

**FPTU Event Management**

**Software Design Document**

**Version 1.0**

**Prepared by**

**Nguyễn Thành Trung**

**Phan Minh Nhựt**

**Trịnh Nam Anh**

– Ho Chi Minh, November 2021 –

**Table of Contents**

[I. Overview 6](#_Toc88733621)

[1. Code Packages/Namespaces 6](#_Toc88733622)

[2. Database Schema 6](#_Toc88733623)

[II. Code Designs 8](#_Toc88733624)

[1. Login/Register 8](#_Toc88733625)

[a. Class Diagram 8](#_Toc88733626)

[b. Sequence Diagram(s) 8](#_Toc88733627)

[c. Database queries 9](#_Toc88733628)

[2. Log out 10](#_Toc88733629)

[a. Class Diagram 10](#_Toc88733630)

[b. Sequence Diagram(s) 10](#_Toc88733631)

[c. Database queries 10](#_Toc88733632)

[3. View Main Page 11](#_Toc88733633)

[a. Class Diagram 11](#_Toc88733634)

[b. Sequence Diagram(s) 11](#_Toc88733635)

[c. Database queries 12](#_Toc88733636)

[4. View Events 13](#_Toc88733637)

[a. Class Diagram 13](#_Toc88733638)

[b. Sequence Diagram(s) 13](#_Toc88733639)

[c. Database queries 14](#_Toc88733640)

[5. View Event’s Detail 15](#_Toc88733641)

[a. Class Diagram 15](#_Toc88733642)

[b. Sequence Diagram(s) 15](#_Toc88733643)

[c. Database queries 16](#_Toc88733644)

[6. Follow/Unfollow Event 17](#_Toc88733645)

[a. Class Diagram 17](#_Toc88733646)

[b. Sequence Diagram(s) 17](#_Toc88733647)

[c. Database queries 18](#_Toc88733648)

[7. Join Event/Unjoin Event 19](#_Toc88733649)

[a. Class Diagram 19](#_Toc88733650)

[b. Sequence Diagram(s) 19](#_Toc88733651)

[c. Database queries 20](#_Toc88733652)

[8. View Followed Events 21](#_Toc88733653)

[a. Class Diagram 21](#_Toc88733654)

[b. Sequence Diagram(s) 21](#_Toc88733655)

[c. Database queries 22](#_Toc88733656)

[9. View Posts 23](#_Toc88733657)

[a. Class Diagram 23](#_Toc88733658)

[b. Sequence Diagram(s) 23](#_Toc88733659)

[c. Database queries 24](#_Toc88733660)

[10. View Post’s Detail 25](#_Toc88733661)

[a. Class Diagram 25](#_Toc88733662)

[b. Sequence Diagram(s) 25](#_Toc88733663)

[c. Database queries 26](#_Toc88733664)

[11. Setting Profile 27](#_Toc88733665)

[a. Class Diagram 27](#_Toc88733666)

[b. Sequence Diagram(s) 27](#_Toc88733667)

[c. Database queries 28](#_Toc88733668)

[12. View User’s Accounts 29](#_Toc88733669)

[a. Class Diagram 29](#_Toc88733670)

[b. Sequence Diagram(s) 29](#_Toc88733671)

[c. Database queries 29](#_Toc88733672)

[13. Update User’s Accounts 30](#_Toc88733673)

[a. Class Diagram 30](#_Toc88733674)

[b. Sequence Diagram(s) 30](#_Toc88733675)

[c. Database queries 30](#_Toc88733676)

[14. View Locations 31](#_Toc88733677)

[a. Class Diagram 31](#_Toc88733678)

[b. Sequence Diagram(s) 31](#_Toc88733679)

[c. Database queries 31](#_Toc88733680)

[15. Add New Location 32](#_Toc88733681)

[a. Class Diagram 32](#_Toc88733682)

[b. Sequence Diagram(s) 32](#_Toc88733683)

[c. Database queries 32](#_Toc88733684)

[16. Update Location 33](#_Toc88733685)

[a. Class Diagram 33](#_Toc88733686)

[b. Sequence Diagram(s) 33](#_Toc88733687)

[c. Database queries 33](#_Toc88733688)

[17. Remove Location 34](#_Toc88733689)

[a. Class Diagram 34](#_Toc88733690)

[b. Sequence Diagram(s) 34](#_Toc88733691)

[c. Database queries 34](#_Toc88733692)

[18. Create Event 35](#_Toc88733693)

[a. Class Diagram 35](#_Toc88733694)

[b. Sequence Diagram(s) 35](#_Toc88733695)

[c. Database queries 35](#_Toc88733696)

[19. View Created Event 36](#_Toc88733697)

[a. Class Diagram 36](#_Toc88733698)

[b. Sequence Diagram(s) 36](#_Toc88733699)

[c. Database queries 37](#_Toc88733700)

[20. Edit Event 38](#_Toc88733701)

[a. Class Diagram 38](#_Toc88733702)

[b. Sequence Diagram(s) 38](#_Toc88733703)

[c. Database queries 38](#_Toc88733704)

[21. Create Post 39](#_Toc88733705)

[a. Class Diagram 39](#_Toc88733706)

[b. Sequence Diagram(s) 39](#_Toc88733707)

[c. Database queries 39](#_Toc88733708)

[22. View Created Post 40](#_Toc88733709)

[a. Class Diagram 40](#_Toc88733710)

[b. Sequence Diagram(s) 40](#_Toc88733711)

[c. Database queries 41](#_Toc88733712)

[23. Edit Post 42](#_Toc88733713)

[a. Class Diagram 42](#_Toc88733714)

[b. Sequence Diagram(s) 42](#_Toc88733715)

[c. Database queries 42](#_Toc88733716)

[24. Upload File 43](#_Toc88733717)

[a. Class Diagram 43](#_Toc88733718)

[b. Sequence Diagram(s) 43](#_Toc88733719)

[c. Database queries 43](#_Toc88733720)

# I. Overview

## 1. Code Packages/Namespaces

*Graphical user interface

Description automatically generated with low confidenceA picture containing diagram

Description automatically generated*

***Package descriptions & package class naming conventions***

|  |  |  |
| --- | --- | --- |
| **No** | **Package** | **Description** |
| *01* | *src.java.Controller* | *Containing servlets* |
| *02* | *src.java.DAO* | *Containing underlying tools that help with the Data Access Object processes with database* |
| *03* | *src.java.DTO* | *Containing underlying tools that help with the Data Transfer Object.* |
| *04* | *src.java.Extension* | *Containing underlying tools that help with data formatting and common action supporting.* |
| *05* | *src.java.GoogleAPI* | *Containing GoogleAPI which is used for authentication features.* |
| *05* | *src.java.Utils* | *Containing underlying tools that help with connection or disconnect to the database.* |
| *07* | *web.CSS* | *Containing Cascading Style Sheets files for client’s view.* |
| *06* | *web.JS* | *Containing JavaScript files for client’s view.* |
| *08* | *web.about* | *Containing prepared data for specified client’s page.* |
| *09* | *web.images* | *Containing title images for events or posts.* |
| *09* | *web.img* | *Containing default images for client’s view.* |

## 2. Database Schema



***Table descriptions & package class naming conventions are as below***

|  |  |  |
| --- | --- | --- |
| **No** | **Table** | **Description** |
| *01* | *tblRoles* | *Specify role for each account.*  *- Primary key: roleId* |
| *02* | *tblStatusUser* | *Specify status (permit or unpermit) for each account.*  *- Primary key: statusId* |
| *03* | *tblUsers* | *Specify user’s identification for each account*  *- Primary key: userId*  *- Foreign keys: statusId, roleId* |
| *04* | *tblSlots* | *Specify start time and end time for each slot (6 slots per day).*  *- Primary key: slotId* |
| *05* | *tblLocations* | *Specify location for hosted events*  *- Primary key: locationId* |
| *06* | *tblEvents* | *Specify event’s details for each existed event.*  *- Primary key: eventId*  *- Foreign keys: locationId, startSlot, endSlot, statusId, userId* |
| *07* | *tblFollowedEvents* | *Specify events which are followed or used to be followed by users.*  *- Primary key: userId, eventId*  *- Foreign keys: userId, eventId* |
| *08* | *tblParticipants* | *Specify users who join in or were in charged of joining in specified event.*  *- Primary key: userId, eventId*  *- Foreign keys: userId, eventId* |
| *09* | *tblStatusPost* | *Specify status for each existed post.*  *- Primary key: statusId* |
| *10* | *tblPosts* | *Specify post’s details*  *- Primary key: postId*  *- Foreign keys: userId , statusId* |
| *11* | *tblComments* | *Specify comments in specified events or posts.*  *- Primary key: userId*  *- Foreign keys: eventide, userId, postId* |

