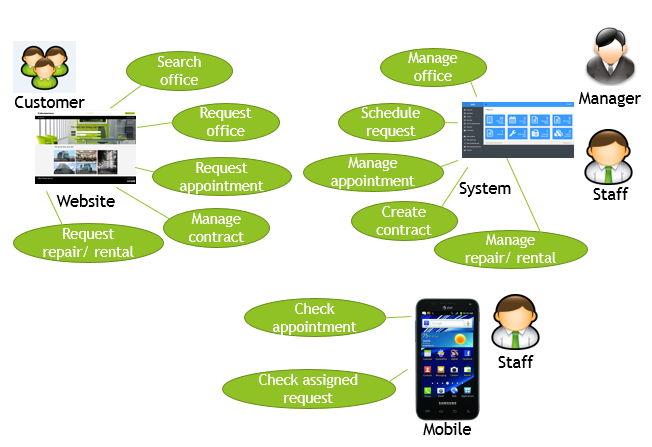


Figure 1: Proposed System

In more detail, the system would contain following features:

##### Website

* Guess can search office and view detail. If they want to make appointment, they need to register.
* Customer can review office, request appointment, request office (if they can’t find ones), request rental equipment and request repair.
* Admin can manage staff and manager.
* Manager can create and delete contract. They also can assign staff to repair for customer and view statistic of system.
* Staff can view list, create, update and delete office. When staff finish repair for customer, they can report to manager. They also can approve or delete rental equipment.

##### Mobile Application

Staff can check appointment task and update status request repair.

#### Boundaries of the System

* The system is intended for customers who want to rent office (such as businessman, medium or small company,…).
* The language of the system is Vietnamese
* The complete product includes:
  + The website, for admin, staff, customers and guest to interactive with the system.
  + Mobile application for staff to check task and update status request repair.
  + All the process involved document.

#### Development Environment

* **Hardware requirement for server computer**

|  |  |  |
| --- | --- | --- |
| Windows | Minimum Requirements | Recommended |
| Internet Connection | Cable, Wi-Fi (2Mbps) | Cable, Wi-Fi (8 Mbps) |
| Operating System | Window 7 | Window 8.1 |
| Computer Processor | Intel Core i5 | Intel Core i7 |
| Computer Memory | 4GB RAM | 6GB or more |

Table 2: Hardware Requirement for Server

* **Hardware Requirement for Web User**

|  |  |  |
| --- | --- | --- |
| **Web** | **Minimum Requirements** | **Recommended** |
| **Internet Connection** | 2Mbps | 4Mbps |
| **Web Browser** | Firefox, Chrome, IE 8 | Firefox, Chrome, IE 9 |

Table 3: Hardware Requirement for Web User

All computers must be connected to the Internet.

* **Hardware requirement for mobile app**

|  |  |  |
| --- | --- | --- |
| Windows | Minimum Requirements | Recommended |
| Internet Connection | Wifi (2Mbps) | Wifi (4Mbps) |
| Operating System | Android 4.0 | Android 4.2 |
| Hardware | Touchscreen | Touchscreen |
| Memory | 512 MB or more | 1 GB or more |

Table 4: Requirement for mobile app

* **Software requirements**
  + Operating system: Windows 7, or above;
  + Framework: Hibernate with Java Persistence;
  + Modeling Tool: Visual Paradigm Community;
  + IDE: Intellji Idea;
  + DBMS: Microsoft SQL Server;
  + Source Control: GitHub, Git for windows, Tortoise Git.

## 2. Project Organization

### 2.1 Software Process Model

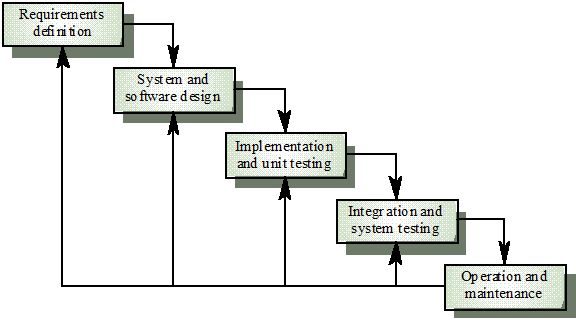


Figure 2: Software Process Model

This model is easy to manage and understand. For our project, we don’t have a lot of time so we use this model to help us release our project on time. This model uses for short project and it suitable for our project, which is small with 4 months and requirements are easy to clear.

### 2.2 Tools and Techniques

* *Front-end*: CSS3, HTML5, JavaScript, JQuery, Ajax, AngularJS, Java Android, PhoneGap.
* *Back-end*: Java 1.7, Hibernate, Java Persistence.
* *Web Server*: Tomcat 7.
* *Developing Tool*: JetBrains IntelliJ IDEA 14, Android Studio.
* *Database Management System:* Microsoft SQL Server 2008.
* *Source Control:* Git-1.9.5-preview, TortoiseGit-1.8.14.0.
* *Modeling Tool*: Visual Paradigm 12.0 Commnunity.
* *Document Tool*: Microsoft Office 2013.

# Software Requirement Specification

## User Requirement Specification

The system has five actors include guest, customer, staff, admin and system.

|  |  |
| --- | --- |
| **Actor** | **Description** |
| Guest | Person join to website but not login into system. |
| Customer | Person who want to rent offices and logged into system |
| Admin | Person who manage account and ban/ unban account |
| Manager | Person who manage request, contact customer, schedule staff…. |
| Staff | Person who in charge of create office, receive and process the appointment/ repair/ rental/ extend/ return office request |
| System | System will manage the schedule to process contract time, send notification, suggest office for customer |

### Guest requirement

* **Register:** guest need to register to become member of system.
* **Search office:** guest can search suitable office by criteria such as price and location.
* **Login**: guest uses email and password to login into the system to search or request office.

### Customer requirement

* **Search office:** customer can search suitable office by criteria such as price and location.
* **Send request:** customer can send request about book appointment, repair something in office, rental, request return before expire or request extend the contract.
* **Request office:** customer can send request office if they don’t find a suitable office.
* **Logout:** when finish all activities at website they can log out of system.

### Admin requirement

* **Logout:** when finish all activities at website they can log out of system.
* **Manage account:** admin can add, edit, remove or ban/ unban account.

### Manager requirement

* **Logout:** when finish all activities at website they can log out of system.
* **Contact with customer:** manager can notify customer when task is done.
* **Manage contract:** manager can create, edit contract.
* **Manage request:** manager can confirm about request repair, rental and appointment.
* **Assign staff:** manager can assign staff to contact with customer when manager has approved request.

