**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

**DOCUMENT REPORT**

Capstone Project Document

**VN Habit Tracker**

|  |  |
| --- | --- |
| **Group 3** | |
| **Group members** | Lưu Thành Đạt - SE61124  Nguyễn Quang Tuyến - SE62069  Nguyễn Hữu Thắng - SE62447  Phạm Thanh Tùng - SE61628 |
| **Supervisor** | Lại Đức Hùng |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | VHT |

- Ho Chi Minh city, September 14th 2018

# Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Name** | **Definition** |
| VHT | VN Habit Tracker |
|  |  |

1. **Report No. 4 Software Design Description**
2. 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.

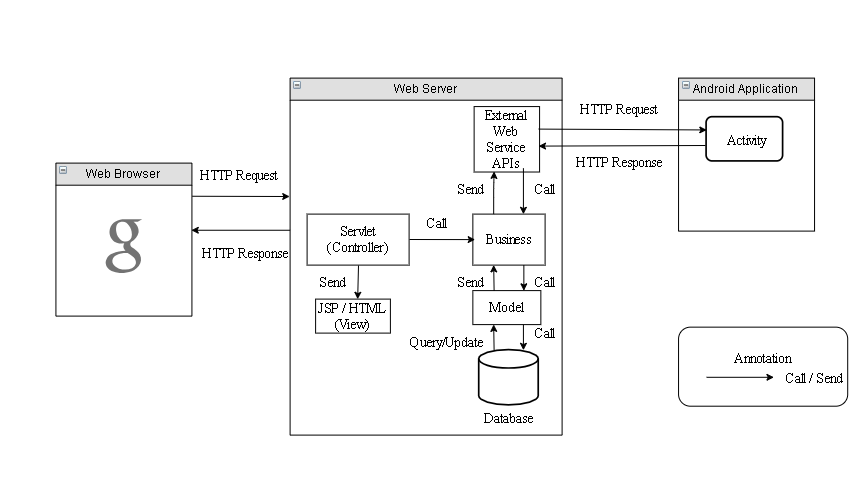
- The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

- 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: describe screens design.
  + Section 6: describe a fully attributed ERD.
  + Section 7: describe algorithms.

1. System architecture design

****

*Figure 9 System architecture design*

* 1. Web Application architecture description

In Web Application, the system is developed under J2EE MVC architecture style. We choose this architecture for Web application because of in scope of 4-member team, MVC architecture make it easier to split the big project into small modules and make it easier to assign each module for members in our team.

This project follows MVC architecture with following components:

* **Controller** is the parts of the application that acts like event handler to handles user interaction. Typically, controller read data from a request and calls appropriate Business’s method then selects view to return to user.
* **JSP/HTML (View)** is the parts of the application that handles the display of the data. The selection of View is under control of Controller.
* **Business** is the parts of the application that do business processing to solve domain problems.
* **Model** is the parts of the application that acts like a data transfer object between the system and database.
* **Web Service** is the parts of the application that acts like event handler for web and mobile communication via REST method.

1. Component Diagram



Figure 1: Component Web Diagram

| **Components Dictionary: Describes components** | |
| --- | --- |
| Web Apllicaiton | Component to controll the system and process request from mobile. Contains sub component: Model, Web Service, Controller. |
| View | Component that display data |
| Controller | Component of website to handle request from web |
| Service | Component to handle business logic of Website component |
| Repository | Component that communicate with database |
| Data Model | Component that do the interaction between the system and database. Contain sub components: Repository and Service |
| Server Database | Component that store data of system |

