Contents

[Use Case Diagram 2](#_Toc159906742)

[Database Class Diagram 3](#_Toc159906743)

[Guest: User Registration Use Case 4](#_Toc159906744)

[Guest: User Login Use Case 5](#_Toc159906745)

[Guest: Browse Events Use Case 6](#_Toc159906746)

[Community member: Join Event Use Case 7](#_Toc159906747)

[Community member: Leave Event Use Case 8](#_Toc159906748)

[Community member: View Event Dashboard Use Case 9](#_Toc159906749)

[Community member: Feedback Event Use Case 10](#_Toc159906750)

[Event owner: Create Event Use Case 11](#_Toc159906751)

[Event owner: Delete Event Use Case 12](#_Toc159906752)

[Event owner: Update Event Use Case 13](#_Toc159906753)

[Event owner: Add Participant Use Case 14](#_Toc159906754)

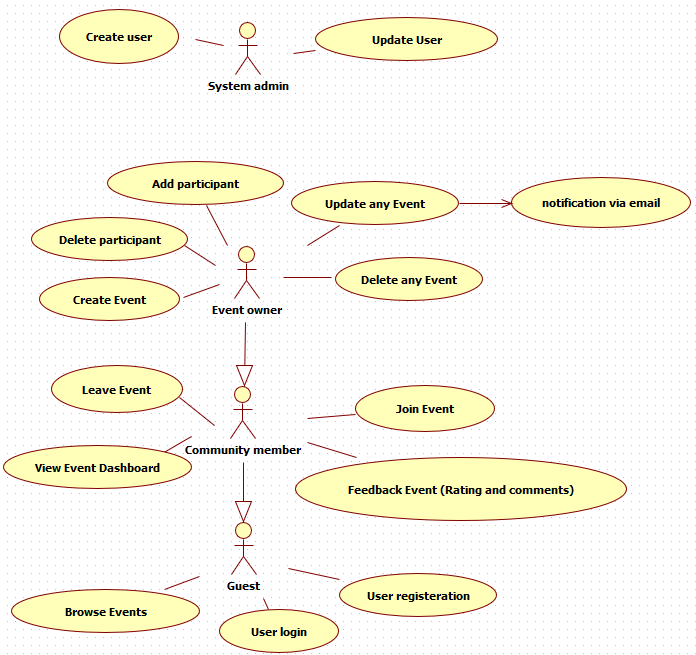
[Event owner: Delete Participant Use Case 15](#_Toc159906755)

[System admin: Create User Use Case 16](#_Toc159906756)

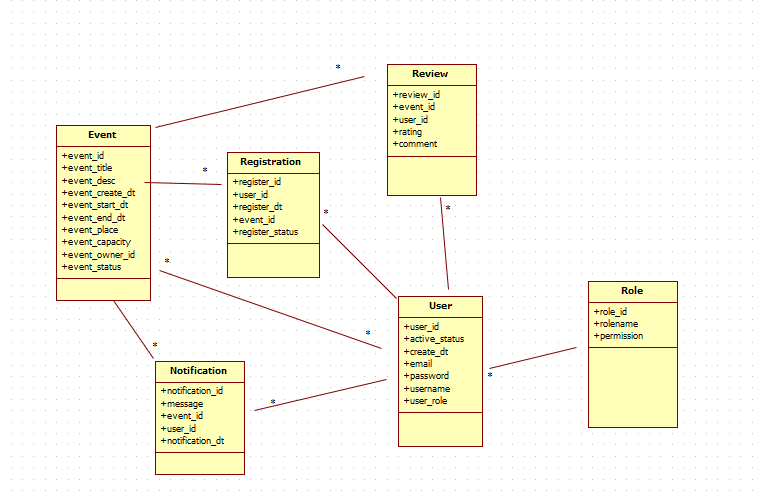
[System admin: Update User Use Case 17](#_Toc159906757)

# Use Case Diagram

Event owner inherits from Community member, which inherits from Guest. The child classes have all the attributes and methods of their parent classes. Thus, the inherited use cases are not listed again in the following part.



# Database Class Diagram



# Guest: User Registration Use Case

**Brief Description**

In this system all the user must be registered through the user registration function. System administrators has the privilege of admin page and able to update the user role through the admin panel. User registration is not required for system administrators.

**Flow of Events**

**Basic/Standard Flow of events**

User registers an account

1. The use case starts when the user clicks the “Sign Up” tab on the navigation bar/Login Page.
2. The system displays the signup page that allow users to fill in their username/email address, first and last name and their password.
3. The user keys in the details and clicks “Sign Up” button.
4. The system displays the user’s profile page.

