

**Software Design Description**

**Hanoi, April 12th, 20201**

**Table of Contents**

[**Chapter 4:** **Detailed Description of Components** 3](#_Toc70063540)

[**4.1. System Architecture Design** 3](#_Toc70063541)

**4.2.**  **Component Diagram** [4](#_Toc70063543)

[**4.3. Detailed Description of Component** 5](#_Toc70063544)

[**4.4. Class Diagram** 6](#_Toc70063545)

[***4.5.4.2 InfoController*** 21](#_Toc70063546)

[***4.5.4.3 ProfileController*** 21](#_Toc70063547)

[***4.5.4.4 PostController*** 22](#_Toc70063548)

[***4.5.4.5 NotificationController*** 23](#_Toc70063549)

[***4.5.4.6 MessageController*** 24](#_Toc70063550)

[***4.5.4.7 ConversationController*** 24](#_Toc70063551)

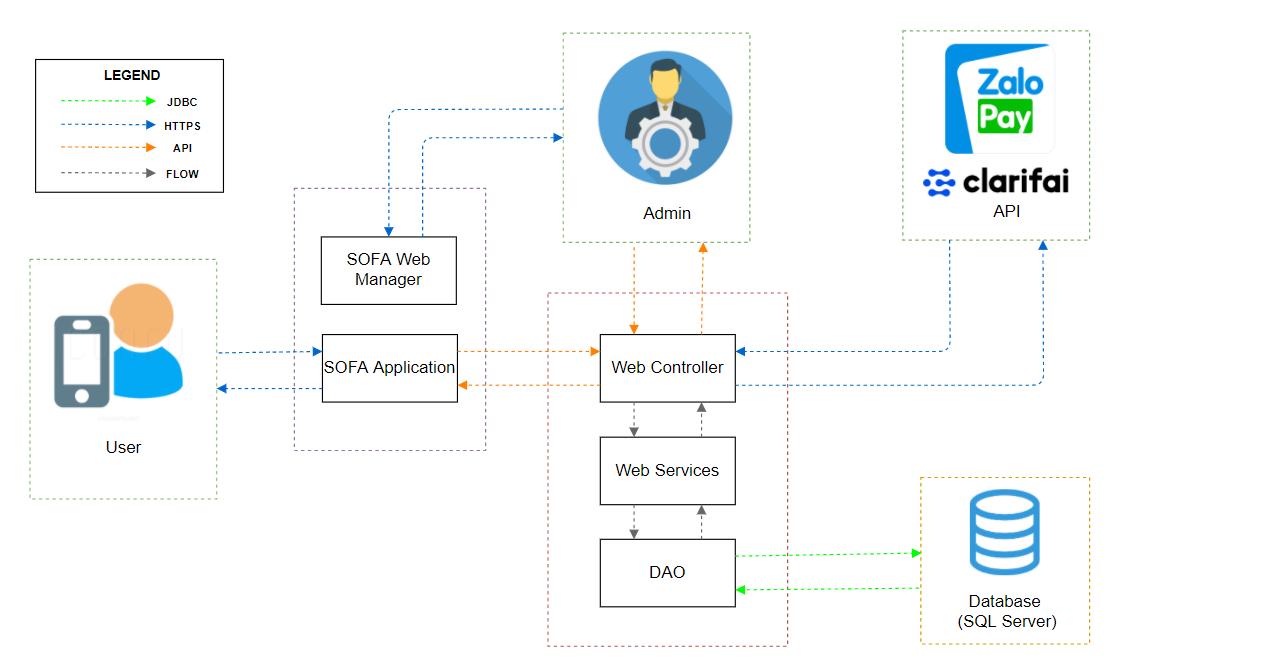
[**4.6. Sequence Diagram** 26](#_Toc70063552)

[**4.7. User Interface Design** 32](#_Toc70063553)

[**4.8. Database Design** 46](#_Toc70063556)

# **Detailed Description of Components**

## **4.1.0 *System Architecture Design***



*Figure 4.1 – System Architecture Design*

## **4.1.1 *Software Architecture Design***

## 

*Figure 4.2 – Software Architecture Design*

## **4.1.2 *Description of System Interface***

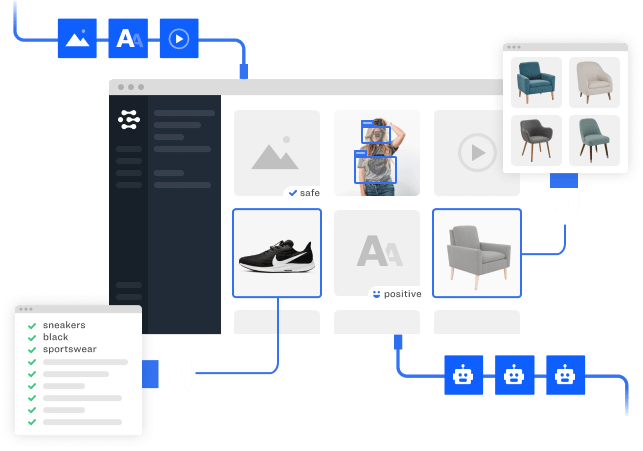
***4.1.2.1. Database management system***

## 

*Database management system*

Database management system is a software which is used to manage the database. For example: SQL Server, MySQL, Oracle, etc are a very popular commercial database which is used in different applications. DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more. It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

***4.1.2.2. Clarifai***

******

## **Clarifai** is an Image Recognition Software solution with functions and costs suitable for small and medium enterprises (SMEs) to large enterprises. **[Clarifai](https://sotaydoanhtri.com/softwares/clarifai-2335/)** is highly appreciated by both users and experts in the field of Deep Learning Software.

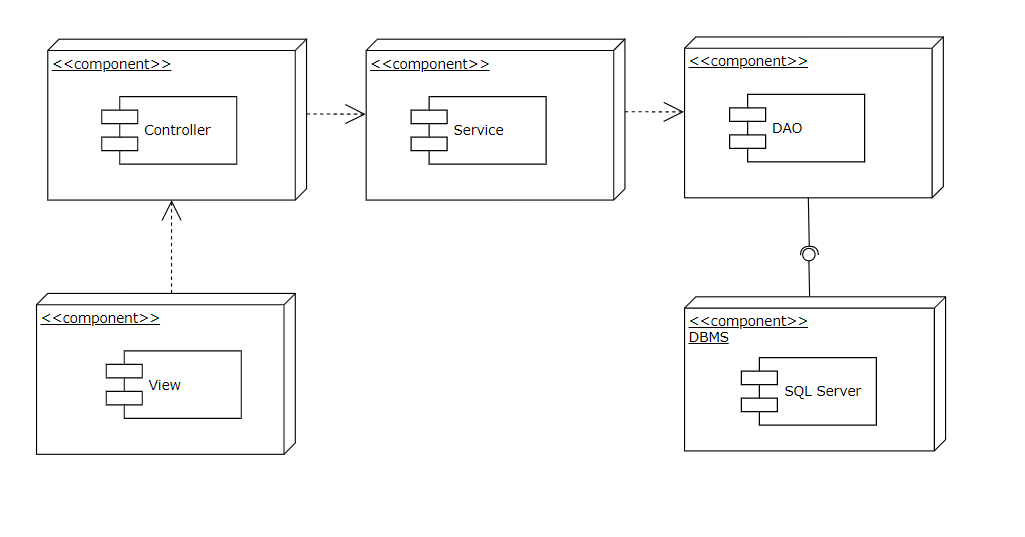
The platform includes the ability to moderate content, perform visual search, visual similarity, and organize media collections. It has pre-built recognition models that can identify a specific set of concepts like food or travel, NSFW, and its general model which can identify a range of concepts including objects, ideas, and emotion.[18] It also has the ability to create custom models which can identify other arbitrary objects such as cars or breeds of dogs.