Table 1- Components Web Dictionary

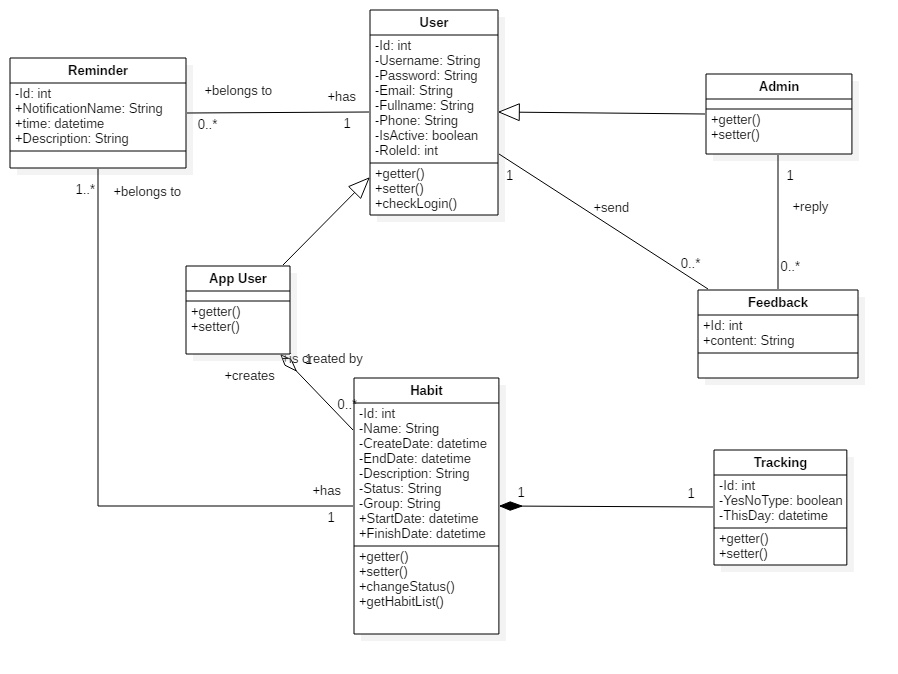


Figure 2: Component Android Application Diagram

| **Components Dictionary: Describes components** | |
| --- | --- |
| Android Apllicaiton | Component to controll the system and process request from mobile. Contains sub component: Model, Android Service, Controller. |
| View | Component that display data. |
| Controller | Component of application to handle request from application. |
| Android Service | Component to handle business logic of application component. |
| Model | Component that do the interaction between the system and database. Contain sub components: Repository and Service |

Table 2- Components Application Dictionary

1. **Detailed Description**
   1. Class diagram



*Figure 11 Class Diagram*

|  |  |  |
| --- | --- | --- |
| **CLASS DICTIONARY: DESCRIBE CLASS** | | |
| **Class Name** | **Mapping column with Conceptual diagram** | **Description** |
| **App User** | N/A | Contains the app user information |
| **Admin** | Admin | Contains the admin information |
| **User** | User | Contains the user information |
| **Habit** | Habit | Contains the habit information |
| **Reminder** | reminder | Contain the reminder information |
| **Feedback** | Feedback | Contains the feedback information |
| **Tracking** | Tracking | Contains the Tracking information |

* 1. Class Diagram Explanation
     1. App User

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | Private | Unique identifier of a user |
| username | String | Private | User’s username |
| password | String | Private | User’s password |
| fullName | String | Private | User’s full name |
| phone | String | Private | User’s phone |
| email | String | Private | User’s email |
| roleId | Integer | Private | Id of user’s role |
| isActive | Boolean | Private | User’s status |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter | Attribute type | Public | Get value of attribute |
| setter | Void | Public | Set value for attribute |
| checkLogin | Boolean | public | Check user login the system |

* + 1. User

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | Private | Unique identifier of a user |
| username | String | Private | User’s username |
| password | String | Private | User’s password |
| fullName | String | Private | User’s full name |
| phone | String | Private | User’s phone |
| email | String | Private | User’s email |
| roleId | Integer | Private | Id of user’s role |
| isActive | Boolean | Private | User’s status |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter | Attribute type | Public | Get value of attribute |
| setter | Void | Public | Set value for attribute |

* + 1. Admin

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | Private | Unique identifier of a user |
| username | String | Private | User’s username |
| password | String | Private | User’s password |
| fullName | String | Private | User’s full name |
| phone | String | Private | User’s phone |
| email | String | Private | User’s email |
| roleId | Integer | Private | Id of user’s role |
| isActive | Boolean | Private | User’s status |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter | Attribute type | Public | Get value of attribute |
| setter | Void | Public | Set value for attribute |

* + 1. Habit

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | Private | Unique identifier of a habit |
| name | String | Private | Habit’s name |
| createDate | datetime | Private | Habit’s create date |
| endDate | datetime | Private | Habit’s end date |
| startDate | datetime | Private | Habit’s start date |
| finishDate | datetime | Private | Habit’s finish date |
| group | String | Private | Habit’s group |
| description | String | Private | Habit’s description |
| status | String | Private | Habit’s status |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter | Attribute type | Public | Get value of attribute |
| setter | Void | Public | Set value for attribute |
| getHabitList | List | Public | Get all habit of user |

* + 1. Feedback

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | Private | Unique identifier of user |
| content | String | Private | Feedback’s content |
| **Method** | **Return Type** | **Visibility** | **Description** |