**Alternative (Exception) Flow of events**

User tries to register an account with one or more invalid details which including an existing username/email address

1. The alternative flow starts after step 3 of the main flow
2. The system finds one or more invalid details
3. The system redisplays the signup page with invalid sign up message
4. The user can re-enter the correct details

**Special Conditions**

**Pre-Condition**

**NA**

**Post Condition**

The user’s details are stored in the database – User Table.

# Guest: User Login Use Case

**Brief Description**

Registered users may login to their account and perform further actions such as join/quit event (public users) or create/manage event (event owners). Not applicable to event owner.

**Flow of Events**

**Basic/Standard Flow of events**

Registered users sign in to the account

1. The use case starts when the user clicks the “Login” tab on the navigation bar.
2. The system displays the login page that allow users to fill in their username/email address, first and last name and their password.
3. The user keys in the details and clicks “Login” button.
4. The system displays the user’s profile page.

**Alternative (Exception) Flow of events**

User tries to login an account with unregistered email address/username, or the login details does not match with any account in database:

1. The alternative flow starts after step 3 of the main flow
2. The system finds no matching account in database.
3. The system redisplays the login page with ‘account not found’ error message
4. The user can re-enter the correct details

**Special Conditions**

**Pre-Condition**

User has their account details registered in the database already.

**NA**

**Post Condition**

# Guest: Browse Events Use Case

**Brief Description**

The user can view all events currently active in the system, apply filters to search for events they are interested in, and view the event details by clicking into the event.

**Flow of Events**

**Basic/Standard Flow of events**

Browse events

1. User clicks on ‘Events’ button on the navigation bar or from the dashboard hyperlink.
2. The system displays all events currently active in the system.
3. The user applies filters by clicking on the ‘filter’ sub button.
4. The system displays available filters that user can choose from (e.g. location, time, availability)
5. The user adds in filter details from the page
6. The system displayed filtered results.
7. The user clicks on a certain event title.
8. The system displays the event details page. The event detail page will display different content by the user role.

**Alternative (Exception) Flow of events**

If there is no active events that matches the user’s filter:

1. The alternative flow starts after step 4 of the main flow
2. The system shows ‘no event found, try a different filter’ error message

If the user is a community member who have not joined in this event or the user has not logged in.

1. The alternative flow starts at step 8 of the main flow
2. The system shows button ‘join the event’

If the user is a community member who have joined in this event.

1. The alternative flow starts at step 8 of the main flow
2. The system shows button ‘quit the event’

If the user is the admin of this event.

1. The alternative flow starts at step 8 of the main flow
2. The system shows button ‘delete the event’ , ‘update the event’, ‘view participants’

**Special Conditions**

# Community member: Join Event Use Case

**Brief Description**

This use case is for the community member to join in an event

**Flow of Events**

**Basic/Standard Flow of events**

1. The flow starts from the event detail page which user can access in the Browse Event Use Case or Dashboard Use case
2. The user clicks on the ‘join event’ button
3. The system saves user’s status to the database.
4. The system shows ‘You have joined the event!’

**Alternative (Exception) Flow of events**

**Special Conditions**

**Pre-Condition**

Join Event Button will only be available when the user is

1. The user is already logged in,
2. The user is not an admin of this event,
3. The user is not a participant of this event,
4. The event is active and still have participant slots (has not reached its maximum capacity)

**Post Condition**

After User has joined the event, they will see ‘leave event’ or ‘give feedback’ buttons when they click into this event detail page.

# Community member: Leave Event Use Case

**Brief Description**

This use case is for the community member to leave an event they have already joined.

**Flow of Events**

**Basic/Standard Flow of events**

1. The flow starts from the event detail page which user can access in the Browse Event Use Case or Dashboard Use case
2. The user click on the ‘leave event’ button
3. The system updates user’s status to the database.
4. The system shows ‘You have left the event. Hope to see you next time!’

**Alternative (Exception) Flow of events**

**Special Conditions**

**Pre-Condition**

Leave Event Button will only be available when the user is

1. The user is already logged in,
2. The user is not an admin of this event,
3. The user is a participant of this event,
4. The event is active

**Post Condition**

# Community member: View Event Dashboard Use Case

**Brief Description**

All logged in members (including community member and event owner) will be able to view the event dashboard in which they can put filter and search for the event they are looking for.