### Staff requirement

* **Logout:** when finish all activities at website they can log out of system.
* **Contact with customer:** staff have to check task list about the appointment with customer and change status of the appointment when finish his task.
* **Manage office:** staff can add or edit office.
* **Check assign request:** staff have to check task list about the request repair amenities and change status of that request after repair complete.

### System requirement

* **Send notification mail:** system will send notification mail to customer when appointment has been approved and scheduled or request repair has been accepted.
* **Send notification SMS (short message service):** system will send notification mail to customer when their contract will be expired in one month.
* **Suggest office:** system will suggest some offices, which is nearly suitable with what customer searching.
* **Schedule request:** system will auto schedule all request about appointment and repair when those request has been approved by manager.

## System Overview Use Case

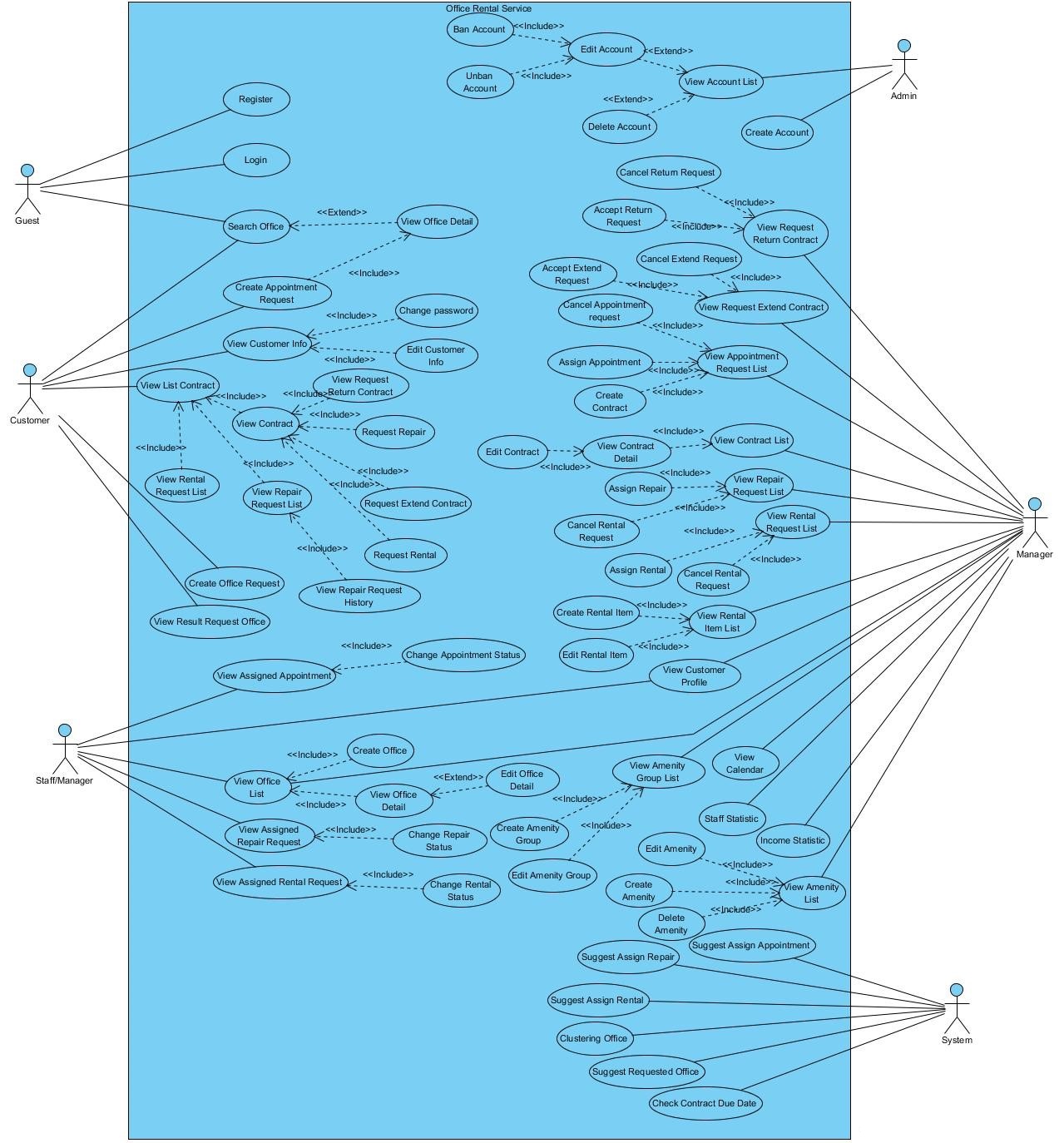


Figure 3: System Overview Use Case



## Conceptual Diagram

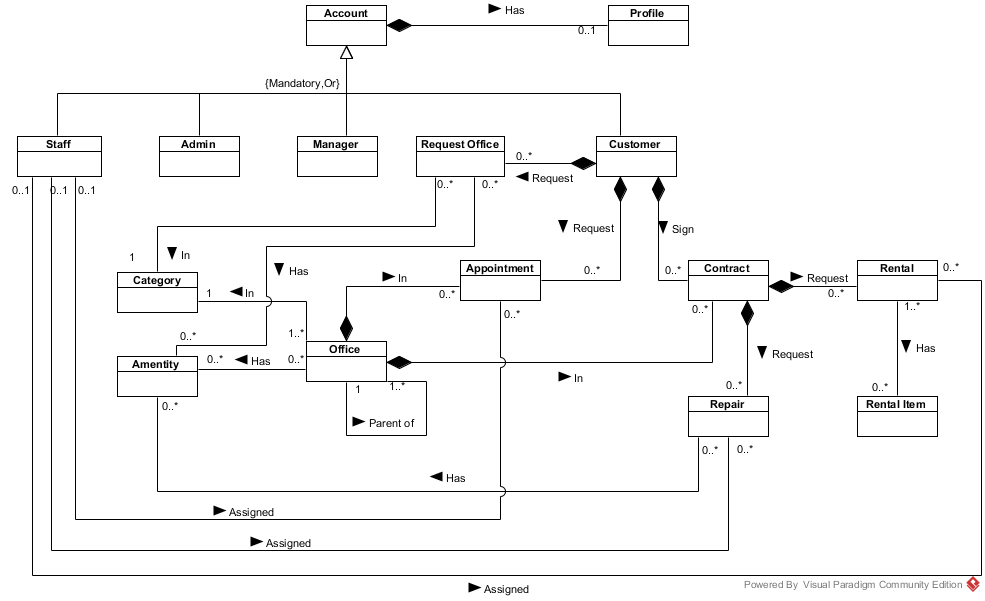


Figure 4: Conceptual diagram

**Data Dictionary**

|  |  |
| --- | --- |
| **Entity Data dictionary: describe content of all entities** | |
| **Entity Name** | **Description** |
| Account | Describe account of user of system. |
| Staff | Describe account type staff |
| Admin | Describe account type admin |
| Manager | Describe account type manager |
| Customer | Describe account type customer |
| Profile | Describe the contact information for customer |
| Office | Describe information of the office |
| Category | Describe the category of the office |
| Amenity | Describe the amenities of the office |
| Appointment | Describe the information of the appointment of customer for the office |
| Contract | Describe the information of contract between customer and office |
| Request Office | Describe the stored office request of customer |
| Repair | Describe the repair request for each contract |
| Rental | Describe the rental request for contract |