# II. Code Designs

## 1. Login/Register

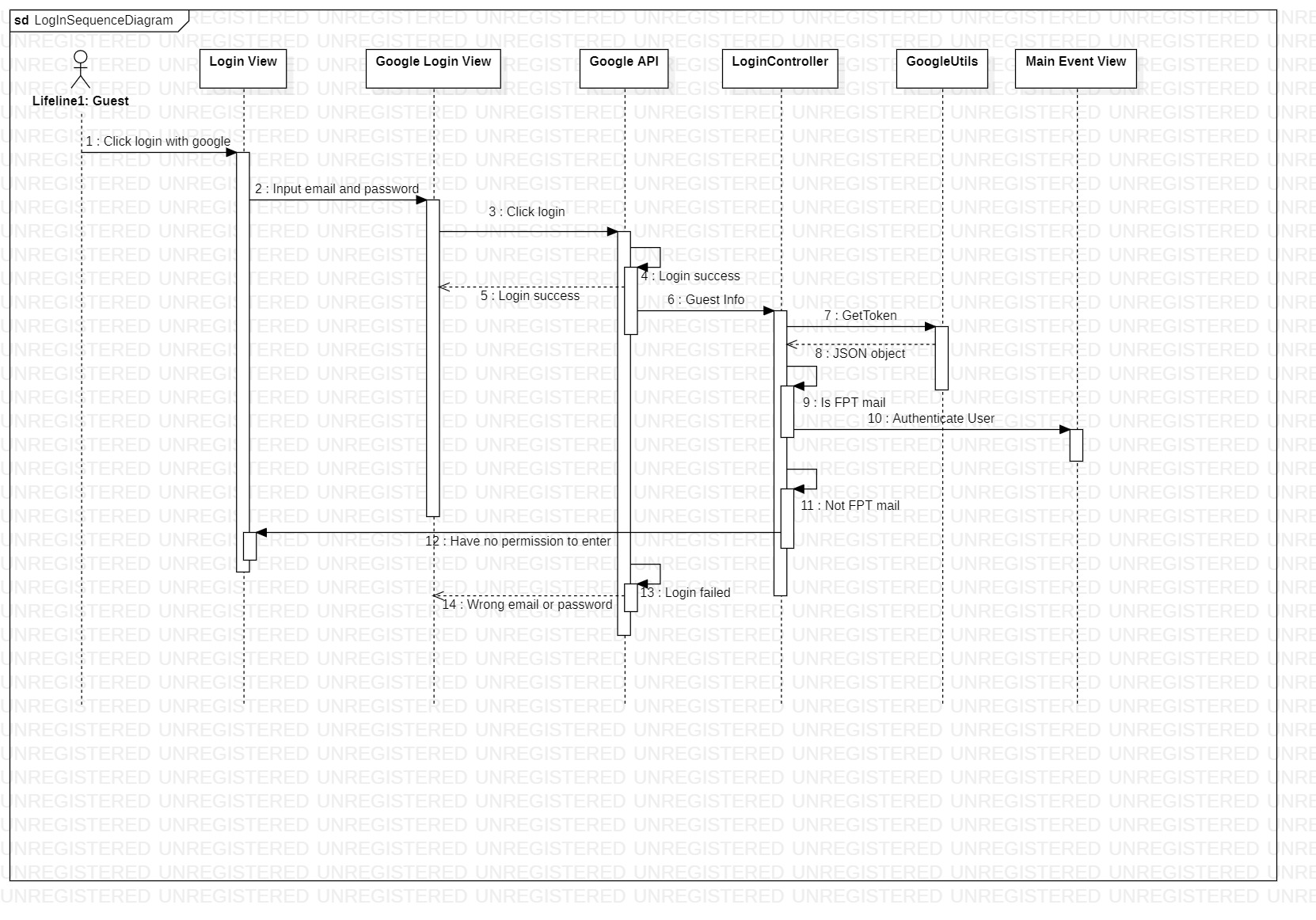
### a. Class Diagram

Timeline

Description automatically generated with medium confidence

### b. Sequence Diagram(s)

Figure 1 - Login



### c. Database queries

- Get user’s data from database:

userID = user’s ID from user’s data at Figure 1 step 10.

*SELECT username, r.roleName, s.statusName*

*FROM tblUsers u, tblStatusUser s, tblRoles r*

*WHERE u.userId =* userID *AND u.statusId = s.statusId AND u.roleId = r.roleId*

## 2. Log out

### a. Class Diagram

Background pattern

Description automatically generated with low confidence

### b. Sequence Diagram(s)

Figure 1 - Logout Diagram

Description automatically generated

### c. Database queries

*- No query for this feature.*

## 3. View Main Page

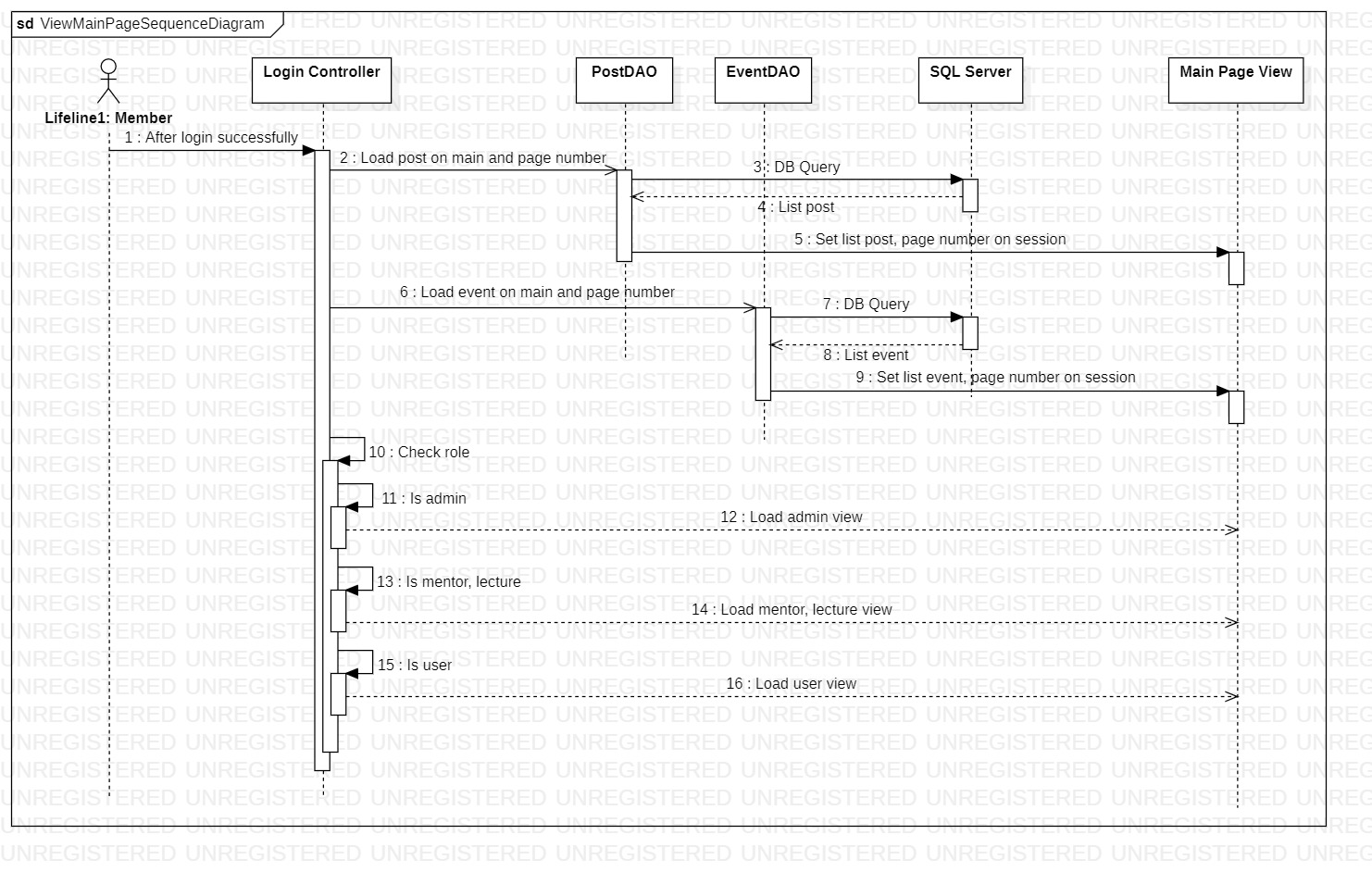
### a. Class Diagram

Diagram, text

Description automatically generated

### b. Sequence Diagram(s)

Figure 3 – View main page



### c. Database queries

- Get events by page

Search = %%

Index = 1

PageSize = 6

*WITH tblEventPage AS*

*(SELECT (ROW\_NUMBER() over (order by startDate) ) AS RowNum,*

*eventId, userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, s.statusName AS status*

*FROM tblEvents e, tblStatusEvent s*

*WHERE* title like search AND *e.statusId = s.statusId)*

*SELECT eventId, userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, status*

*FROM tblEventPage*

*WHERE RowNum BETWEEN* Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

- Update events’ status each time query event’s list

UPDATE tblEvents

SET statusId = (SELECT statusId FROM tblStatusEvent WHERE statusName = ?)

WHERE eventId = ?