* + 1. Reminder

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | Private | Unique identifier of the habit |
| notificationName | String | Private | Notification’s notification name |
| time | Datetime | Private | Notification’s time |
| description | String | Private | Notification’s description |
| **Method** | **Return Type** | **Visibility** | **Description** |

* + 1. Tracking

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| id | Integer | Private | Unique identifier of a habit |
| type | Boolean | Private | Tracking’s type |
| thisDay | datetime | Private | Tracking’s this day |
| **Method** | **Return Type** | **Visibility** | **Description** |
| getter | Attribute type | Public | Get value of attribute |
| setter | Void | Public | Set value for attribute |

* 1. Interaction Diagram
     1. Add new habit

*Summary: this diagram show process of user add new habit*



Figure - Sequence Diagram for add new habit <User>

* + 1. Edit habit

*Summary: this diagram show process of user edits a habit.*

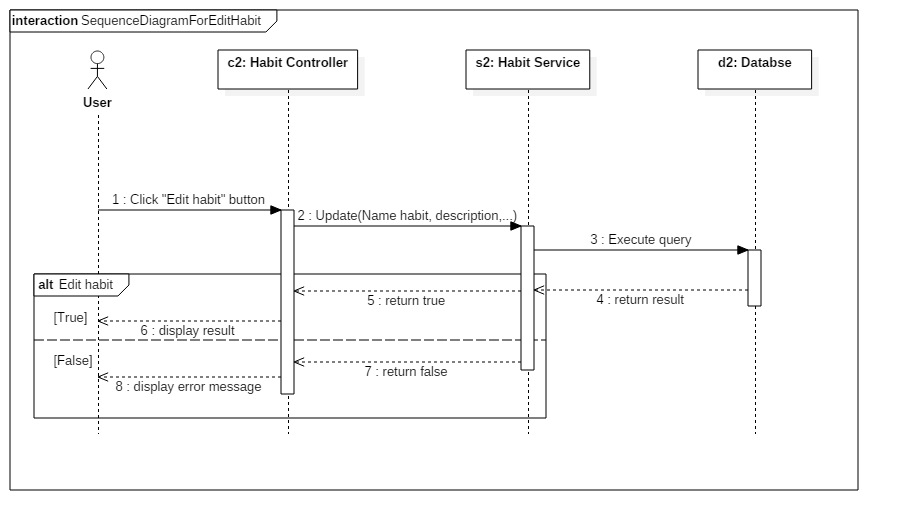


Figure - Sequence Diagram for edit habit <User>

* + 1. Delete habit

*Summary: this diagram show process of user deletes a habit*

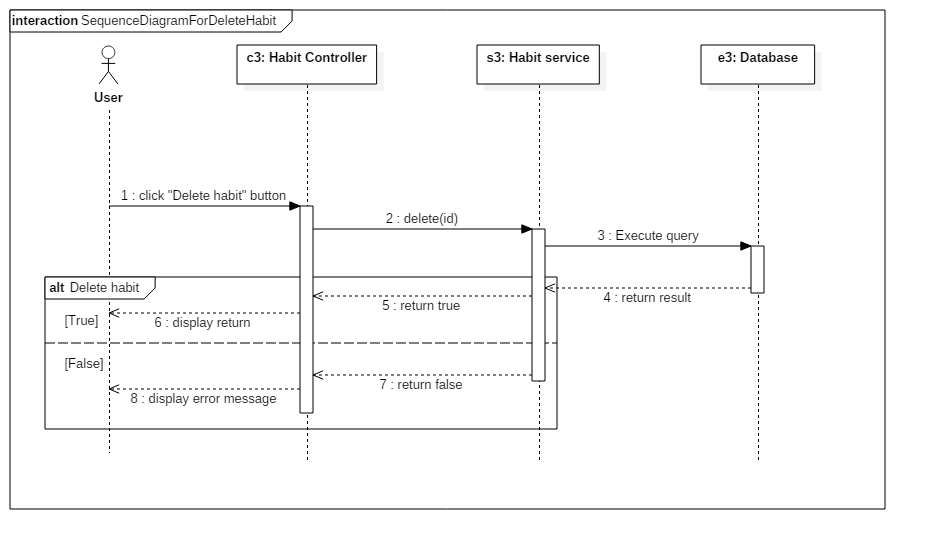


Figure - Sequence Diagram for add delete habit <User>