| Rental Item | Describe the rental items which available for rental request |

Table 5: Conceptual diagram data dictionary

# Software Design Description

## Design Overview

This document describes the technical and user interface design of MSSC System. 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, and sequence diagram to details the application functions.
* Section 5: overview some main user interface of system.
* Section 6: describe fully attributed ERD.
* Section 7: describe in details all algorithms used in the system.

## System Architectural Design

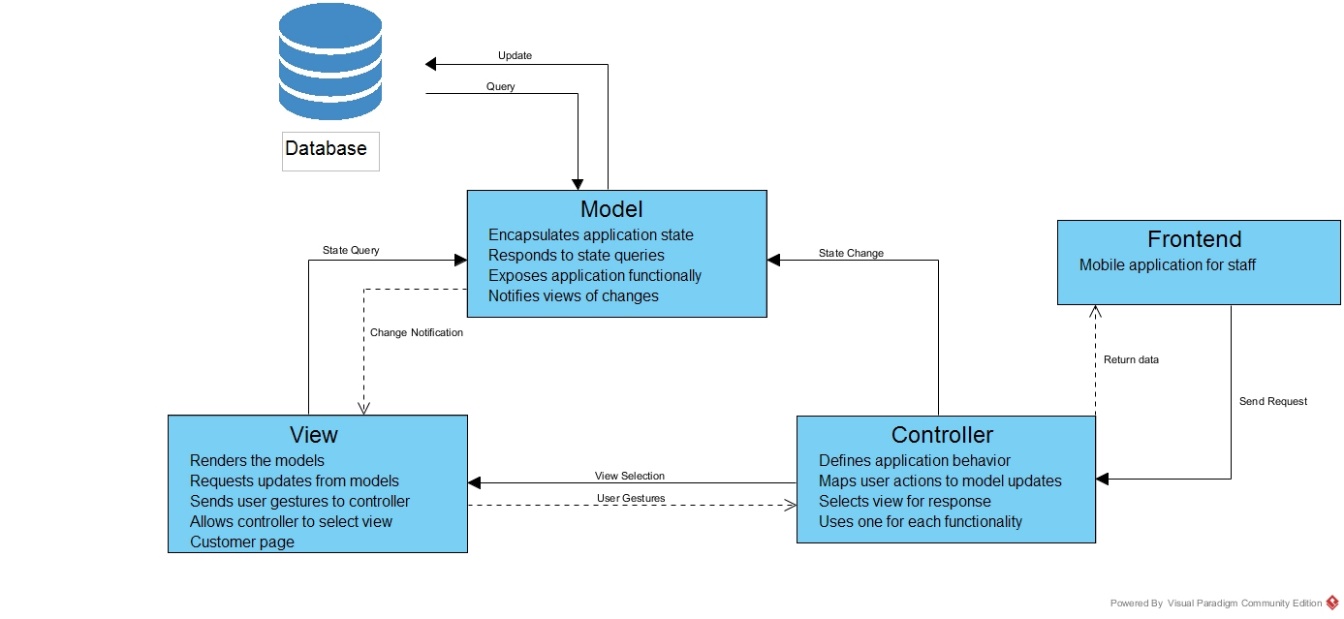


Figure 5: ORS System Architectural



#### Web application architecture description

* **Model:** The model manages the behavior and data of the application domain, responds to requests for information about its state (usually from the view), and responds to instructions to change state (usually from the controller).
* **View:** The view manages the display of information.
* **Controller**: The controller interprets the mouse and keyboard inputs from the user, informing the model and/or the view to change as appropriate.
* **Frontend**: Mobile application for staffs send and receive request through controller.

## Component Diagram

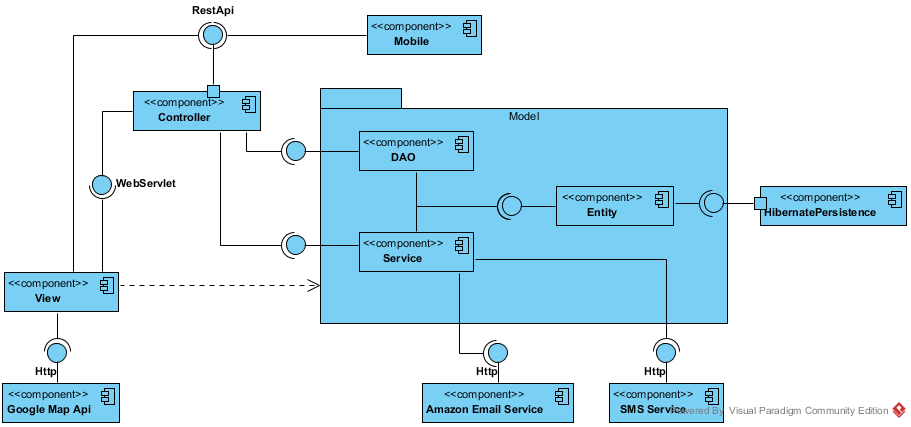


Figure 6: Component Diagram

|  |  |
| --- | --- |
| **Component dictionary: describe component** | |
| **Component Name** | **Description** |
| HibernatePersistence | Component provide by Hibernate to establish database connection |
| Entity | The database mapping classes component |
| DAO | Data Access Classes component, provide function to manipulate data |
| Service | Provice additional data manipulation and external data manipulation |
| Controller | Receive and handle the request from View, then call DAO/ Service to manipulate |
| View | Render the view for user |
| Mobile | The mobile application component |
| Google Map Api | The map component provide by Google |
| Amazon Email Service | The email service component provide by Amazon |
| SMS Service | The SMS service component provide by ESMS (<http://esms.vn/>) |

