CHAPTER 4: SOFTWARE DESIGN DESCRIPTION

4.1. Purpose

4.2. Architecture Overview

4.2.1. System Architecture

#### MVC model

Model–view–controller (MVC) is a software architectural pattern for implementing user interfaces on computers. It divides a given application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented to, and accepted from, the user



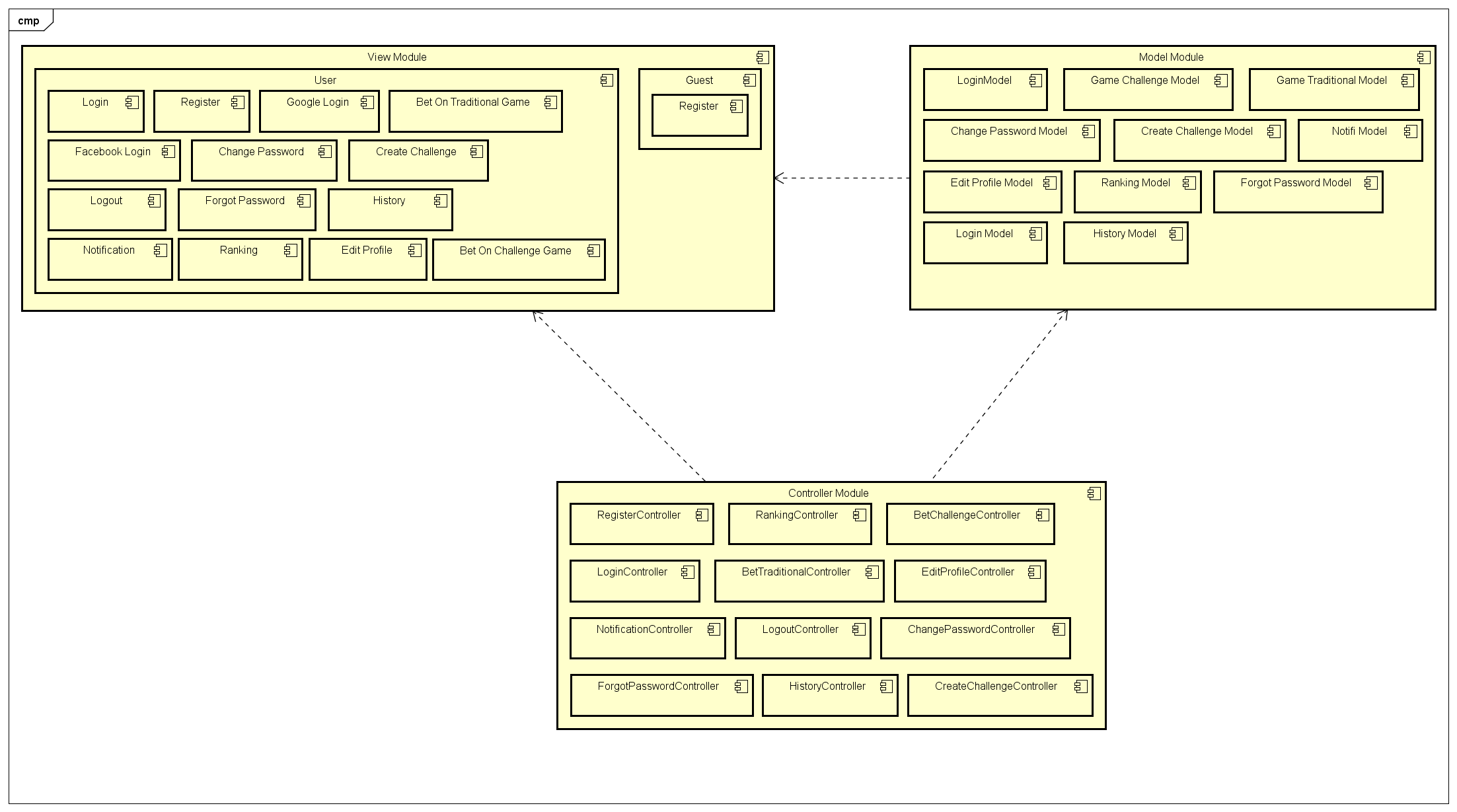
MVC Model

* **Model:** Model represents shape of the data and business logic. It maintains the data of the application. Model objects retrieve and store model state in a database.
* **View:** View is a user interface. View display data using model to the user and also enables them to modify the data.
* **Controller:** Controller handles the user request. Typically, user interact with View, which in-tern raises appropriate URL request, this request will be handled by a controller. The controller renders the appropriate view with the model data as a response.

#### Advantages and disadvantages of MVP model

* **Advantages**
  + **Faster development process:** MVC supports rapid and parallel development. With MVC, one programmer can work on the view while other can work on the controller to create business logic of the web application. The application developed using MVC can be three times faster than application developed using other development patterns.
  + **Ability to provide multiple views:** In the MVC Model, you can create multiple views for a model. Code duplication is very limited in MVC because it separates data and business logic from the display.
  + **Support for asynchronous technique:** MVC also supports asynchronous technique, which helps developers to develop an application that loads very fast.
  + **Modification does not affect the entire model:** Modification does not affect the entire model because model part does not depend on the views part. Therefore, any changes in the Model will not affect the entire architecture.
  + **MVC model returns the data without formatting:** MVC pattern returns data without applying any formatting so the same components can be used and called for use with any interface.
  + **SEO friendly Development platform**: Using this platform, it is very easy to develop SEO-friendly URLs to generate more visits from a specific application.
* **Disadvantages**
  + Increased complexity
  + Inefficiency of data access in view
  + Difficulty of using MVC with modern user interface.
  + Need multiple programmers
  + Knowledge on multiple technologies is required.
  + Developer have knowledge of client side code and html code.

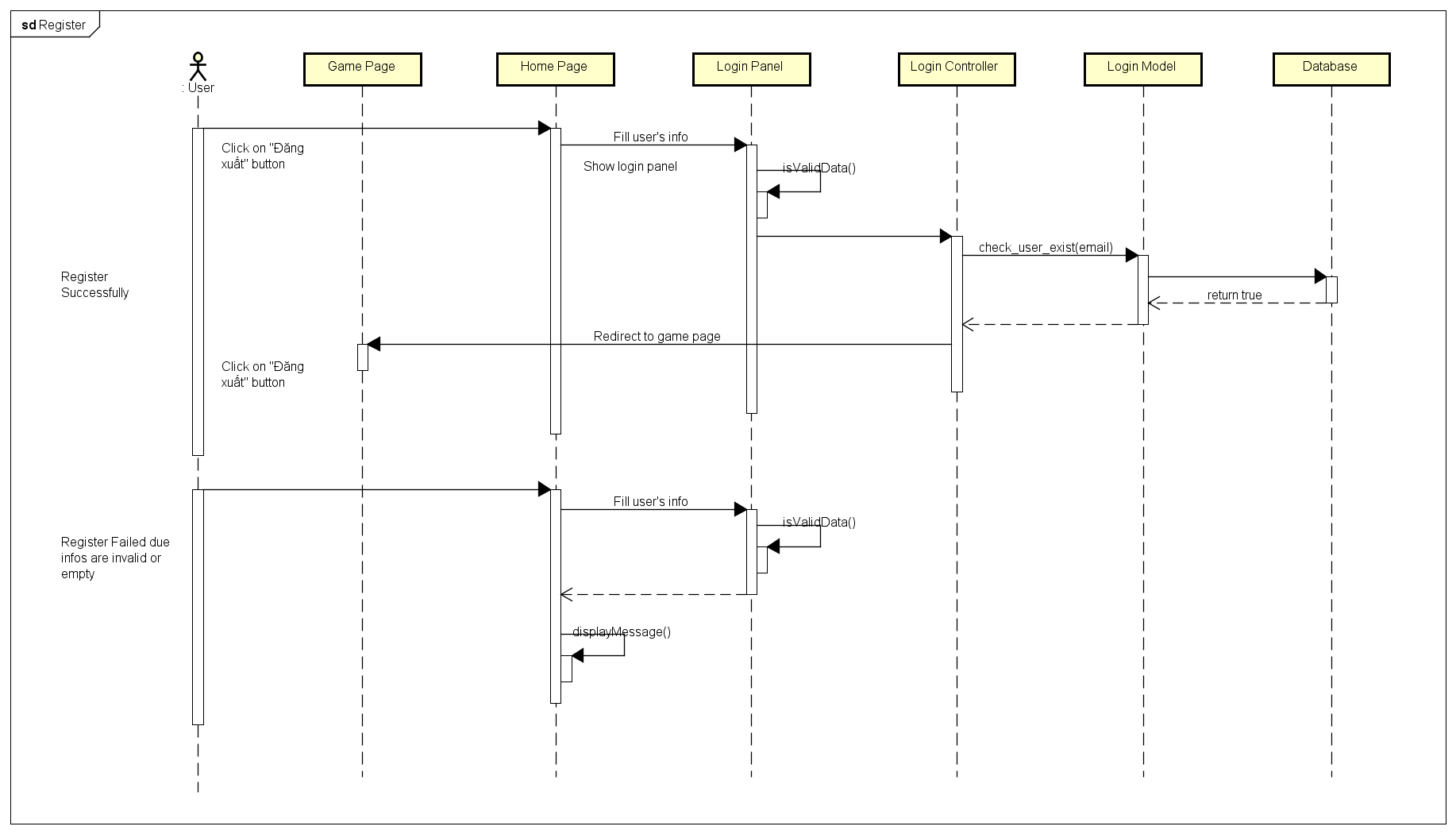
4.3. Component Diagram

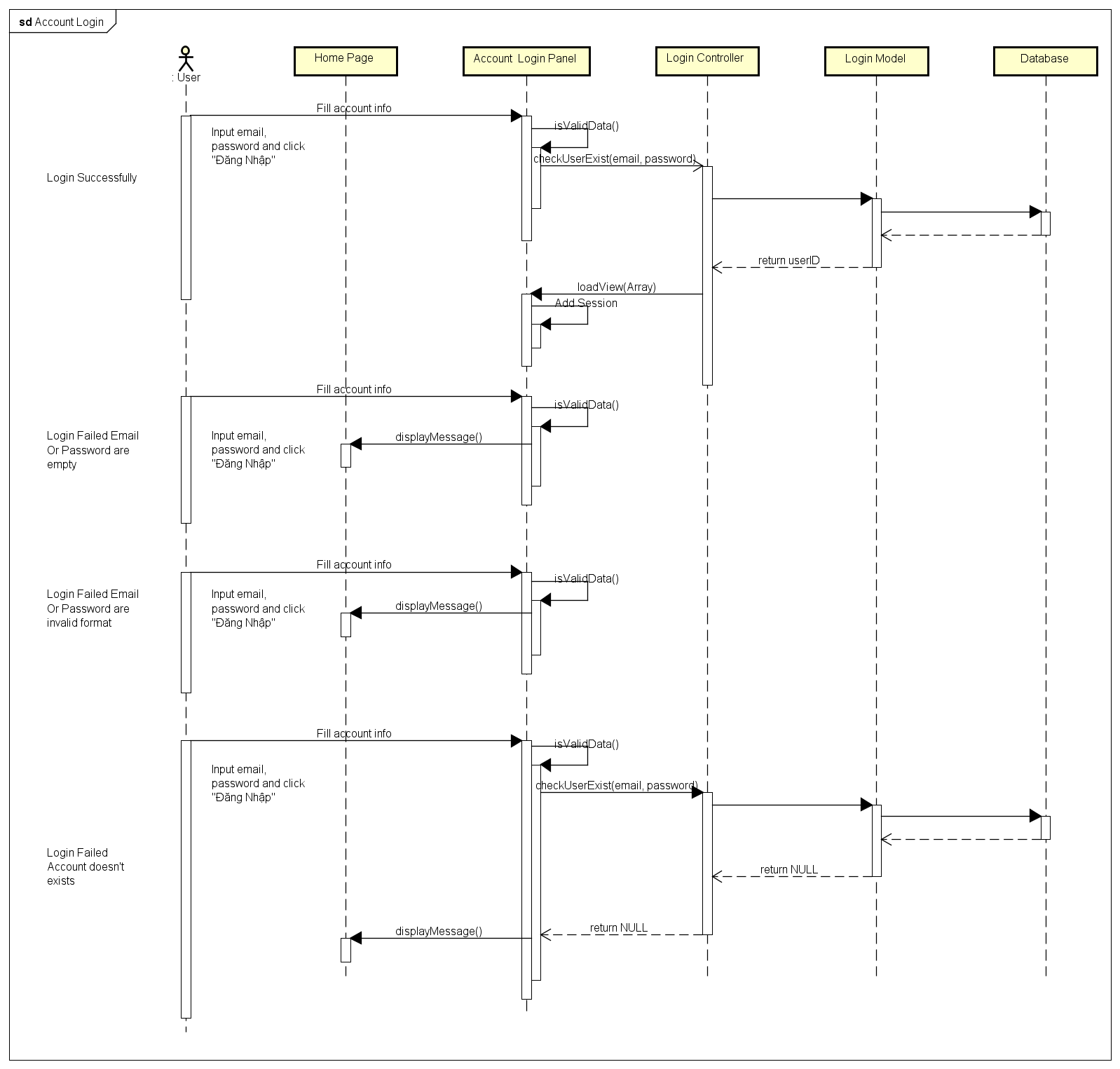


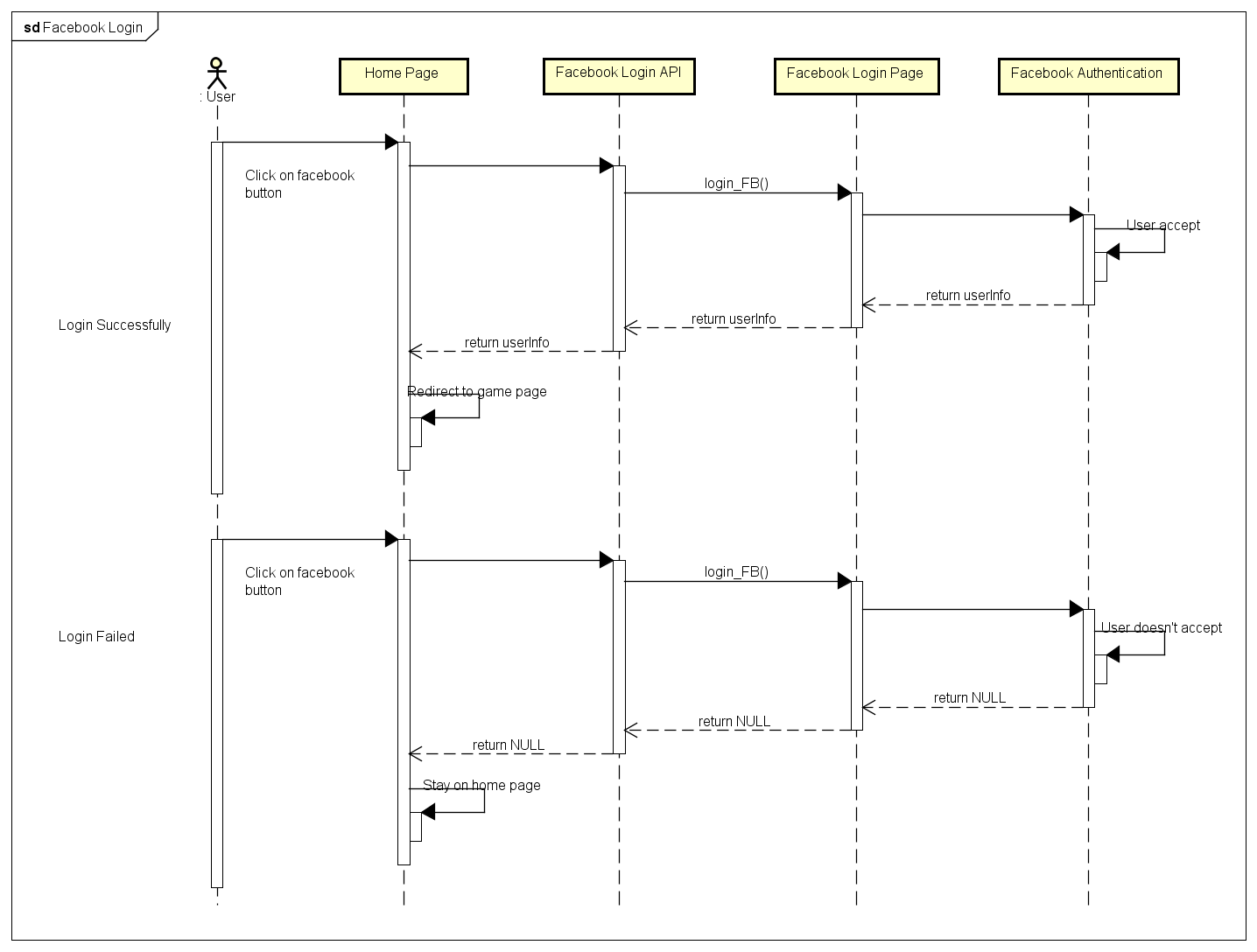
4.4.A. Sequence Diagram

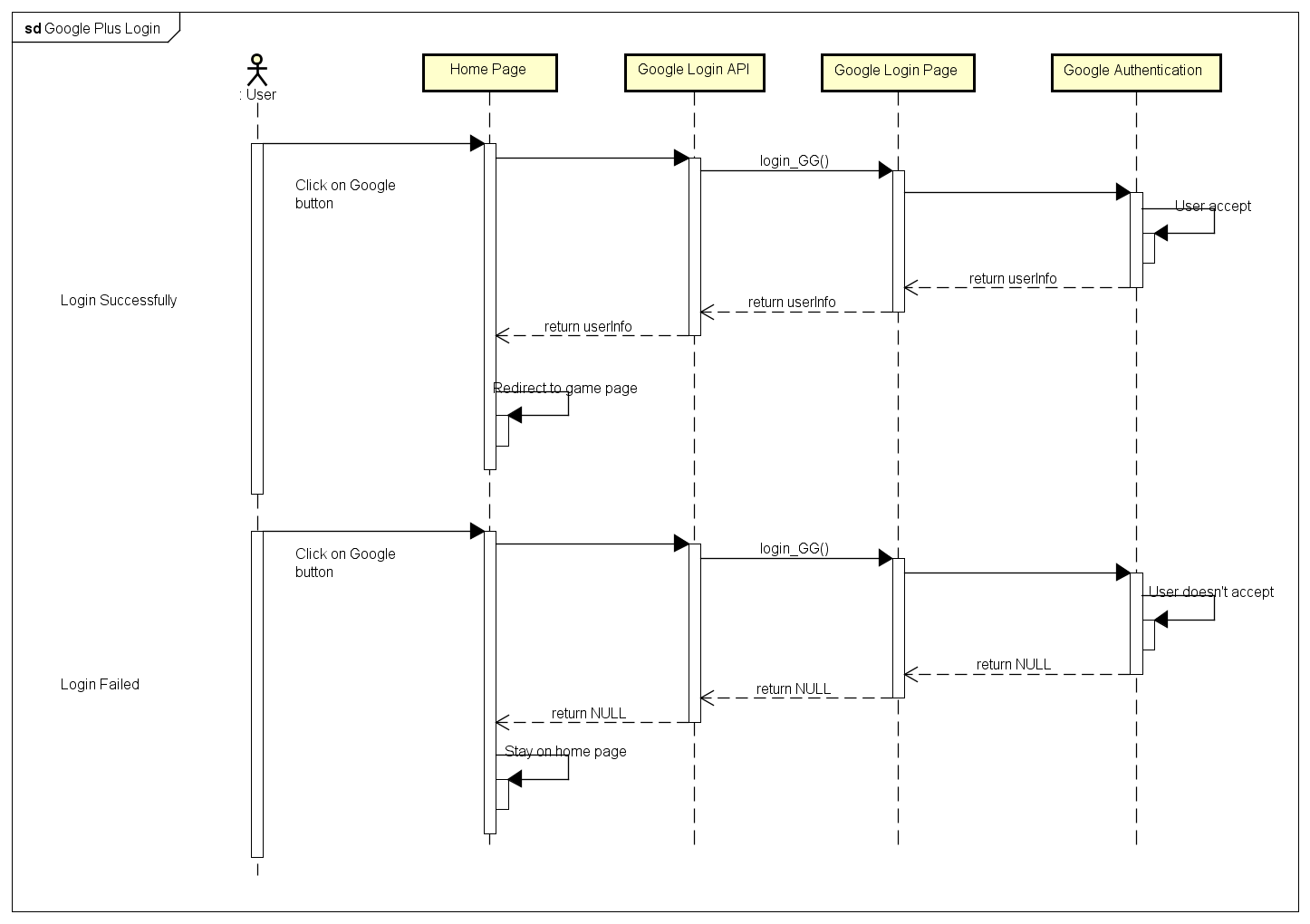
Several sequence diagrams are shown in the following:

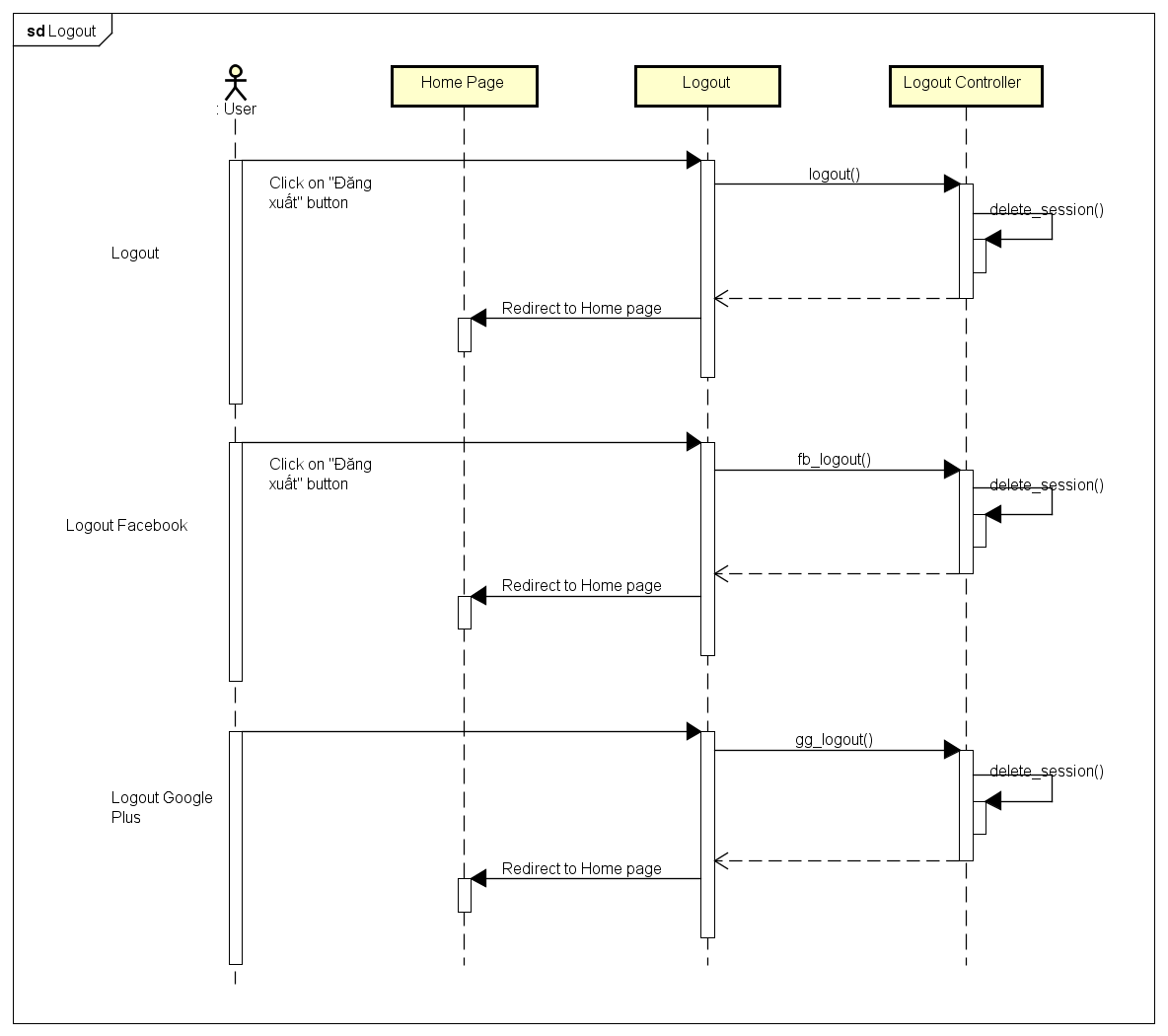
4.4.A.1. Register

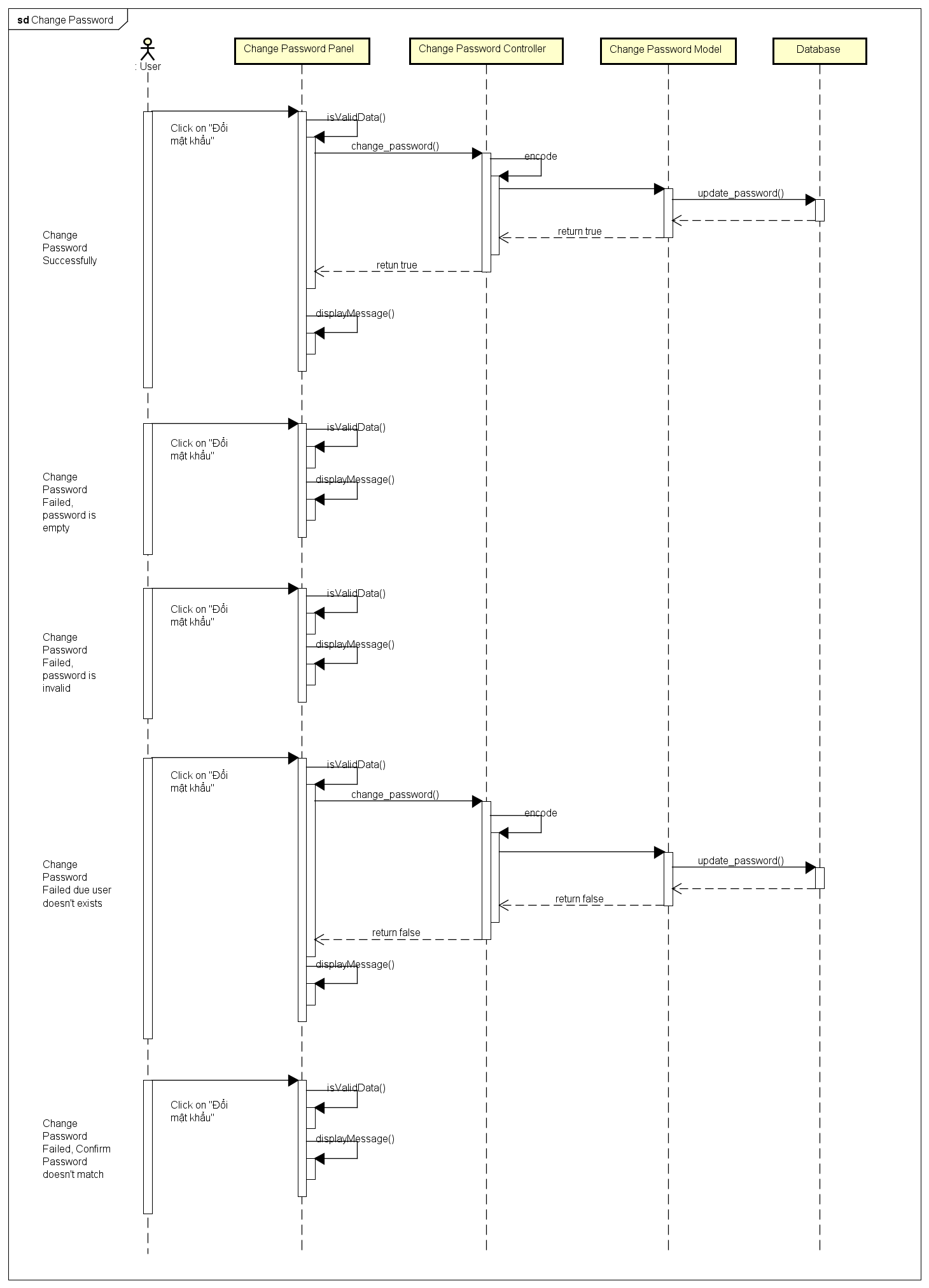


4.4.A.2. Login

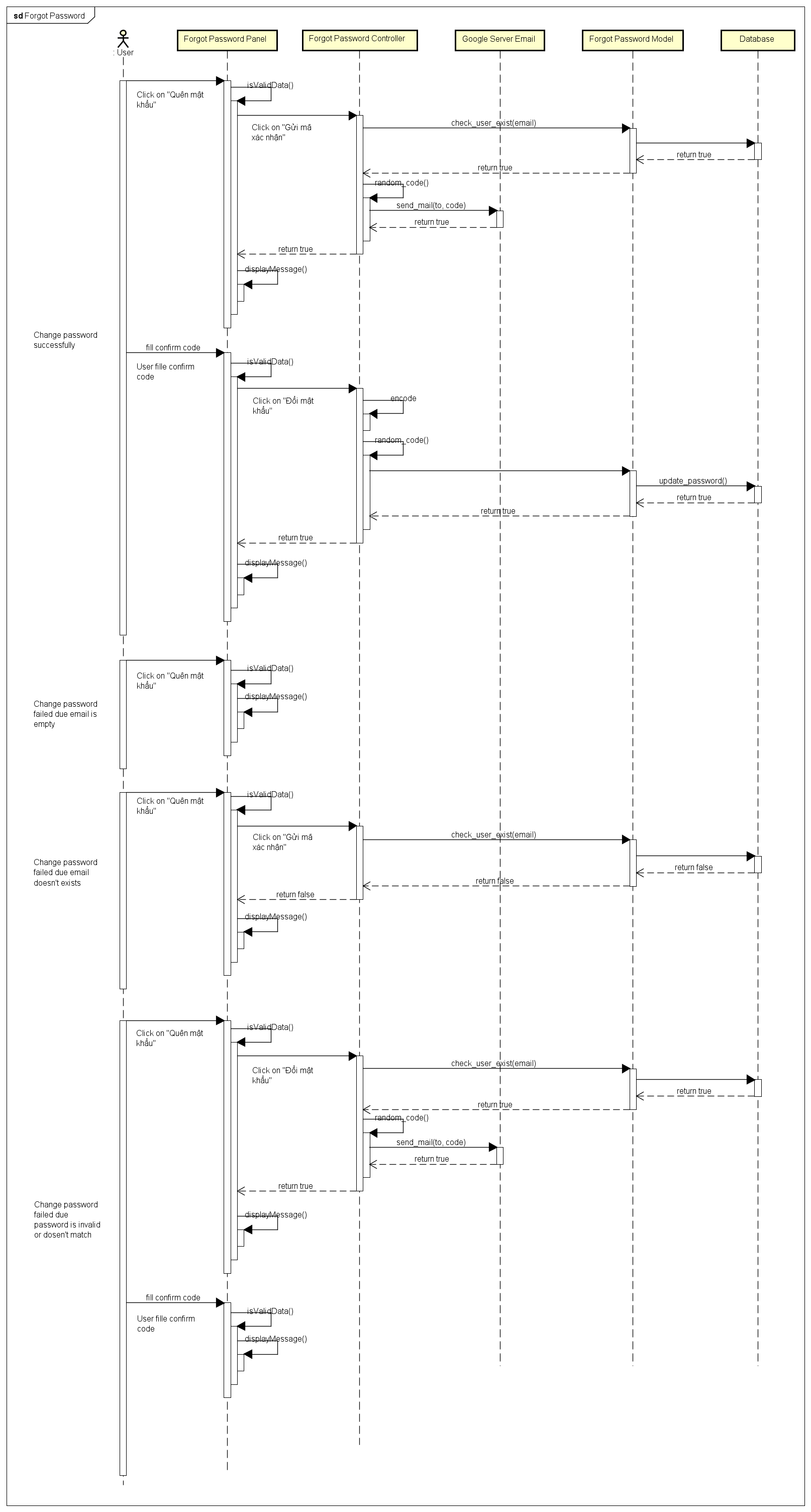
4.4.A.3. Login Using Facebook

4.4.A.4. Login Using Google Plus

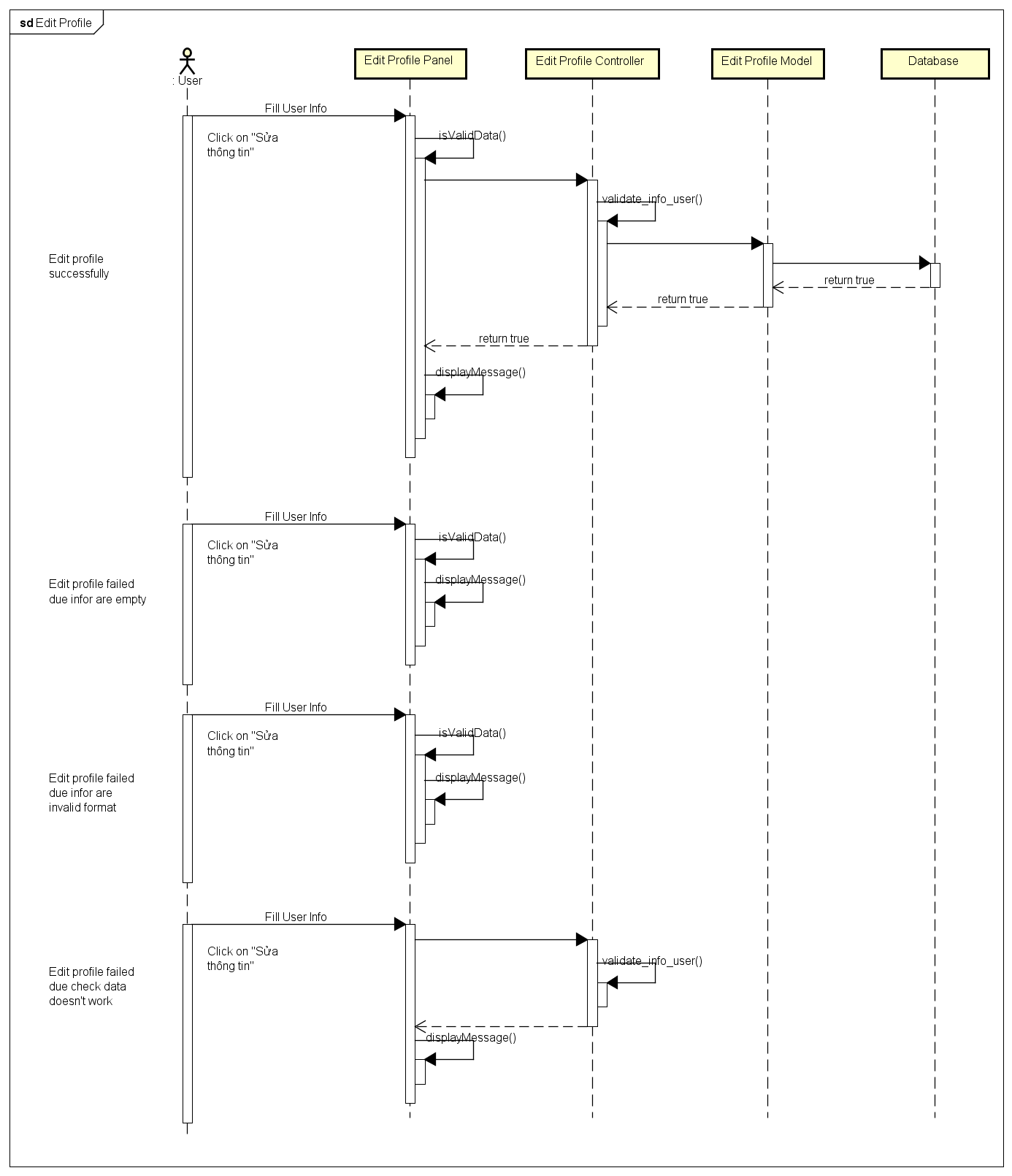
4.4.A.5. Logout

4.4.A.6. Change Password

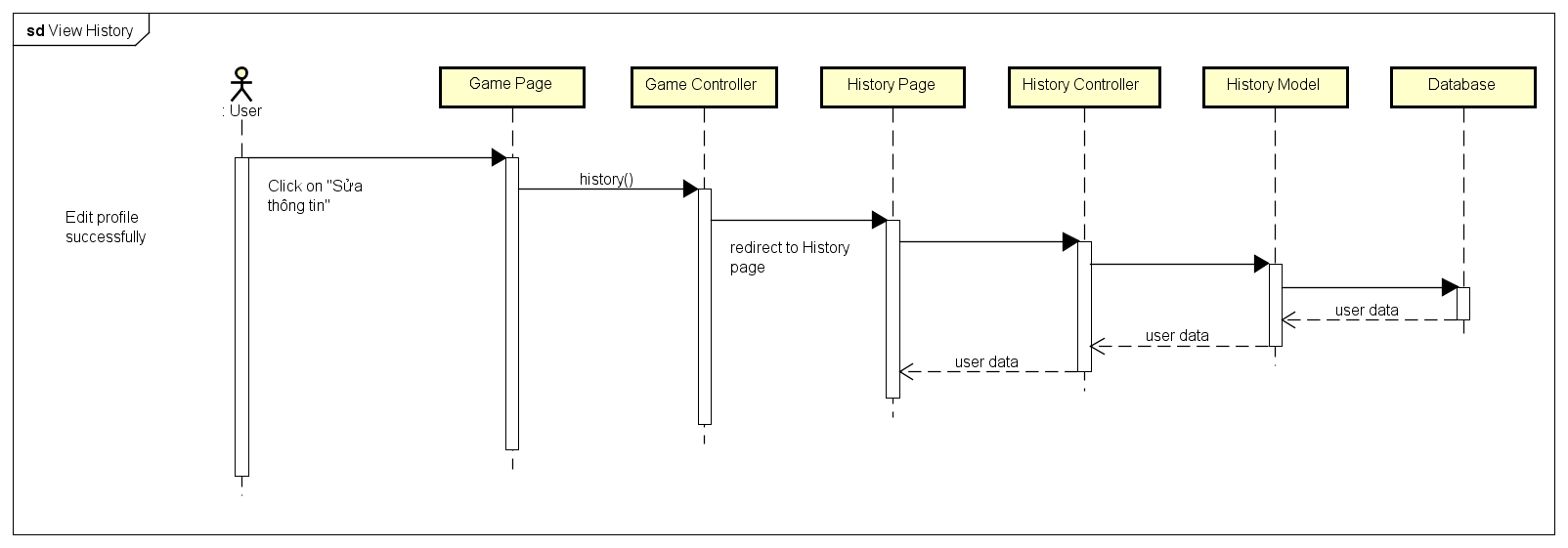
4.4.A.7. Forgot Password



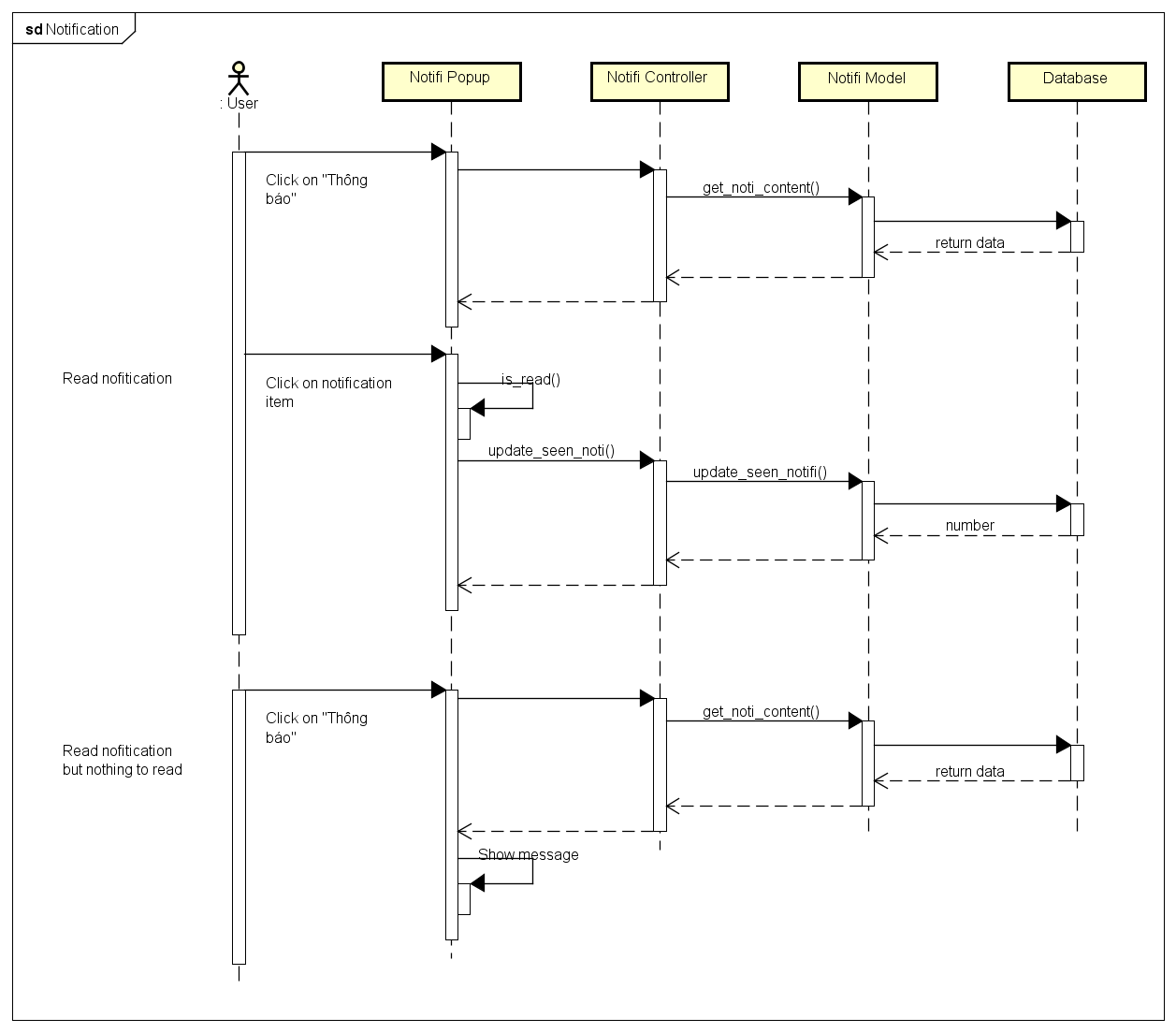
4.4.A.8. Edit Profile



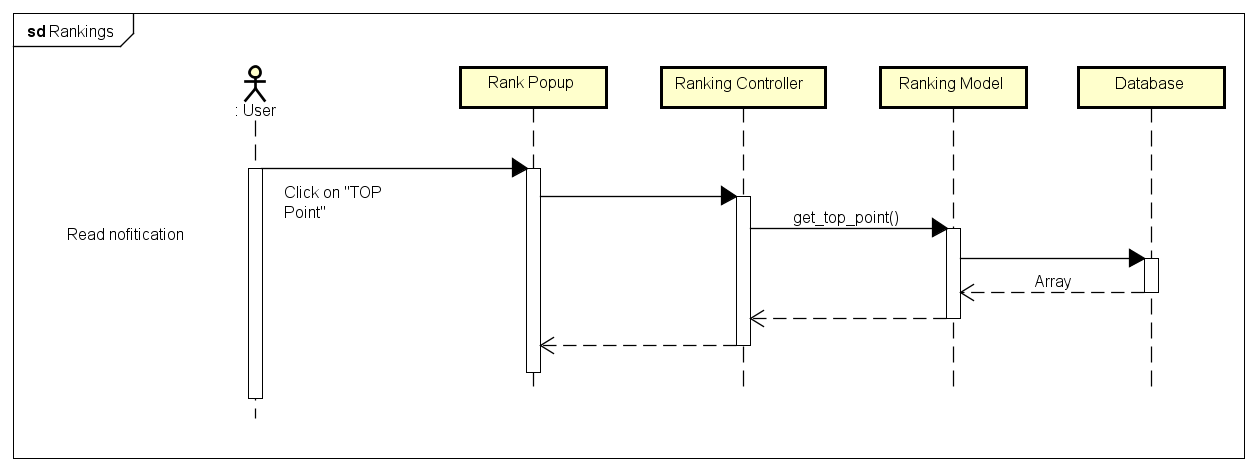
4.4.A.9. History



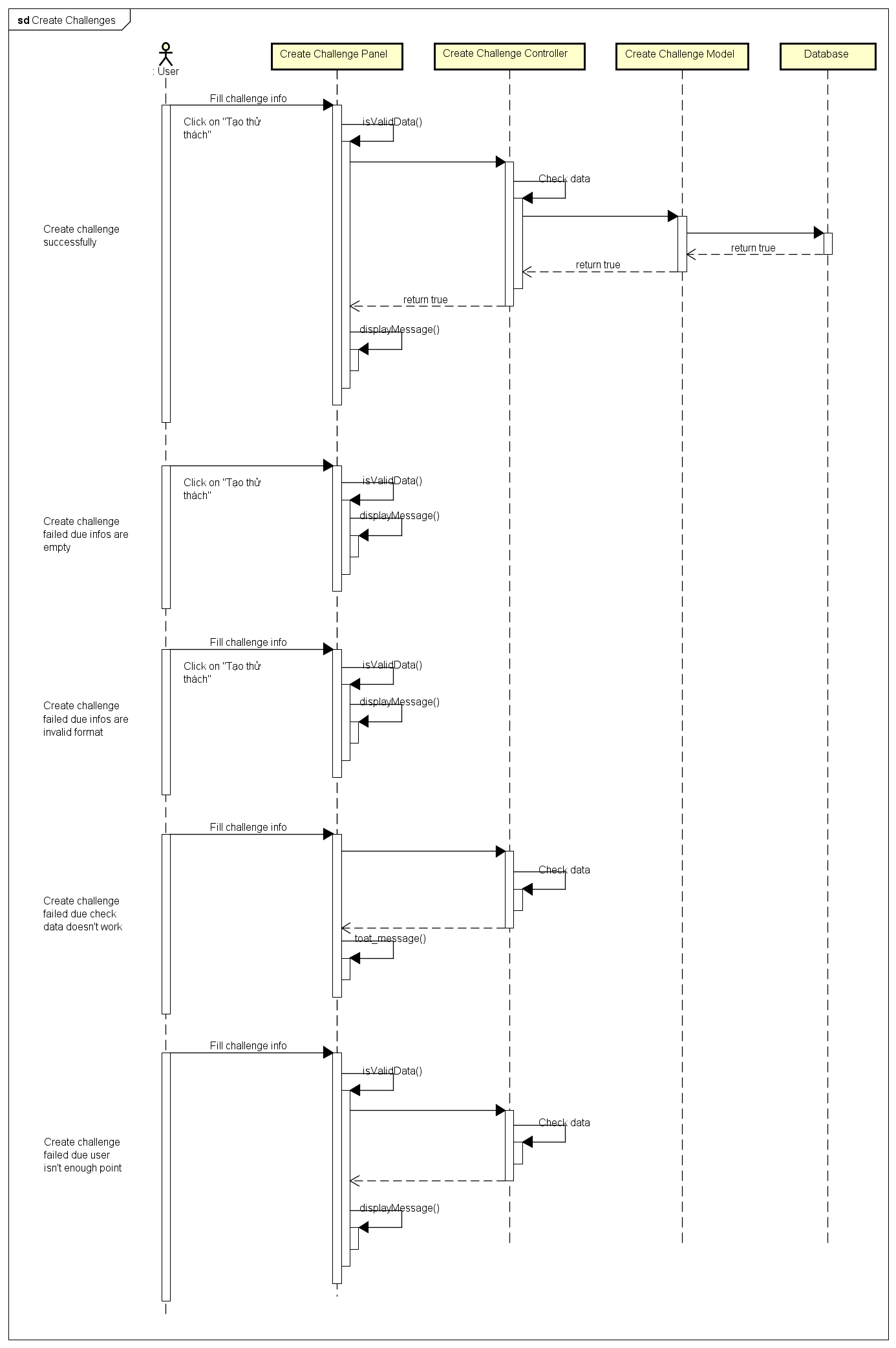
4.4.A.10. Notification



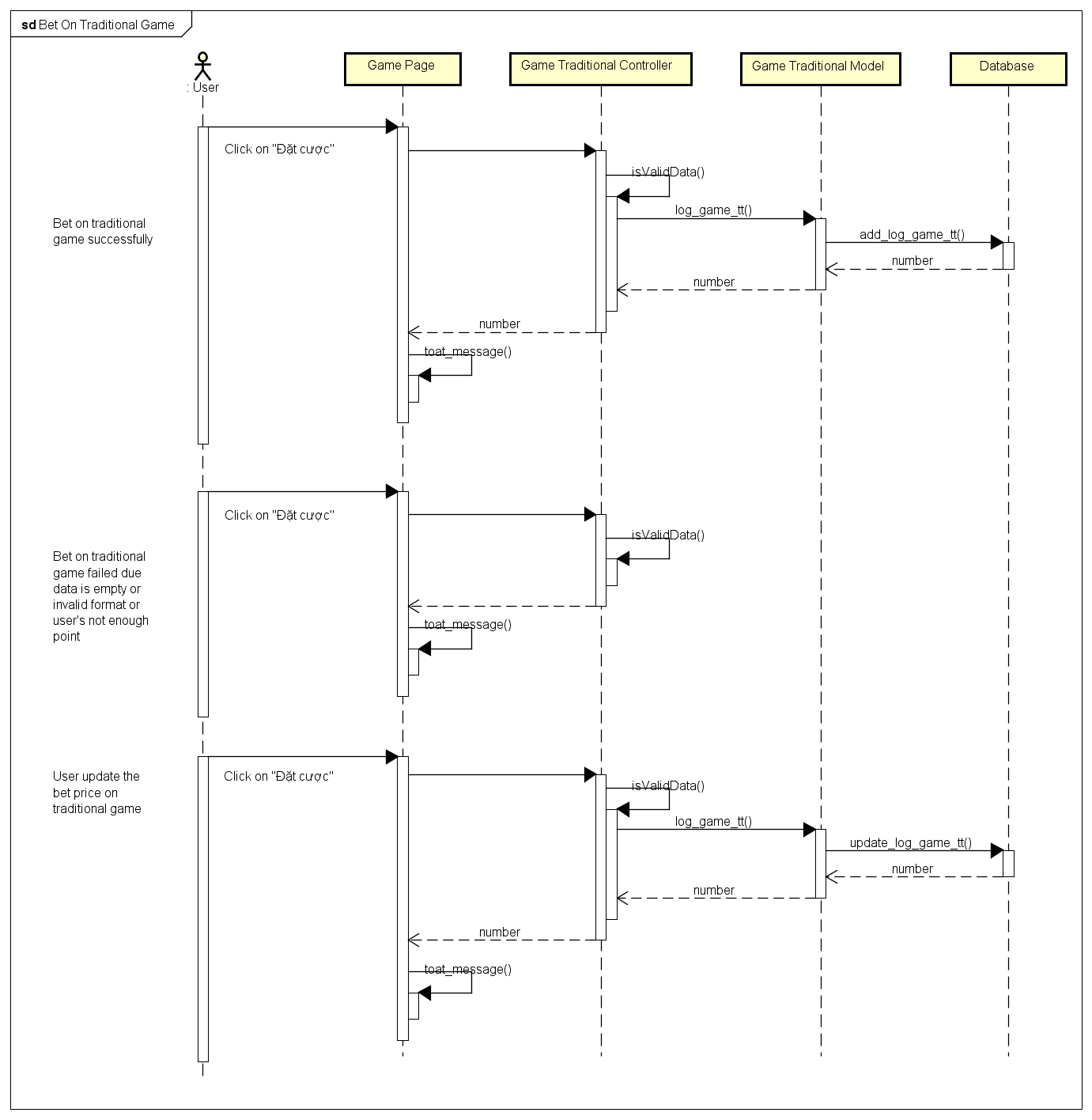
4.4.A.11. User Rankings



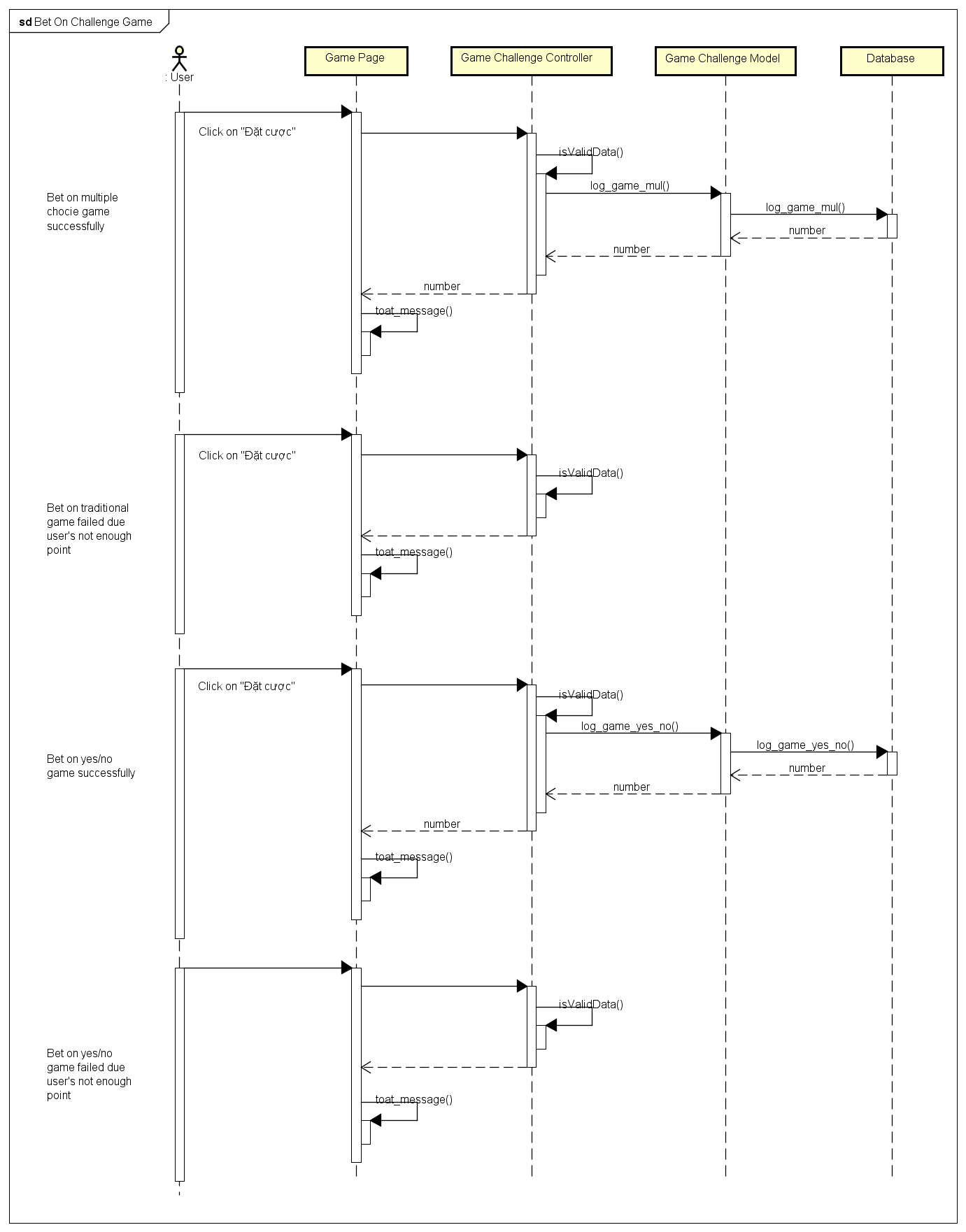
4.4.A.12. Create Challenges



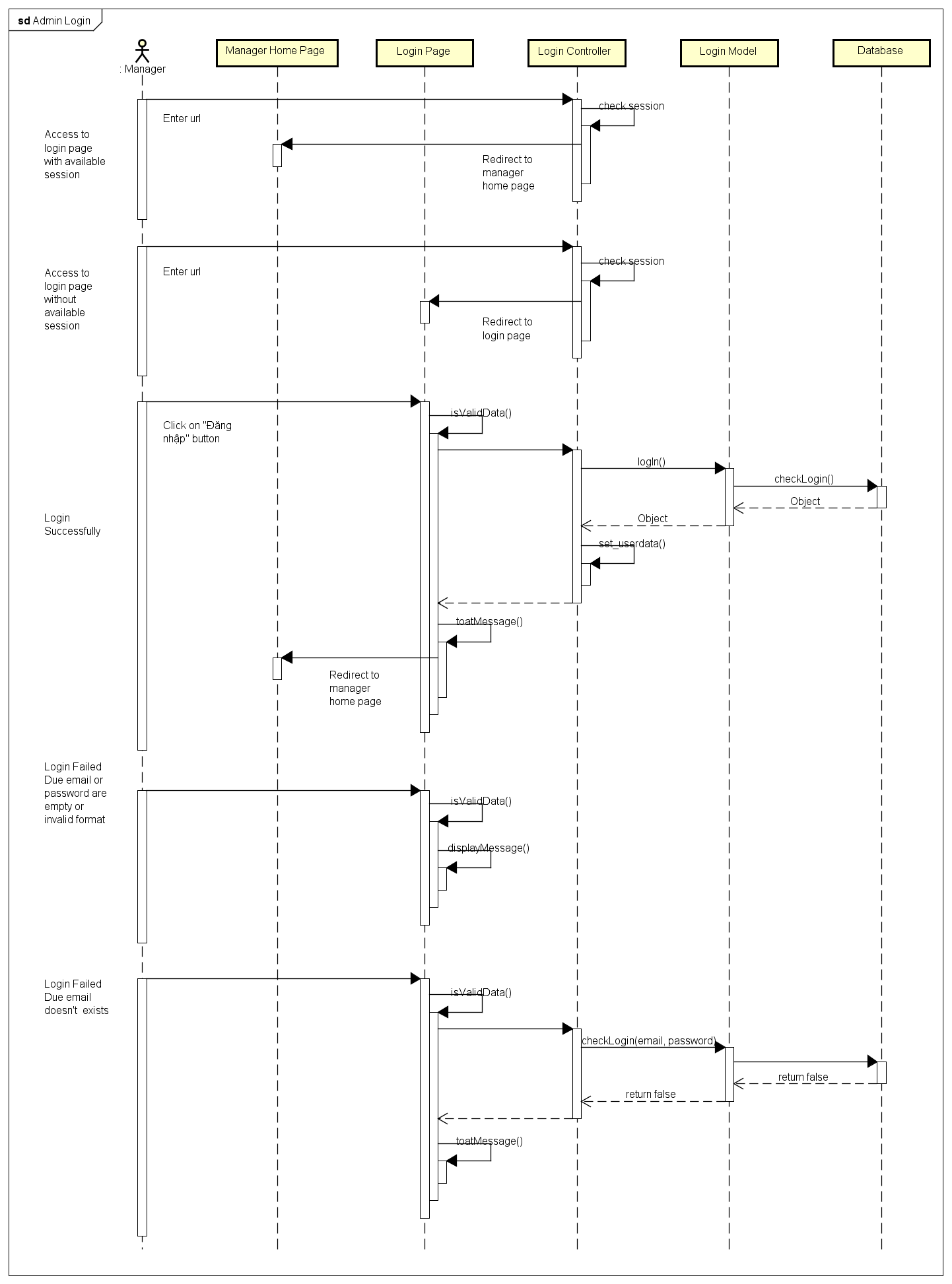
4.4.A.13. Bet On Traditional Game