(Note: parameters’ values is passed from “Get event by page”)

- Get posts by page

Search = %%

Index = 1

PageSize = 3

WITH tblPostPage AS

(SELECT (ROW\_NUMBER() over (order by createDate) ) AS RowNum,

postId, userId, title, content, video, createDate,s.statusName AS status

FROM tblPosts e, tblStatusPost s

WHERE title like Search AND e.statusId = s.statusId)

SELECT postId, userId, title, content, video, createDate, status

FROM tblPostPage

WHERE RowNum Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

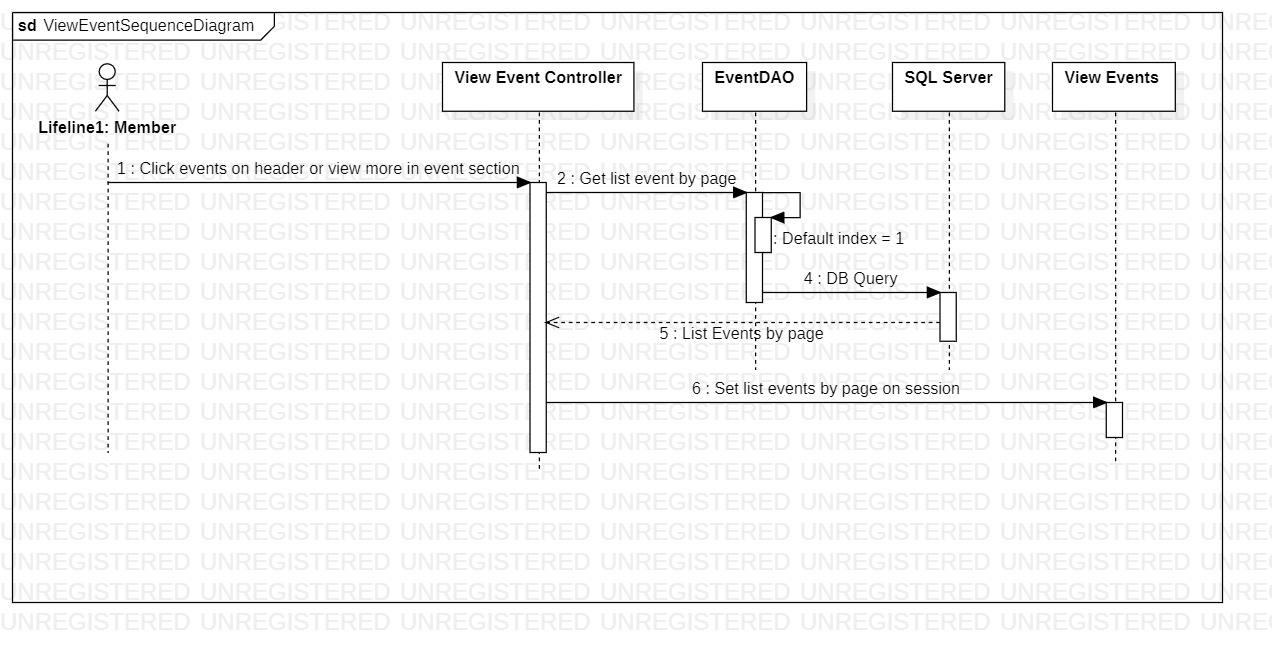
## 4. View Events

### a. Class Diagram

Graphical user interface

Description automatically generated with medium confidence

### b. Sequence Diagram(s)

Figure 4 – View events by page

### c. Database queries

userID = User’s id stored in HttpSession

- Get events by page

Search = %%

Index = selected index page from Figure 4

PageSize = 6

*WITH tblEventPage AS*

*(SELECT (ROW\_NUMBER() over (order by startDate) ) AS RowNum,*

*eventId, userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, s.statusName AS status*

*FROM tblEvents e, tblStatusEvent s*

*WHERE* title like search AND *e.statusId = s.statusId)*

*SELECT eventId, userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, status*

*FROM tblEventPage*

*WHERE RowNum BETWEEN* Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

- Update events’ status each time query event’s list

UPDATE tblEvents

SET statusId = (SELECT statusId FROM tblStatusEvent WHERE statusName = ?)

WHERE eventId = ?

(Note: parameters’ values is passed from “Get event by page”)

- Get count of events for pagination

*SELECT COUNT(eventId) as count*

*FROM tblEvents*

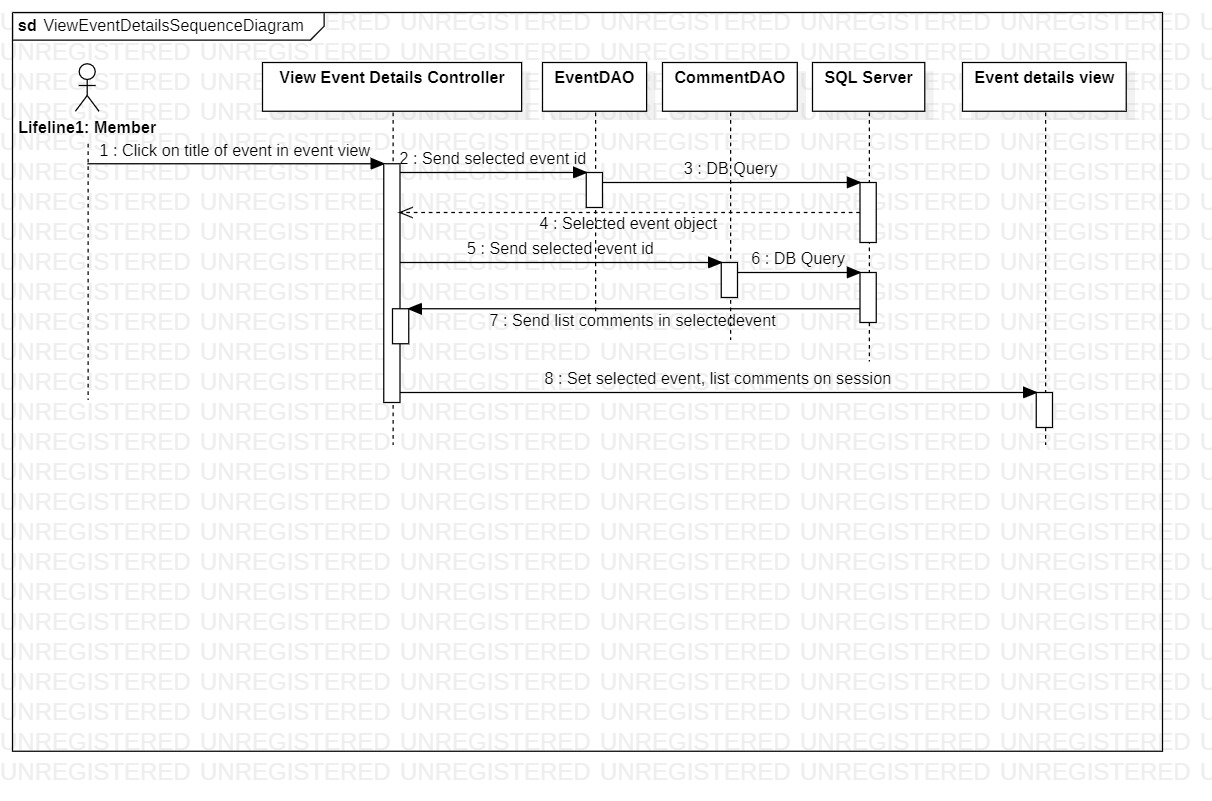
## 5. View Event’s Detail

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 5 – View selected event’s details

### c. Database queries

eventID = Event’s id passed from selected event in events page

userID = User’s id stored in HttpSession

- Get selected event’s details

SELECT userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, s.statusName

FROM tblEvents e, tblStatusEvent s

WHERE eventId = eventID AND s.statusId = e.statusId

- Get comment count in selected event

SELECT userId,commentDatetime, content FROM tblComments WHERE eventId = eventID

- Get comments to load to view

Index = 1 and increment for each “View more comments” click.

PageSize = 6

WITH tblCommentPage AS

(SELECT (ROW\_NUMBER() over (order by commentDatetime desc) ) AS RowNum,

content, commentDatetime, userId

FROM tblComments

WHERE eventId= eventID)

SELECT content, commentDatetime, userId

FROM tblCommentPage

WHERE RowNum BETWEEN Index \* PageSize -(PageSize -1) AND Index \* PageSize

- Get follow status

SELECT follow

FROM tblFollowedEvent

WHERE userId = userID AND eventId = eventID

- Get join status

SELECT joinStatus

FROM tblParticipants

WHERE userId = userID AND eventId = eventID

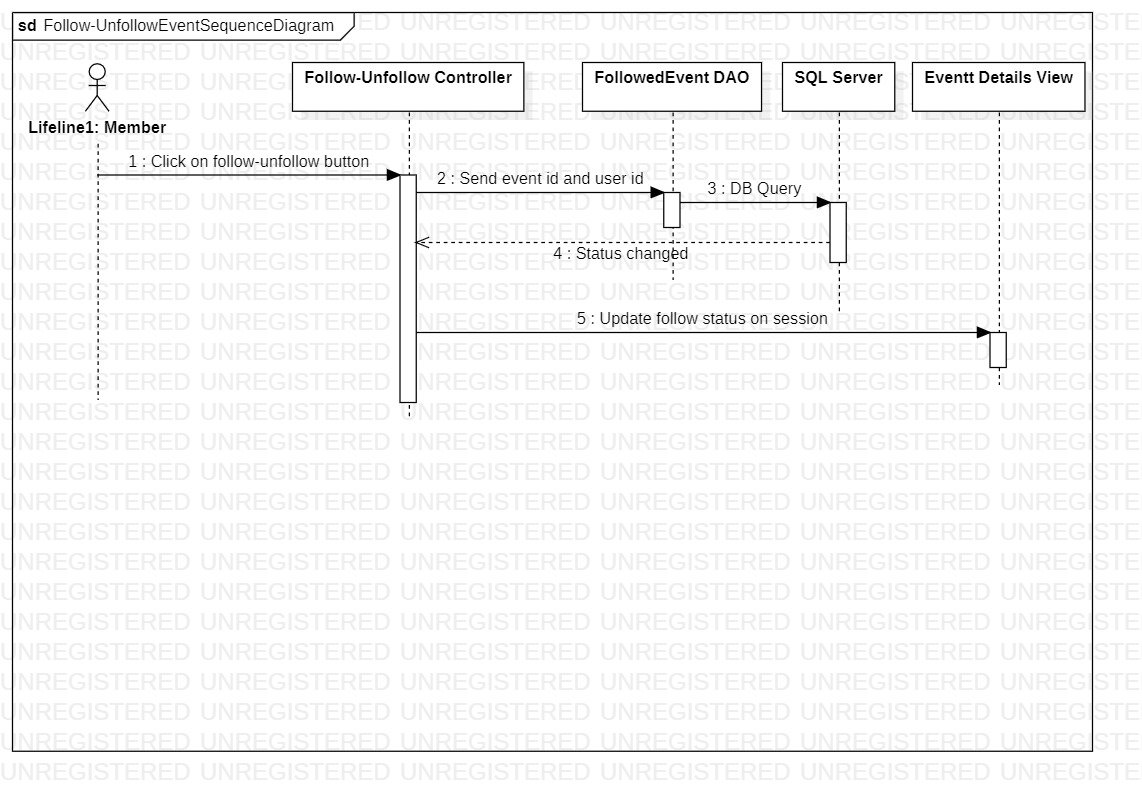
## 6. Follow/Unfollow Event

### a. Class Diagram

Diagram

Description automatically generated with low confidence

### b. Sequence Diagram(s)

Figure 6 – Follow or unfollow selected event

### c. Database queries

eventID = Event’s id passed from selected event in events page

userID = User’s id stored in HttpSession

- Follow for the first time

INSERT INTO tblFollowedEvent (userId, eventId, follow)