***4.1.2.3. Zalopay***

******

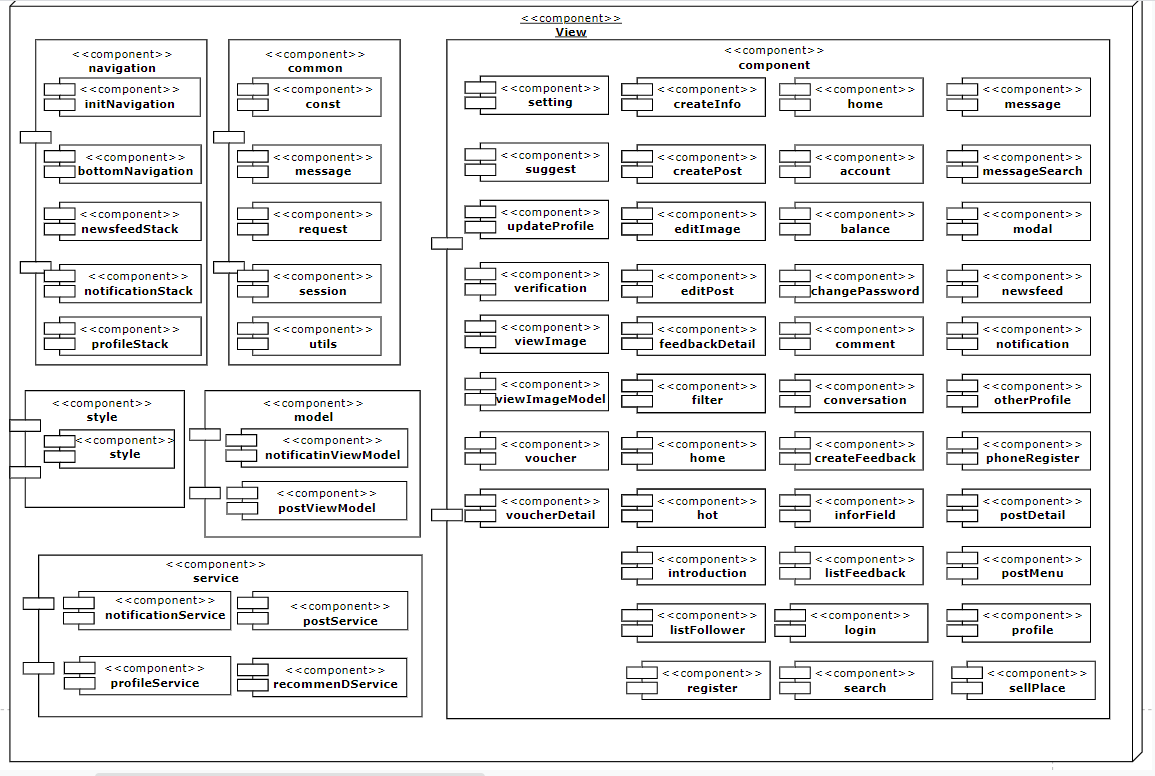
ZaloPay is an e-wallet platform and online payment system developed by VNG to deploy in-app payments and scan QR codes for payment on mobile devices, allowing users to do so. transactions using phones and tablets running Android and iOS. For many years (from 2018 to 2020), many channels have been introduced as the top 5 recommended e-wallets in Vietnam