Table 6: Component dictionary

## Detailed Description Explanation

### Class Diagram

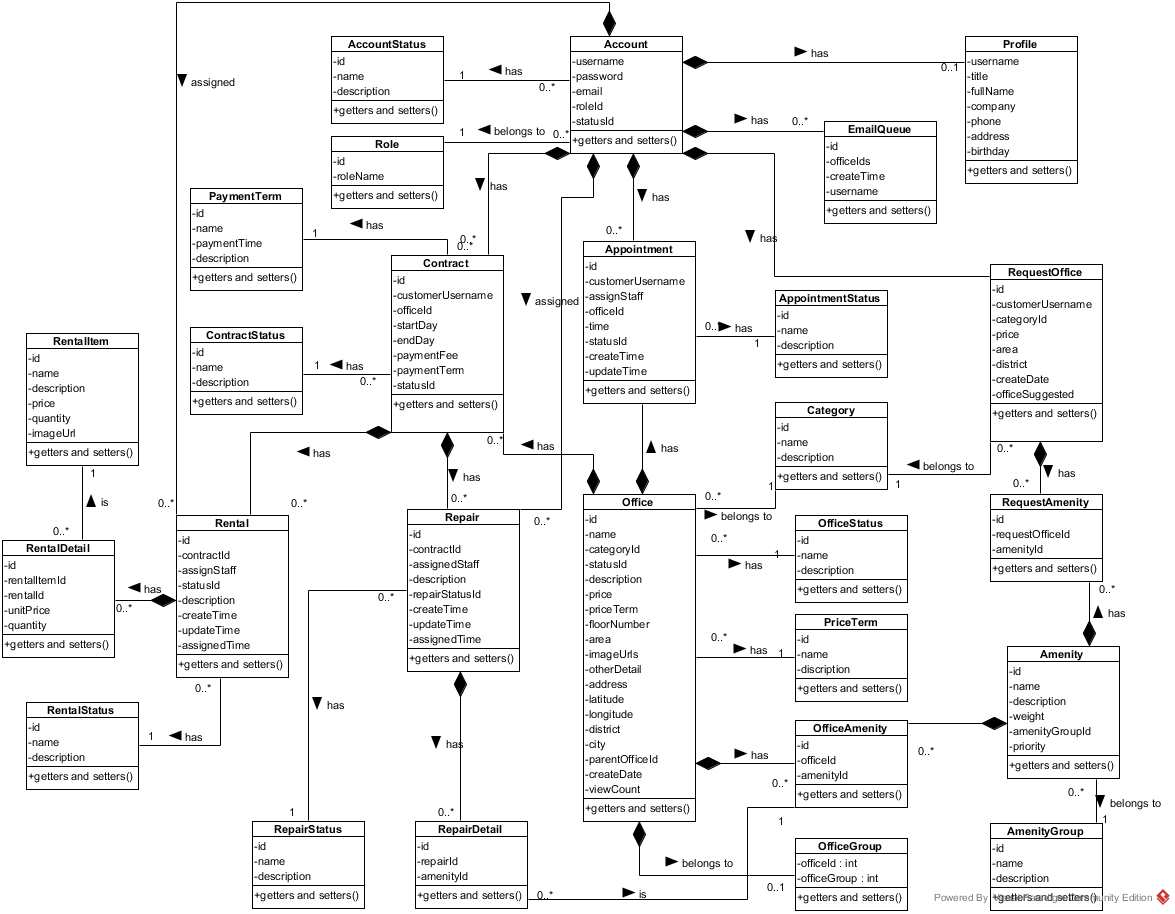


Figure 7: Class diagram

### Sequence Diagram

#### 4.2.1 <Manager> View Repair Request List

**Summary:** This diagram used to describe the process of manager view repair requests.

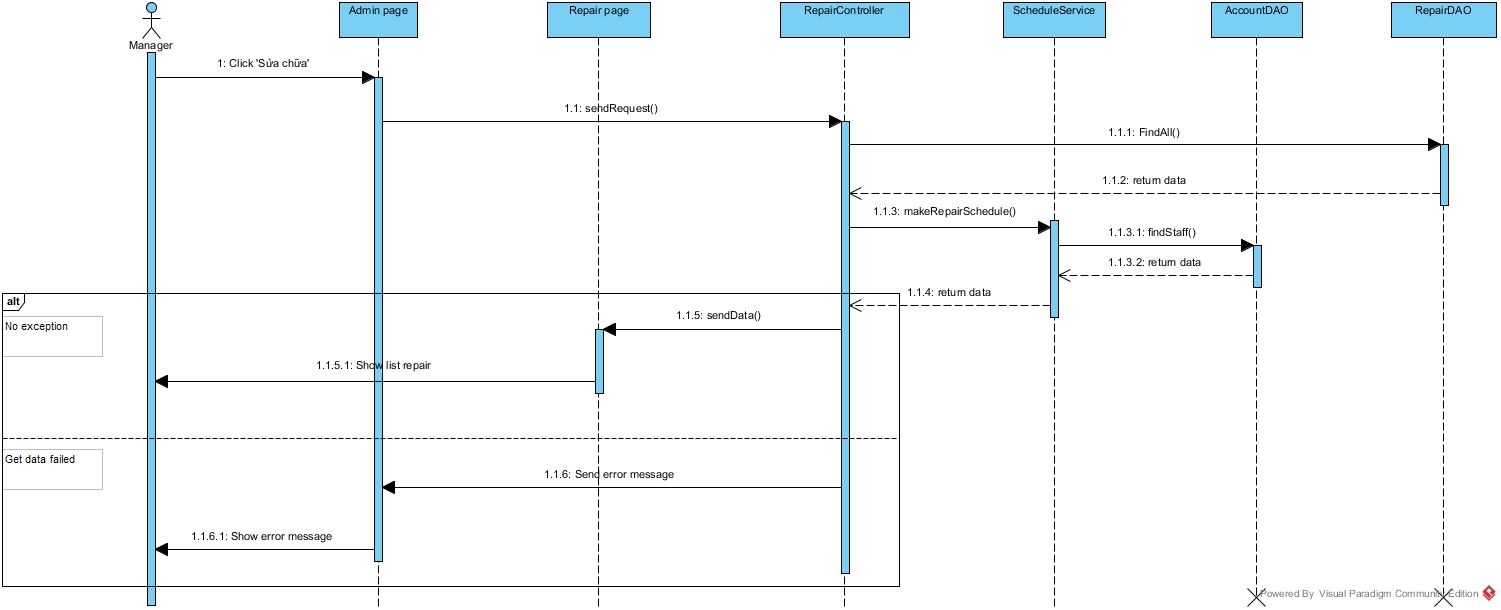


Figure 8: View Repair Request Sequence Diagram

#### <Customer> Create Appointment Request

**Summary:** This diagram used to describe the process of customer request an appointment.

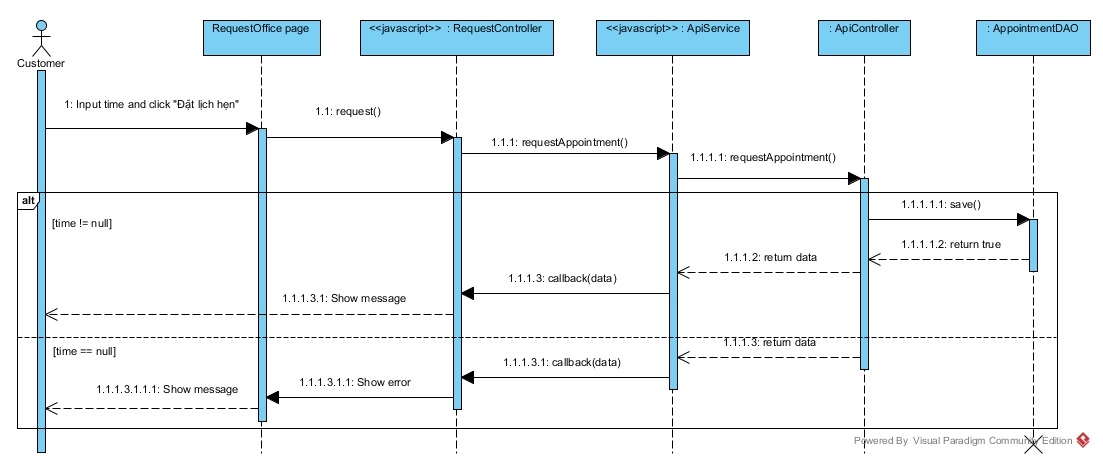


Figure 9: Create Appointment Request Sequence Diagram

#### <Customer> View Repair Request List

**Summary:** This diagram used to describe the process of customer view repair request list

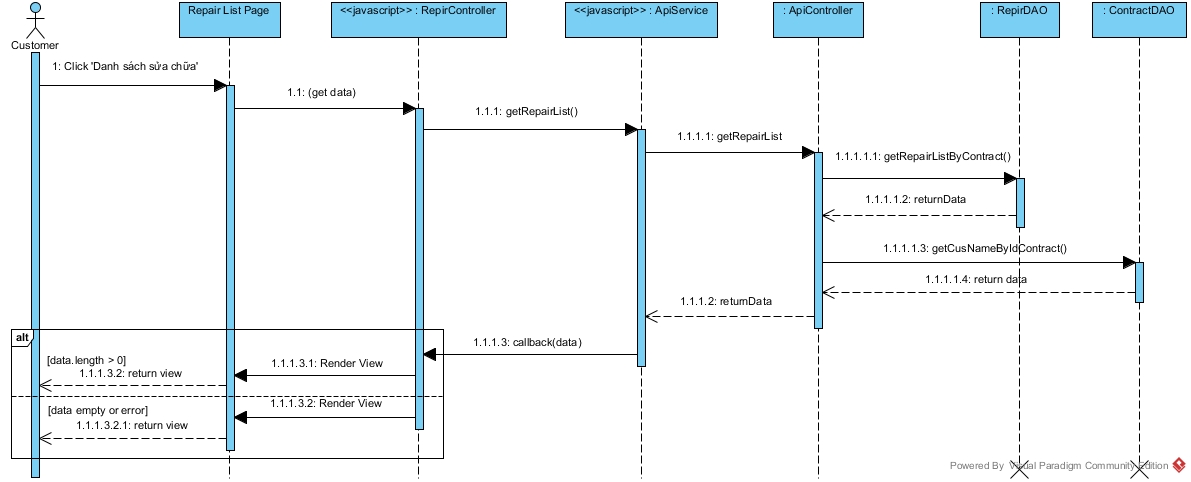


Figure 10: View Repair Request List Sequence Diagram

## Database Design

### Logical Diagram

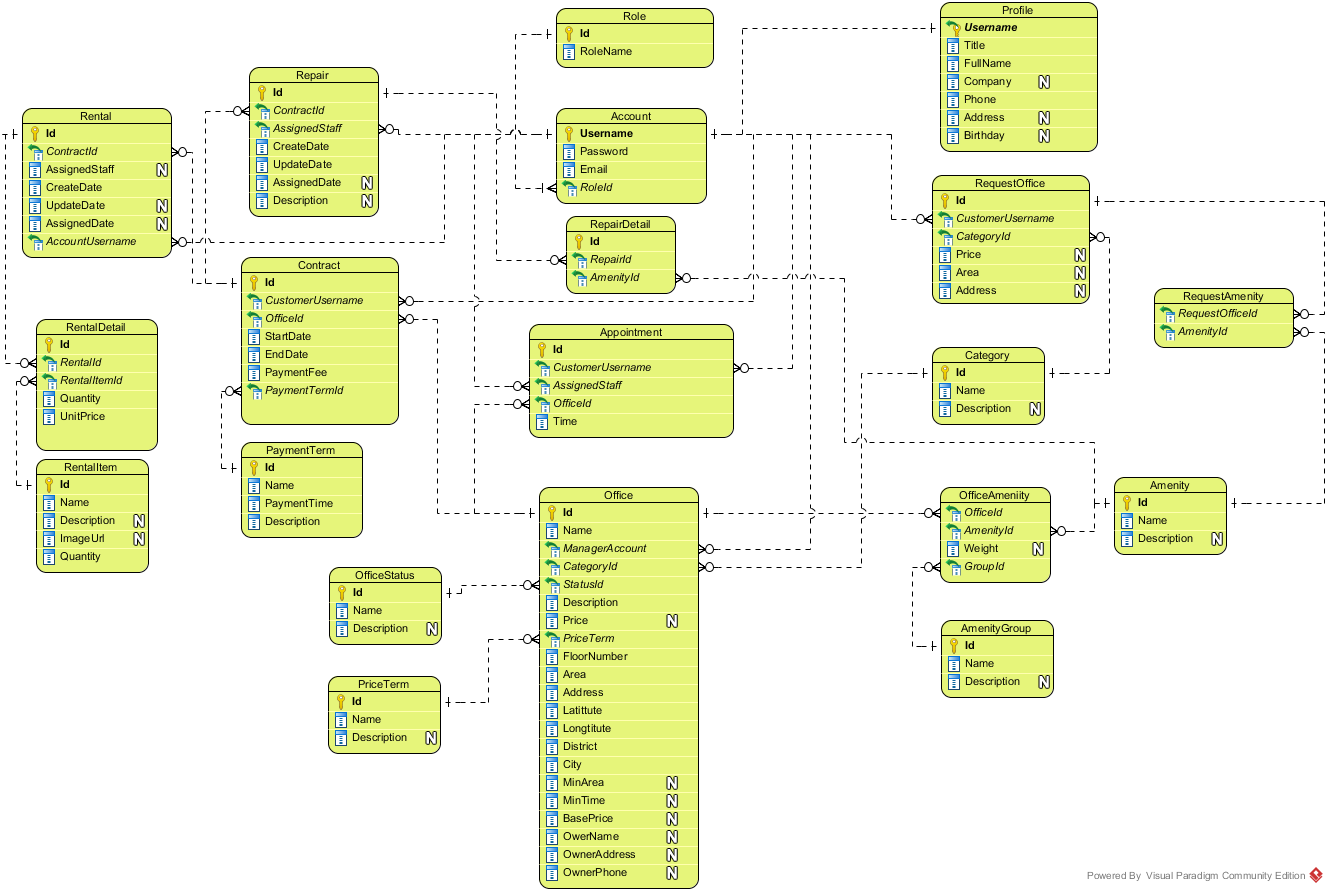


Figure 11: Logical Diagram

### Data Dictionary

|  |  |
| --- | --- |
| **Entity Data dictionary: describe content of all entities** | |
| **Entity Name** | **Description** |
| Account | Describe all accounts of users in the system. |
| Account Status | Describe status of account. One account has only one status. |
| Amenity | Describe all amenities in system. |
| Amenity Group | Describe groups of amenity. One amenity is only in one group. |
| Appointment | Describe all appointment requests of customer. |
| Appointment Status | Describe status of appointment request. One appointment has only one status. |
| Category | Describe category of office. One office is only in one category. |
| Contract | Describe all contracts in the system. |
| Contract Status | Describe status of contract. One contract has only one status. |
| Office | Describe all offices in the system. |
| Office Amenity | Describe all amenities in one office. One office can have more than one amenity. |