VALUES (userID, eventID, 1)

- Follow/Unfollow (Change follow status)

Follow = 0 if previous value was 1 or

1 if previous value was 0

UPDATE tblFollowedEvent

SET follow = Follow

WHERE userId= userID AND eventId = eventID

- Get follow status

SELECT follow

FROM tblFollowedEvent

WHERE userId = userID AND eventId = eventID

- Get join status

SELECT joinStatus

FROM tblParticipants

WHERE userId = userID AND eventId = eventID

## 7. Join Event/Unjoin Event

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 7 – Join or unjoin an event Diagram

Description automatically generated

### c. Database queries

eventID = Event’s id passed from selected event in events page

userID = User’s id stored in HttpSession

- Get number of participants of the event

SELECT COUNT(userId)

FROM tblParticipants

WHERE eventId = eventID AND userId != userID AND joinStatus = 1

- Get join status

SELECT joinStatus

FROM tblParticipants

WHERE userId = userID AND eventId = eventID

- Join in for the first time in selected event

INSERT INTO tblParticipants(eventId, userId, joinStatus)

VALUES (eventID , userID, 1)

- Join/Unjoin (Change join status)

Join = 0 if previous value was 1 or

1 if previous value was 0

UPDATE tblParticipants

SET joinStatus= Join

WHERE userId= userID AND eventId = eventID

- Get follow status

SELECT follow

FROM tblFollowedEvent

WHERE userId = userID AND eventId = eventID

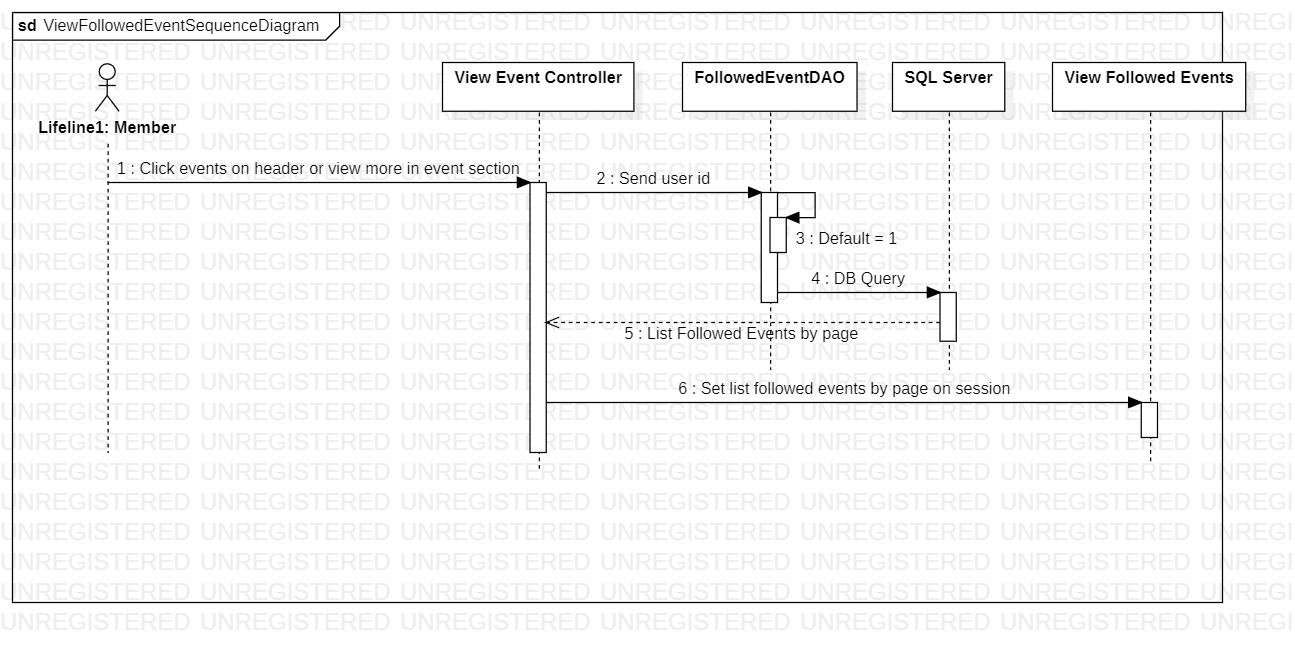
## 8. View Followed Events

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 8 – View followed events

### c. Database queries

userID = User’s id stored in HttpSession

- Get count of events created by user for pagination

*SELECT COUNT(userId) as count*

*FROM tblFollowedEvent*

*WHERE follow = 1 AND userId =* userID

- Get events by page

Search = %%

Index = selected index page from Figure 8

PageSize = 6

*WITH tblEventPage AS*

*(SELECT (ROW\_NUMBER() over (order by startDate) ) AS RowNum, e.eventId, e.userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, s.statusName AS status*

*FROM tblEvents e, tblStatusEvent s, tblFollowedEvent f*

*WHERE title like* Search *AND e.statusId = s.statusId AND f.userId = ? AND f.eventId = e.eventId AND f.follow = 1)*

*SELECT eventId, userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, status*

*FROM tblEventPage*

*WHERE RowNum BETWEEN* Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

- Update events’ status each time query event’s list

UPDATE tblEvents

SET statusId = (SELECT statusId FROM tblStatusEvent WHERE statusName = ?)

WHERE eventId = ?

(Note: parameters’ values is passed from “Get event by page”)

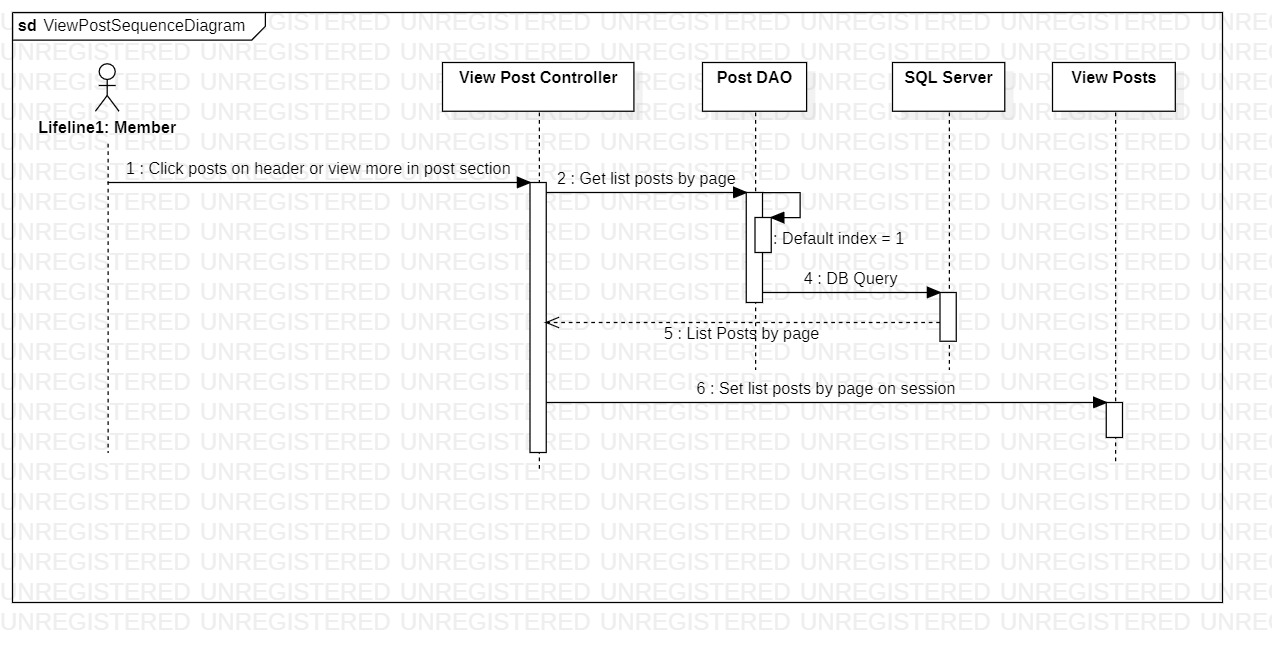
## 9. View Posts

### a. Class Diagram

A picture containing diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 9 – View posts by page