**4.2.**  **Component Diagram**

****

## **4.3. Detailed Description of Component**

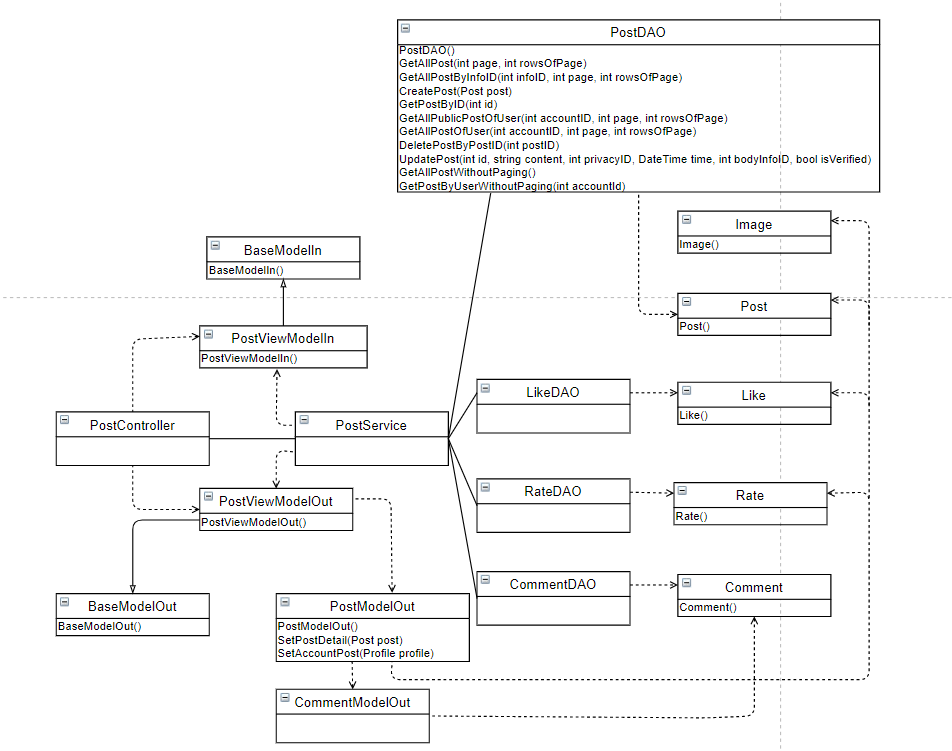
***4.3.1. View***



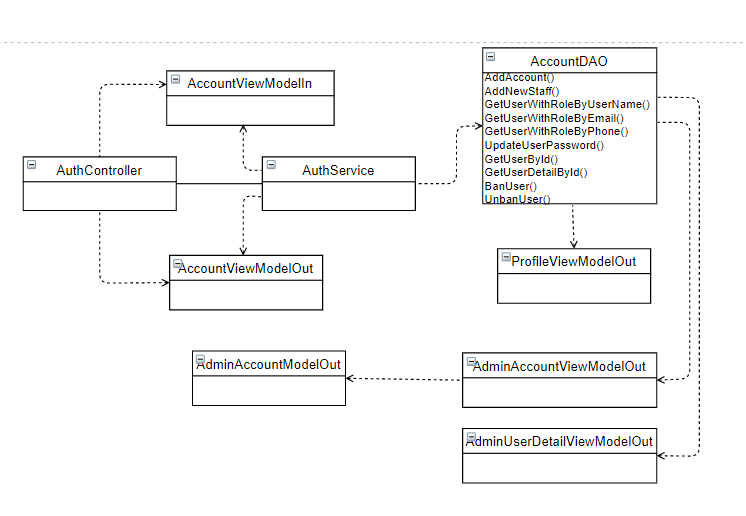
## **4.4. Class Diagram**

***4.4.1. Class diagram***

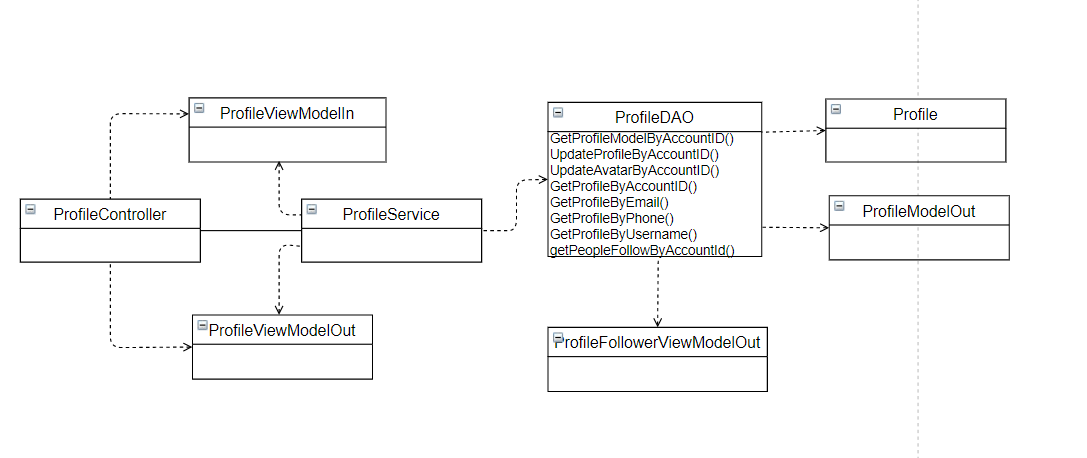
***4.4.1.1. Post Class Diagram***

******

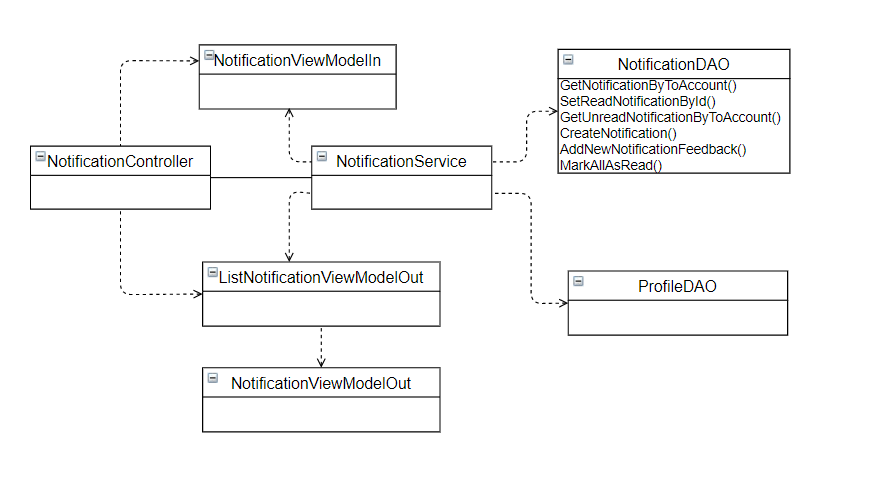
***4.4.1.2. Account Class Diagram***

******

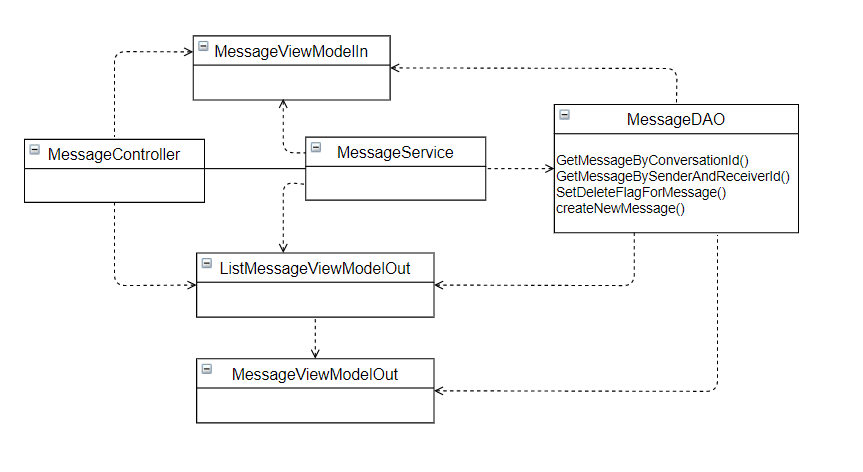
***4.4.1.3. Profile Class Diagram***

******

***4.4.1.4. Notification Class Diagram***

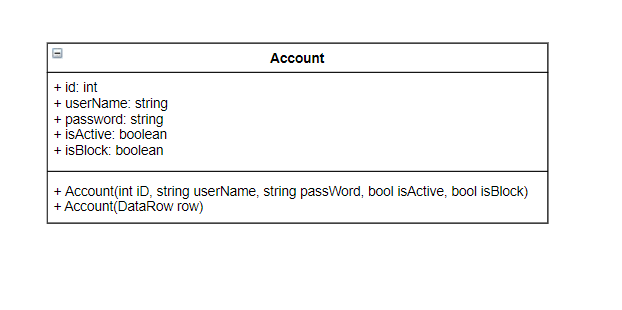
******

***4.4.1.5. Message Class Diagram***

******

***4.4.2. Models***

***4.4.2.1 Account***

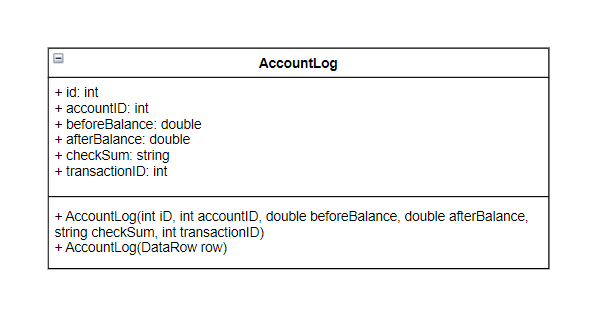
****

**Class specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | id | int | ID of account |
| 02 | username | string | User Name of account |
| 03 | password | string | Password of account |
| 04 | isActive | boolean | Status active of account |
| 05 | isBlock | boolean | Status block of account |

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | Account(int iD, string userName, string password, bool isActive, bool isBlock) | Default constructor |
| 02 | Account(DataRow row) | Another constructor |

***4.4.2.2 AccountLog***

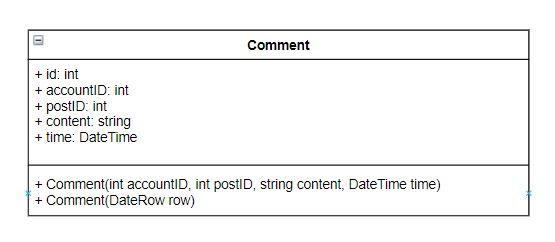
****

**Class specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | id | int | ID of log |
| 02 | accountID | int | ID of account log |
| 03 | beforeBalance | double | Balance of account before transaction |
| 04 | afterBalance | double | Balance of account after transaction |
| 05 | transactionID | int | ID of transacation |

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | AccountLog(int iD, int accountID, double beforeBalance, double afterBalance, int transactionID) | Default constructor |
| 02 | AccountLog(DataRow row) | Another constructor |

***4.4.2.3 Comment***

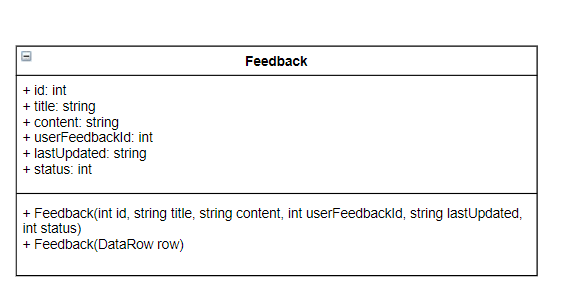
******

**Class specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | id | int | Id of comment |
| 02 | accountID | int | Id of account comment |
| 03 | postID | int | Id of the post |
| 04 | content | string | Content of the comment |
| 05 | time | DateTime | Time comment is created |

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | Comment(int id, int accountID, int postID, string content, Datetime time) | Default constructor |
| 02 | Comment(DataRow row) | Another constructor |

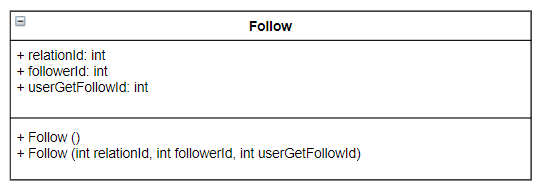
***4.5.2.4 Feedback***

**Class specification****

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | id | int | Id of the feedback |
| 02 | title | string | Title of the feedback |
| 03 | content | string | Content of the feedback |
| 04 | userFeedbackId | id | Id of the user have send feedback |
| 05 | lastUpdated | string | The final date update the feedback |
| 06 | status | int | Status of Feedback :   * Status = 1 : Resolved * Status = 2 : Unresolved * Status = 3 : Rejected |

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | Feedback(int id,string title, string content, int userFeedbackId, string lastUpdated, int status) | Default constructor |
| 02 | Feedback(DataRow row) | Another constructor |

***4.4.2.5 Follow***

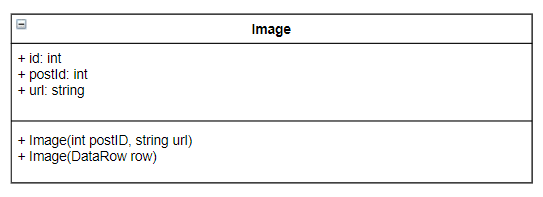
******

**Class specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | relationId | int | Relation of the Following :   * Relation = 1 : Follow * Relation = 2 : Block |
| 02 | followerId | int | Id of the user follow |
| 03 | userGetFollowId | int | Id of the user is followed |

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | Follow() | Default constructor |
| 02 | Follow(int relationId, int followerId, int userGetFollowId) | Another constructor |

***4.4.2.6 Image***

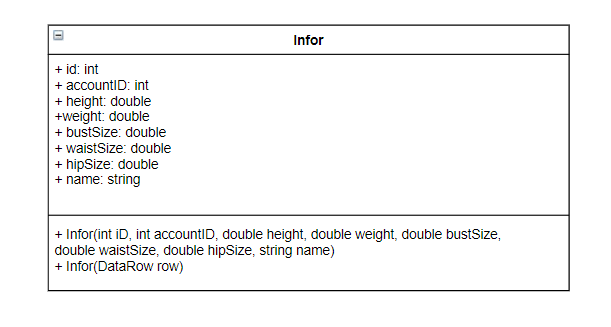
******

**Class specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | id | int | Id of the image |
| 02 | postId | int | Id of the post contain the image |
| 03 | url | string | The path to the image file |