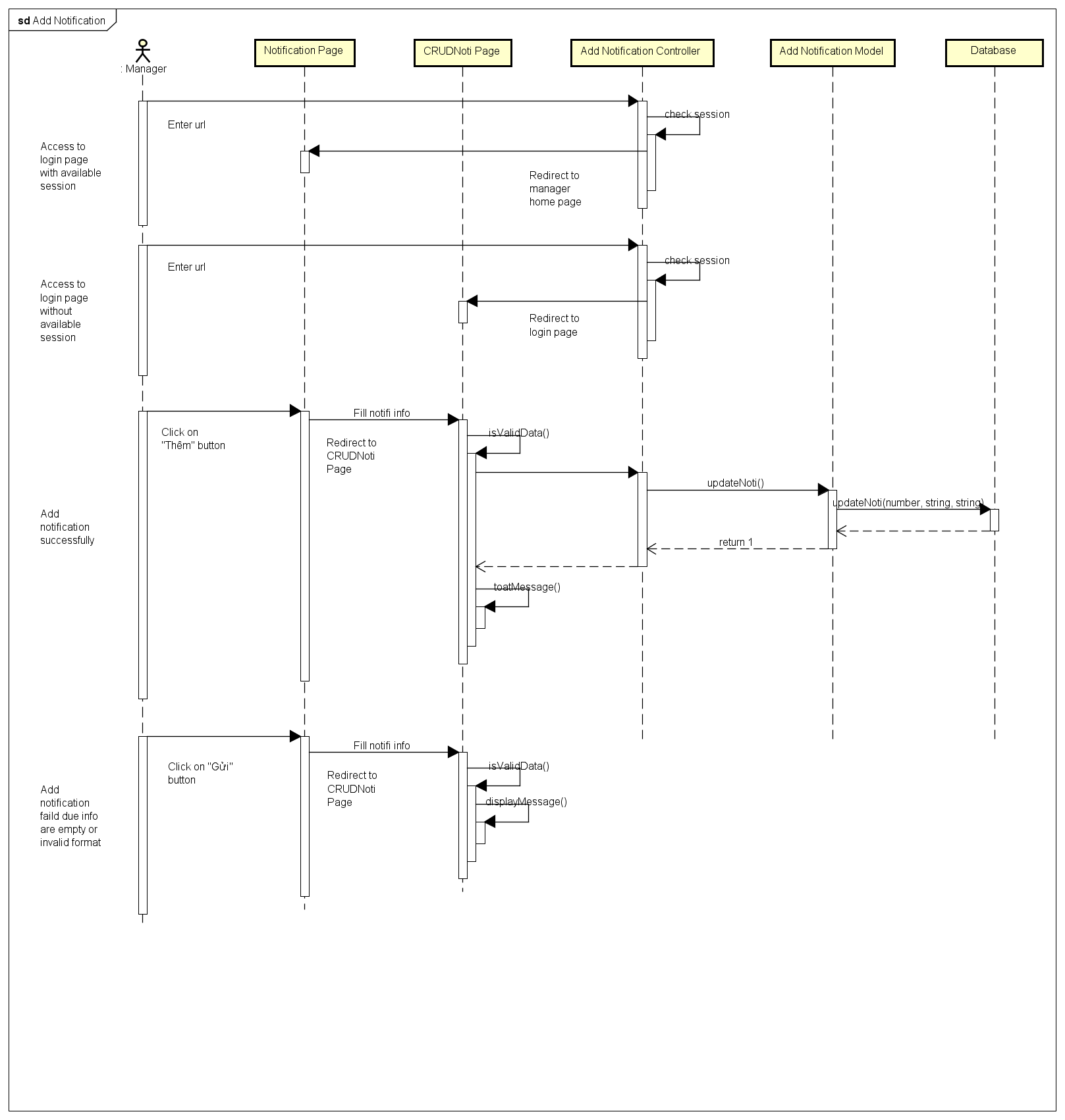


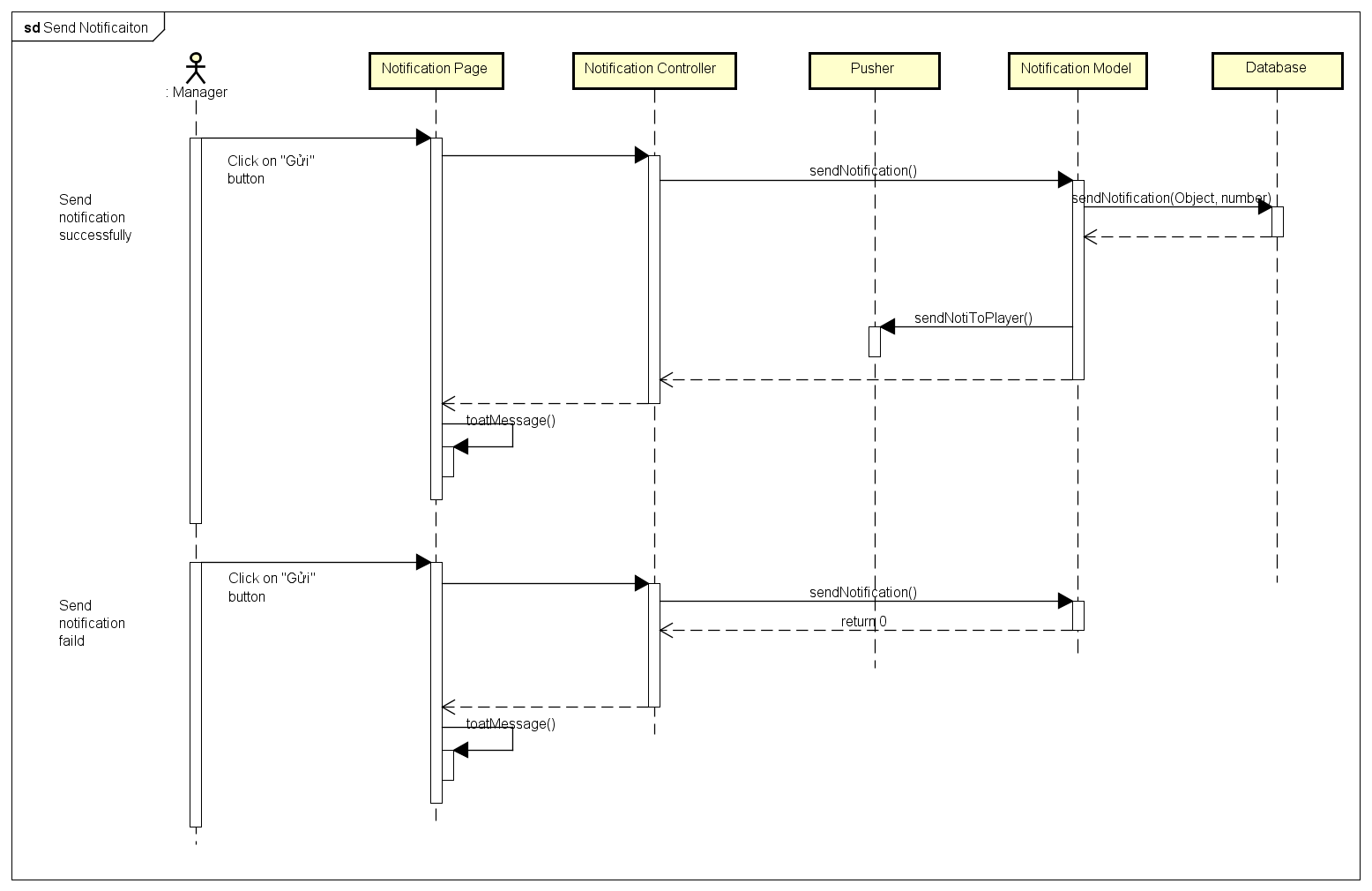
4.4.A.14. Bet On Challenge Game

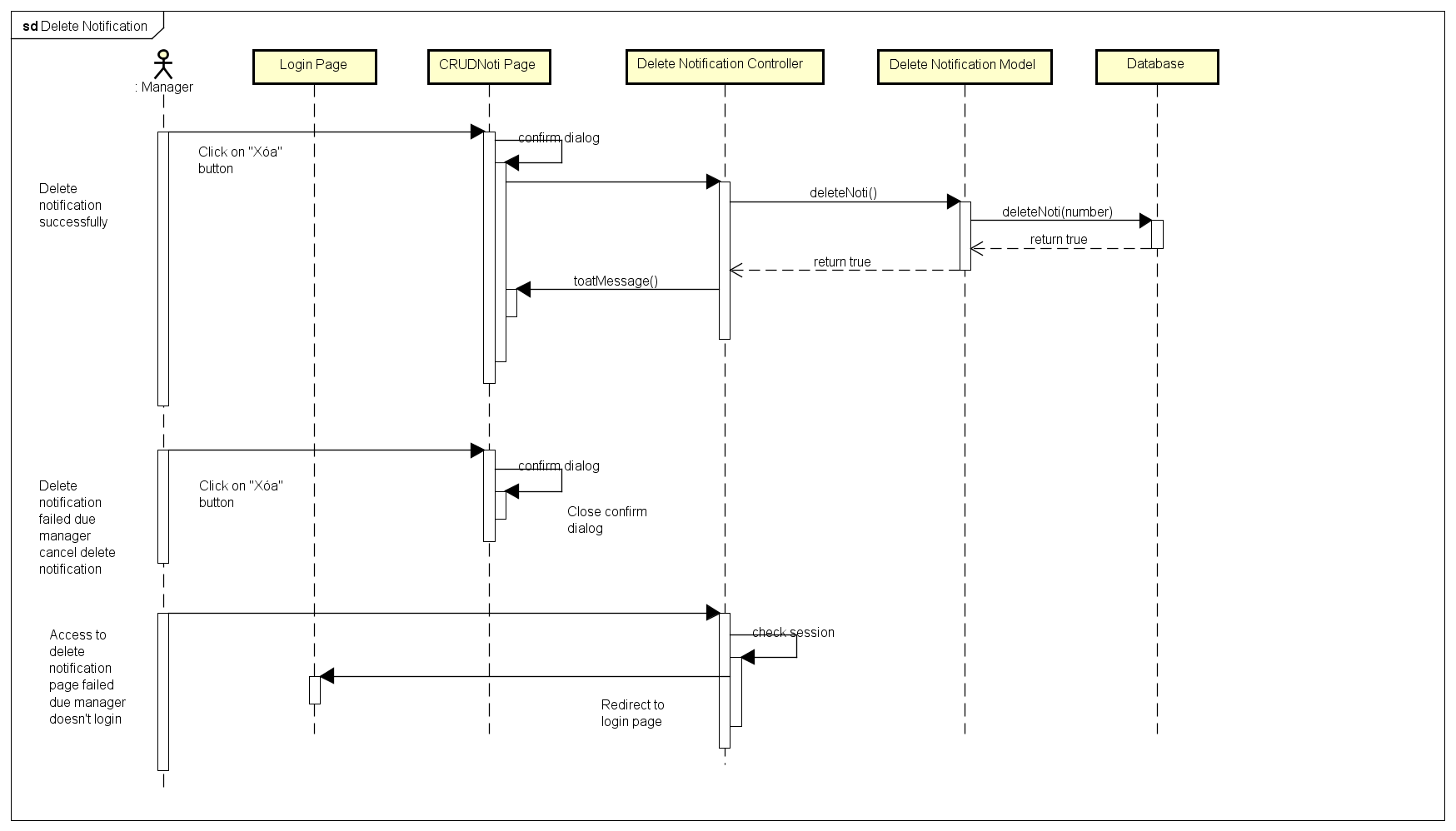


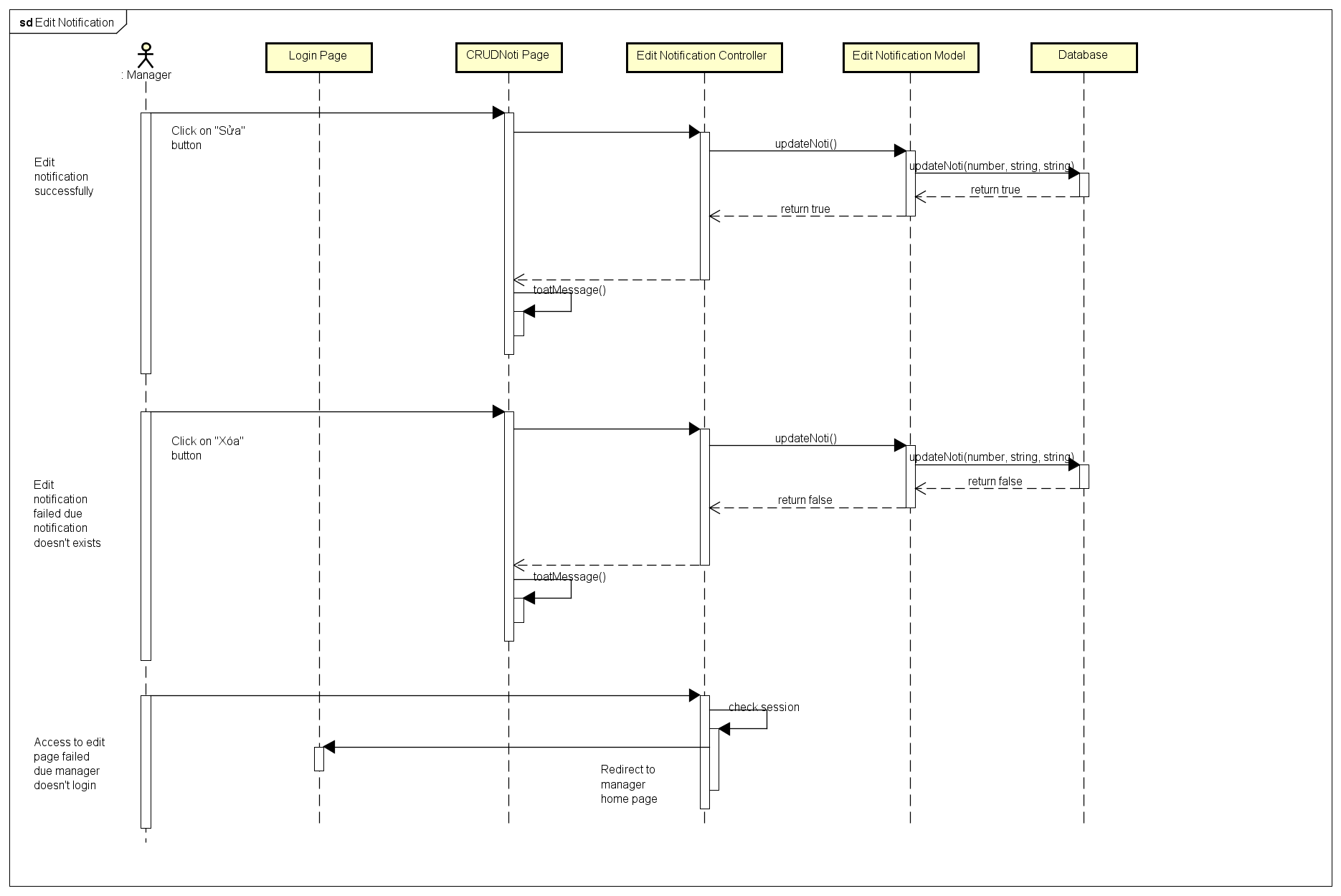
4.4.B. Admin Sequence Diagram

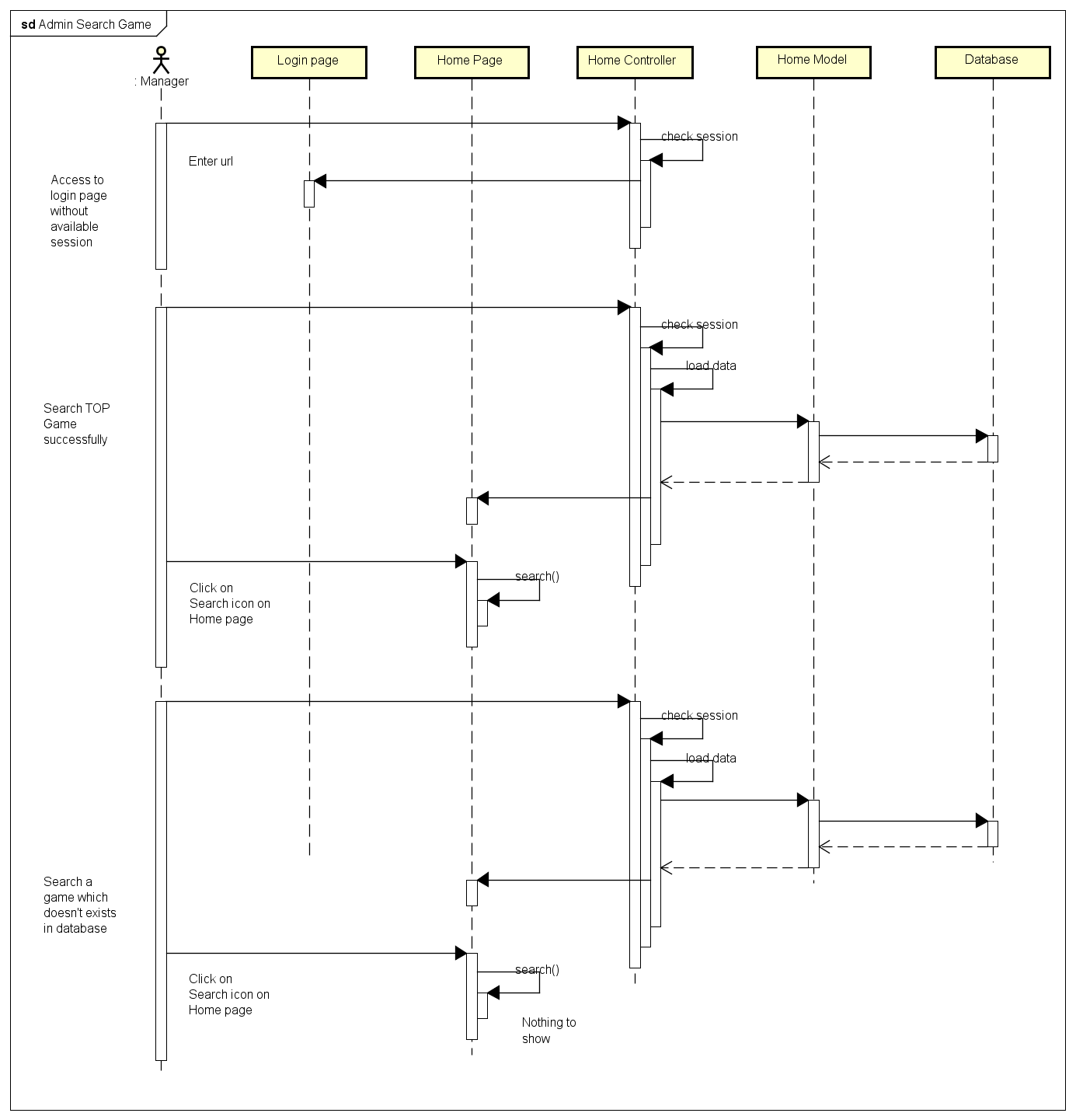
4.4.B.1. Login

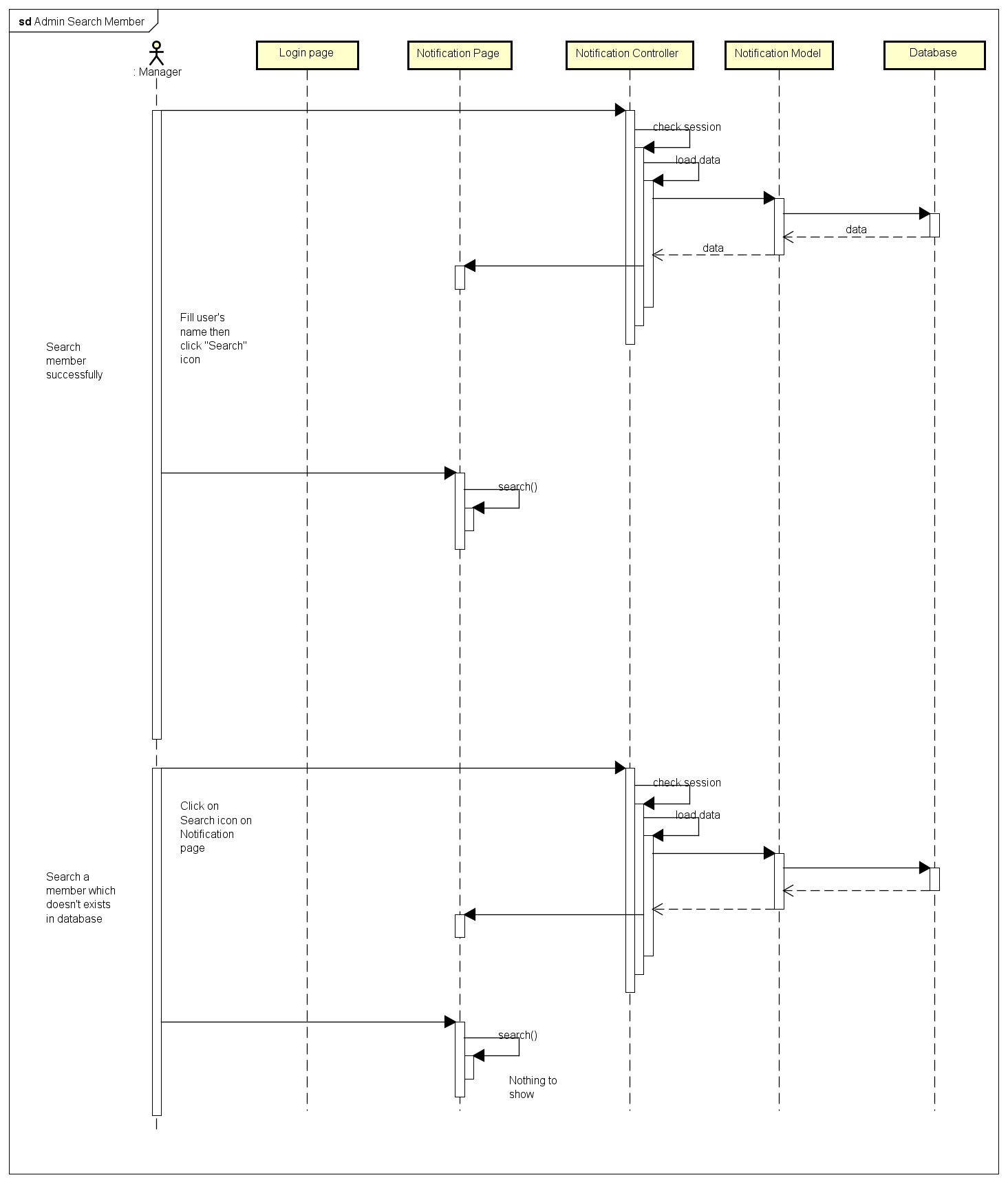
4.4.B.2. Add Notification

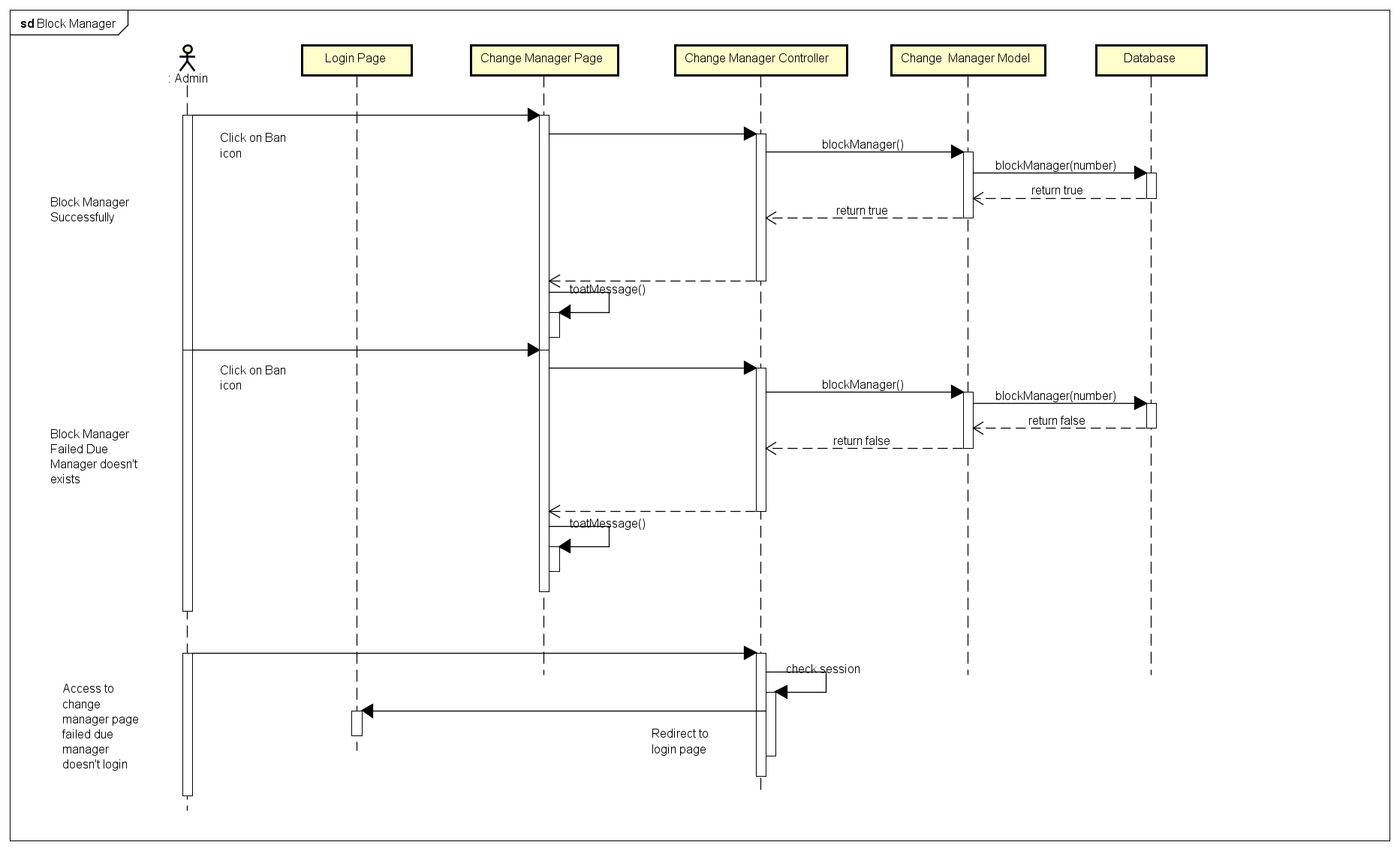
4.4.B.3. Send Notification

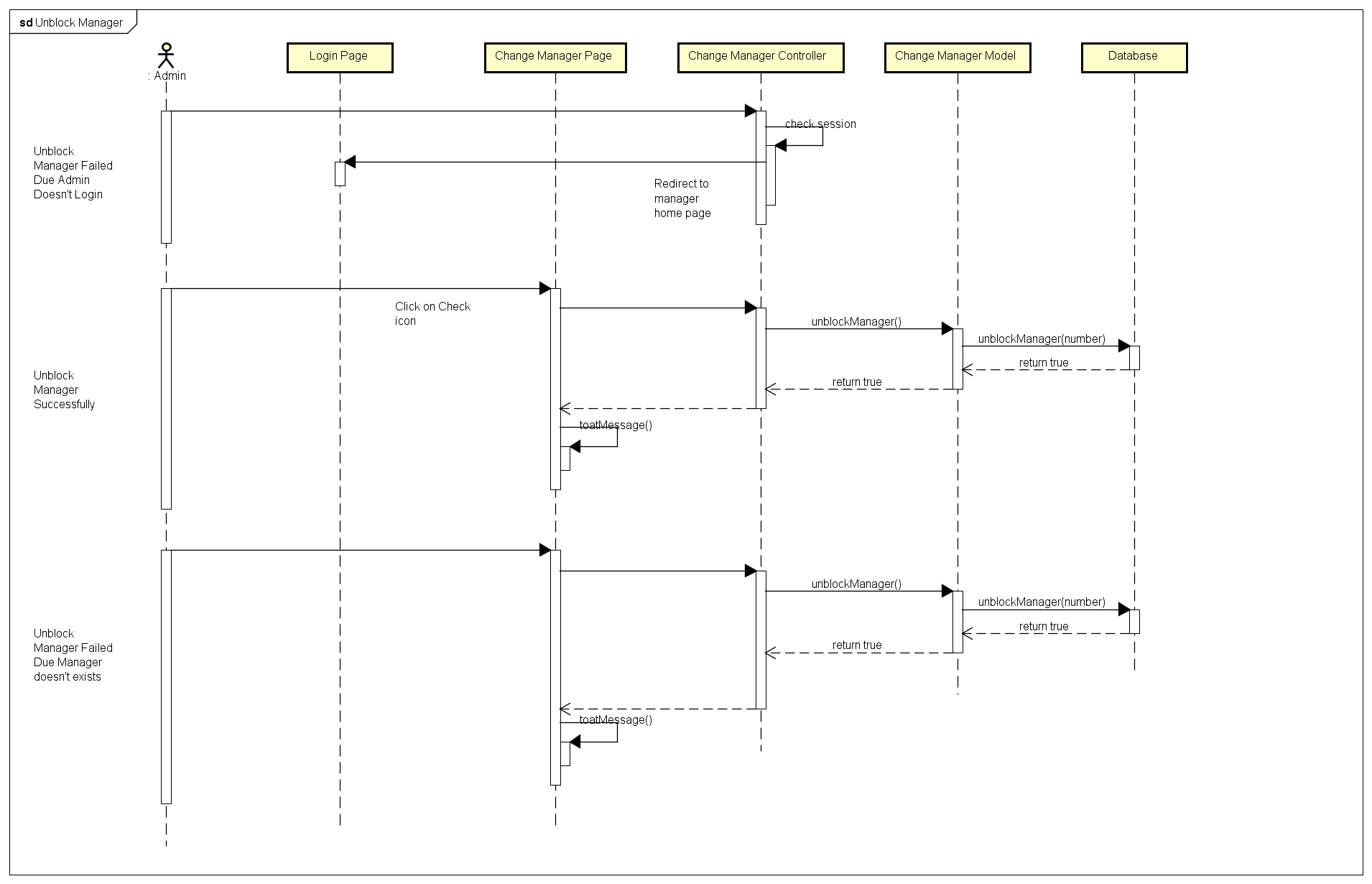
4.4.B.4. Delete Notification

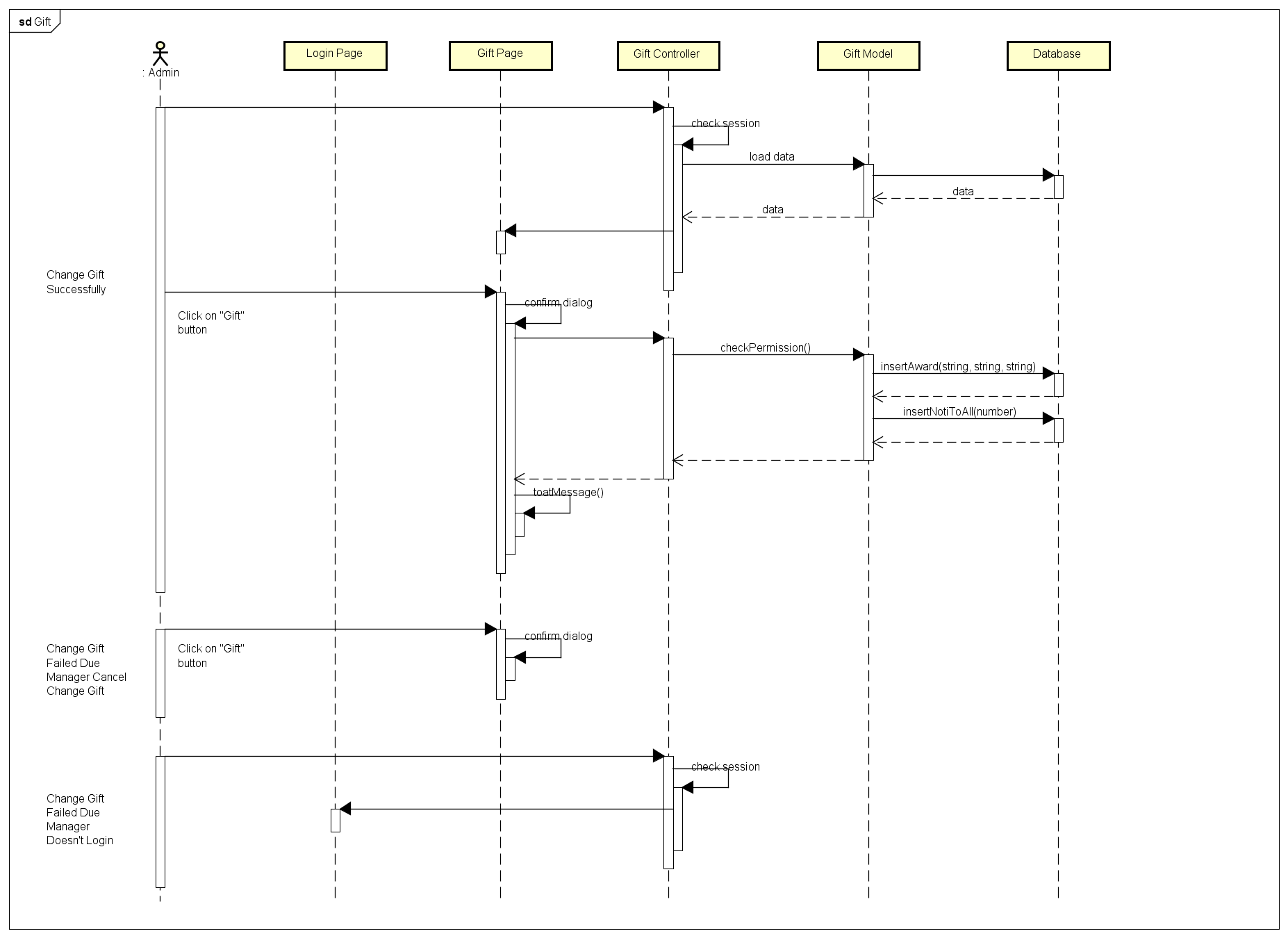
4.4.B.5. Edit Notification

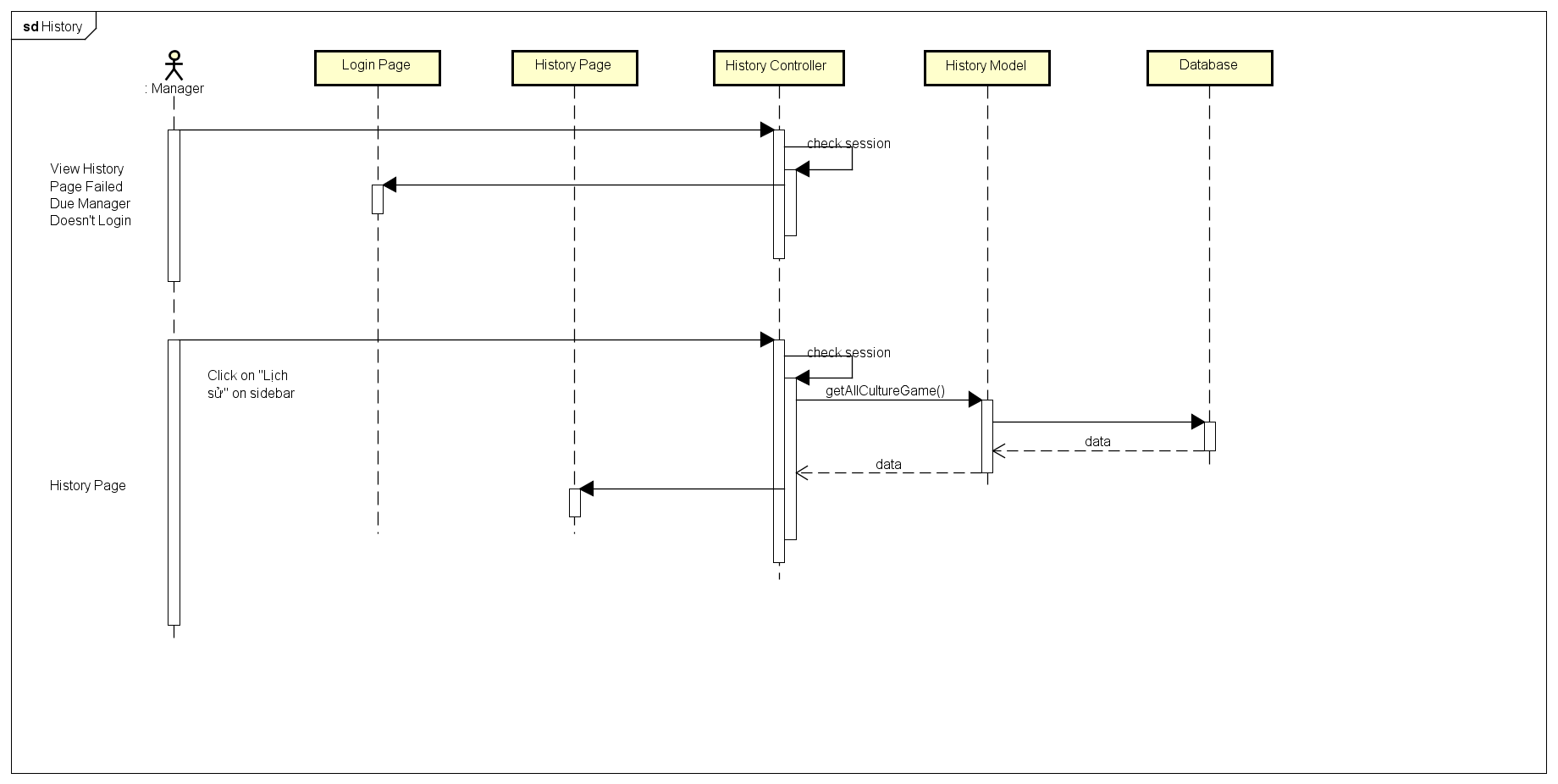
4.4.B.6. Search Game

4.4.B.7. Search Member

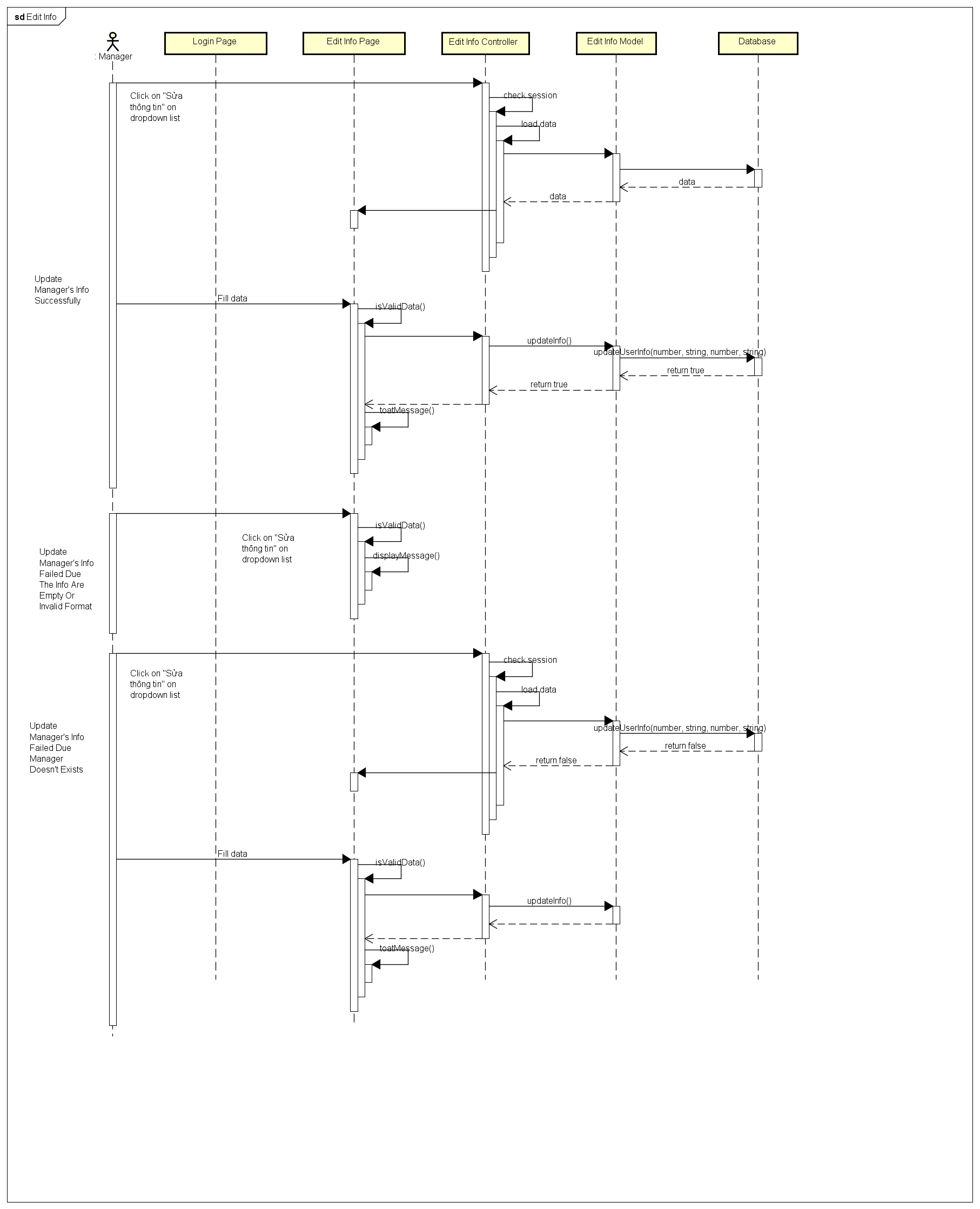
4.4.B.8. Block Manager

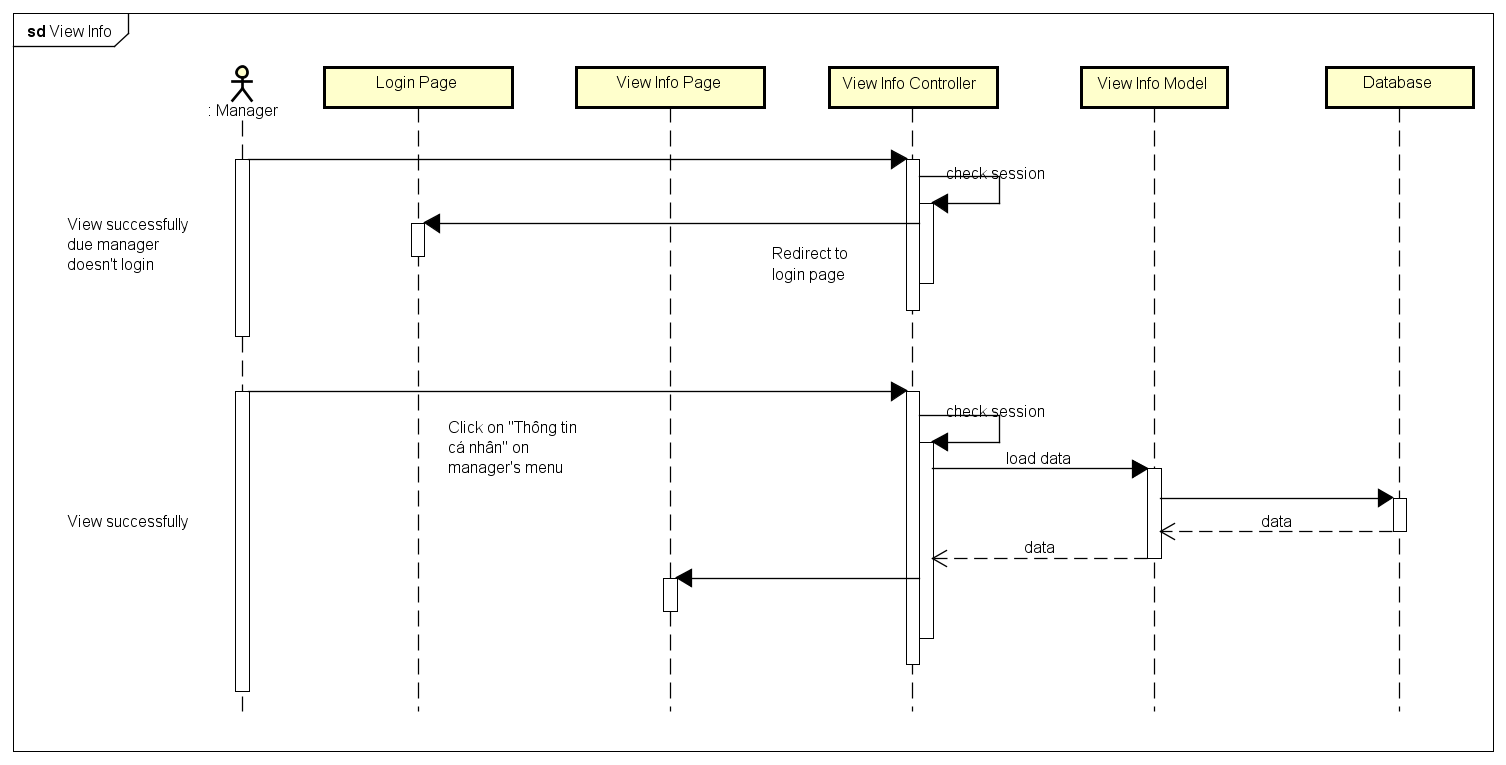
4.4.B.9. Unblock Manager