### c. Database queries

userID = User’s id stored in HttpSession

- Get events by page

Search = %%

Index = selected index page from Figure 9

PageSize = 6

*WITH tblPostPage AS*

*(SELECT (ROW\_NUMBER() over (order by createDate) ) AS RowNum, postId, userId, title, content, video, createDate,s.statusName AS status*

*FROM tblPosts e, tblStatusPost s*

*WHERE title like Search AND e.statusId = s.statusId*

*SELECT postId, userId, title, content, video, createDate, status*

*FROM tblPostPage*

*WHERE RowNum BETWEEN* Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

- Get count of posts for pagination

*SELECT COUNT(postId) as count*

*FROM tblPosts*

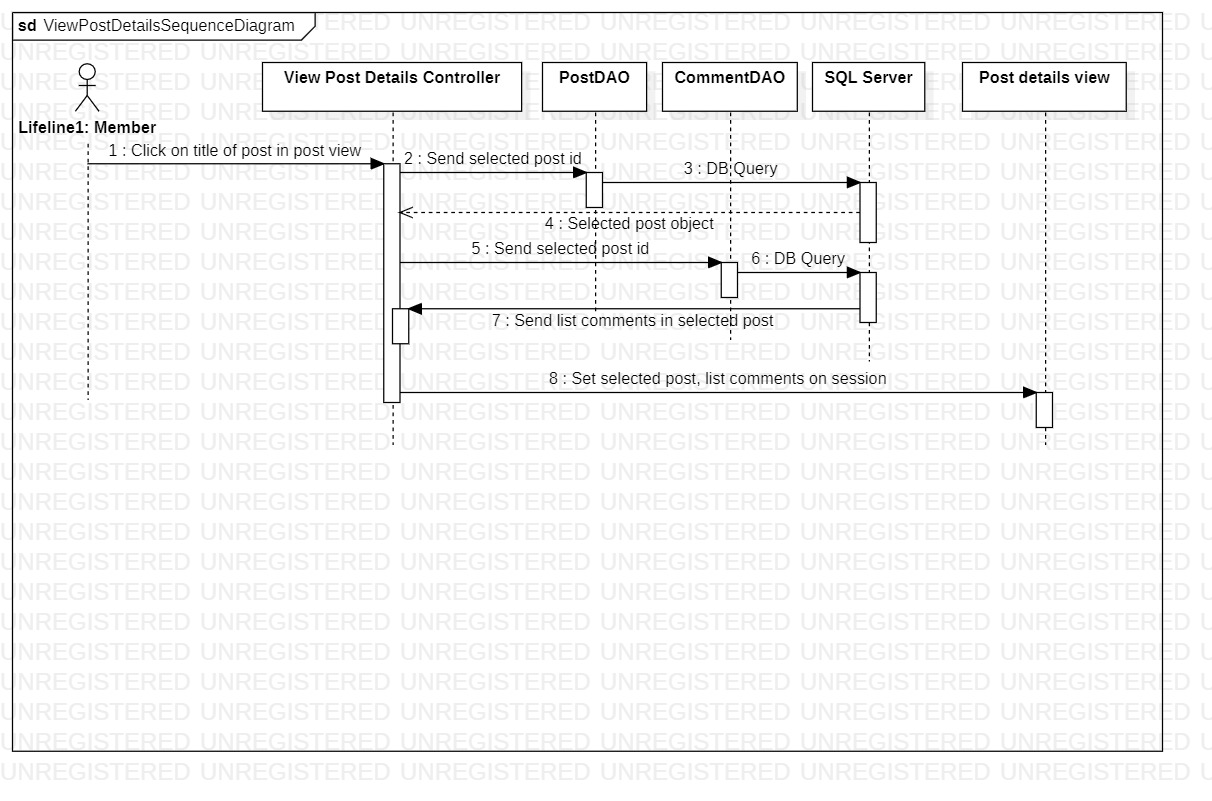
## 10. View Post’s Detail

### a. Class Diagram

Graphical user interface

Description automatically generated

### b. Sequence Diagram(s)

Figure 10 – View selected post’s details

### c. Database queries

postID = Post’s id passed from selected post in posts page

userID = User’s id stored in HttpSession

- Get selected post’s details

SELECT userId, title, content, video, createDate, s.statusName

FROM tblPosts e, tblStatusPost s"

WHERE e.postId= postID AND e.statusId = s.statusId

- Get comment count in selected post

SELECT userId, commentDatetime, content FROM tblComments WHERE postId = postID

- Get comments to load to view

Index = 1 and increment for each “View more comments” click.

PageSize = 6

WITH tblCommentPage AS

(SELECT (ROW\_NUMBER() over (order by commentDatetime desc) ) AS RowNum,

content, commentDatetime, userId

FROM tblComments

WHERE eventId= eventID)

SELECT content, commentDatetime, userId

FROM tblCommentPage

WHERE RowNum BETWEEN Index \* PageSize -(PageSize -1) AND Index \* PageSize

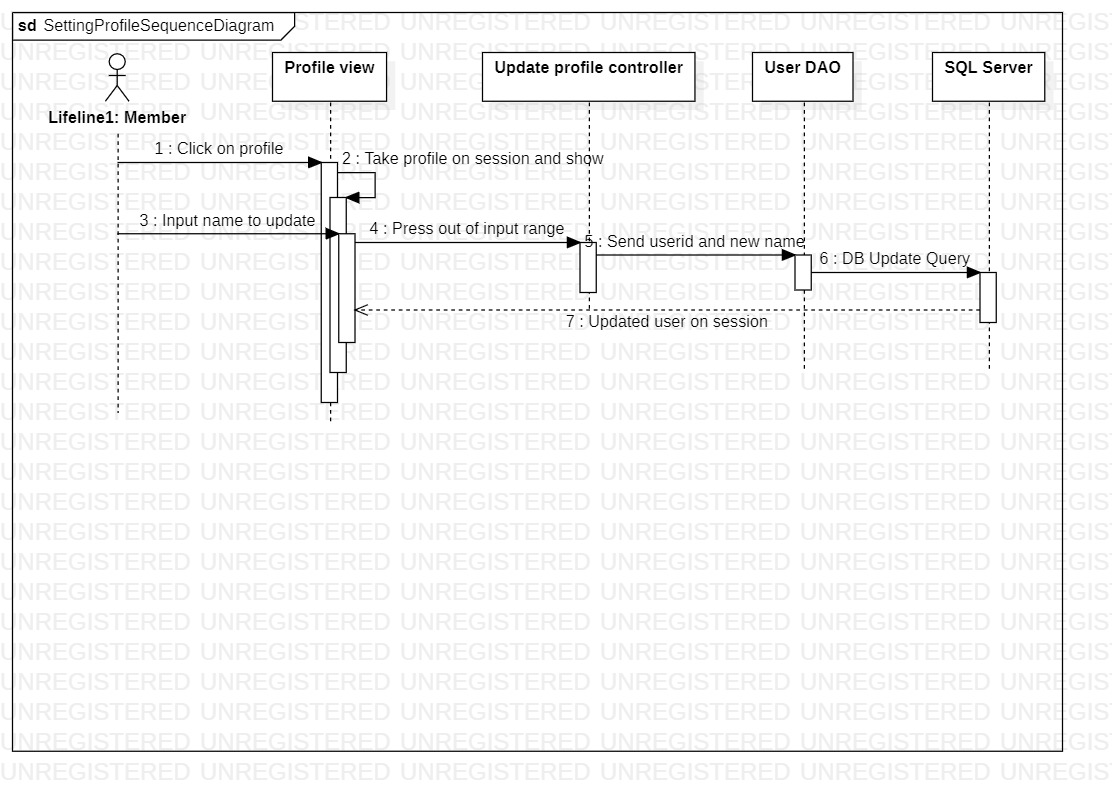
## 11. Setting Profile

### a. Class Diagram

Diagram

Description automatically generated with medium confidence

### b. Sequence Diagram(s)

Figure 11 – Setting your profile

### c. Database queries

userID = User’s id stored in HttpSession

username = User’s new name from Figure 11

- Update profile (username only)

UPDATE tblUsers

SET username = username

WHERE userId = userID

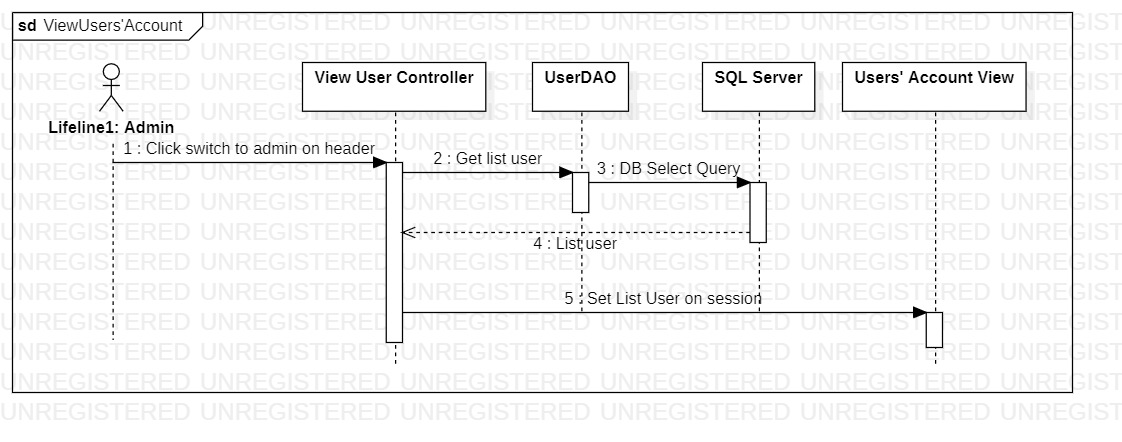
## 12. View User’s Accounts

### a. Class Diagram

Graphical user interface, text, application

Description automatically generated

### b. Sequence Diagram(s)

Figure 12 – View user accounts list (Administrator)