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | Image(int postID, string url) | Default constructor |
| 02 | Image(DataRow row) | Another constructor |

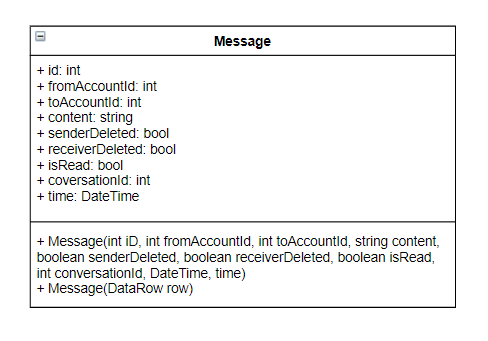
***4.4.2.7 Infor***

******

**Class specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | id | int | Id of the Infor |
| 02 | accountID | int | Id of the account have the infor |
| 03 | height | double | Height of the person |
| 04 | weight | double | Weight of the person |
| 05 | bustSize | double | The size of the butt |
| 06 | waistSize | double | The size of the waist |
| 07 | hipSize | double | Size of the hip |
| 08 | name | string | Name of the Infor |

***4.4.2.8 Message***

******

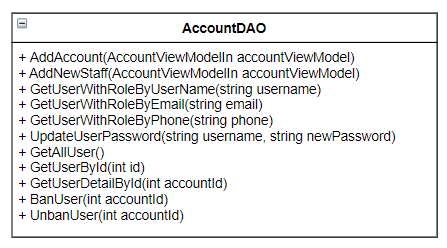
**Class specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **NO** | **Attribute** | **Type** | **Description** |
| 01 | id | int | Id of the message |
| 02 | fromAccountId | int | Id of sender |
| 03 | toAccountId | int | Id of the receiver |
| 04 | content | string | Content of the message |
| 05 | senderDeleted | boolean | Status of the message on sender :   * Status = 0 : UnDeleted * Status = 1 : Deleted |
| 06 | receiverDeleted | boolean | Status of the message on receiver:   * Status = 0 : Undeleted * Status = 1 : Deleted |
| 07 | isRead | boolean | Status of the message on receiver is read:   * Relation = 0 : Unread * Relation = 1 : Read |
| 08 | conversationId | int | Id of the conversation |
| 09 | time | DateTime | Time of the message |

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | Message(int iD, int fromAccountId, int toAccountId, string content,boolean senderDeleted, boolean receiverDeleted, boolean isRead,int conversationId, DateTime, time) | Default constructor |
| 02 | Message(DataRow row) | Another constructor |

**4.5.3. DAO**

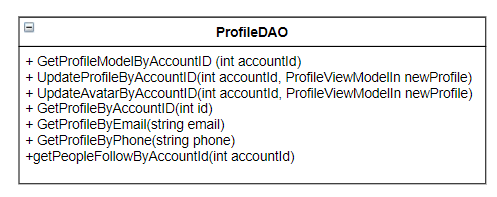
***4.5.3.1 AccountDAO***

****

**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | AddAccount(AccountViewModelIn accountViewModel) | Create an account for users |
| 02 | AddNewStaff(AccountViewModelIn) | Create a new account for staff manager |
| 03 | GetUserWithRoleByUserName(string username) | Get information of users with their username |
| 04 | GetUserWithRoleByEmail(string username) | Get information of users with their Email |
| 05 | GetUserWithRoleByPhone(string username) | Get information of users with their Phone |
| 06 | UpdateUserPassword(string username, string newPassword) | Change user password |
| 07 | GetAllUser() | Get all user in system |
| 08 | GetUserById(int id) | Get an user’s account and profile by his/her ID |
| 09 | GetUserDetailById(int id) | Get an user’s profile and statistics |
| 10 | BanUser(int accountId) | Ban an user |
| 11 | UnbanUser(int accountId) | Unban an user |

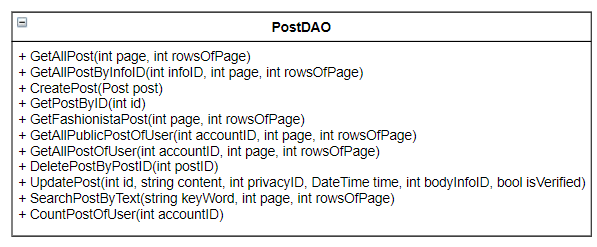
***4.5.3.2. ProfileDAO***

****

**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetProfileModelByAccountID (int accountId) | Get an custom profile by account id |
| 02 | UpdateProfileByAccountID(int accountId, ProfileViewModelIn newProfile) | Update an profile by account id |
| 03 | UpdateAvatarByAccountID(int accountId, ProfileViewModelIn newProfile) | Update an user’s profile avatar by account id |
| 04 | GetProfileByAccountID(int id) | Get user’s profile by account id |
| 05 | GetProfileByEmail(string email) | Get user’s profile by user’s email |
| 06 | GetProfileByPhone(string phone) | Get user’s profile by user’s phone |
| 07 | getPeopleFollowByAccountId(int accountId) | Get people follow an account by account id |

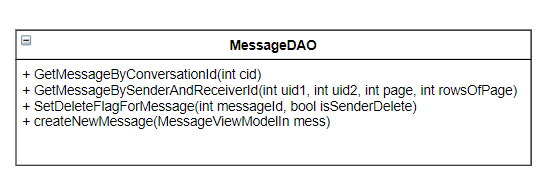
***4.5.3.3 PostDAO***

******

**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetAllPost(int page, int rowsOfPage) | Get all public post in the system |
| 02 | GetAllPostByInfoID(int infoID, int page, int rowsOfPage) | Get all public post have body measurement equal infor id |
| 03 | CreatePost(Post post) | Create an post |
| 04 | GetPostByID(int id) | Get an post by its id |
| 05 | GetFashionistaPost(int page, int rowsOfPage) | Get fashionista’s post |
| 06 | GetAllPublicPostOfUser(int accountID, int page, int rowsOfPage) | Get all public post of user in the system |
| 07 | DeletePostByPostID(int postID) | Delete a post by its id |
| 08 | UpdatePost(int id, string content, int privacyID, DateTime time, int bodyInfoID, bool isVerified) | Update a post |
| 09 | SearchPostByText(string keyWord, int page, int rowsOfPage) | Search a post by keyword |
| 10 | CountPostOfUser(int accountID) | Get number of an user’s post |

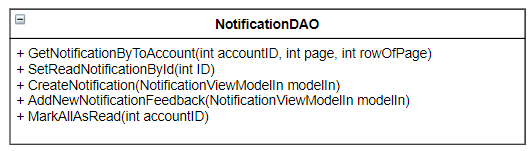
***4.5.3.4 MessageDAO***

****

**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetMessageByConversationId(int cid) | Get messages by conversation id |
| 02 | GetMessageBySenderAndReceiverId(int uid1, int uid2, int page, int rowsOfPage) | Get messages by sender and receiver |
| 03 | SetDeleteFlagForMessage(int messageId, bool isSenderDelete) | Delete an message in an conversation |
| 04 | createNewMessage(MessageViewModelIn mess) | Create a new message then create an conversation if it not belong into an existed conversation |

***4.5.3.5 NotificationDAO***

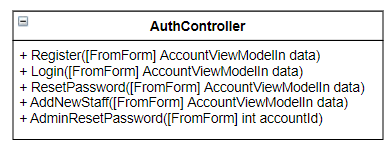
****

**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetNotificationByToAccount(int accountID, int page, int rowOfPage) | Get messages by conversation id |
| 02 | SetReadNotificationById(int ID) | Get messages by sender and receiver |
| 03 | CreateNotification(NotificationViewModelIn modelIn) | Delete an message in an conversation |
| 04 | AddNewNotificationFeedback(NotificationViewModelIn modelIn) | Create a new message then create an conversation if it not belong into an existed conversation |
| 05 | MarkAllAsRead(int accountID) |  |

**4.5.4 Controller**

***4.5.4.1 AuthController***

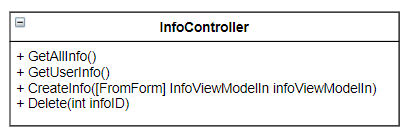
******

**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | Register([FromForm] AccountViewModelIn data) | Register for an account |
| 02 | Login([FromForm] AccountViewModelIn data) | Login for an user to the system |
| 03 | ResetPassword([FromForm] AccountViewModelIn data) | Reset password for an user |
| 04 | AddNewStaff([FromForm] AccountViewModelIn data) | Add new staff for admin web server |
| 05 | AdminResetPassword([FromForm] int accountId) | Reset password for admin |