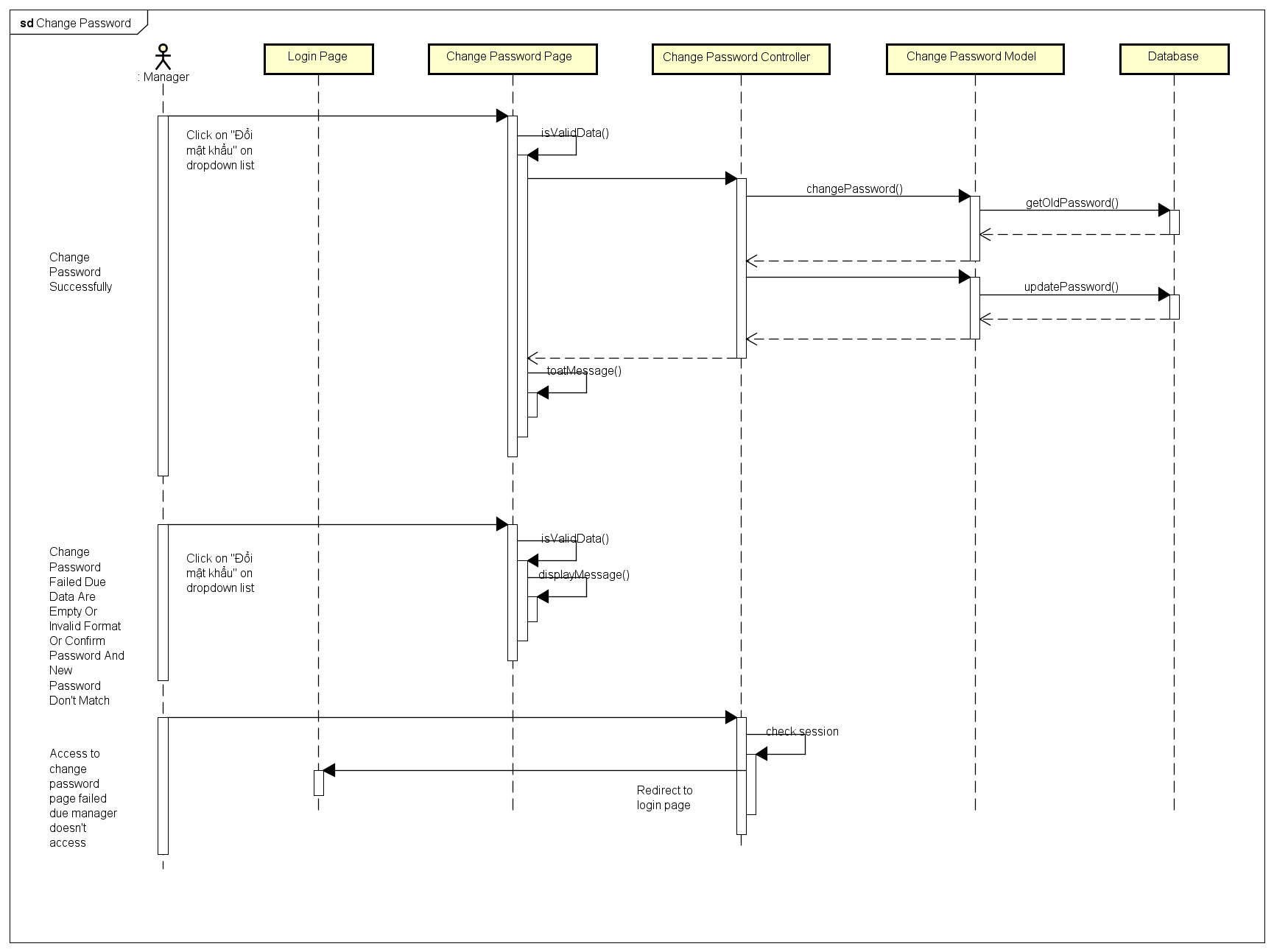
4.4.B.10. Change Gift

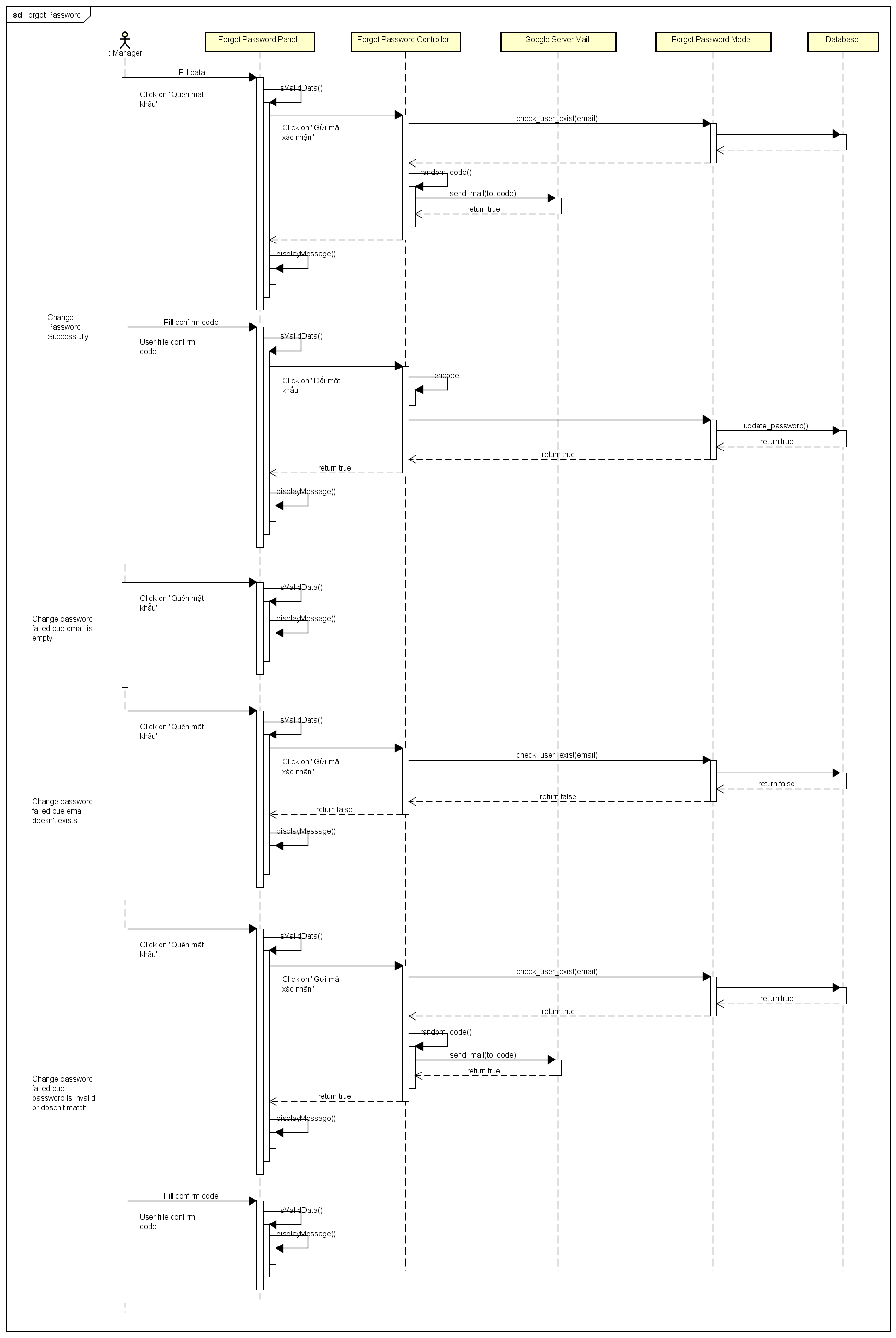
4.4.B.11. History

4.4.B.12. Create Game

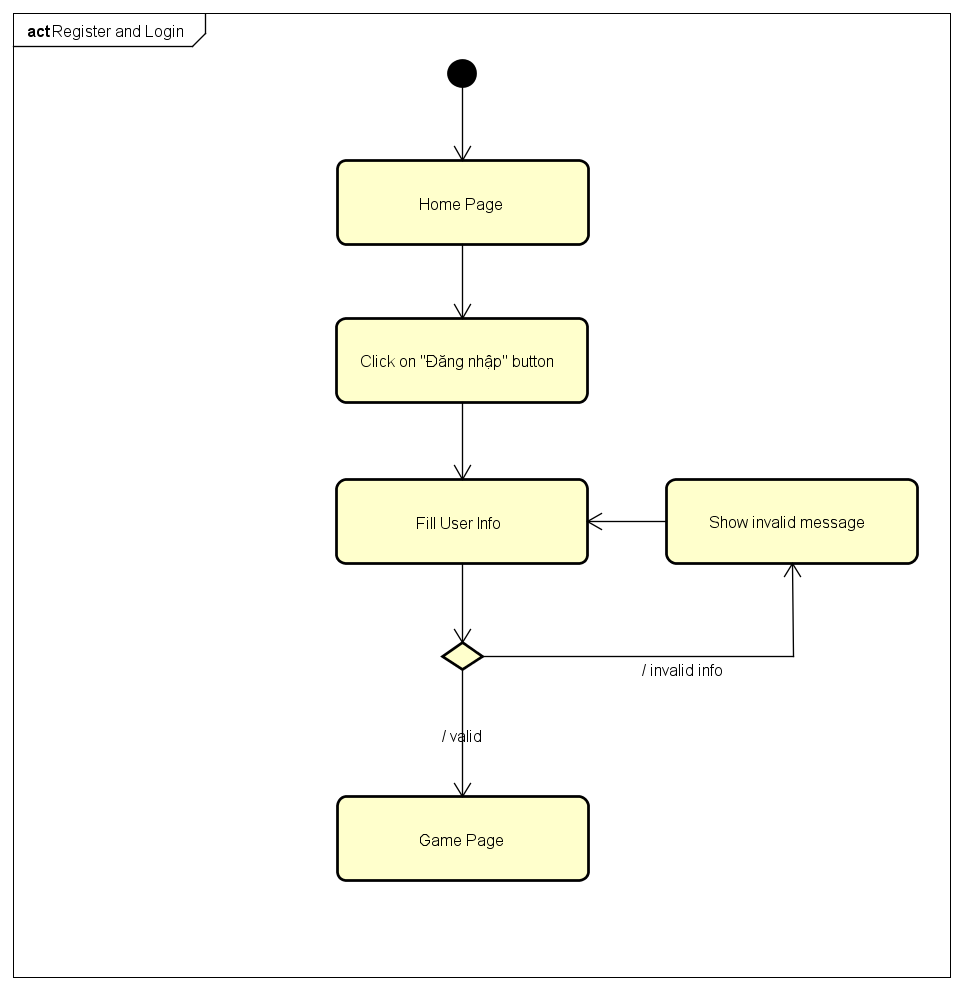
4.4.B.13. Edit Manager Information

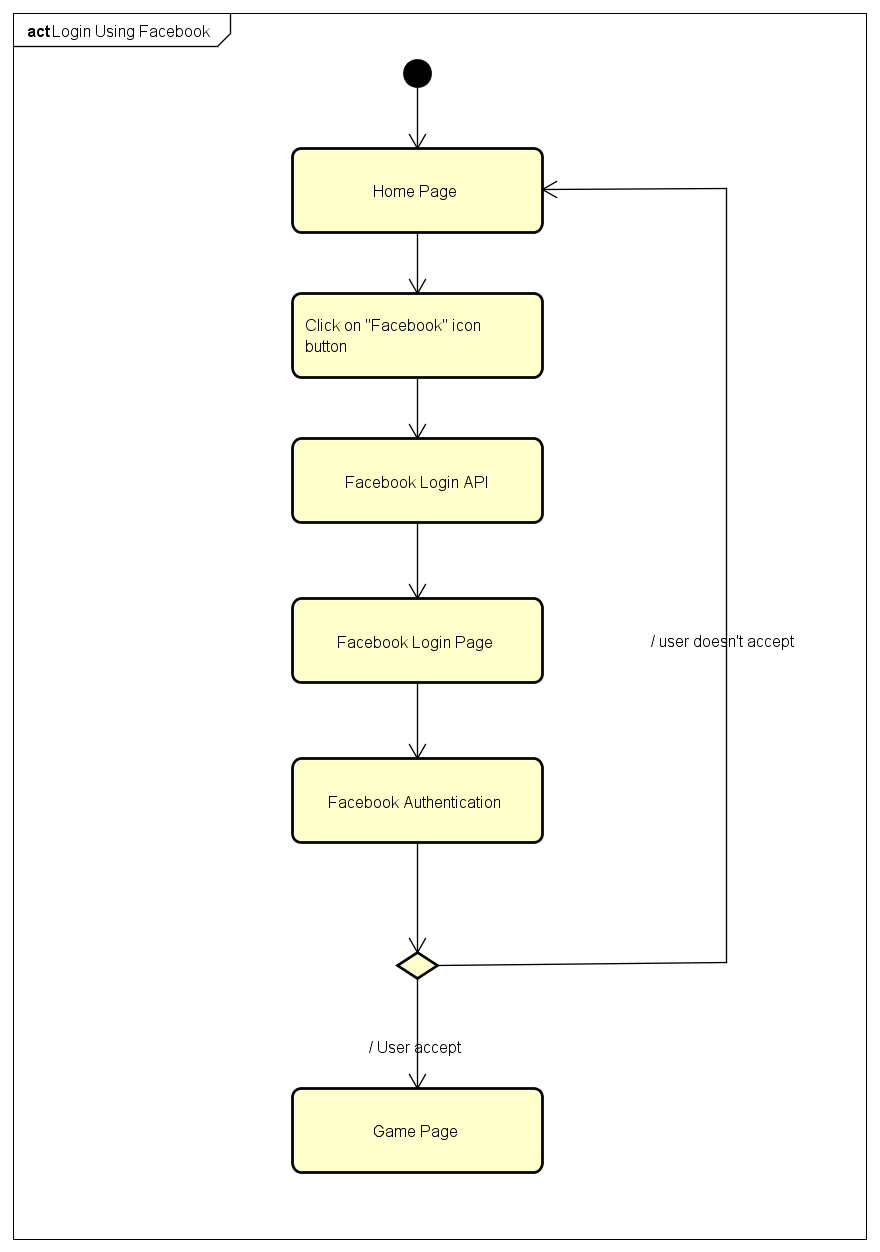
4.4.B.14. View Manager Information

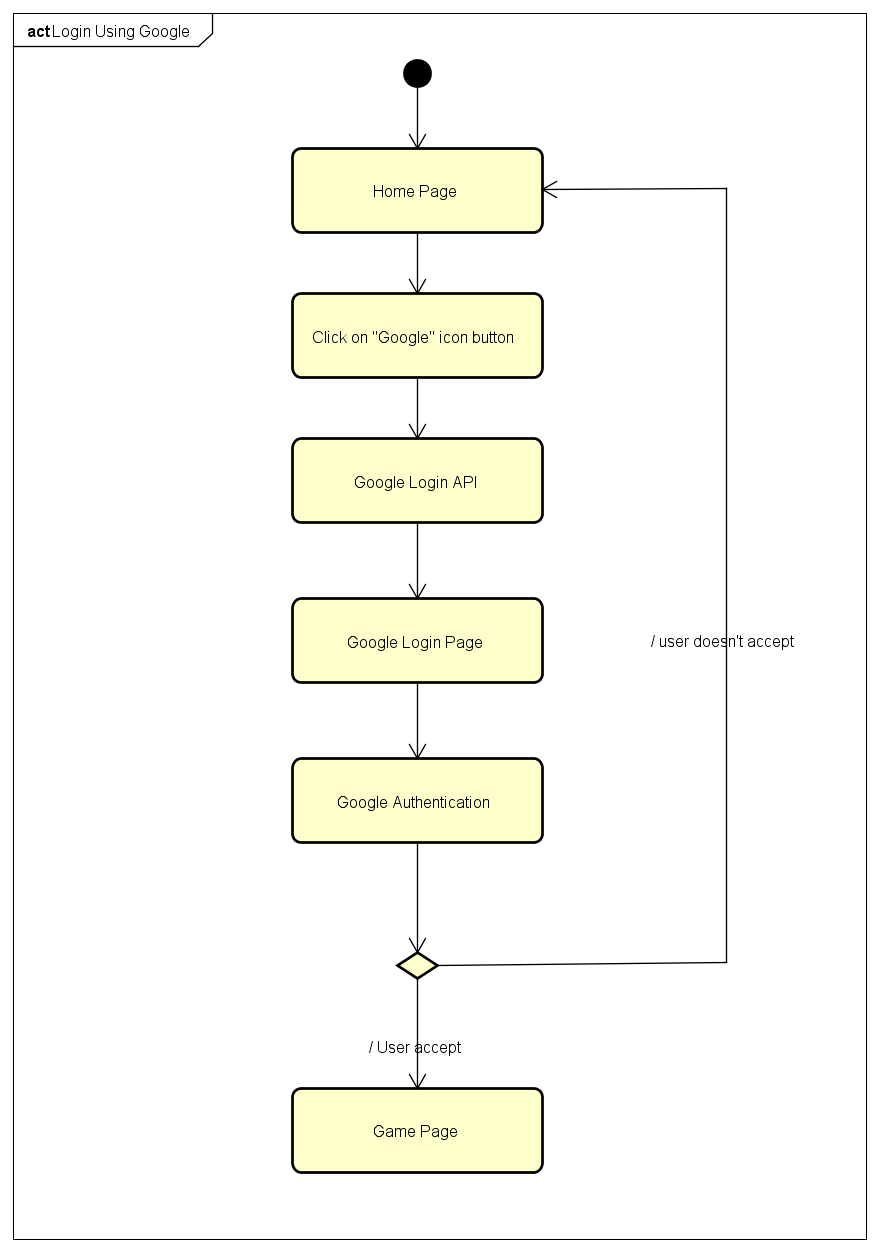
4.4.B.15. Change Password

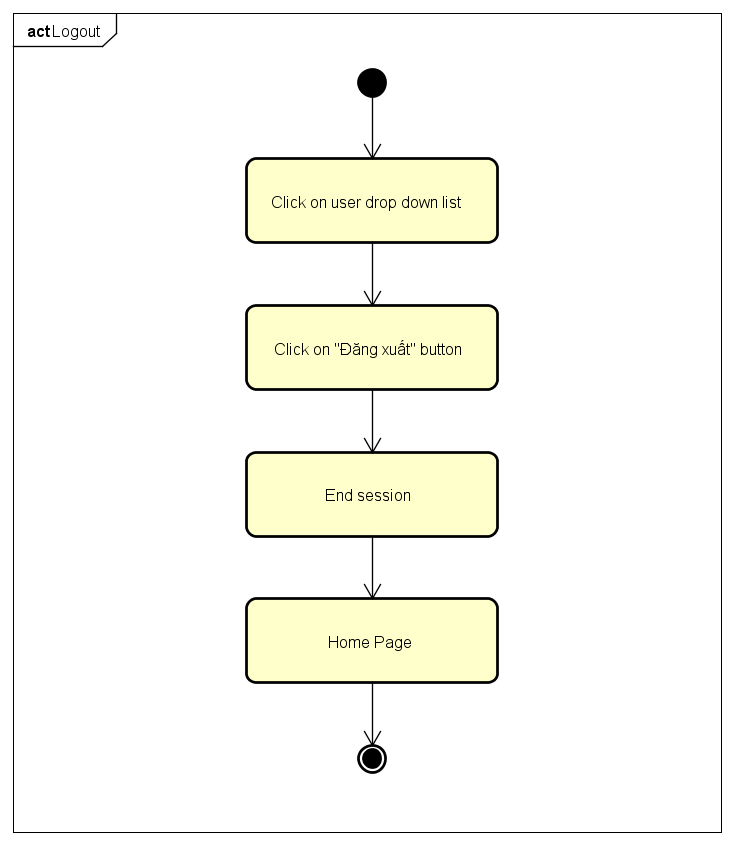
4.4.B.16. Forgot Password

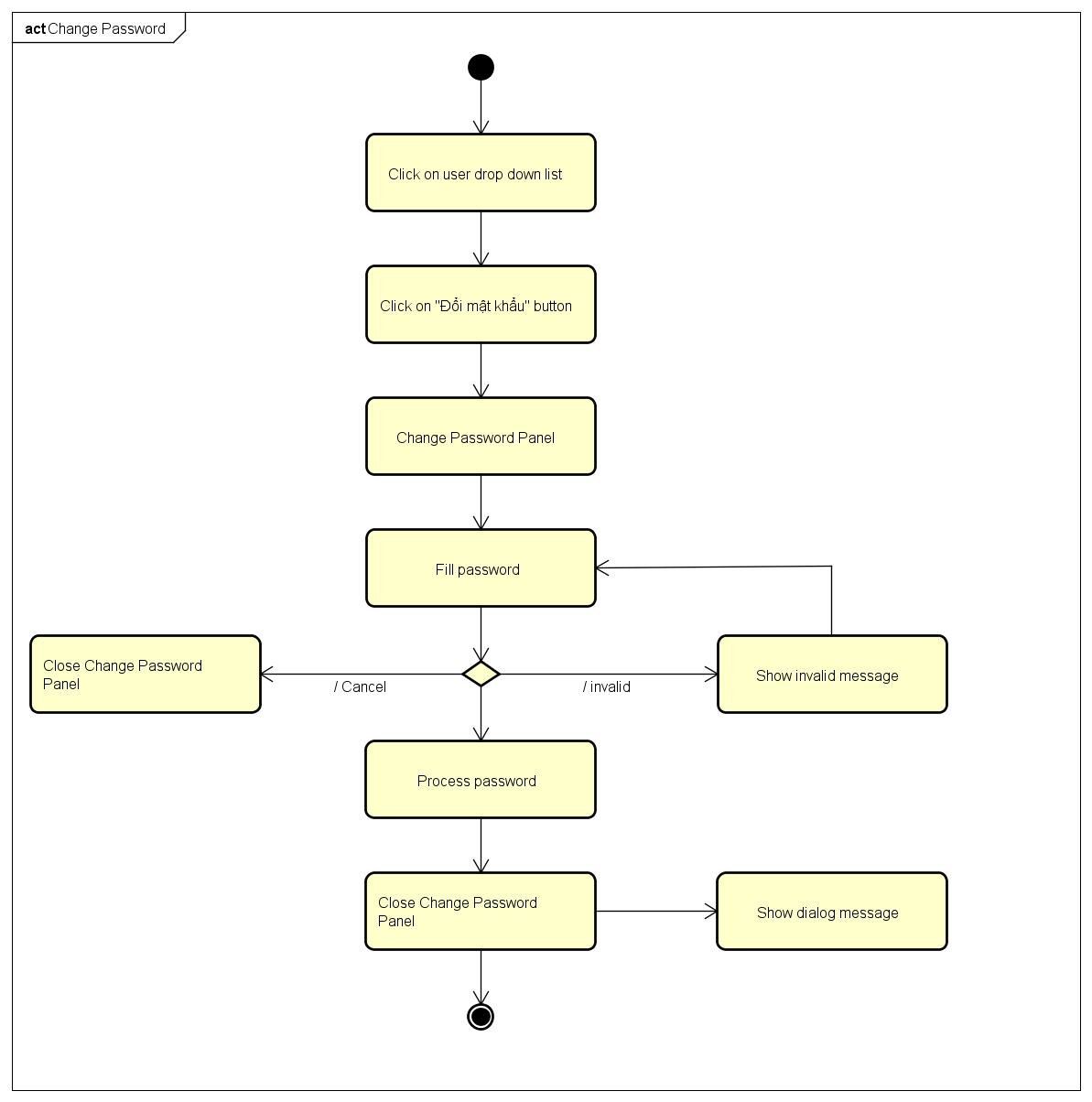
4.5.A. Activities Diagram

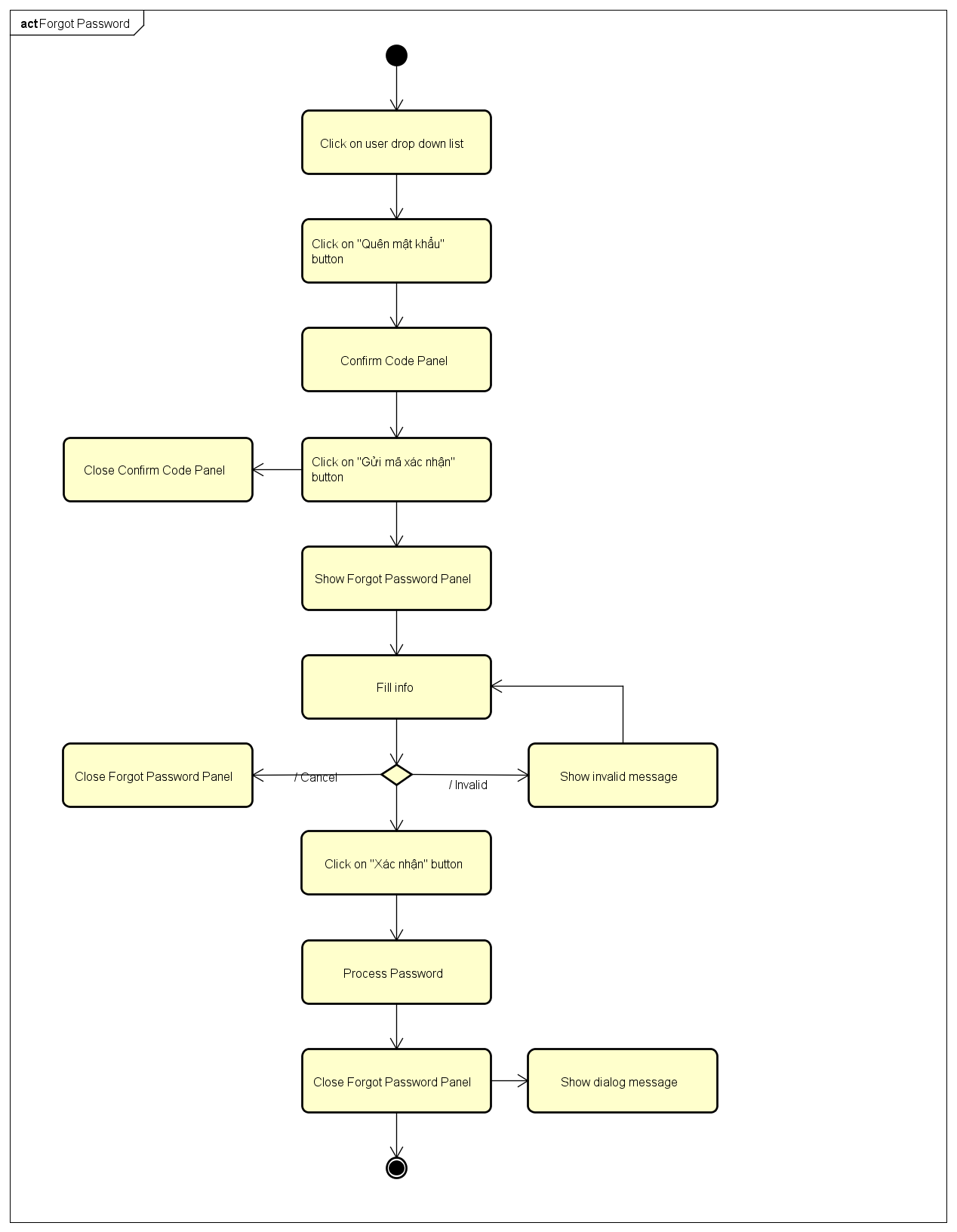
4.5.A.1. Register and Login

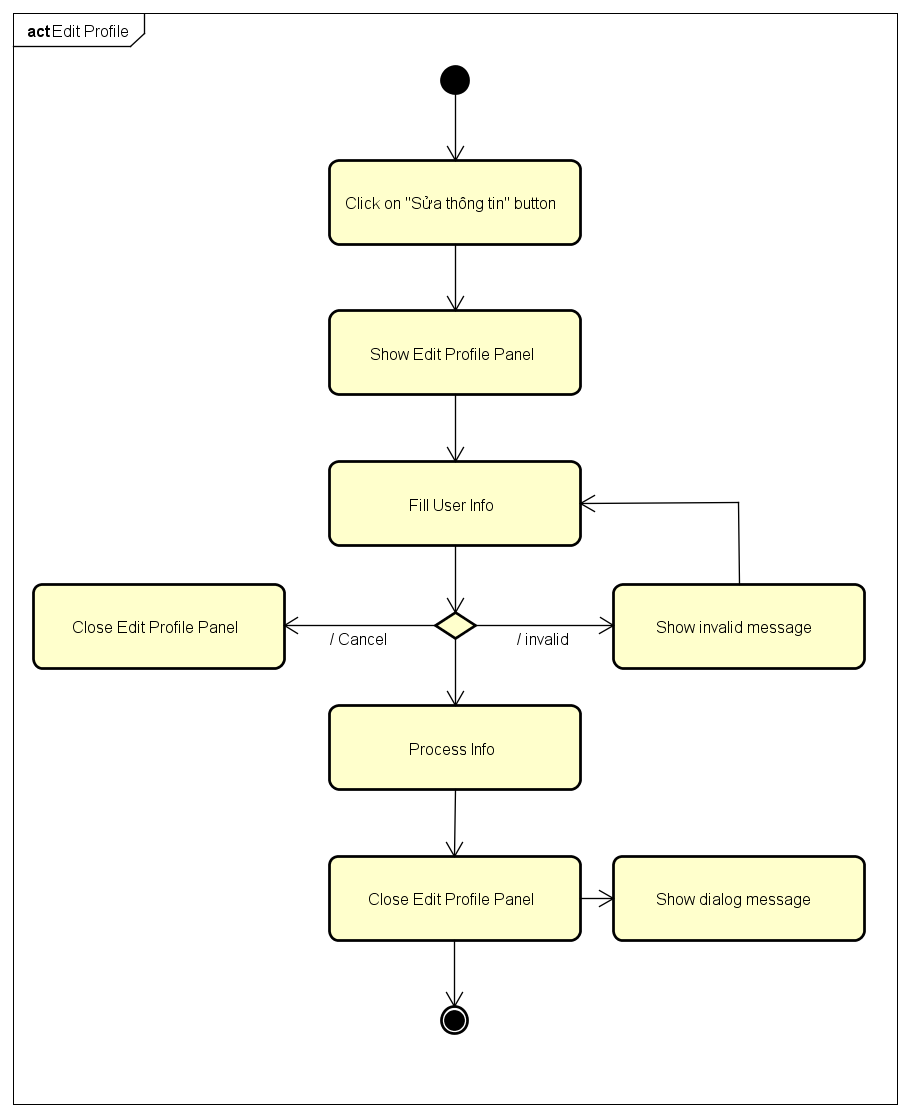
4.5.A.2. Login Using Facebook

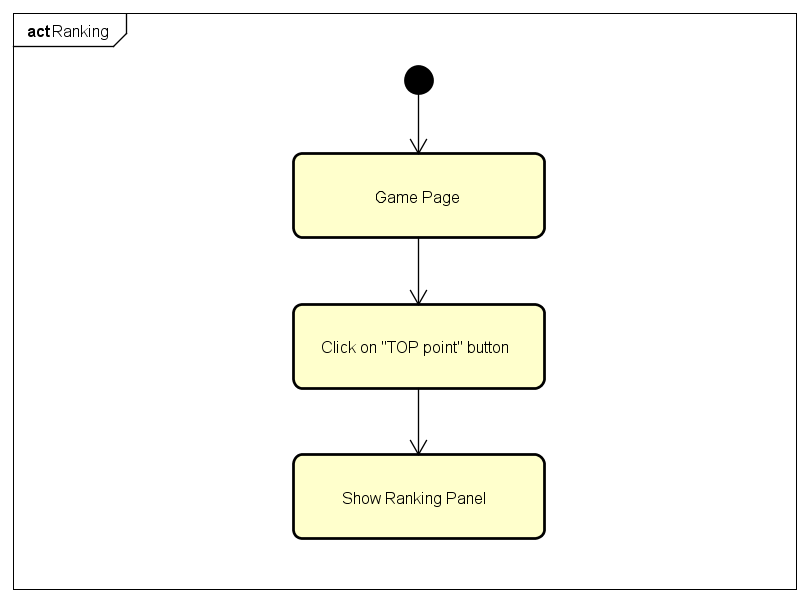
4.5.A.3. Login Using Google Plus

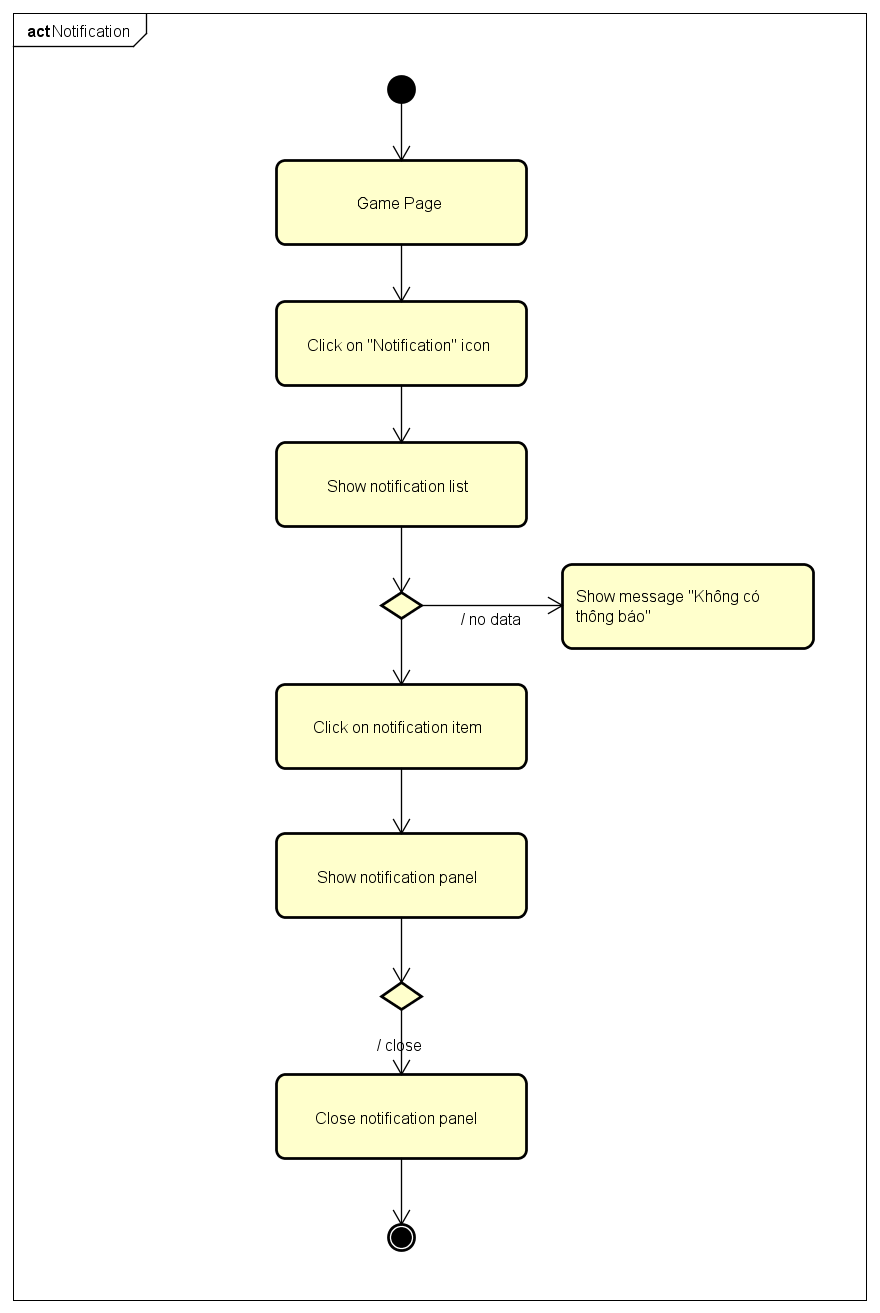
4.5.A.4. Logout