| Office Group | Describe group of the office for searching office. One office is only in one group |
| Office Status | Describe status of the office. One office has only one status. |
| Payment Term | Describe time for the payment |
| Price Term | Describe the way that money is calculated base on. |
| Profile | Describe information of the account |
| Rental | Describe all rental requests from customer |
| Rental Detail | Describe all rental items for one rental request |
| Rental Item | Describe all rental items in system |
| Rental Status | Describe status of rental request. One rental request has only one status. |
| Repair | Describe all repair requests from customer |
| Repair Detail | Describe all amenities need to repair for one repair request |
| Repair Status | Describe status of repair request. One repair request has only one status. |
| Request Amenity | Describe all amenities in one request office |
| Request Office | Describe all requests office from customer |
| Role | Describe role of account. One account has only one role. |

Table 7: Data dictionary

## Algorithms

### Clustering data

#### Definition

K-means is a prototype based clustering technique defining the prototype in terms of a centroid which is considered to be the mean of a group of points and is applicable to objects in a continuous n-dimensional space. (Refer website http://www.hypertextbookshop.com)

#### Define Problem

* Office in near location may have similarity in price range. But the location may vary and hard to define all location.
* Group the office by location and price range will make the user easier to find the similarity offices

#### Solution

To solve the problem, we use the well-known K-Means algorithm.

The step to perform the algorithm:

1. Defined the k number (number of cluster)
   * The k number is calculate by the square root of total of data (n). This make sure that there is group, and each group has about data