**Flow of Events**

**Basic/Standard Flow of events**

View event dashboard

1. User clicks on ‘Dashboard’ button on the navigation bar.
2. The system displays all events the user enrolled and/or managed by the user, with some brief event details.
3. The user clicks on a certain event title.
4. The system displays the event details page. The event detail page will display different content by the user role.

**Alternative (Exception) Flow of events**

If there is no active events joined by the user or managed by the user:

1. The alternative flow starts after step 1 of the main flow
2. The system shows ‘you have no active events’ with hyperlinks to
   1. the ‘view all events’ page
   2. the ‘become an event owner’ page

**Special Conditions**

**Pre-Condition**

User has logged in to their account

**NA**

**Post Condition**

# Community member: Feedback Event Use Case

**Brief Description**

This use case is for the community member to give feedback to participated events.

**Flow of Events**

**Basic/Standard Flow of events**

1. The flow starts from the event detail page which user can access in the Browse Event Use Case or Dashboard Use case
2. The user click on the ‘Feedback’ button
3. The system shows a page in which user can give rate and comment about the event.
4. The user fills in the feedback form.
5. The system updates user’s feedback to the database.
6. The system shows ‘Thank you for your feedback!’

**Alternative (Exception) Flow of events**

**Special Conditions**

**Pre-Condition**

Leave Event Button will only be available when the user is

1. The user is already logged in,
2. The user is not an admin of this event,
3. The user is a participant of this event,
4. The user has not left a feedback to this event before,
5. The event is inactive (past date, or cancelled by admin).

**Post Condition**

# Event owner: Create Event Use Case

**Brief Description**

This use case is for the event owner to create an event for public to join.

**Flow of Events**

**Basic/Standard Flow of events**

1. The user click on the create event button in the Dashboard Page.
2. The system shows the create event page in which the admin user can provide various details. (e.g. title, location, time, description, capacity)
3. The user fills in the create event form.
4. The user clicks on ‘Create Event Button’
5. The System saves the event to db.
6. The system shows: ‘Event created.’, and jumps to this events’ detail page

**Alternative (Exception) Flow of events**

If the user enters invalid Event details in step 2.

1. This alternative flow happens at step 5. The system validates the information user has keyed in.
2. If user has keyed in invalid information
   1. Empty title
   2. Empty location
   3. Empty dates
   4. Past start date
   5. Empty Capacity
3. The system stops processing and returns to step 2, with extra error messages on the invalid fields.

**Special Conditions**

**Pre-Condition**

Create Event Button will only be available when the user is

1. The user is already logged in,
2. The user has the event owner role.

**Post Condition**

After Event is created, event owner can send notification as announcement.

After Event is created, event owner can delete this event.

# Event owner: Delete Event Use Case

**Brief Description**

This use case is for the event owner to delete an event.

**Flow of Events**

**Basic/Standard Flow of events**

1. The user clicks on the delete event button in the event details page.
2. The system shows ‘Are you sure you want to delete this event? All participants will be notified about the deletion of this event.’
3. The user clicks ‘Yes’.
4. The system updates the database.
5. The System triggers email notification webhook/api to all registered participants.
6. The system shows: ‘Event deleted.’

**Alternative (Exception) Flow of events**

**Special Conditions**

**Pre-Condition**

Delete Event Button will only be available when the user is

1. The user is already logged in,
2. The user has the event owner role.
3. The event’s start date is in future.

**Post Condition**

After Event is deleted, give feedback option is made available to all participants.

# Event owner: Update Event Use Case

**Brief Description**

This use case is for the event owner to update an event.

**Flow of Events**

**Basic/Standard Flow of events**

1. The user clicks on the update event button in the event details page.
2. The system shows an update event form for the user to fill in new details of the event.
3. The user fills in the form and clicks ‘Update’.
4. The system updates the database.
5. The System triggers email notification webhook/api to all registered participants with update details.
6. The system shows: ‘Event Updated.’

**Alternative (Exception) Flow of events**

**Special Conditions**

**Pre-Condition**

Delete Event Button will only be available when the user is

1. The user is already logged in,
2. The user has the event owner role.
3. The event’s start date is in future.

**Post Condition**

# Event owner: Add Participant Use Case

**Brief Description**

This use case is for the event owner to add (invite) a participant to his event.