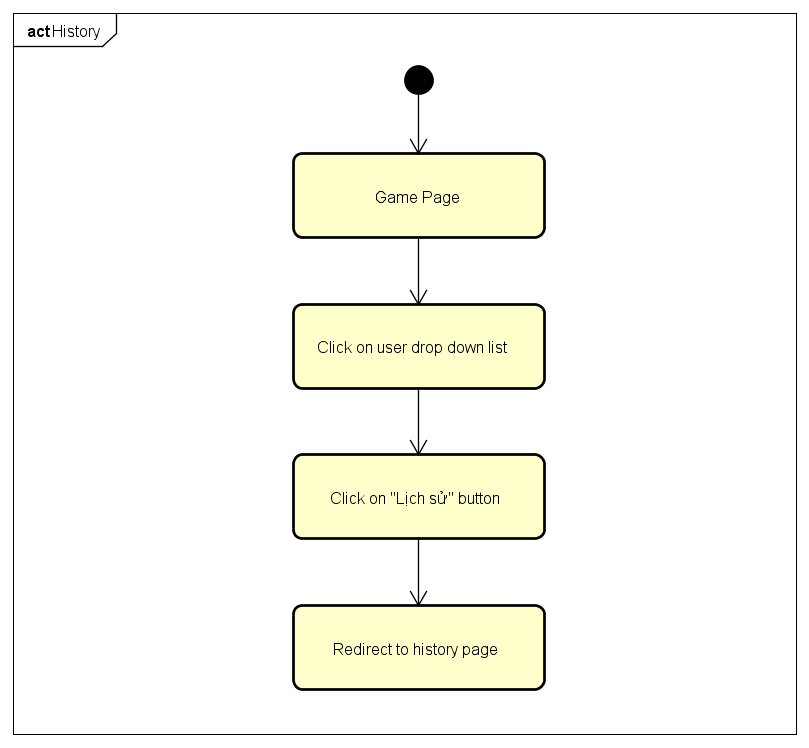
4.5.A.5. Change Password

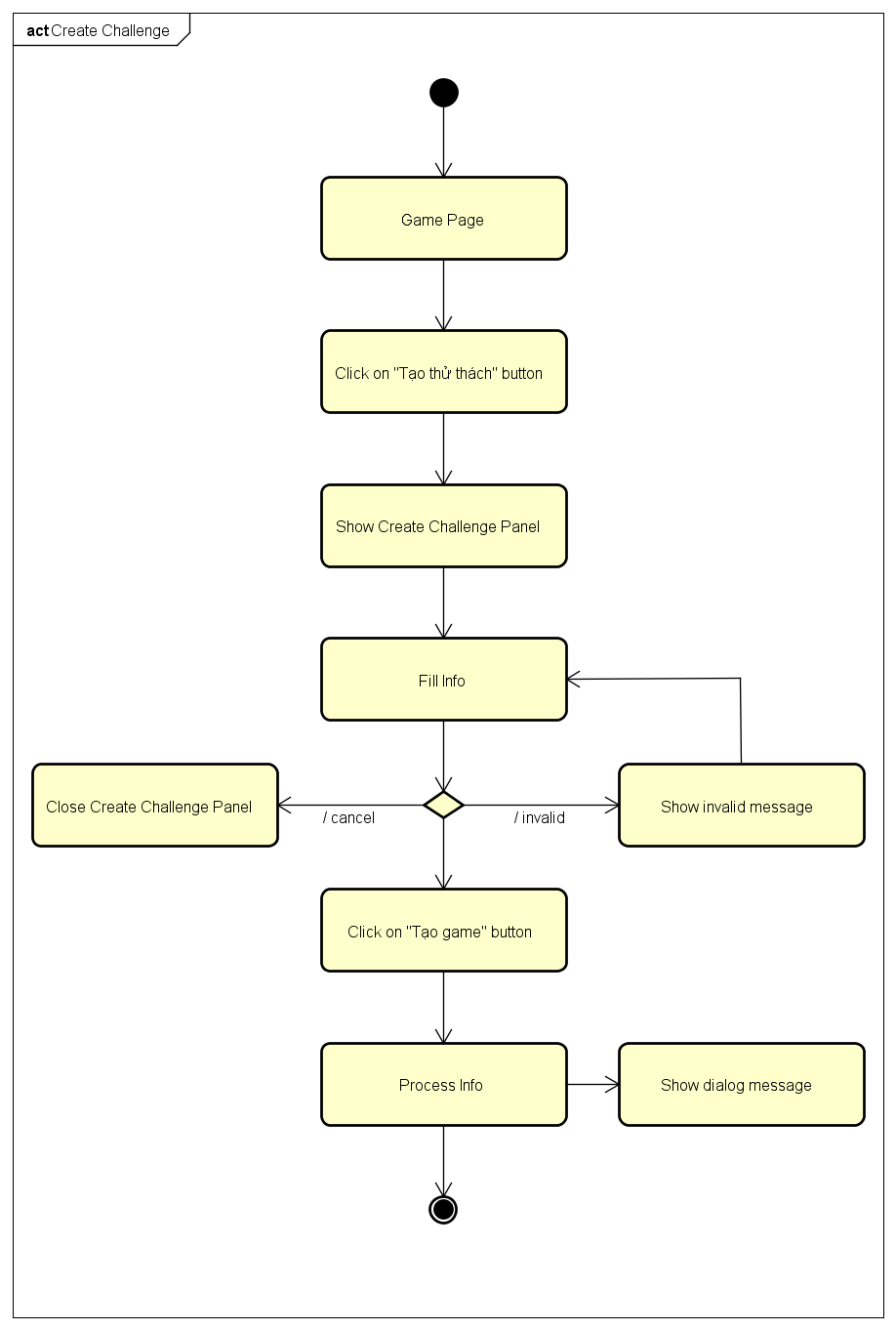
4.5.A.6. Forgot Password

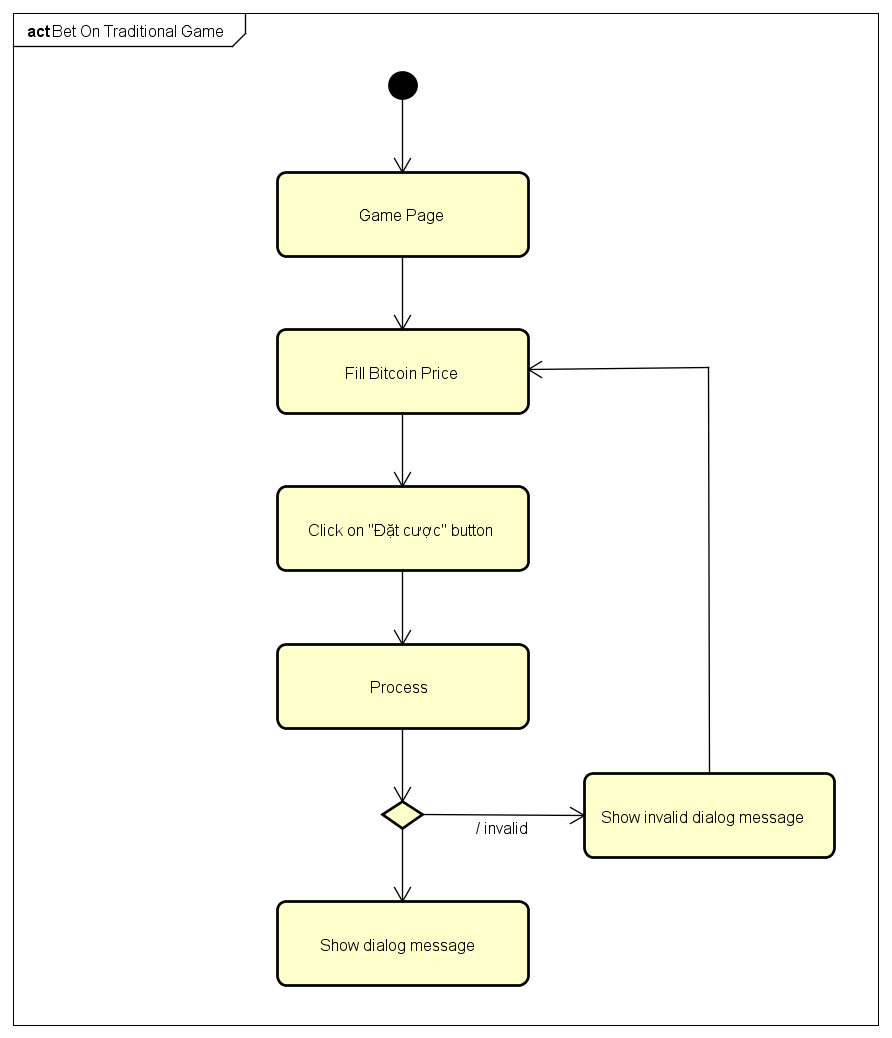
4.5.A.7. Edit Profile

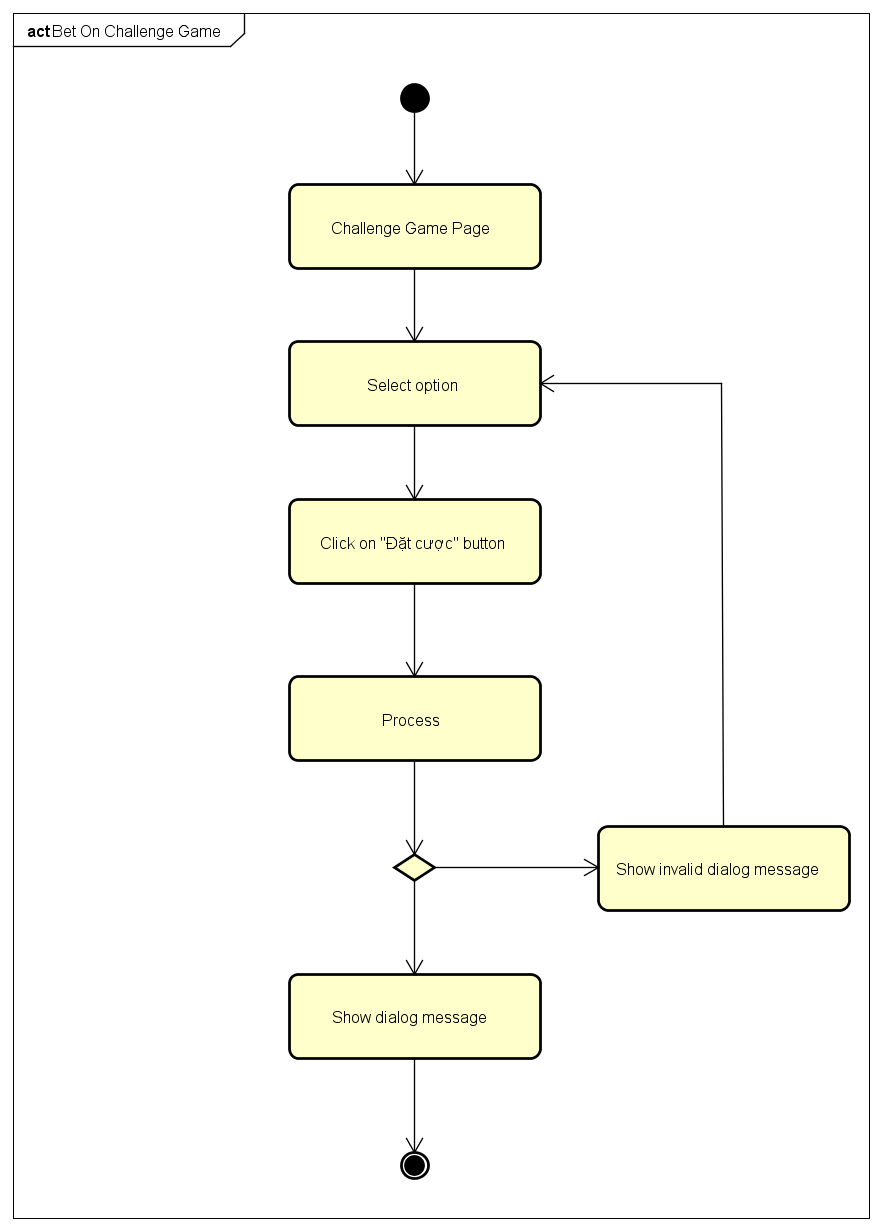
4.5.A.8. History

4.5.A.9. Notification

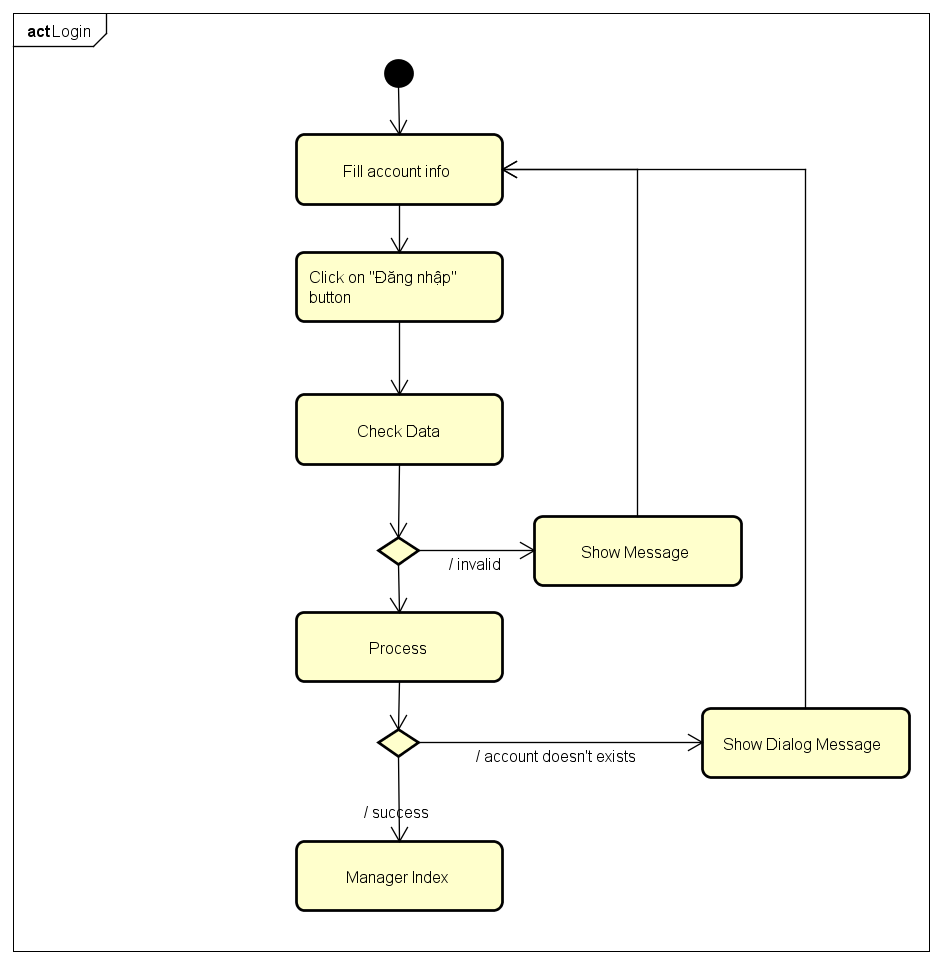
4.5.A.10. Ranking

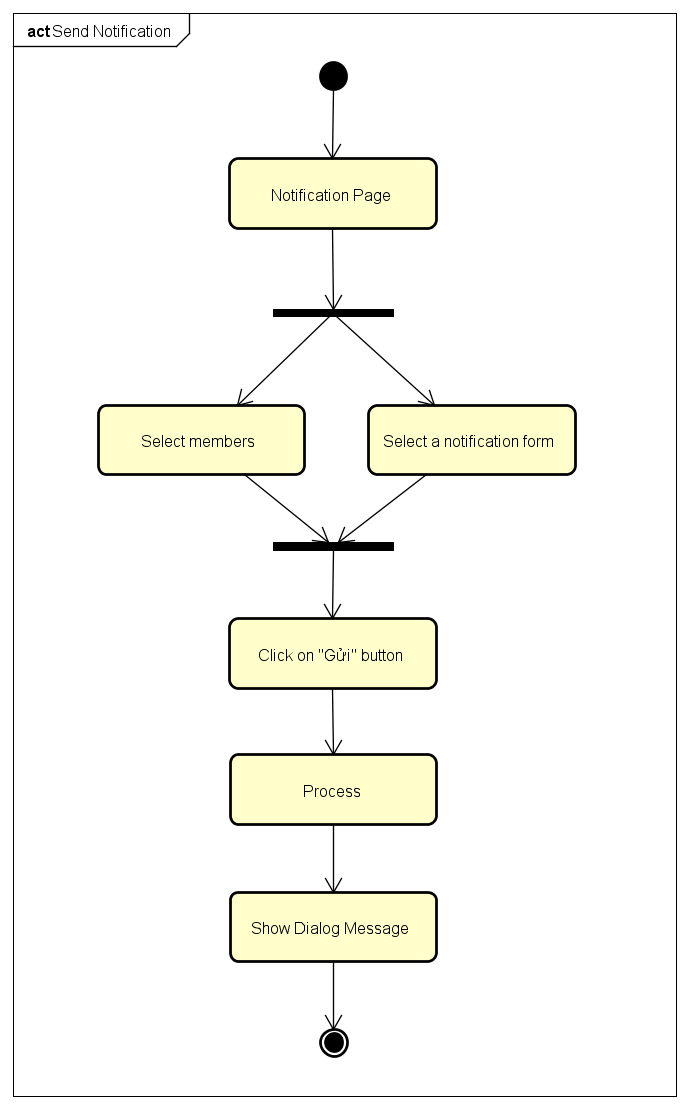
4.5.A.11. Create Challenges

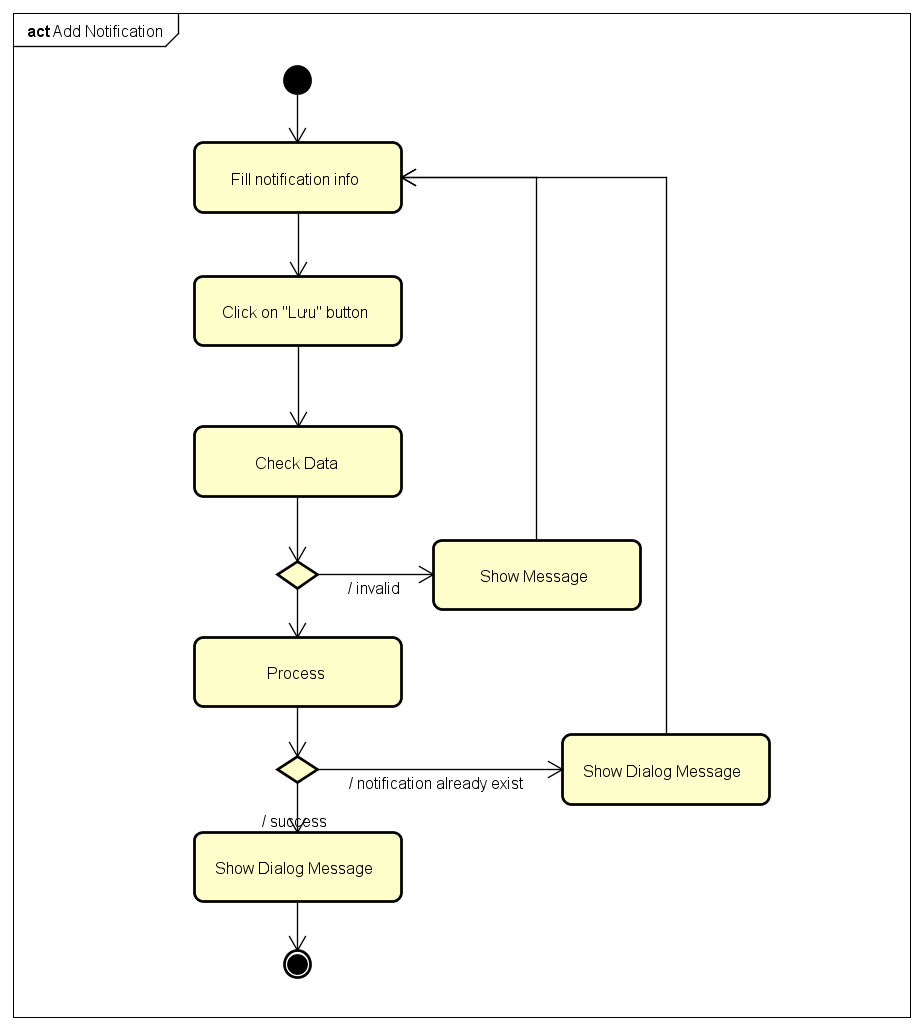
4.5.A.12. Bet On Traditional Game

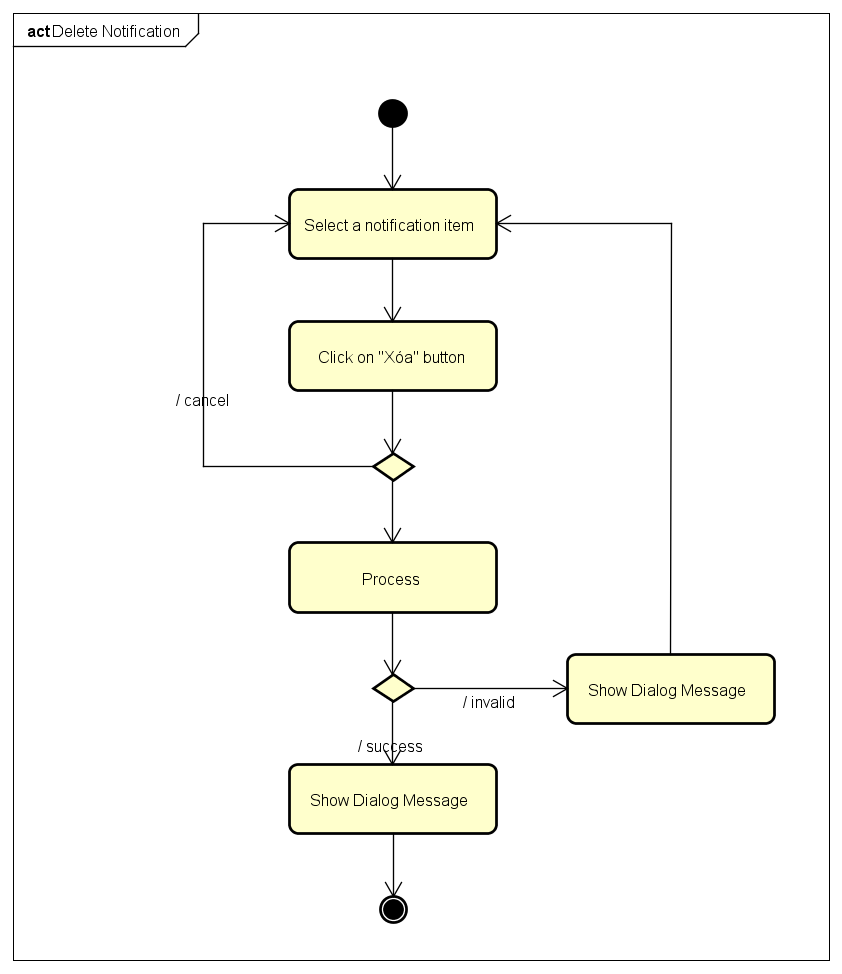
4.5.A.13. Bet On Challenge Game

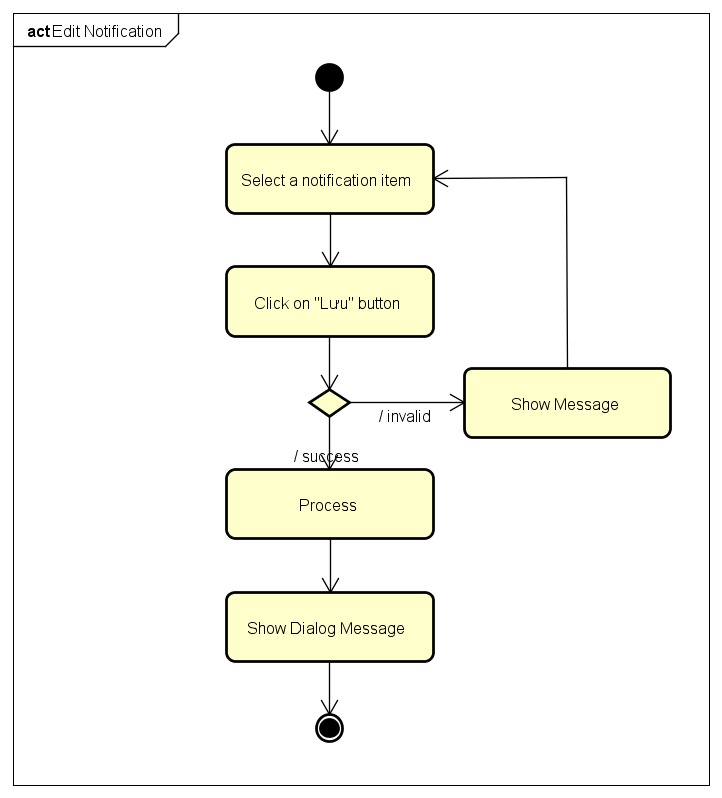
4.5.B. Admin Activities Diagram

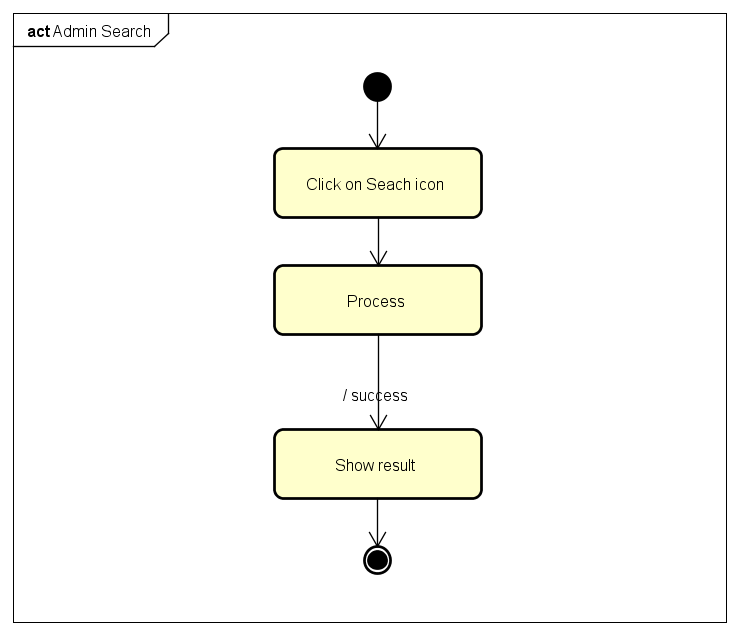
4.5.B.1. Login

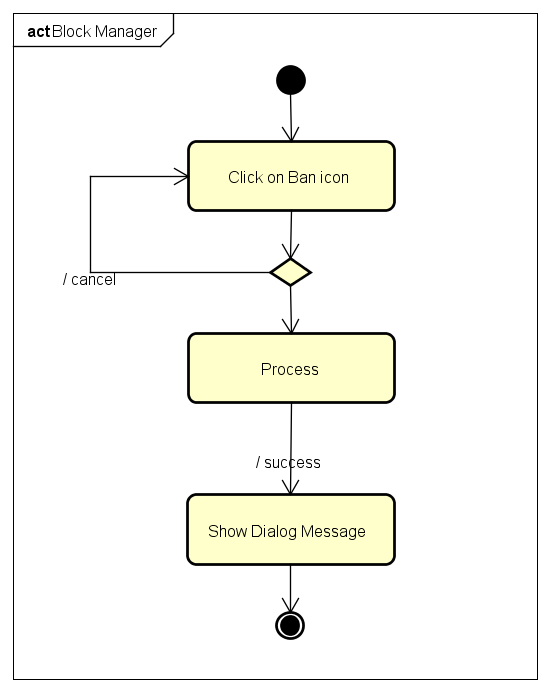
4.5.B.2.Send Notification

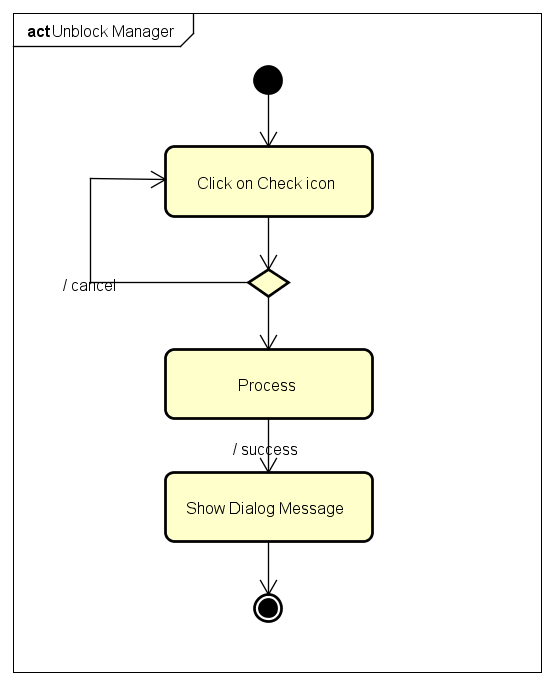