## 

## ***4.5.4.2 InfoController***

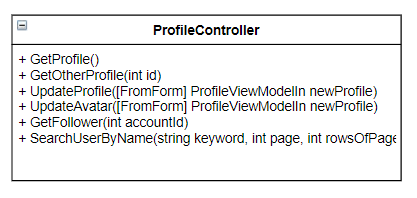
****

**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetAllInfo() | Get all body measurement of all users |
| 02 | GetUserInfo() | Get body measurement of an user |
| 03 | CreateInfo([FromForm] InfoViewModelIn infoViewModelIn) | Create an body measurement for an user |
| 4 | Delete(int infoID) | Delete an body measurement of an user |

## 

## ***4.5.4.3 ProfileController***



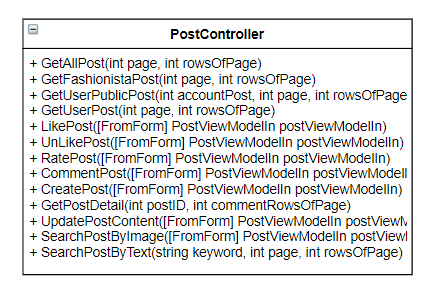
**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetProfile() | Get user’s profile |
| 02 | GetOtherProfile(int id) | Get another user’s profile |
| 03 | UpdateProfile([FromForm] ProfileViewModelIn newProfile) | Update user’s profile |
| 04 | UpdateAvatar([FromForm] ProfileViewModelIn newProfile) | Update user’s avatar |
| 05 | GetFollower(int accountId) | Get follower of user |
| 06 | SearchUserByName(string keyword, int page, int rowsOfPage) | Search user by name |

## 

## 

## ***4.5.4.4 PostController***

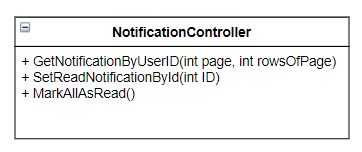


**Class specification**

|  |  |  |
| --- | --- | --- |
| NO | Method | Description |
| 01 | GetAllPost(int page, int rowsOfPage) | Get all users’s posts |
| 02 | GetFashionistaPost(int page, int rowsOfPage) | Get fashionistas’s posts |
| 03 | GetUserPublicPost(int accountPost, int page, int rowsOfPage) | Get public user’s posts |
| 05 | LikePost([FromForm] PostViewModelIn postViewModelIn) | Like a post |
| 06 | UnLikePost([FromForm] PostViewModelIn postViewModelIn) | Unlike a post |
| 07 | RatePost([FromForm] PostViewModelIn postViewModelIn) | Rate a post |
| 08 | CommentPost([FromForm] PostViewModelIn postViewModelIn) | Comment a post |
| 09 | CreatePost([FromForm] PostViewModelIn postViewModelIn) | Create a new post |
| 10 | GetPostDetail(int postID, int commentRowsOfPage) | Get detail of a post |
| 11 | UpdatePostContent([FromForm] PostViewModelIn postViewModelIn) | Update a post’s content |
| 12 | SearchPostByImage([FromForm] PostViewModelIn postViewModelIn) | Search a post by Image |
| 13 | SearchPostByText(string keyword, int page, int rowsOfPage) | Search a post by keyword |

## 

## ***4.5.4.5 NotificationController***

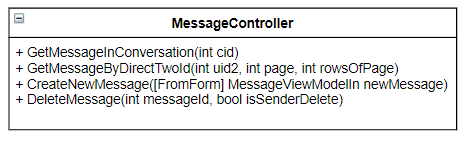


**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetNotificationByUserID(int page, int rowsOfPage) | Get notification of user |
| 02 | SetReadNotificationById(int ID) | Set a notification is read |
| 03 | MarkAllAsRead() | Set all notifications are read |

## 

## ***4.5.4.6 MessageController***

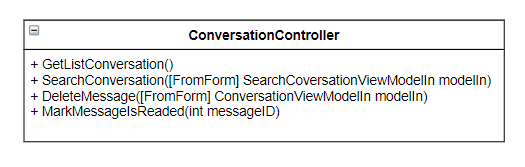


**Class specification**

|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetMessageInConversation(int cid) | Get messages in a conversation |
| 02 | GetMessageByDirectTwoId(int uid2, int page, int rowsOfPage) | Get messages by user |
| 03 | CreateNewMessage([FromForm] MessageViewModelIn newMessage) | Send a message to another user |
| 04 | DeleteMessage(int messageId, bool isSenderDelete) | Delete a message |

## 

## ***4.5.4.7 ConversationController***



**Class specification**

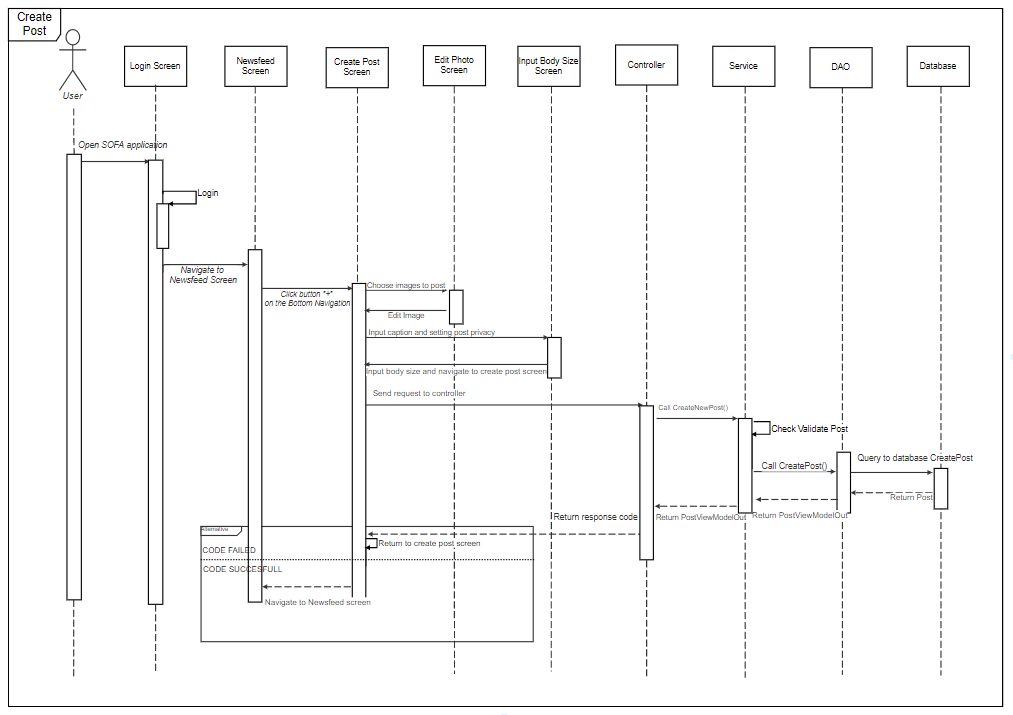
|  |  |  |
| --- | --- | --- |
| **NO** | **Method** | **Description** |
| 01 | GetListConversation() | Get list user’s conversation |
| 02 | SearchConversation([FromForm] SearchCoversationViewModelIn modelIn) | Search conversation by username |
|  | DeleteMessage([FromForm] ConversationViewModelIn modelIn) | Delete a conversation |
|  | MarkMessageIsReaded(int messageID) | Mark conversation is read |