2. Normalize the data into the n-dimensions Euclidean space, all data based on [0, 1] distance. Each of office data is called “point”
   * In this problem, we use 3 statics: latitude, longitude and price range of the office for 3-dimensions
   * The price range is divided into 5 range: Call as 0, 100.000 VND – 200.000 VND as 1, etc., larger than 500.000 VND as 5
   * In each dimension, calculate the min and max. Then the normalize dimension data is calculate by:

normalize: dimension normalized data

data: the raw data in dimension

1. Select random point into k group, each group has at least 1 point
2. Calculate the mean points of each group
3. Calculate the distance of each data point to each group’s mean point, and change the group of data point to the group that has the nearest mean
   * The distance between A(a1, a2, … an) and X(x1, x2, … xn) is calculate by Euclidean distance:

Where n is the number of dimension

1. Repeat step 4 until the group in step 5 is unchanged

#### Complexity

* In total, the complexity of this algorithm is NP-hard. But with current algorithm maximum complexity limit to n\*10 (n is total of data)

#### Flow chart

Calculate K

Normalize data

Group data in K groups

Calculate the mean of each group

Group based on minimum distance to mean

Group changed?

End

Start

Yes

No

### Matching data

#### Definition

K-Nearest Neighbor algorithm (KNN) is a method for classifying objects based on the closest training examples in the feature space.  KNN is a type of instance-based learning, or lazy learning where the function is only approximated locally and all computation is deferred until classification. (Refer website http://www.datasciencecentral.com)

#### Define Problem

* The search office may be different from all of the office in system, and user want to make their decision with vary of result.

#### Solution

With the clustering data using K-Means, the best solution is KNN (K-Nearest-Neighbor) algorithm.