4.5.B.3.Add Notification

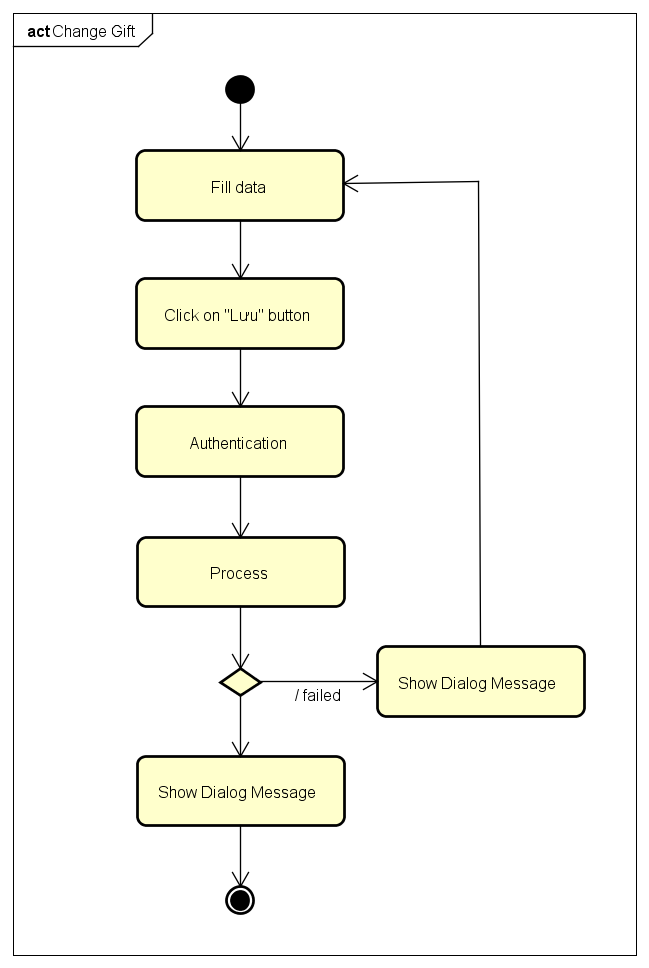
4.5.B.4.Delete Notification

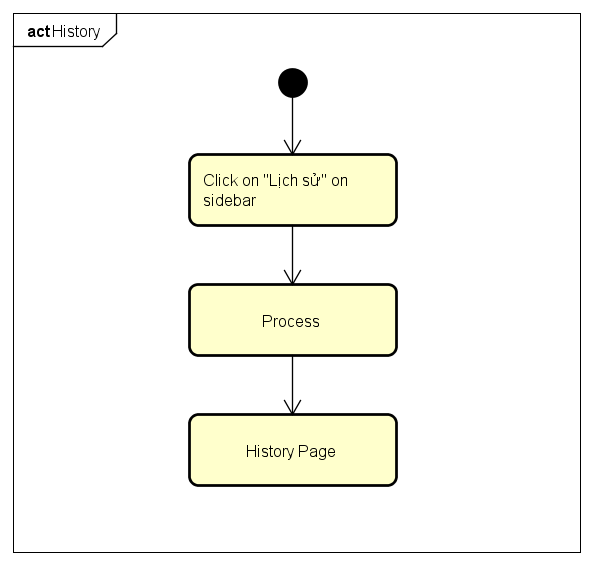
4.5.B.5.Edit Notification

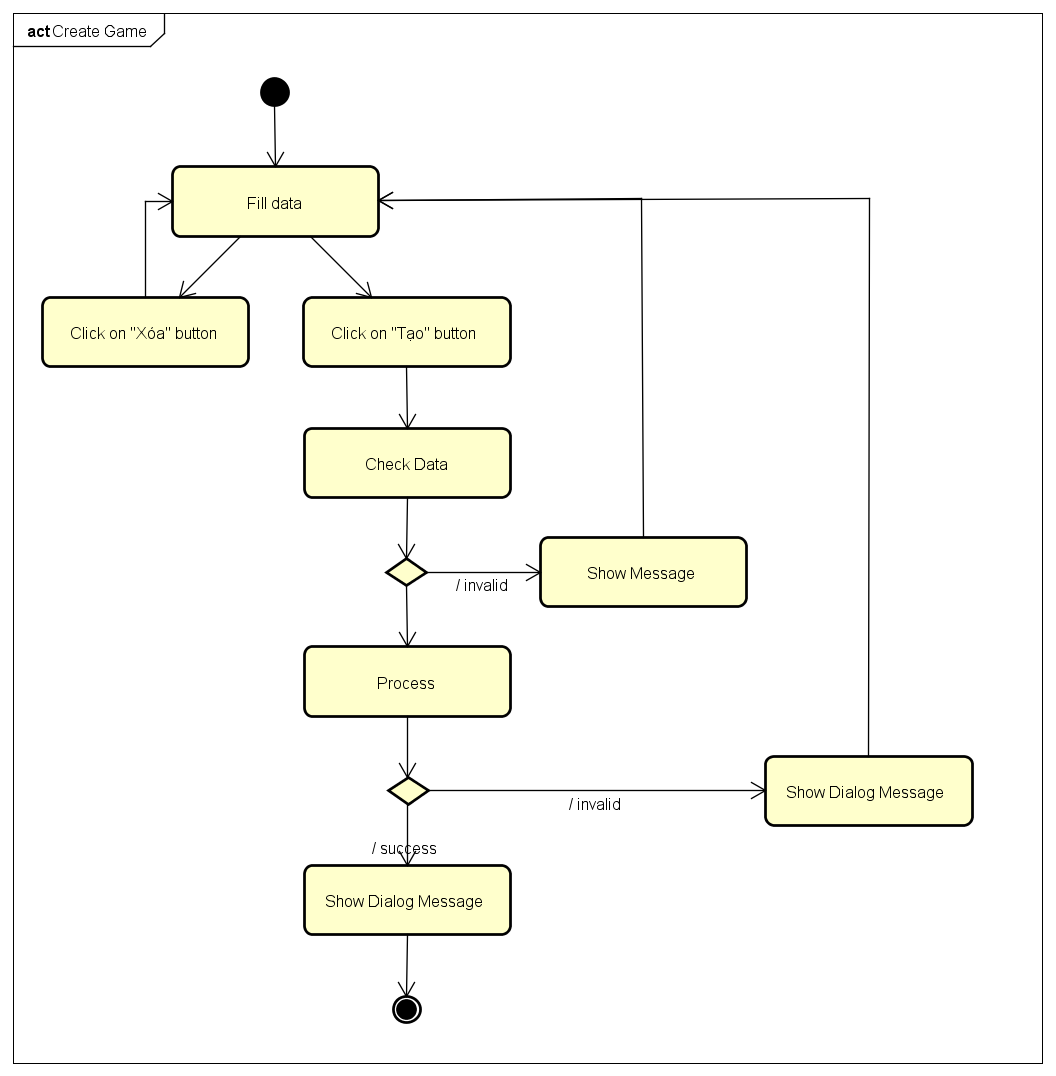
4.5.B.6.Search

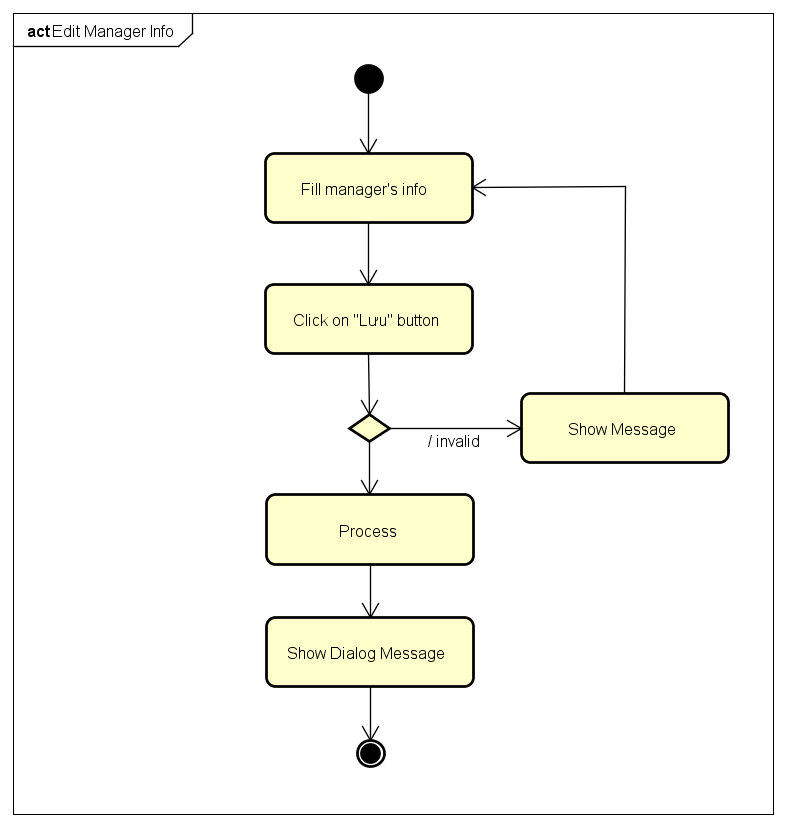
4.5.B.7.Block Manager

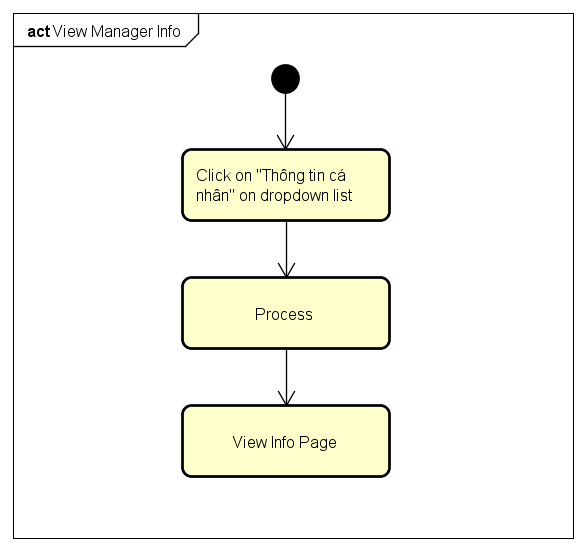
4.5.B.8.Unblock Manager

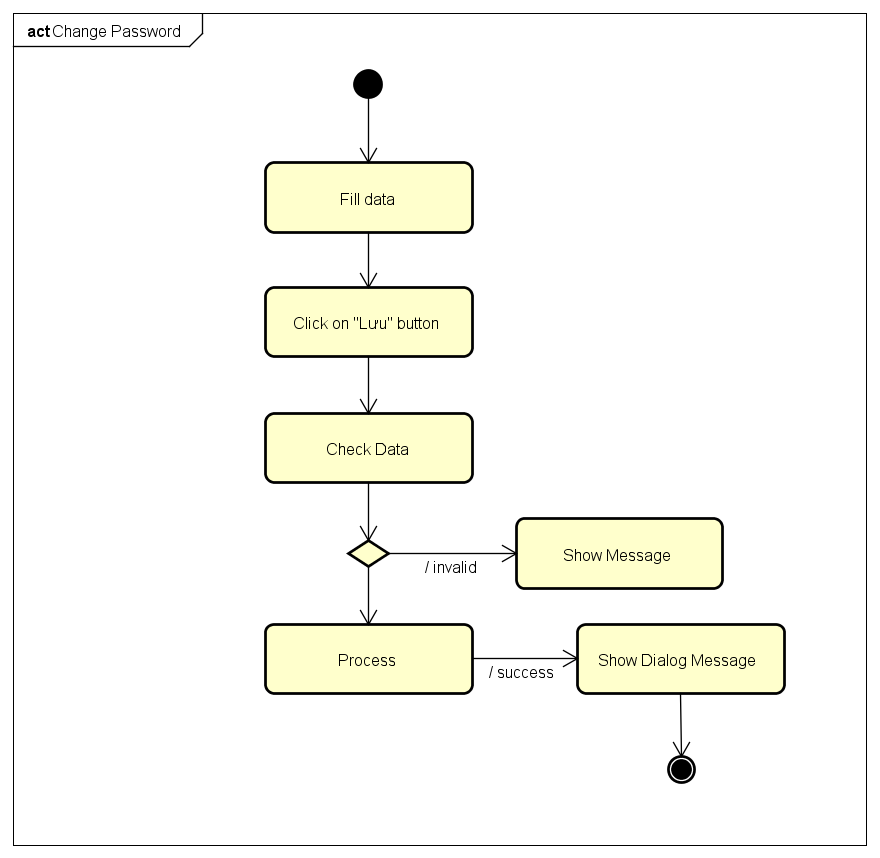
4.5.B.9.Change Gift

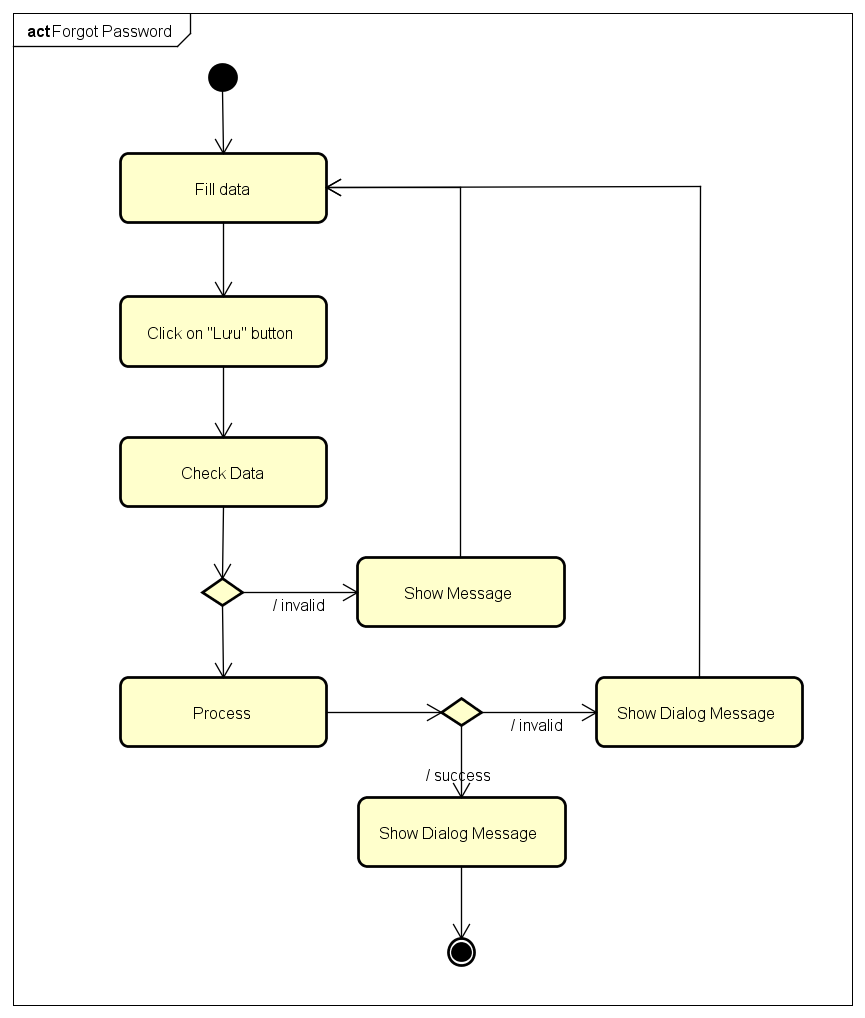
4.5.B.10.History

4.5.B.11.Create Game

4.5.B.12.Edit Manager Information

4.5.B.13.View Manager Information

4.5.B.14.Change Password

4.5.B.15.Forgot Password

4.6. CRC Cards and Class Diagram

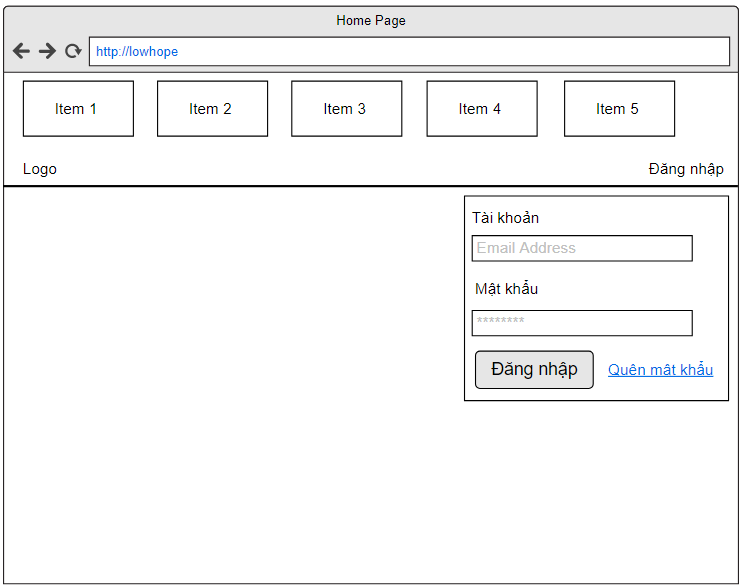
4.6.1. CRC Cards

| AdminAppointmentController | |
| --- | --- |
| Responsibilities | Collaborators |
| Get list of appointments | GetAppointmentList  Param  NormalRequest  GetAppointmentListRequestData  View |
| Get list of rooms | GetRoomList  Param  NormalRequest  GetRoomListRequestData  View |
| Get list of appointments by doctor | GetAppointmentListByDoctor  Param  NormalRequest  GetByIDRequestData  View |
| Update validate | UpdateValidate  Param  NormalRequest  UpdateValidateRequestData  Response |
| Cancel validate | CancelValidate  Param  NormalRequest  CancelValidateRequestData  Response |
| Update status of appointment | UpdateStatusOfAppointment  Param  NormalRequest  UpdateStatusRequestData  Response |
| Cancel appointment | CancelAppointmentByID  Param  NormalRequest  GetByIDRequestData  Reponse |
| Check in | CheckInById  Param  NormalRequest  GetByIDRequestData  Response |