## 

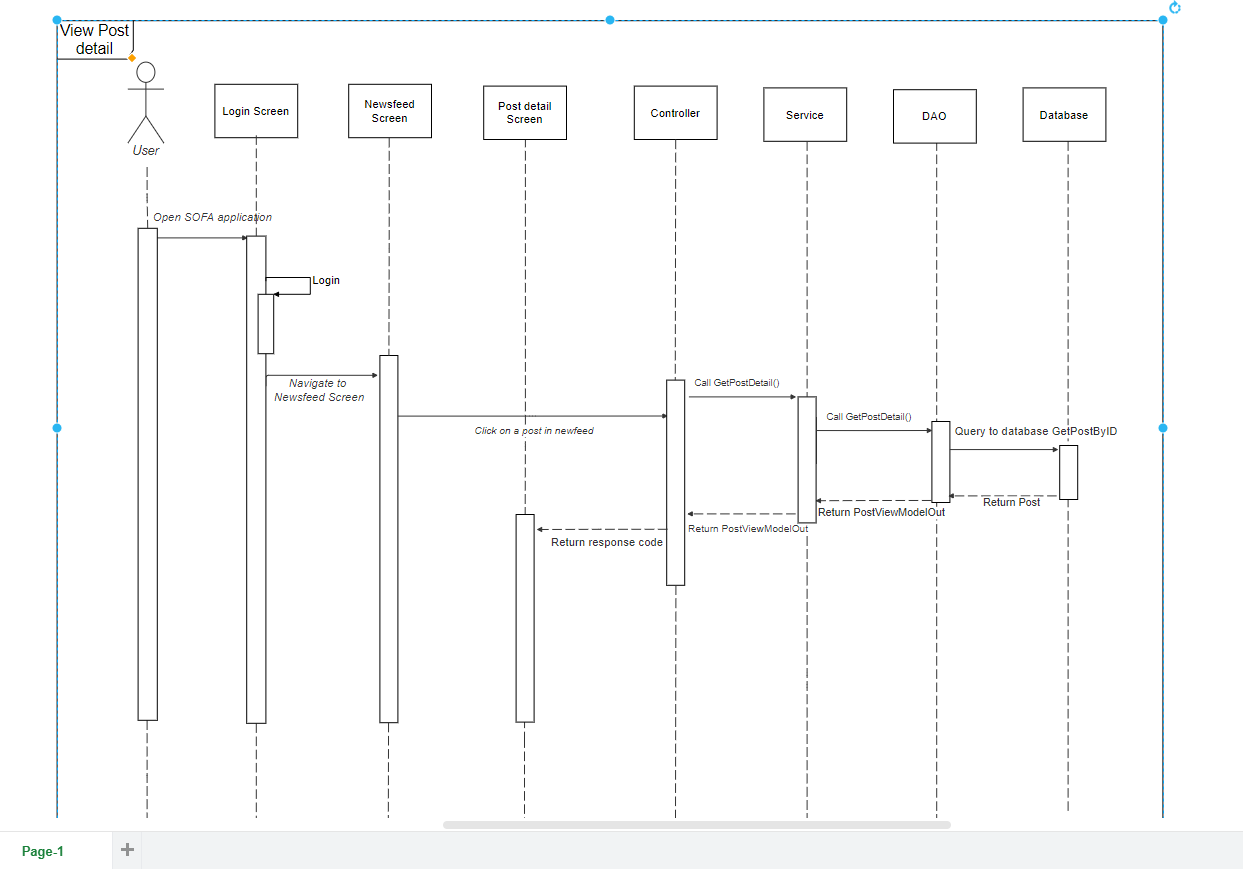
## 

## **4.6. Sequence Diagram**

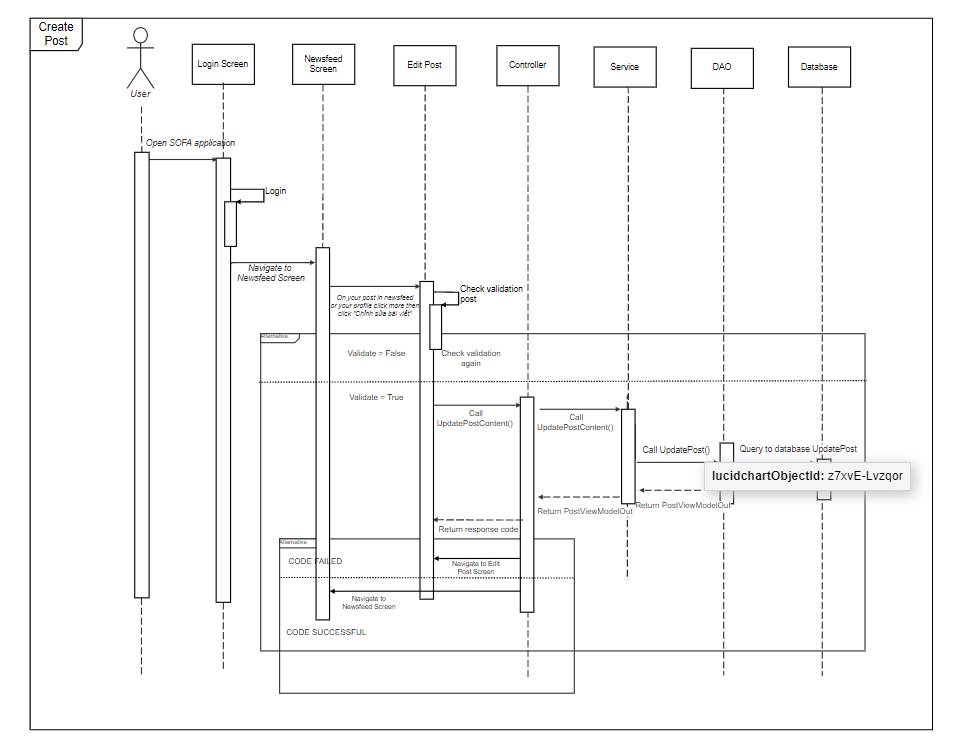
***4.6.1. Create New Post***

******

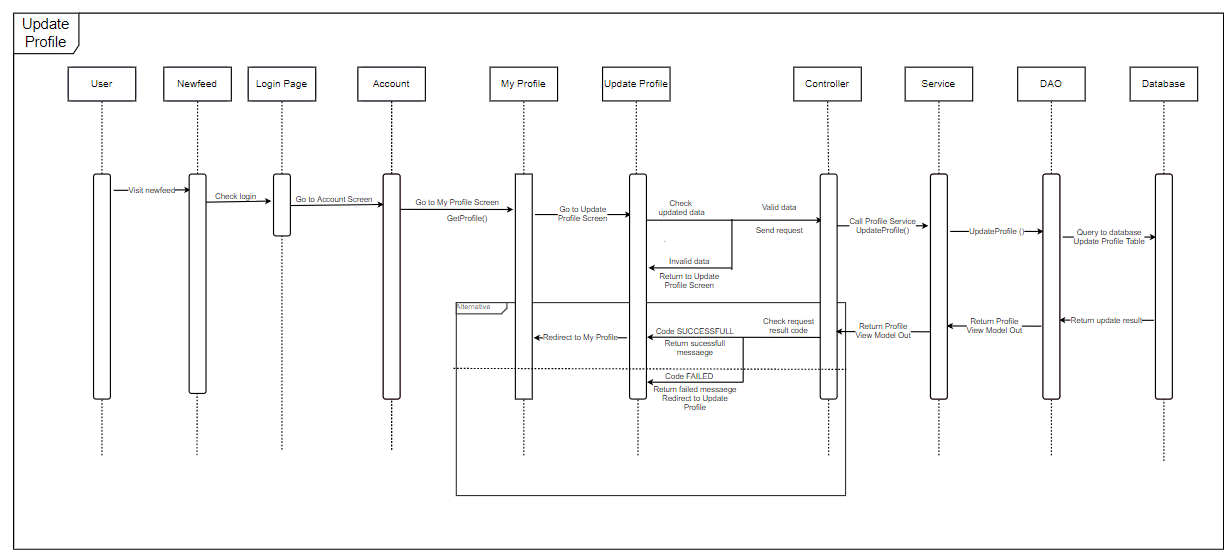
***4.6.2. View post detail***

******

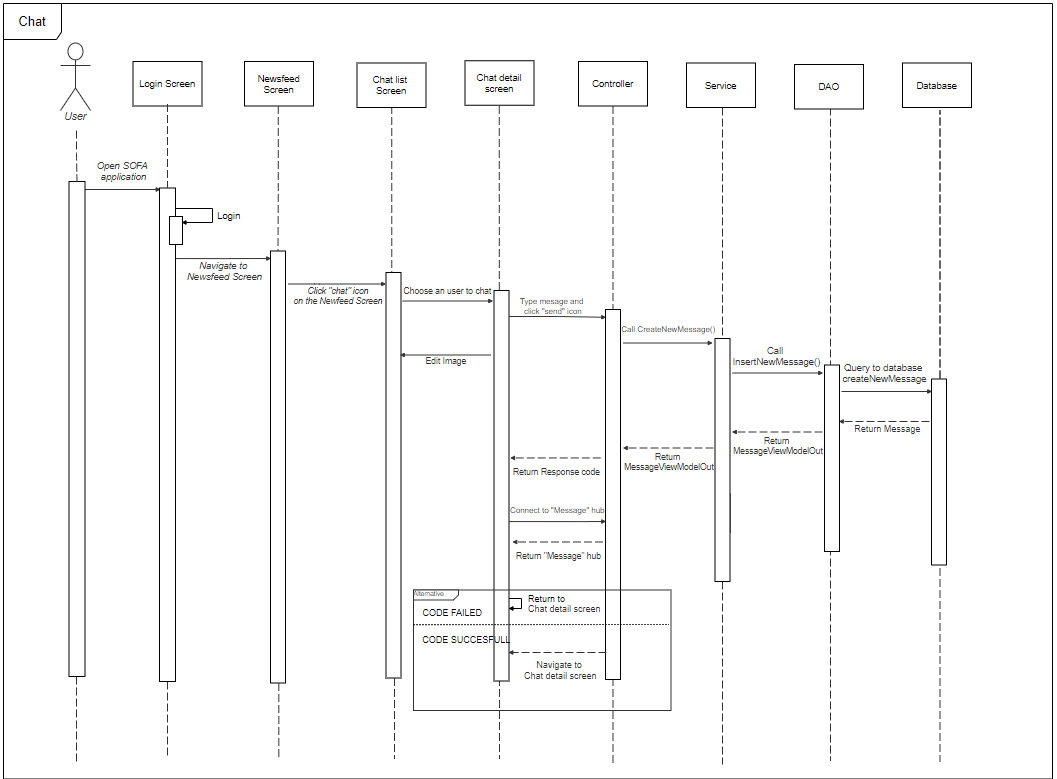
***4.6.3. View edit post***

******

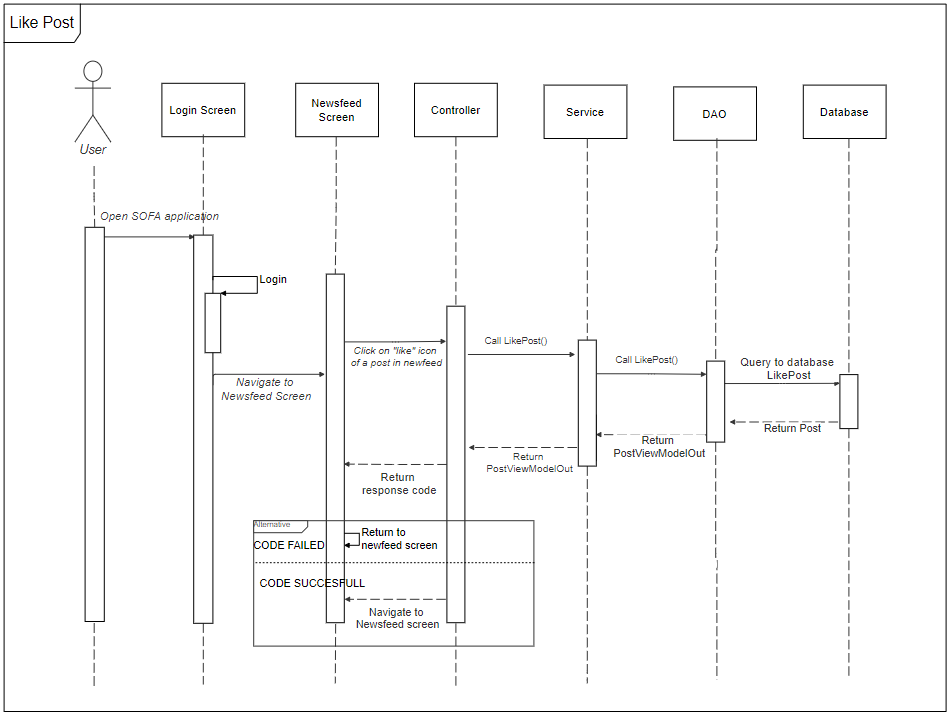
***4.6.4. Update Profile***

******

***4.6.5. Chat***

******

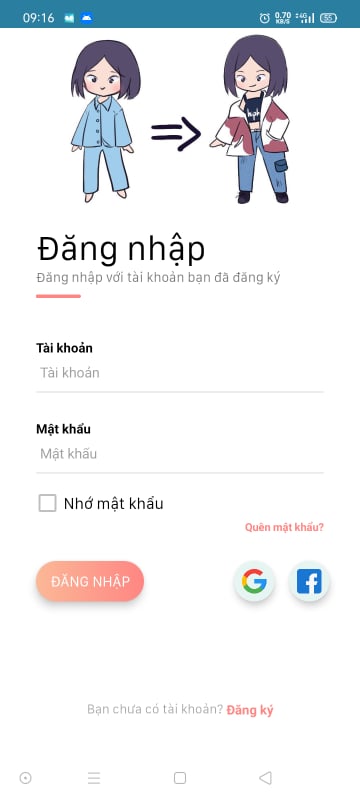