The steps to perform the algorithm:

1. Define k number. In this solution, we choose k at least by haft of the average office in each group of K-Means algorithm. Particular 2\* .
2. Normalize the query point (input data) to the same dimension of sample data (by above function)
3. Calculate the distance between query point and each of data in sample (all office) by Euclidean distance (as above).
4. Sort the calculated distance, select the minimum k distances for comparison.
5. The group of query point will be group that have the most frequent existence in above k distances. If there is more than 1 group that have largest existence, the selected group will be the group that have minimum total distance (in the k selected distances only)

#### Complexity

* In total, the complexity of this algorithm is O(n)

# System Implementation & Test

## Physical Diagram

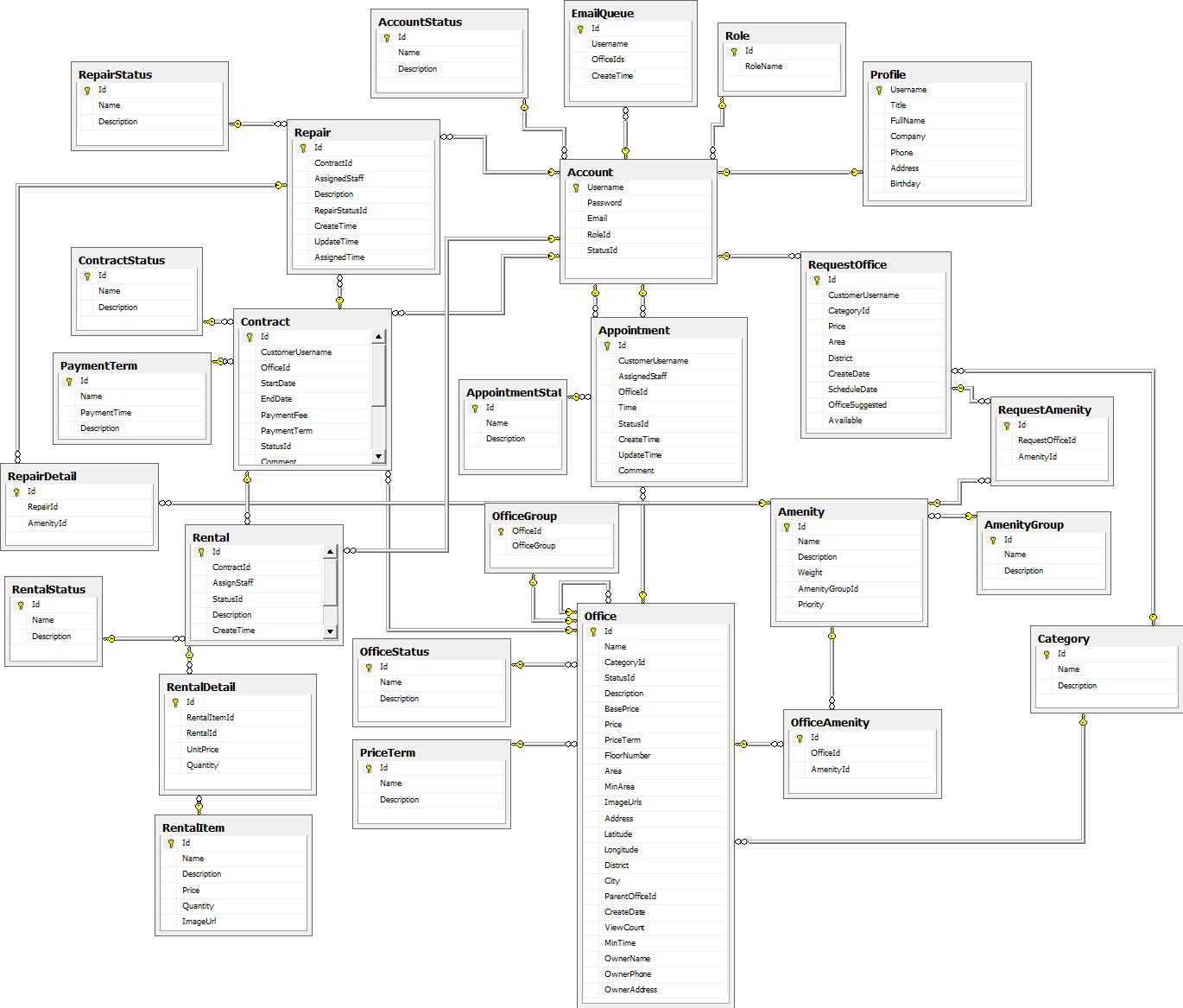


Figure 12: Physical Database Diagram

# Demonstrations Workflow

## <Manager> Create contract

**Summary:** This diagram shows how contract has been created.

[Staff] Check appointment list

Customer decide

Delete request

No

Create contract

Decide

Yes

[Manager] Assign appointment

Manager and Customer check contract

Delete request

No

Yes

Figure 13: <Manager> Create contract

## <Customer> Request repair

**Summary:** This diagram shows how request repair has been handled

Request repair

Manager decide

Notify customer

Reject

Accept

(Date, time, staff…)

OK

Customer decide

[Staff] Report to manager

Accept

Make schedule

[Staff] Check schedule

Staff check office

Repairing…

Reject

Reject

Figure 14: <Customer> Request repair

## <Customer> Request appointment

**Summary:** This diagram shows how customer find and get appointment

Input search info

Search result

No

Request office

View

Customer decide

Yes

Search

No

Yes

Request appointment

Manager decide

Notify (Message)

Yes

No

Notify (Message)