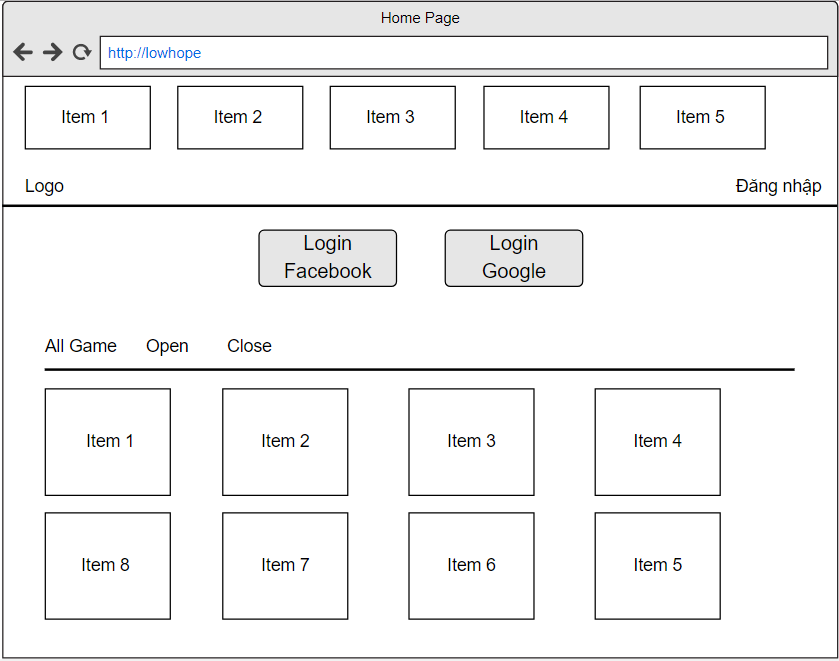
4.6.2. Class Diagram

4.7.A. User Interface Design

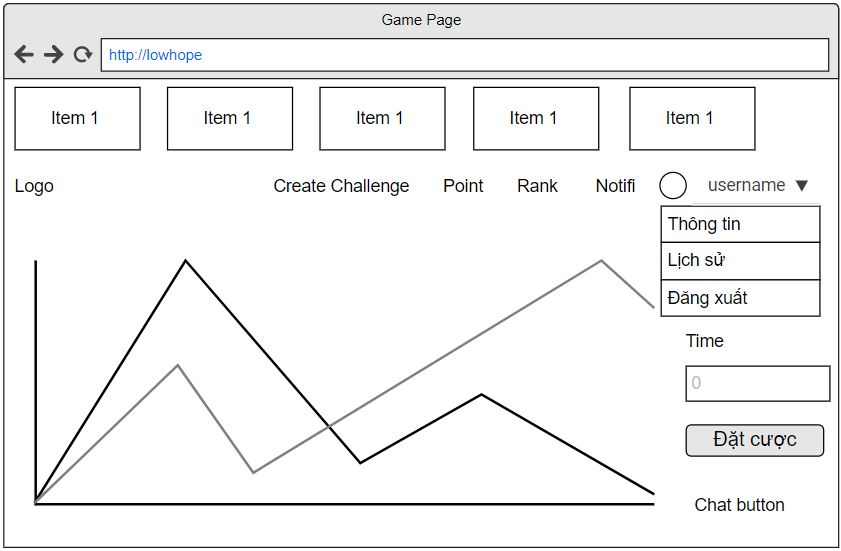
4.7.A.1. Register and Login



4.7.A.2. Login Using Facebook And Google Plus

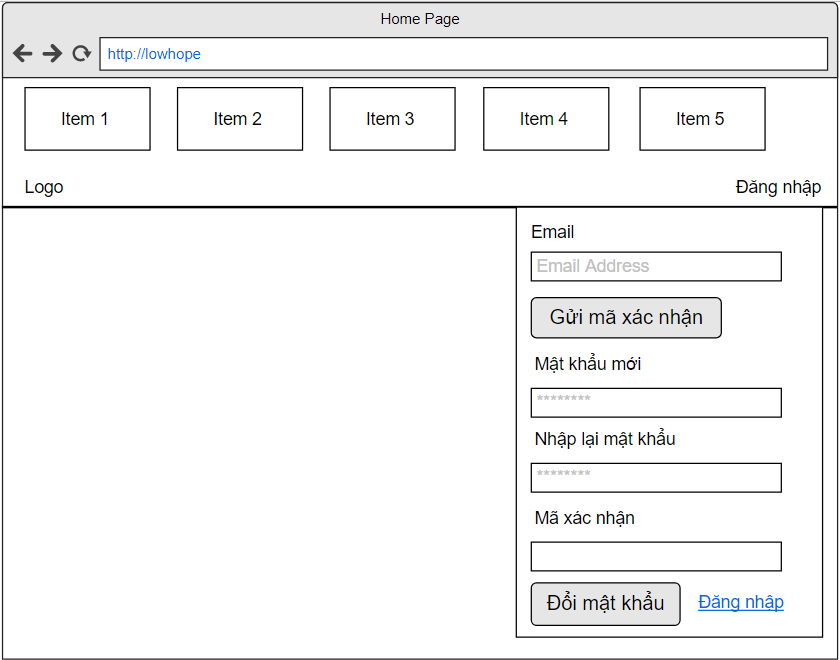


4.7.A.4. Logout

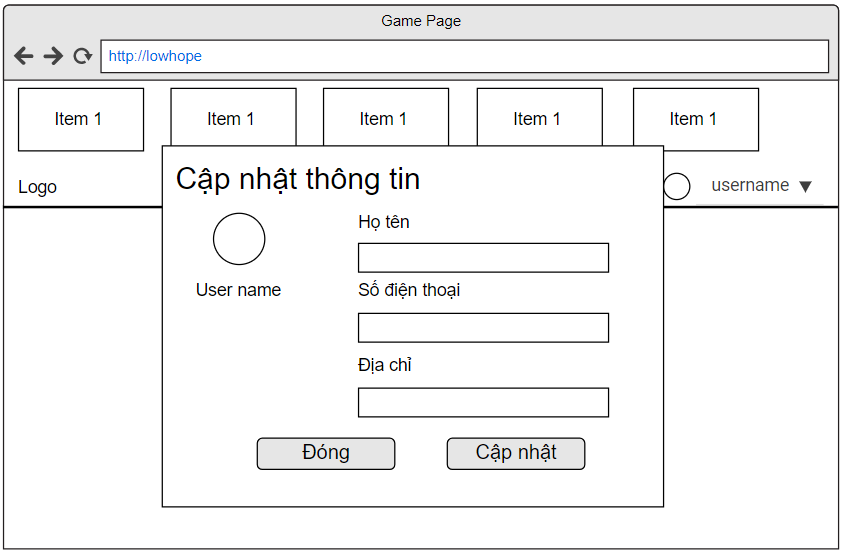


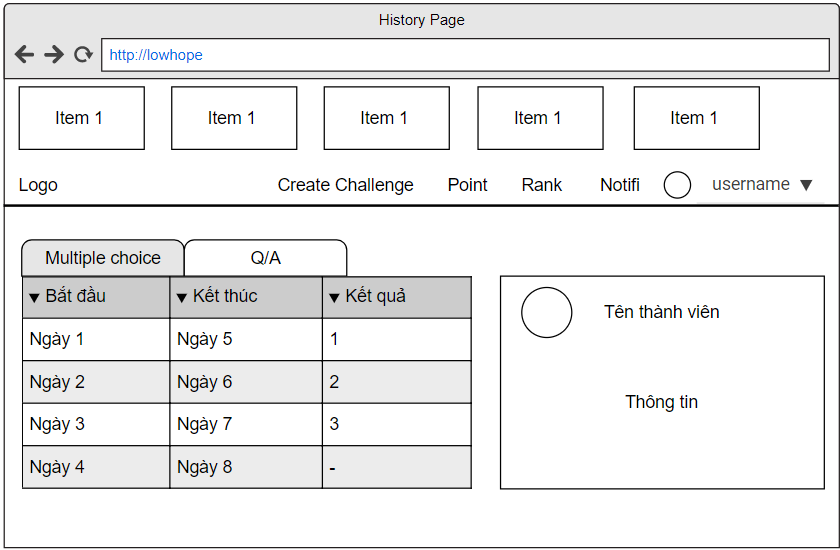
4.7.A.5. Change Password

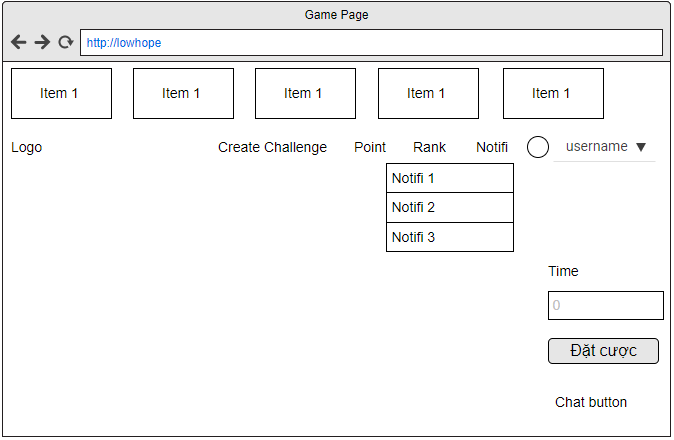
4.7.A.6. Forgot Password

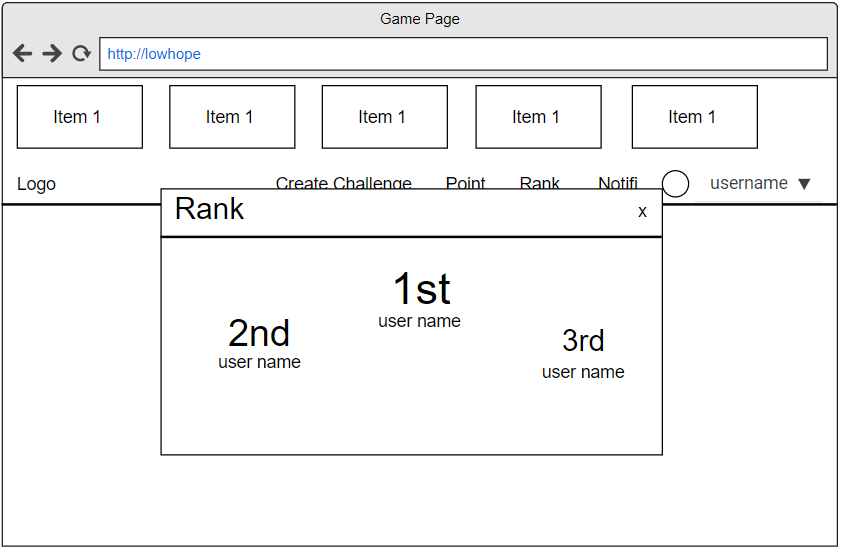


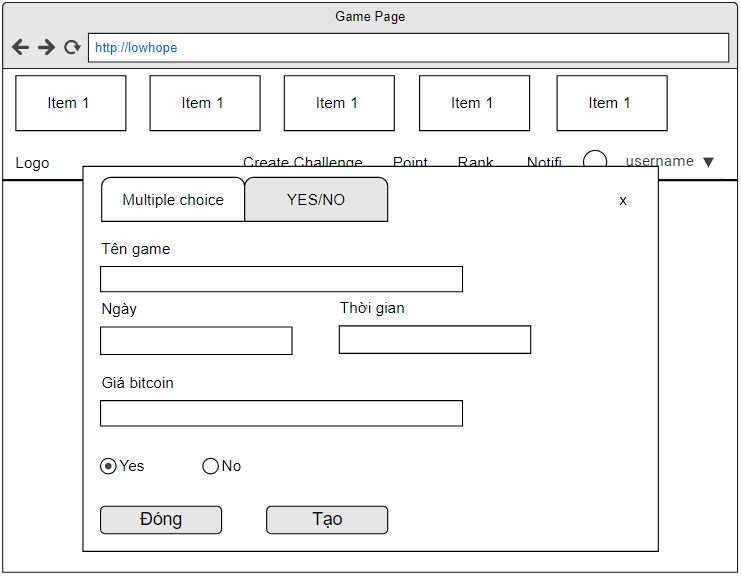
4.7.A.7. Edit Profile

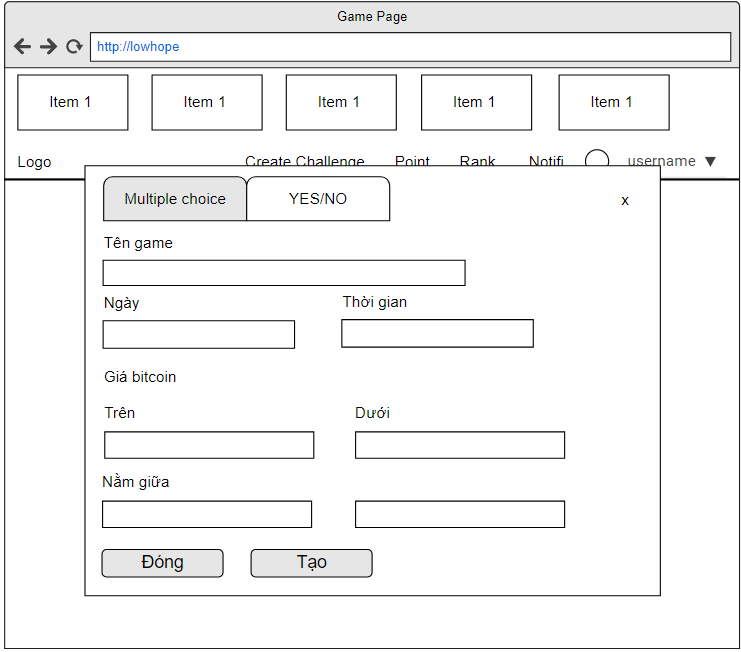


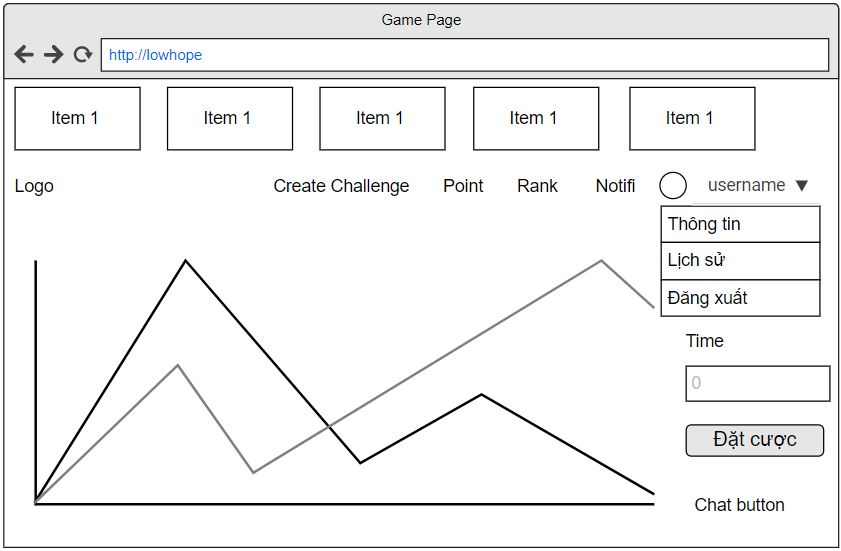
4.7.A.8. History

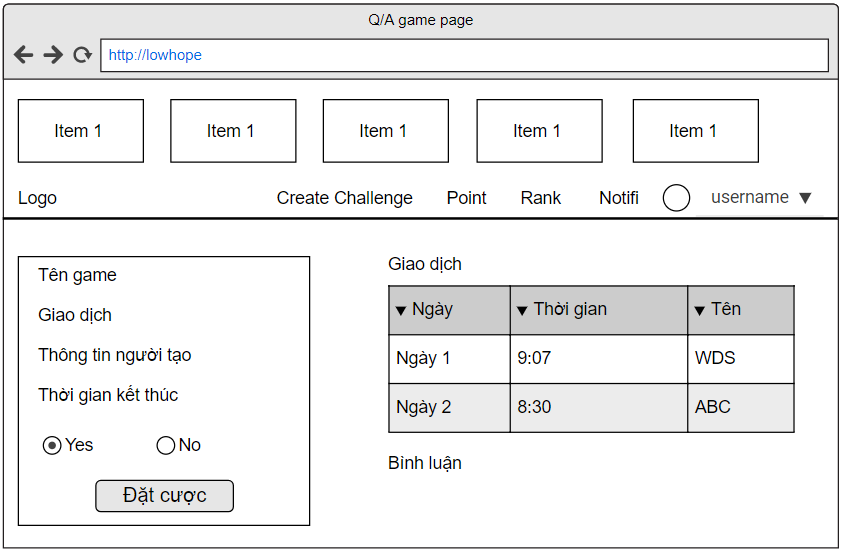
4.7.A.9. Notification

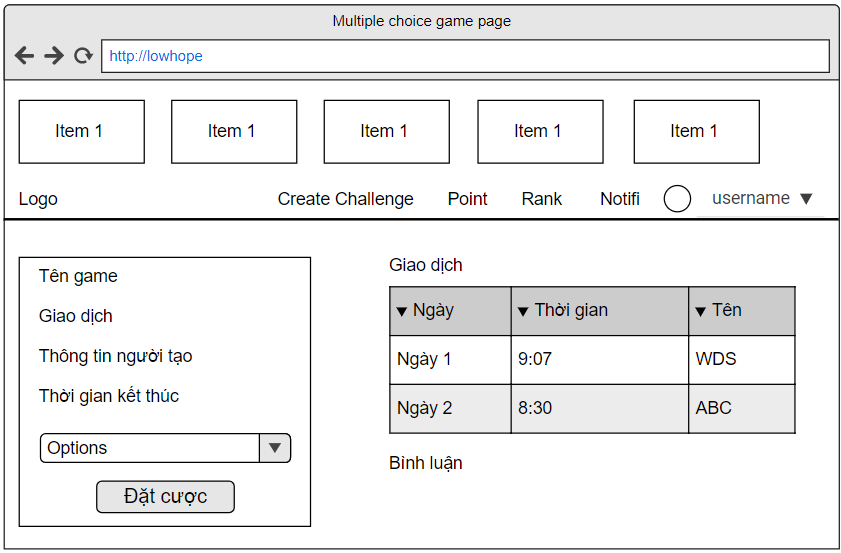
4.7.A.10. Ranking

4.7.A.11. Create Yes/No Game

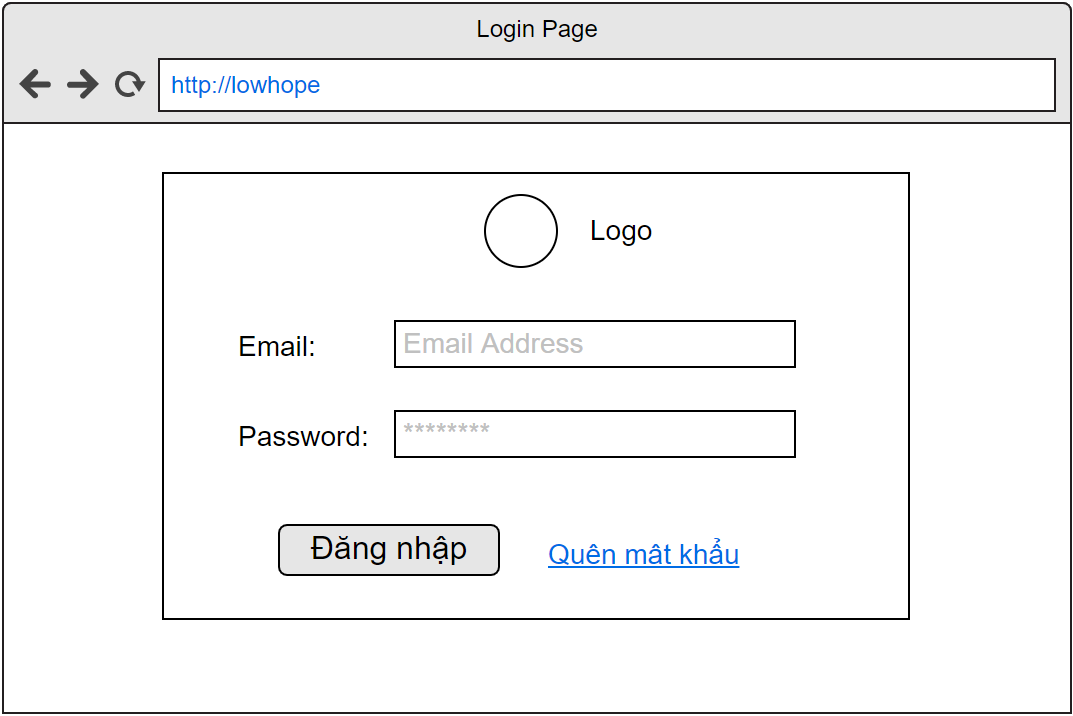
4.7.A.12. Create Multiple Choice Game

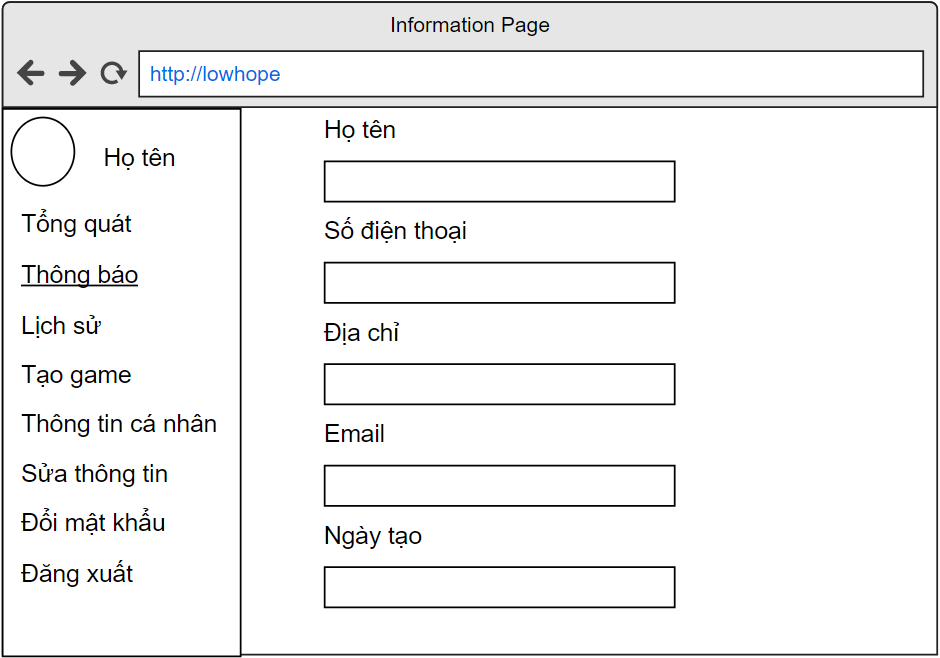
4.7.A.13. Bet On Traditional Game

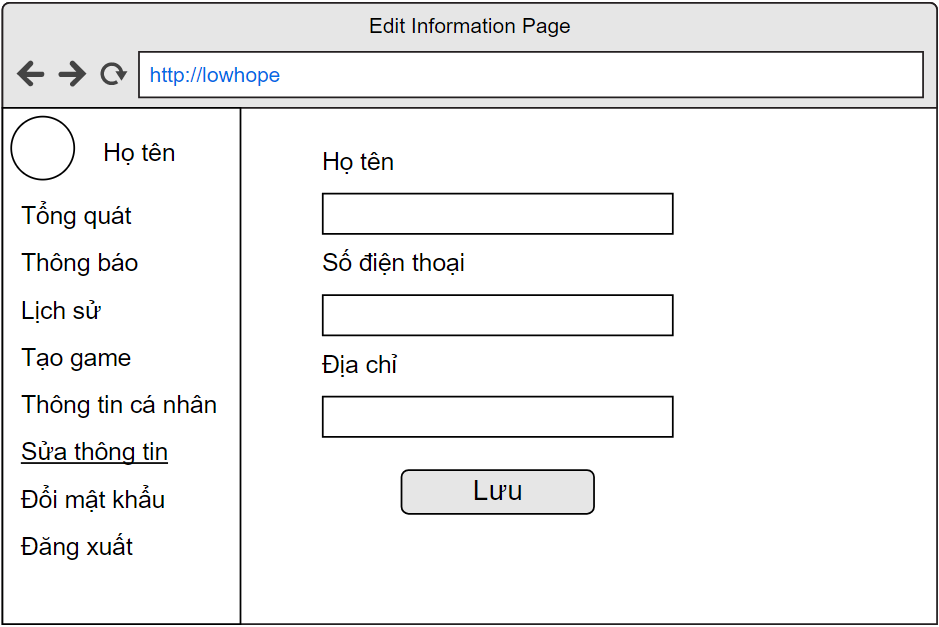
4.7.A.14. YES/No Game

4.7.A.15. Multiple Choice Game

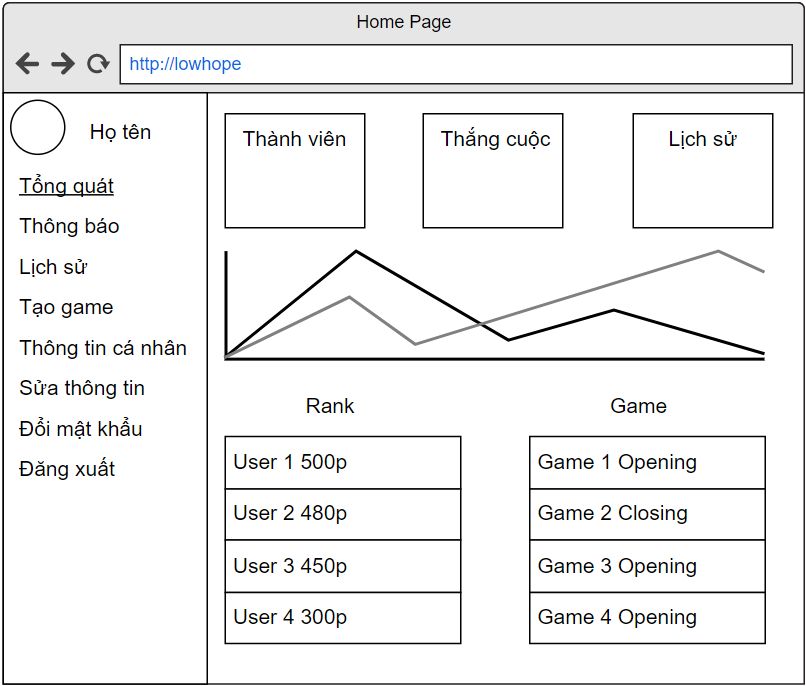
4.7.B. Admin Interface Desgin

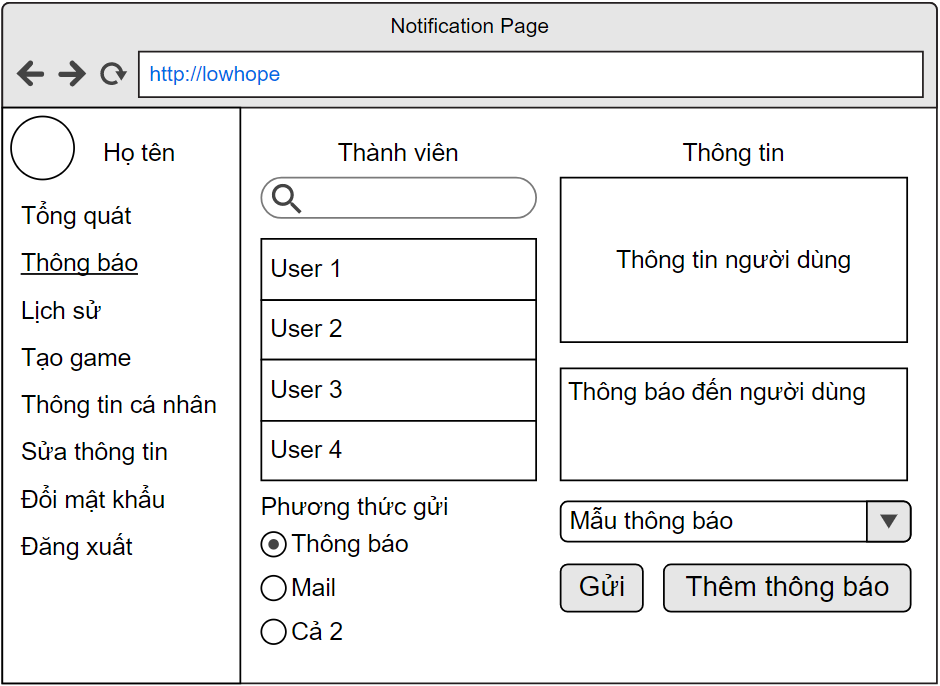
4.7.B.1. Login

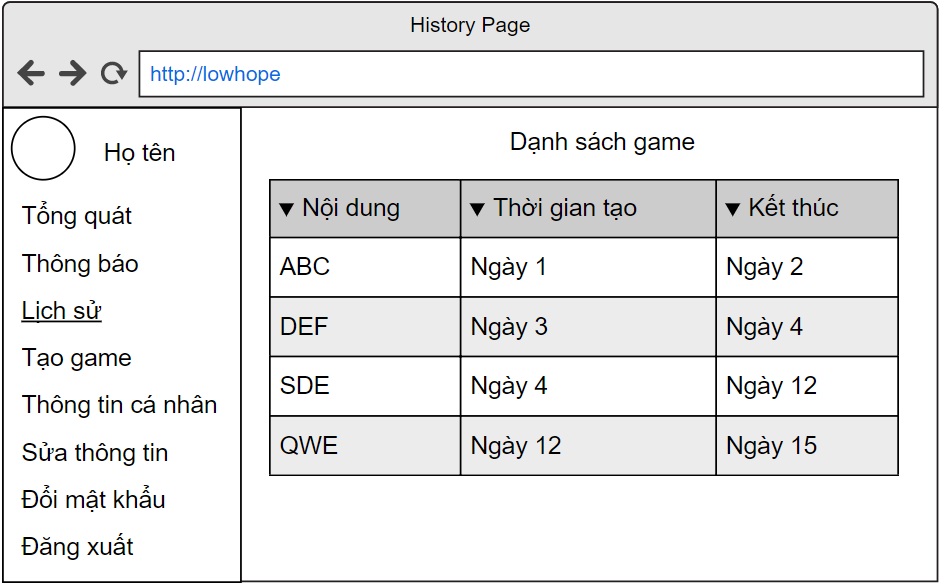
4.7.B.2. Information

4.7.B.3. Edit Information

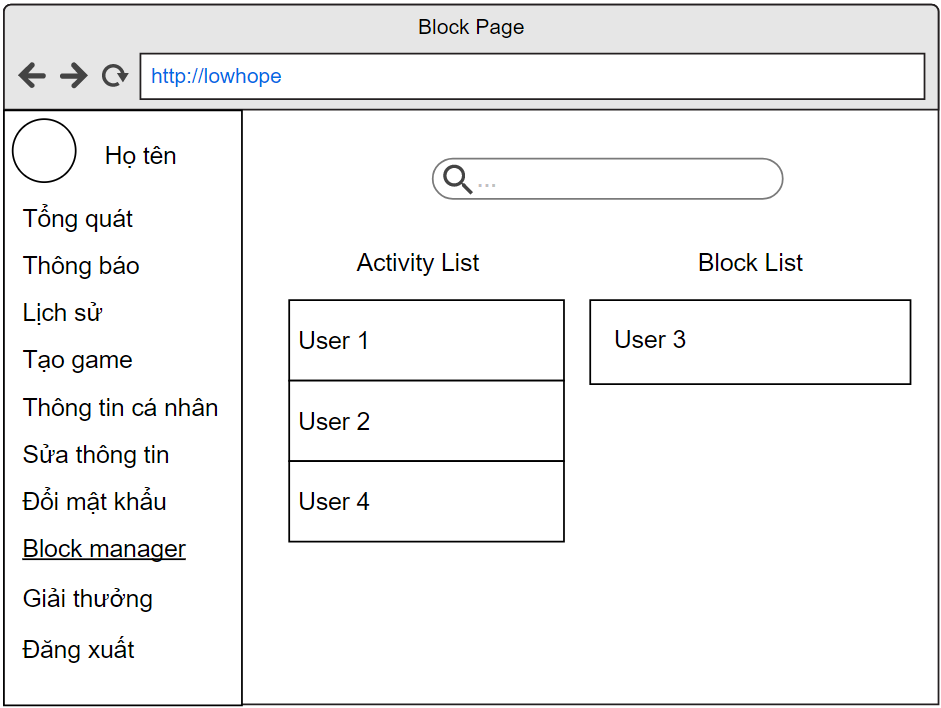
4.7.B.4. Change Password

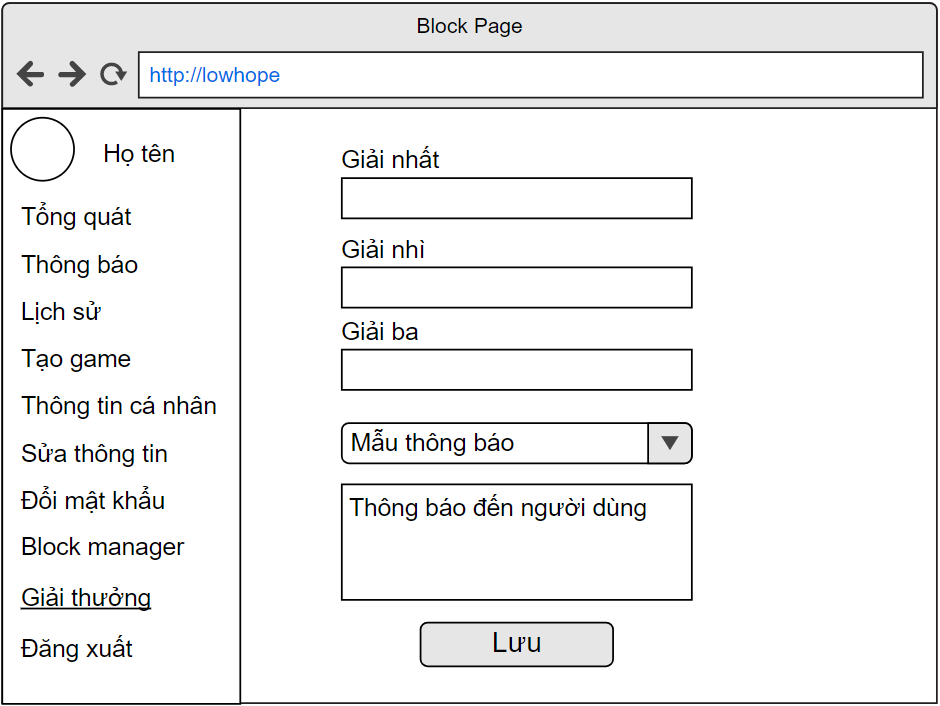
4.7.B.5. Overview

4.7.B.6. Send Notification

4.7.B.7. History

4.7.B.8. Create Challenges

4.7.B.9. Block or Unblock Manager

4.7.B.10. Change Gift

4.8. Database Design

4.8.1. Overview

Figure 1 Database Design Screen

4.8.2. ROLE table

ROLE table store users’ role information

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | ROLE\_ID (Primary key) | int | 11 | The unique identity field for role | X |
| 2 | ROLE\_NAME | varchar | 30 | Name of the role | X |

4.8.3. USERS table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | USER\_ID  (Primary key) | int | 11 | The unique identity field for user | X |
| 2 | ROLE\_ID | int | 11 | The unique identity field for role  REFERENCES ROLE(ROLE\_ID) | X |
| 3 | USER\_CIF | varchar | 50 | The user’s id is returned from facebook or google while login with these providers |  |
| 4 | USER\_NAME | nvarchar | 100 | Name of user | X |
| 5 | PASSWORD | varchar | 100 | Password to login |  |
| 6 | USER\_POINT | int | 11 | The point to join the created by system or users | X |
| 7 | EMAIL | varchar | 100 | Email of user | X |
| 8 | PHONE\_NUMBER | varchar | 30 | Phone number of user |  |
| 9 | ADDRESS | nvarchar | 255 | Address of user |  |
| 10 | CREATE\_DATE | date |  | The day that user join us | X |
| 11 | ATTENDANCE | bool |  | This field to check if user use service daily | X |
| 12 | ACTIVE | bool |  | To block or active user | X |

4.8.4. NOTIFICATION table

NOTIFICATION table store the message sent to user when necessary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | NOTICE\_ID  (PRIMARY KEY) | int | 11 | The unique identity field for message | X |
| 2 | TITLE | nvarchar | 255 | Title of message | X |
| 3 | CONTENT | text |  | Content of message | X |
| 4 | CREATE\_DATE | datetime |  | Date when message is created | X |

4.8.5. NOTIFICATION\_TYPE table

NOTIFICATION\_TYPE table store type of notice

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | TYPE\_ID  (PRIMARY KEY) | int | 11 | The unique identity field for message type | X |
| 2 | TYPE\_NAME | varchar | 20 | Name of type | X |

4.8.6. NOTIFICATION\_DETAILS table

NOTIFICATION\_DETAILS table store logs when messages sent to users

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | NOTICE\_ID  references NOTIFICATION(NOTICE\_ID) | int | 11 | The unique identity field for message | X |
| 2 | USER\_ID  references USERS(USER\_ID) | int | 11 | The unique identity field for user | X |
| 3 | TYPE\_ID  references NOTIFICATION\_TYPE(TYPE\_ID) | int | 2 | The unique identity field for type | X |
| 4 | GAME\_ID | int | 11 | To determine what game message responsible for | X |
| 5 | SEND\_DATE | datetime |  | Date that message was sent to user | X |
| 6 | SEEN | bool |  | Status of message when received by user | X |

4.8.7. CHAT\_ROOMS table

CHAT\_ROOMS table store the channel that message’s sent to

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | ROOM\_ID  (Primary key) | int | 11 | The unique identity field for channel | X |
| 2 | GAME\_ID  references SYSTEM\_GAMES (GAME\_ID) | int | 11 | Game that chat channel is created for | X |
| 3 | ROOM\_NAME | varchar | 100 | The name of channel | X |
| 4 | CREATE\_DATE | datetime |  | The day that room is created | X |

4.8.9. CHAT\_MESSAGES table

CHAT\_MESSAGES table store message that is commented by user to system game

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | MESSAGE\_ID  (Primary key) | int | 11 | The unique identity field for comment | X |
| 2 | USER\_ID  references USERS(USER\_ID) | int | 11 | The owner of comment | X |
| 3 | ROOM\_ID  references CHAT\_ROOMS(ROOM\_ID) | int | 11 | The channel that comment in | X |
| 4 | CONTENT | nvarchar | 255 | Content of comment | X |
| 5 | SEND\_DATE | dateitme |  | Date that user leave their comment | X |

4.8.10. CATEGORIES table

CATEGORIES table store currency category information

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | CATEGORY\_ID  (Primary key) | int | 11 | The unique identity field for category | X |
| 2 | CATEGORY\_NAME | varchar | 30 | Category name | X |

4.8.11. CURRENCY\_TYPE table

CURRENCY\_TYPE table store currency type

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | TYPE\_ID  (Primary key) | int | 11 | The unique identity field for type | X |
| 2 | TYPE\_NAME | varchar | 20 | Type name | X |
| 3 | CATEGORY\_ID  references CATEGORIES(CATEGORY\_ID) | int | 11 | Category of type | X |

4.8.12. CURRENCY\_DETAILS table

CURRENCY\_DETAILS table store information about a specificed currency unit at a time

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | CURRENCY\_ID  (Primary key) | int | 11 | The unique identity field for currency | X |
| 2 | PRICE | double |  | Price of currency unit at a time | X |
| 3 | UPDATE\_AT | datetime |  | The time that currency is to be value at | X |
| 4 | TYPE\_ID  references CURRENCY\_TYPE(TYPE\_ID) | int |  | Type of currency | X |

4.8.13. SYSTEM\_GAMES table

SYSTEM\_GAMES table store the game automatically created by system

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | GAME\_ID  (Primary key) | int | 11 | The unique identity field for game | X |
| 2 | TITLE | nvarchar | 255 | Game title | X |
| 3 | CONTENT | nvarchar | 255 | Game content | X |
| 4 | START\_DATE | datetime |  | Time that game starts | X |
| 5 | END\_DATE | datetime |  | Time that game ends | X |
| 6 | ACTIVE | bool |  | Status of game | X |
| 7 | POINT\_TO\_BET | double |  | Point spent to join game | X |
| 8 | RESULT | double |  | Result of game |  |
| 9 | CUR\_TYPE\_ID  references CURRENCY\_TYPE(TYPE\_ID) | int | 11 | Type of game corresponding to type of currency | X |

4.8.14. SYSTEM\_GAME\_LOGS table

SYSTEM\_GAME\_LOGS store history of system game

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | USER\_ID  references USERS(USER\_ID) | int | 11 | Game’s player | X |
| 2 | GAME\_ID  references SYSTEM\_GAMES(GAME\_ID) | int | 11 | Game that logs belong to | X |
| 3 | PRICE\_GUESS | double |  | Value of currency unit user guess | X |
| 4 | DATE\_GUESS | datetime |  | Time user leave their prediction | X |

4.8.15. AWARD table

AWARD table store gift for the 3 first prizes of system game

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | AWARD\_ID  (Primary key) | int | 11 | The unique identity field for award | X |
| 2 | PRIZE | int | 1 | The prize that the gift belongs to | X |
| 3 | AWARD\_NAME | nvarchar | 100 | Name | X |
| 4 | PRICE | double |  | Value of gift | X |
| 5 | IMAGE\_URL | varchar | 255 | Image of gift | X |
| 6 | ACTIVE | bool |  | Status of gift | X |

4.8.16. ACHIEVEMENT table

ACHIEVEMENT table store winners of system game

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | A\_ID | int | 11 | The unique identity field for achievement | X |
| 2 | USER\_ID  references USERS(USER\_ID) | int | 11 | User get achievement | X |
| 3 | AWARD\_ID  references AWARD(AWARD\_ID) | int | 11 | The ward for winner | X |
| 4 | GAME\_ID  references SYSTEM\_GAMES(GAME\_ID) | int | 11 | Game that user get achievement | X |
| 5 | GET\_AT | datetime |  | Time that user get achievement | X |

4.8.17. YN\_GAMES table

YN\_GAMES table store binary option game (true/faslse) created by users

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | GAME\_ID  (Primary key) | int | 11 | The unique identify field for game | X |
| 2 | OWNER\_ID  references USERS(USER\_ID) | int | 11 | The owner of game | X |
| 3 | CUR\_TYPE\_ID  references CURRENCY\_TYPE(TYPE\_ID) | int | 11 | The currency unit that game use to bet | X |
| 4 | TITLE | nvarchar | 255 | Game title | X |
| 5 | CONTENT | nvarchar | 255 | Game content | X |
| 6 | START\_DATE | datetime |  | Time to start game | X |
| 7 | END\_DATE | datetime |  | Time to end game | X |
| 8 | POINT\_TO\_BET | int | 5 | Point to join game | X |
| 9 | PRICE\_BET | double |  | Value of currency unit that user raise to bet | X |
| 10 | RESULT | double |  | Result of game |  |
| 11 | PLAYER\_COUNT | int | 2 | Number of players join game.  (<= 40) | X |
| 12 | ACTIVE | bool |  | Status of game. It’s set to false if game end | X |
| 13 | TOTAL\_AMOUNT | int | 11 | Total point get from players  (PLAYER\_COUNT\*POINT\_TO\_BET) | X |

4.8.18. YN\_GAME\_LOGS table

YN\_GAME\_LOGS table store history of yn\_game

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | USER\_ID  references USERS(USER\_ID) | int | 11 | Players in game | X |
| 2 | GAME\_ID  references YN\_GAMES(GAME\_ID) | int | 11 | The unique identify field for game that log is saved | X |
| 3 | ANSWER | bool |  | The answer players choose | X |
| 4 | ANS\_TIME | datetime |  | Date that user join game | X |
| 5 | IS\_WINNER | bool |  | Result of game that point out who is winner or looser |  |

4.8.19. MULTI\_CHOICE\_GAMES table

MULTI\_CHOICE\_GAMES table store multichoice game created by user

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | GAME\_ID  (Primary key) | int | 11 | The unique identify field for game | X |
| 2 | OWNER\_ID | int | 11 | The owner of game | X |
| 3 | CUR\_TYPE\_ID | int | 11 | The currency unit that game use to bet | X |
| 4 | TITLE | nvarchar | 255 | Game title | X |
| 5 | CONTENT | nvarchar | 255 | Game content | X |
| 6 | START\_DATE | datetime |  | Time to start game | X |
| 7 | END\_DATE | datetime |  | Time to end game | X |
| 8 | POINT\_TO\_BET | int | 5 | Point to join game | X |
| 9 | PRICE\_BELOW | double |  | Minimum value of currency unit that user raise to bet | X |
| 10 | PRICE\_ABOVE | double |  | Maximum value of currency unit that user raise to bet | X |
| 11 | RESULT | double |  | Value of currency unit when game ends |  |
| 12 | PLAYER\_COUNT | int | 2 | Number of players join game.  (<= 40) | X |
| 13 | ACTIVE | bool |  | Status of game. It’s set to false if game end | X |
| 14 | TOTAL\_AMOUNT | int | 11 | Total point get from players  (PLAYER\_COUNT\*POINT\_TO\_BET) |  |

4.8.20. MULTI\_CHOICE\_GAME\_LOGS table

MULTI\_CHOICE\_GAME\_LOGS table store history of multi choice game

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Field Name** | **Type** | **Size** | **Description** | **Not Null** |
| 1 | USER\_ID  references USERS(USER\_ID) | int | 11 | Players in game | X |
| 2 | GAME\_ID  references YN\_GAMES(GAME\_ID) | int | 11 | The unique identify field for game that log is saved | X |
| 3 | PRICE\_BELOW | bool |  | Option that player choose. True if the real price < MULTI\_CHOICE\_GAMES (PRICE\_BELOW) | X |
| 4 | PRICE\_BETWEEN | bool |  | Option that player choose. True if the MULTI\_CHOICE\_GAMES (PRICE\_BELOW) <= real price <= MULTI\_CHOICE\_GAMES (PRICE\_ABOVE) | X |
| 5 | PRICE\_ABOVE | bool |  | Option that player choose. True if the real price > MULTI\_CHOICE\_GAMES (PRICE\_ABOVE) | X |
| 6 | ANS\_TIME | datetime |  | Date that user join game | X |
| 7 | IS\_WINNER | bool |  | Result of game that point out who is winner or looser |  |