***4.6.6. Like Post***

******

## **4.7. User Interface Design**

## ***4.7.1. Screen Image***

4.7.1.1 Login



4.7.1.2 Register



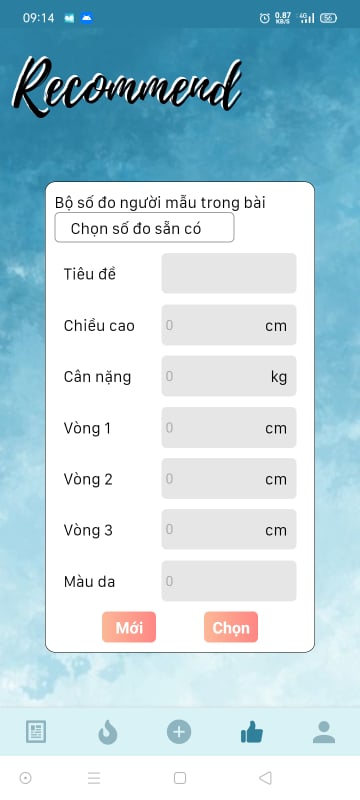
4.7.1.3 Newsfeed



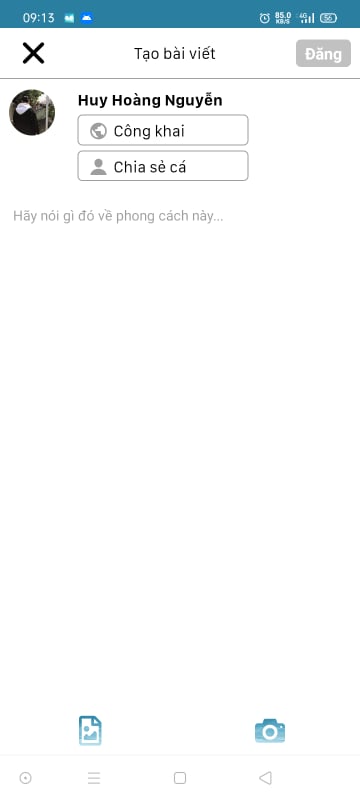
4.7.1.4 Hot

## 

4.7.1.5 Recommend



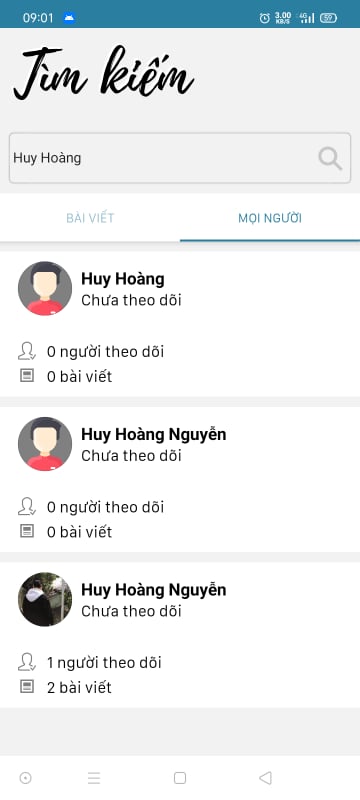
4.7.1.6 Create Post



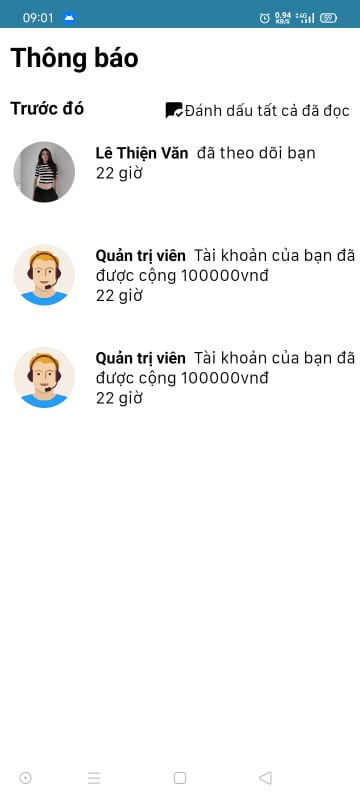
4.7.1.7 Post Detail