### c. Database queries

- Get list of user accounts

SELECT userId, email, username, s.statusName, r.roleName

FROM tblUsers u, tblStatusUser s, tblRoles r

WHERE u.statusId = s.statusId AND u.roleId = r.roleId

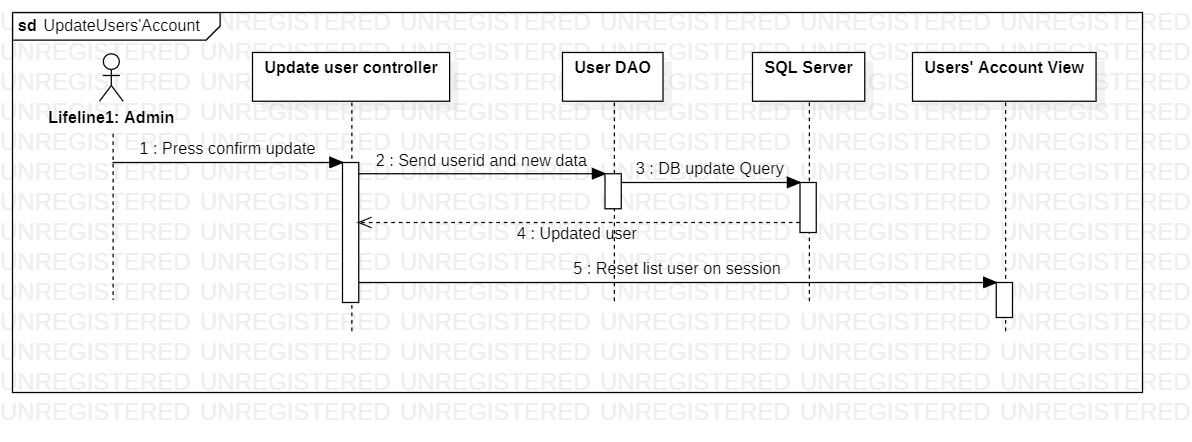
## 13. Update User’s Accounts

### a. Class Diagram

Graphical user interface

Description automatically generated

### b. Sequence Diagram(s)

Figure 13 – Update edited user account (Administrator)

### c. Database queries

userID = User’s id retrieved from Figure 13

roleName = User’s role retrieved from Figure 13

- Update edited account to the database

UPDATE tblUsers

SET roleId = (SELECT roleId FROM tblRoles WHERE roleName = roleName)

WHERE userId = userID

or Change account’s status (ban/unban)

UPDATE tblUsers

SET statusId = (SELECT statusId FROM tblStatusUser WHERE statusName = (1))

WHERE userId = userID

(1) is “Deactive” if statusName is “Active” and vise versa

- Get list of user accounts to the view

SELECT userId, email, username, s.statusName, r.roleName

FROM tblUsers u, tblStatusUser s, tblRoles r

WHERE u.statusId = s.statusId AND u.roleId = r.roleId

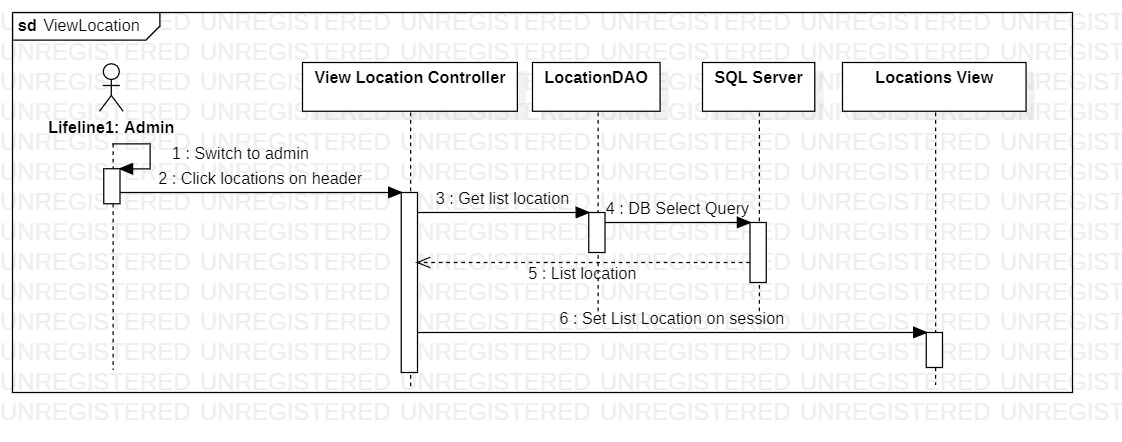
## 14. View Locations

### a. Class Diagram

Graphical user interface

Description automatically generated

### b. Sequence Diagram(s)

Figure 14 – View locations list (Administrator)

### c. Database queries

- Get count of locations for pagination

SELECT COUNT(locationId) as amount

FROM tblLocations

- Get locations by page

Search = %%

Index = selected index page from Figure 14

PageSize = 10

*WITH tblLocationPage AS*

*(SELECT (ROW\_NUMBER() over (order by locationName) ) AS RowNum,*

*locationId, seat, locationName*

*FROM tblLocations*

*WHERE locationName like* Search*)*

*SELECT locationId, seat, locationName \n"*

*FROM tblLocationPage*

*WHERE RowNum BETWEEN* Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

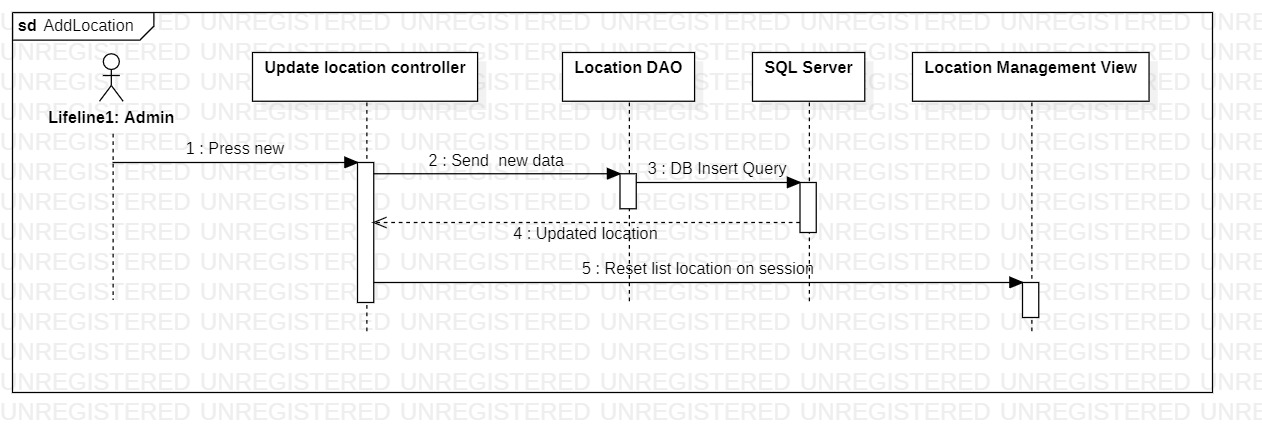
## 15. Add New Location

### a. Class Diagram

Graphical user interface, text, application

Description automatically generated

### b. Sequence Diagram(s)

Figure 15 – Add new location (Administrator)

### c. Database queries

locationName = inputted locationName from Figure 16

seat = inputted seat from Figure 16

- Add inserted location into database

INSERT INTO tblLocations (locationName, seat)

VALUES (locationName, seat)

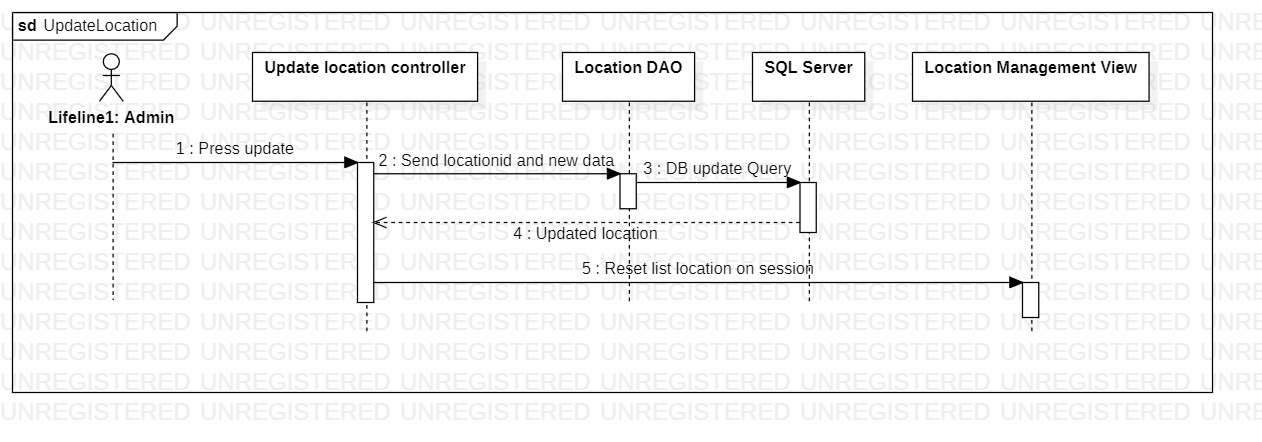
## 16. Update Location

### a. Class Diagram

Graphical user interface

Description automatically generated

### b. Sequence Diagram(s)

Figure 16 – Update edited location (Administrator)