**Flow of Events**

**Basic/Standard Flow of events**

1. The user clicks on the ‘participants’ button in the event details page.
2. The system shows a list of the registered participants of the event with a button “manage participants”
3. The user clicks on the ‘manage participants’ button.
4. The system shows interactable ‘delete’ button next to all enrolled participants’ names. The system also shows an ‘add user’ button above all users.
5. The user clicks ‘add user’
6. The system shows an add user page in which event owner can type in user id, username or User Email to look for a user.
7. The user key in the search information and click on ‘look for user’
8. The system shows the search result.
9. The user clicks on the ‘add to this event’ button next to the username.
10. The system updates the database and make the user a participant of the event.
11. The System triggers email notification webhook/api to this user.
12. The system shows: ‘You have invited <username> to your event <event Id>.’

**Alternative (Exception) Flow of events**

If user is not found:

1. This alternative flow happens at step 7.
2. If no user matches the details the event owner has put, the system goes back to step 6, with an extra error message ‘The user does not exist’.

**Special Conditions**

**Pre-Condition**

‘Participant’ Button will only be interactable when

1. The user is already logged in,
2. The user has the event owner role of this event.
3. The event’s start date is in future.

**Post Condition**

The user that is added to this event will be able to leave/give feedback to the event.

# Event owner: Delete Participant Use Case

**Brief Description**

This use case is for the event owner to delete a participant from his event.

**Flow of Events**

**Basic/Standard Flow of events**

1. The user clicks on the ‘participants’ button in the event details page.
2. The system shows a list of the registered participants of the event with a button “manage participants”
3. The user clicks on the ‘manage participants’ button.
4. The system shows interactable ‘delete’ button next to all enrolled participants’ names. The system also shows an ‘add user’ button above all users.
5. The user clicks ‘delete user’ next to one of the participant’s name.
6. The system shows ‘Are you sure you want to delete user <user name> from your event?’
7. The user clicks ‘Yes’.
8. The system updates the database and remove the user from the event.
9. The System triggers email notification webhook/api to this user.
10. The system shows: ‘You have removed <username> from your event <event Id>.’

**Alternative (Exception) Flow of events**

**Special Conditions**

**Pre-Condition**

‘Participant’ Button will only be interactable when

1. The user is already logged in,
2. The user has the event owner role of this event.
3. The event’s start date is in future.

**Post Condition**

# System admin: Create User Use Case

**Brief Description**

This use case is for the system admin to create a user with a certain role.

**Flow of Events**

**Basic/Standard Flow of events**

1. The user clicks on the ‘create user’ button in the admin interface.
2. The system shows a form for User Creation
   1. User Email
   2. User Name
   3. Password
   4. User Role: Community member/ Event owner.
3. The user fills in the form and clicks on ‘create user’.
4. The system updates the database and add the user.
5. The System triggers email notification webhook/api to this user.
6. The system shows: ‘You have created a new user <username>’

**Alternative (Exception) Flow of events**

Admin tries to create an user with invalid details which including an existing username/email address

1. The alternative flow starts after step 3 of the main flow
2. The system finds one or more invalid details
3. The system redisplays the create user page(step 2) with invalid sign up message
4. The user can re-enter the correct details

**Special Conditions**

**Pre-Condition**

**Post Condition**

The user can login with the details as sent to their email.

# System admin: Update User Use Case

**Brief Description**

This use case is for the system admin to update a user’s role from community member to event owner or vice versa.

**Flow of Events**

**Basic/Standard Flow of events**

1. The user clicks on the ‘update user role’ button in the admin interface.
2. The system shows a form for User Role Update
   1. User Id/email/username
   2. New Role: Community member/ Event owner.
   3. New Active Status: Active/ Closed.
3. The user fills in the form and clicks on ‘Update user’.
4. The system updates the database accordingly.
5. The System triggers email notification webhook/api to this user.
6. The system shows: ‘You have update the role of user <username>’

**Alternative (Exception) Flow of events**

Admin tries to update a user from event owner to community member, when the event owner still have active events under his management:

1. The alternative flow starts after step 3 of the main flow
2. The system finds active events under the user’s name.
3. The system redisplays the update user page (step 2) with invalid update message: ‘You cannot make <username> a community member as he/she still owns at least one active event(s)!’
4. The user can re-enter the user name details.

Admin tries to update a non-exist user role:

1. The alternative flow starts after step 3 of the main flow
2. The system finds no user from the detail provided by the system admin.
3. The system redisplays the update user page (step 2) with invalid update message: ‘No user exists!’
4. The user can re-enter the user name details.

**Special Conditions**

**Pre-Condition**

**Post Condition**

If a community member is updated to event owner:

* He/she can create event from their dashboard and have access to all event management functions, and will still have access to the event they joined

If an event owner user is updated to community member:

* He/she will still have access to the event they joined, but lose access to all create/update event functions.