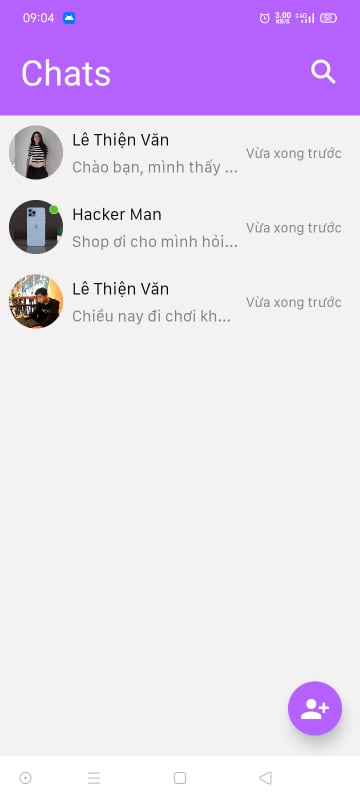
4.7.1.8 Search



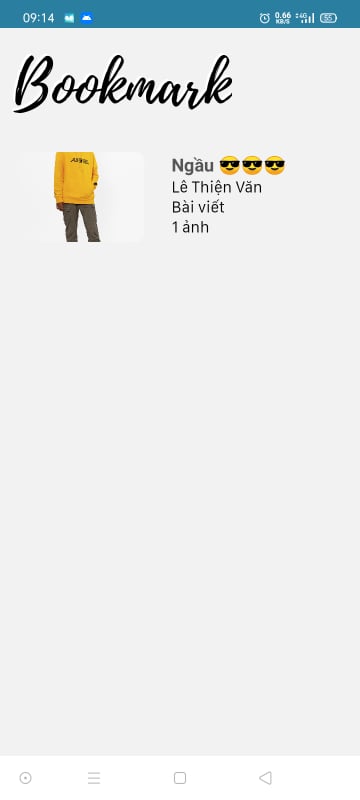
4.7.1.9 Notification



4.7.1.10 Chat



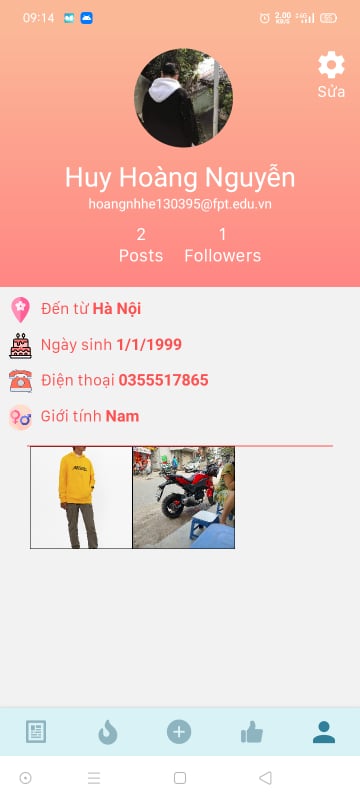
4.7.1.11 Bookmark



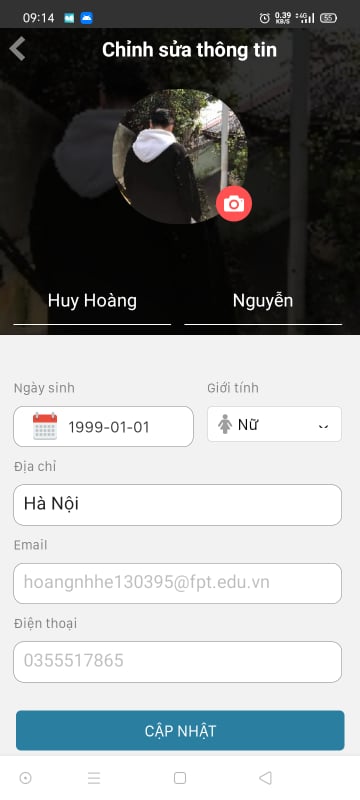
4.7.1.12 Account



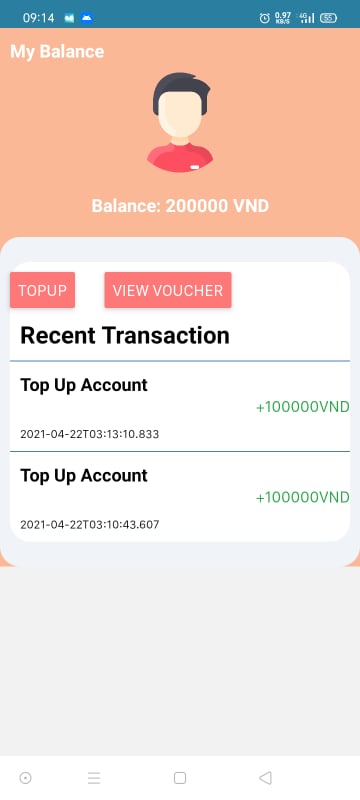
4.7.1.13 Profile



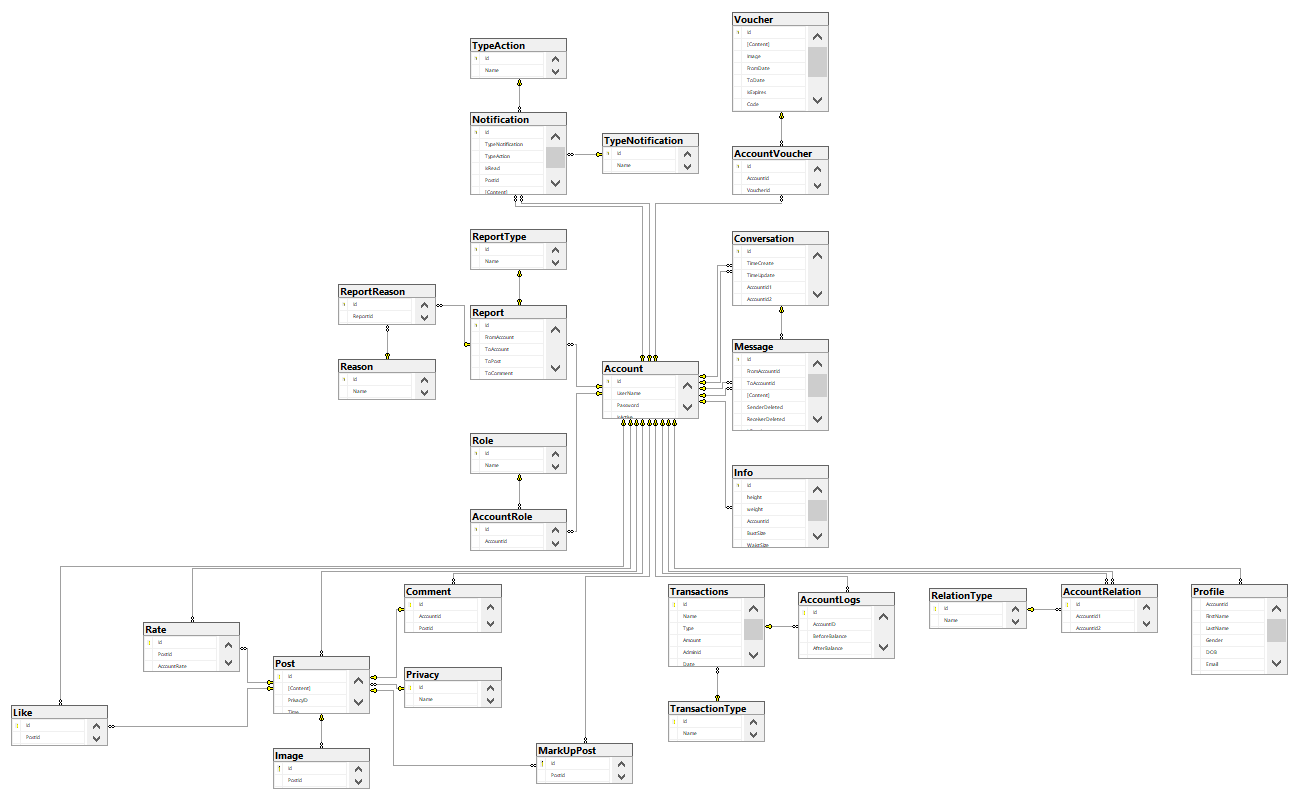
4.7.1.14 Edit Profile



4.7.1.15 Balance



## **4.8. Database Design**

**