### c. Database queries

locationId = changed location’s id from Figure 16

locationName = inputted locationName from Figure 16

seat = inputted seat from Figure 16

- Update location’s data

UPDATE tblLocations

SET locationName = locationName, seat = seat

WHERE locationId = locationId

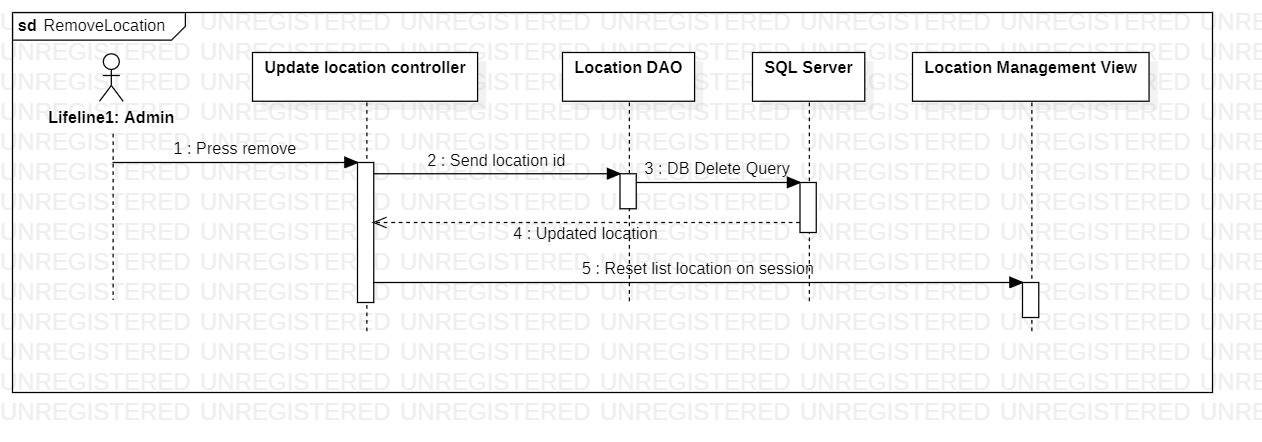
## 17. Remove Location

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 17 – Remove selected location (Administrator)

### c. Database queries

locationId = selected location’s id from Figure 18

- Remove selected location

DELETE FROM tblLocations

WHERE locationId = locationId

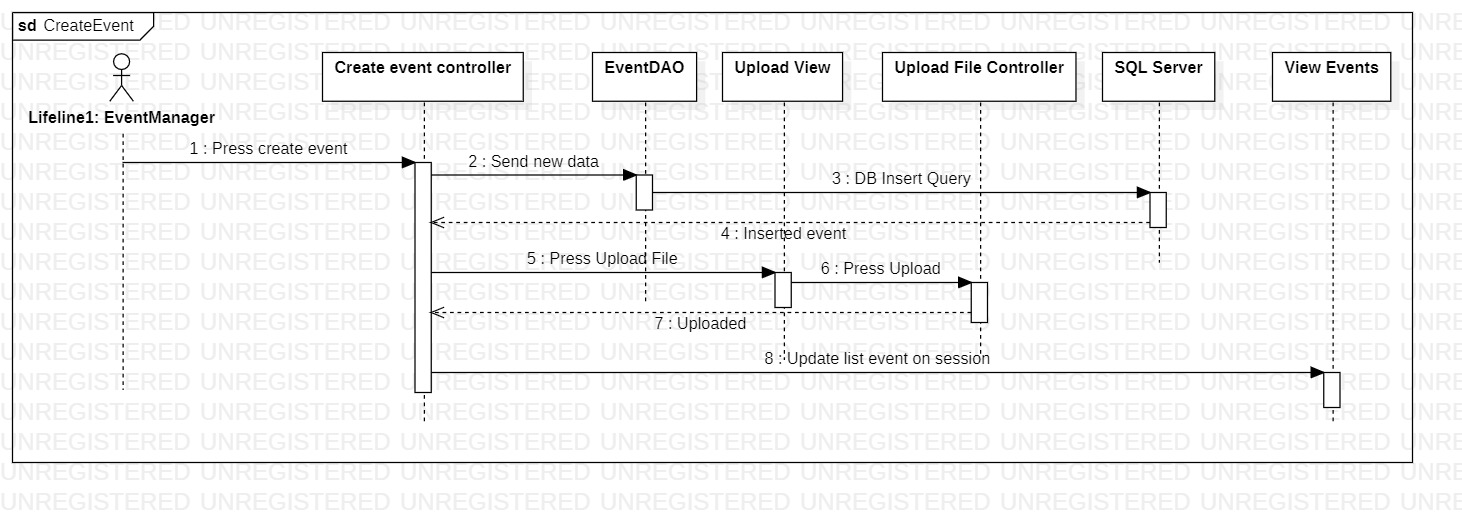
## 18. Create Event

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 18 – Create new event (Event Manager)

### c. Database queries

- Insert new event into database

INSERT INTO tblEvents

(userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, statusId)

VALUES (?,?,?,?,?,?,?,?, (SELECT statusId FROM tblStatusEvent WHERE statusName = ?))

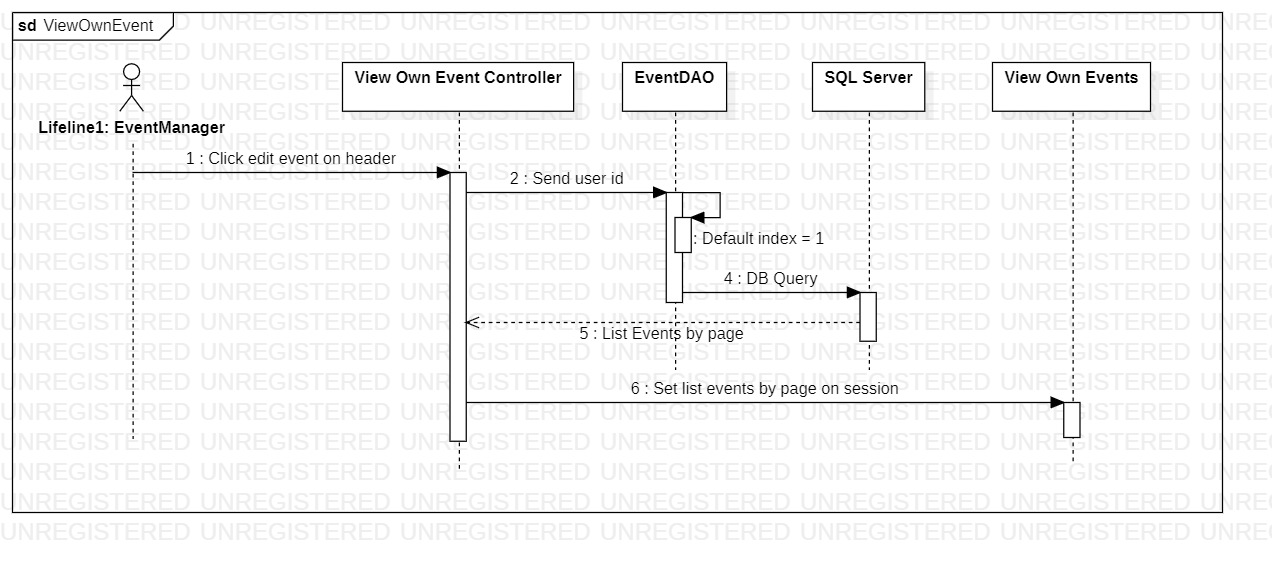
## 19. View Created Event

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 19 – View your created events by page (Event Manager)

### c. Database queries

userID = User’s id stored in HttpSession

- Get owned events by page

Search = %%

Index = selected index page from Figure 4

PageSize = 6

*WITH tblEventPage AS*

*(SELECT (ROW\_NUMBER() over (order by startDate) ) AS RowNum,*

*eventId, userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, s.statusName AS status*

*FROM tblEvents e, tblStatusEvent s*

*WHERE userId =* userID *AND title like* Search *AND e.statusId = s.statusId)*

*SELECT eventId, userId, title, description, locationId, createDatetime, startDate, startSlot, endSlot, status*

*FROM tblEventPage WHERE RowNum BETWEEN* Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

- Get count of owned events for pagination

*SELECT COUNT(eventId) as count*

*FROM tblEvents*

*WHERE userId =* userID

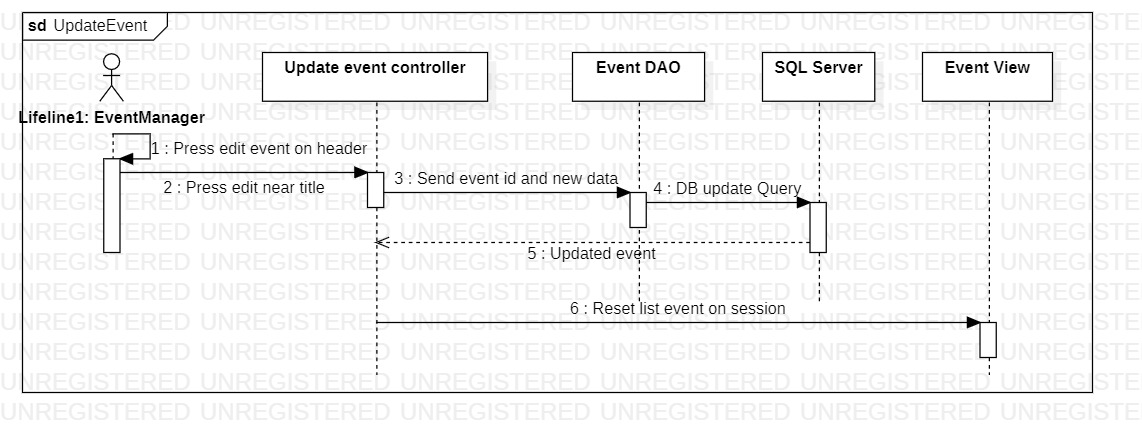
## 20. Edit Event

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 20 – Update edited event (Event Manager)

### c. Database queries

eventId = changed event’s id

- Update selected event’s changes to database

UPDATE tblEvents

SET title= ?, description= ?, locationId= ?, startDate= ?, startSlot= ?, endSlot= ?,

statusId = (SELECT statusId FROM tblStatusEvent WHERE statusName = ?)