Figure 15: <Customer> Request appointment

## <System> Check request office

**Summary:** This diagram shows how system checks request office of customers.

Send email

Customer decide

Remove Request Office

Unsubscribe

Accept

Check requested office

Found

Request appointment

Figure 16: <System> Check request office

# Task sheet

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Product Deliverables** | **Task** | **TienLX** | **QuocNVH** | **ThanhTT** | **TuanTL** | **Unit** | **Size** |
| 1 | Report1 - Introduction | Project Information | **O** |  |  |  |  |  |
| Introduction |  | **O** |  |  |  |  |
| Current Situation | **O** |  |  |  |  |  |
| Problem Definition |  |  |  | **O** |  |  |
| **Proposed Solution** |  |  |  |  |  |  |
| Feature functions |  |  | **O** |  |  |  |
| Advantages and disadvantages | **O** | **O** | **O** | **O** |  |  |
| Functional Requirements | **O** |  |  |  |  |  |
| Role and Responsibility | **O** |  |  |  |  |  |
| 2 | Report2- Software Project Management Plan | Problem Definition |  | **O** |  | **O** |  |  |
| **Project organization** |  |  |  |  |  |  |
| Software Process Model |  | **O** |  |  |  |  |
| Roles and Responsibilities | **O** |  |  |  |  |  |
| Tools and Techniques | **O** |  |  |  |  |  |
| **Project management plan** |  |  |  |  |  |  |
| Software development life cycle | **O** |  |  | **O** |  |  |
| Phase Detail | **O** |  | **O** |  |  |  |
| All Meeting Minutes |  | **O** | **O** |  |  |  |
| Coding Convention |  | **O** |  |  |  |  |
| 3 | Report 3- Software Requirement Specification | User Requirement Specification | **O** |  | **O** | **O** |  |  |
| **System Requirement Specification** |  |  |  |  |  |  |
| External Interface Requirements |  |  | **O** |  |  |  |
| **Functional Requirement** |  |  |  |  |  |  |
| <Guest>Overview Use Case |  |  | **O** |  |  |  |
| <Customer>Overview Use Case |  | **O** | **O** |  |  |  |
| <Staff> Overview Use Case | **O** |  |  |  |  |  |
| <Admin> Overview Use Case |  |  |  | **O** |  |  |
| <Manager> Overview Use Case | **O** | **O** |  | **O** |  |  |
| <System>Overview Use Case | **O** | **O** |  |  |  |  |
| Non-functional Requirement |  |  | **O** |  |  |  |
| Conceptual Diagram | **O** |  |  |  |  |  |
| 4 | Report 4- Software Design Description | Design Overview | **O** |  | **O** |  |  |  |
| System Architectural Design |  | **O** |  |  |  |  |
| Component Diagram | **O** |  |  |  |  |  |
| **Detailed Description Explanation** |  |  |  |  |  |  |
| **Class Diagram** |  |  |  |  |  |  |
| Overview | **O** |  | **O** |  |  |  |
| Entity package | **O** |  | **O** |  |  |  |
| DAO package |  |  |  | **O** |  |  |
| Service Package | **O** |  |  |  |  |  |
| Controller Package |  |  |  | **O** |  |  |
| Mobile Package | **O** |  |  |  |  |  |
| **Interaction Diagram** |  |  |  |  |  |  |
| <Guest> Sequence Diagram |  |  | **O** |  |  |  |
| <Customer> Sequence Diagram |  | **O** | **O** |  |  |  |
| <Staff> Sequence Diagram | **O** |  |  |  |  |  |
| <Admin> Sequence Diagram |  |  |  | **O** |  |  |
| <Manager> Sequence Diagram | **O** | **O** |  | **O** |  |  |
| <System> Sequence Diagram | **O** | **O** |  |  |  |  |
| **User Interface Design** |  |  |  |  |  |  |
| Guest Interface Design | **O** | **O** | **O** |  |  |  |
| Customer Interface Design | **O** | **O** | **O** |  |  |  |
| Manager Interface Design | **O** | **O** | **O** | **O** |  |  |
| Mobile Interface Design | **O** |  |  |  |  |  |
| **Database Design** |  |  |  |  |  |  |
| Logical Diagram |  |  |  | **O** |  |  |
| Data Dictionary |  | **O** |  |  |  |  |
| **Algorithms** |  |  |  |  |  |  |
| Clustering data | **O** |  |  |  |  |  |
| Matching data | **O** |  |  |  |  |  |
| 5 | Report 5 - Software Implementation and Test Document | **Setup Environment** | **O** |  |  |  |  |  |
| **Coding admin functions** |  |  |  |  |  |  |
| Login / Log out |  |  |  | **O** | 1 | 1 |
| Manage account |  |  |  | **O** | 3 | 1 |
| **Coding manager / staff functions** |  |  |  |  |  |  |
| Manage contract |  | **O** |  |  | 2 | 4 |
| Manage appointment |  | **O** |  |  | 2 | 2 |
| Manage repair/rental |  |  | **O** |  | 2 | 2 |
| Manage office | **O** |  |  |  | 2 | 4 |
| Manage rental item |  |  | **O** |  | 2 | 1 |
| Manage amenity |  |  |  | **O** | 2 | 2 |
| Manage amenity group |  |  |  | **O** | 2 | 1 |
| Statistics |  |  |  | **O** | 2 | 3 |
| Calendar | **O** |  |  |  | 2 | 2 |
| Validation |  |  |  | **O** | 5 | 4 |
| Pagination | **O** |  |  |  | 3 | 1 |
| **Customer functions** |  |  |  |  |  |  |
| Login / Register |  |  | **O** |  | 1 | 1 |
| Search office | **O** |  |  |  | 1 | 5 |
| View home page |  | **O** |  |  | 1 | 1 |
| View office detail |  |  | **O** |  | 1 | 2 |
| Request appointment | **O** |  |  |  | 1 | 1 |
| Manage contract |  |  | **O** |  | 2 | 2 |
| Manage repair/rental |  |  | **O** |  | 2 | 2 |
| Request rental |  | **O** |  |  | 1 | 1 |
| Manage request office |  | **O** |  |  | 1 | 2 |
| **System functions** |  |  |  |  |  |  |
| Clustering office | **O** |  |  |  | 2 | 5 |
| Schedule |  | **O** |  |  | 3 | 5 |
| Suggest office |  | **O** |  |  | 2 | 4 |
| Suggest assign job | **O** |  |  |  | 2 | 4 |
| **Mobile functions** |  |  |  |  |  |  |
| Coding front app | **O** |  |  |  | 2 | 1 |
| Coding mobile adapter | **O** |  |  |  | 1 | 2 |
| Coding backend | **O** |  |  |  | 2 | 2 |
| Introduction |  |  | **O** |  |  |  |
| Database Relationship Diagram | **O** |  |  |  |  |  |
| **Performance Measures** |  |  |  |  |  |  |
| Clustering Performance | **O** |  |  |  |  |  |
| Matching Performance | **O** |  |  |  |  |  |
| **Test Plan** |  |  |  |  |  |  |
| Features to be tested | **O** |  |  |  |  |  |
| Features not to be tested | **O** |  |  |  |  |  |
| **System Testing Test Case** |  |  |  |  |  |  |
| Overview |  |  | **O** |  |  |  |
| Guest Test Case | **O** |  |  |  |  |  |
| Customer Test Case |  | **O** | **O** |  |  |  |
| Manager Test Case |  | **O** | **O** | **O** |  |  |
| System | **O** | **O** |  |  |  |  |
| 6 | Report 6 - Software User's Manual | **Installation Guide** |  |  |  |  |  |  |
| Setting up environment at server side | **O** |  |  | **O** |  |  |
| Deployment at server side | **O** |  |  | **O** |  |  |
| Client side environment setting | **O** |  |  | **O** |  |  |
| **User Guide** |  |  |  |  |  |  |
| Guide for customer |  | **O** | **O** | **O** |  |  |
| Guide for manager on web application |  | **O** |  |  |  |  |
| Guide for staff on mobile application | **O** |  |  |  |  |  |
| Appendix |  |  |  | **O** |  |  |