WHERE eventId= eventId

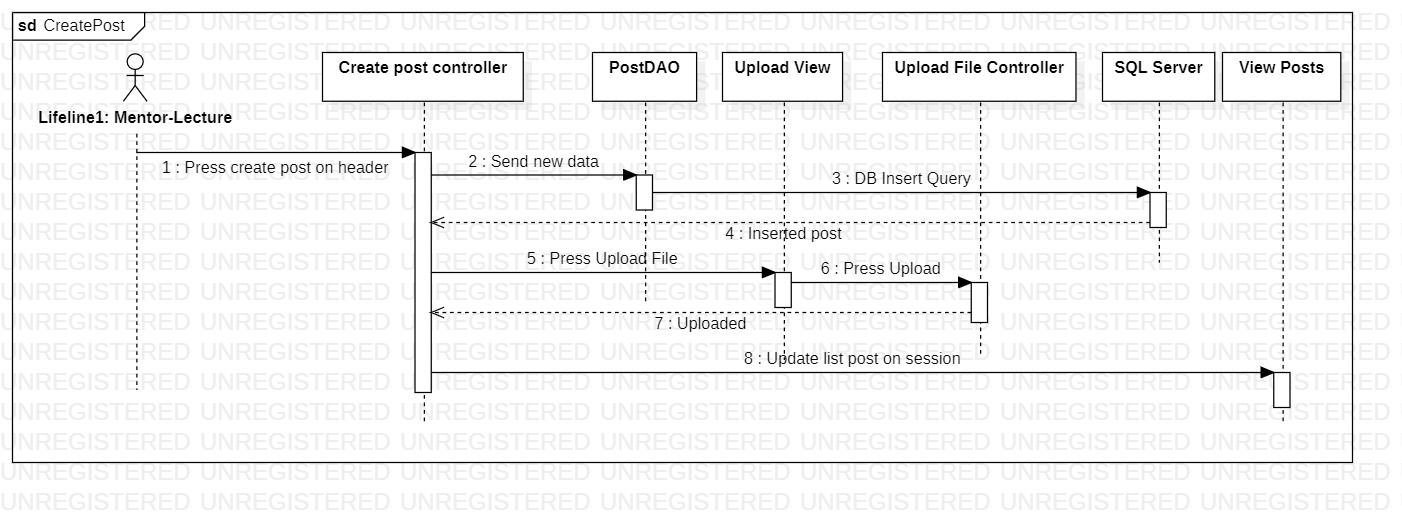
## 21. Create Post

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 21 – Create new post (Mentor/Lecturer)

### c. Database queries

- Insert new post into database

INSERT INTO tblPosts (userId, title, content, video, createDate, statusId)

VALUES (?, ?, ?, ?, ?, (SELECT statusId FROM tblStatusPost WHERE statusName = ?))

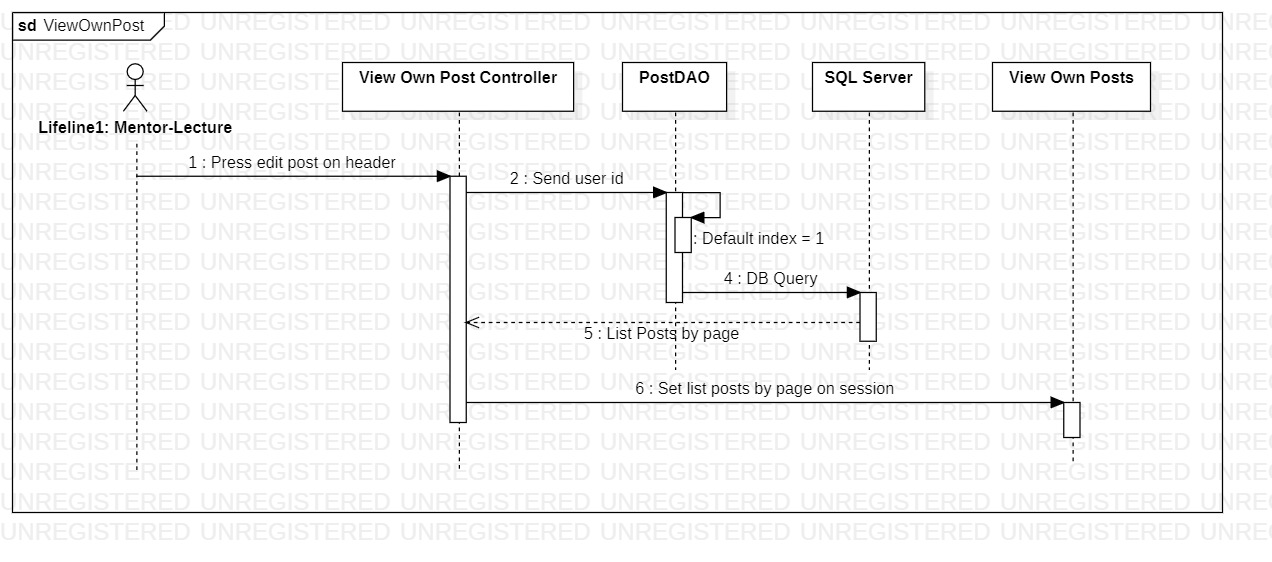
## 22. View Created Post

### a. Class Diagram

Diagram

Description automatically generated

### b. Sequence Diagram(s)

Figure 22 – View your created posts by page (Mentor/Lecturer)

### c. Database queries

userID = User’s id stored in HttpSession

- Get events by page

Search = %%

Index = selected index page from Figure 22

PageSize = 6

*WITH tblPostPage AS*

*(SELECT (ROW\_NUMBER() over (order by createDate) ) AS RowNum,*

*userId, title, content, video, createDate,s.statusName AS status*

*FROM tblPosts e, tblStatusPost s*

*WHERE title like Search AND e.statusId = s.statusId AND e.userId=* userID*)*

*SELECT postId, userId, title, content, video, createDate, status \n"*

*FROM tblPostPage WHERE RowNum BETWEEN* Index *\** PageSize *- (*PageSize *-1) AND* Index *\** PageSize

- Get count of owned posts for pagination

*SELECT COUNT(postId) as count*

*FROM tblPosts*

*WHERE userId=* userID

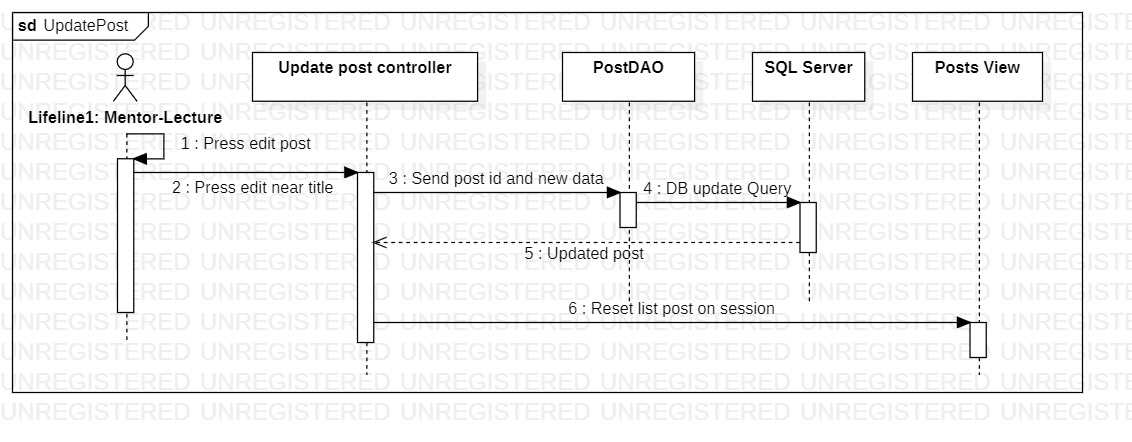
## 23. Edit Post

### a. Class Diagram

A screenshot of a computer

Description automatically generated with medium confidence

### b. Sequence Diagram(s)

Figure 23 – Update edited post (Mentor/Lecturer)

### c. Database queries

userID = User’s id stored in HttpSession

postID = changed post’s id

- Update selected post’s changes to database

*UPDATE tblPosts*

*SET title =?, content =?, video =?, createDate =?,*

*statusId = (SELECT statusId FROM tblStatusPost WHERE statusName=?)*

*WHERE postId =* postID *AND userId=* userID

## 24. Upload File

### a. Class Diagram

Chart

Description automatically generated

### b. Sequence Diagram(s)

*No sequence diagram for this feature.*

### c. Database queries

*No query for this feature*