25-March-2025



**SUPERVISED BY:**

*Dr. ONAIZA MAQBOOL*

Group: **15**

Case Study: **5**

**PRESENTED BY:**

1. *M. Ahmad Hassan*
2. *Ahmad Masood*
3. *M. Waleed (Leader)*
4. *Talha Arif Wains*

***Event Hub***

**Signature Page :**

**Member 1:** M. Ahmad Hassan

**Signature: -------------------------**

**Member 2:** Ahmad Masood

**Signature: --------------------------**

**Member 3:** Muhammad Waleed

**Signature: ------------------------------**

**Member 4:** M. Talha Arif Wains

**Signature: ------------------------------**

CHANGE HISTORY:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **VERSION NO.** | **REVISION DATE** | **MODIFIED BY** | **CHANGES** | **APPROVED BY** |
| 1.0 | 03/03/2025 | Ahmad Masood | Original | DR. ONAIZA MAQBOOL |
| 1.1 | 24/03/2025 | Ahmad Hassan | Use case diagram  Use case description | DR. ONAIZA MAQBOOL |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

REMARKS:

Signature

DR. ONAIZA MAQBOOL

Professor, QAU Islamabad

**TABLE OF CONTENTS**

1. Project overview……………………………………………………………… 5
   1. Project summary.…………………………………….………………. 5
   2. Purpose, Scope and objectives ………………………………………. 5
   3. Assumptions and constraints… ……………………………………… 6
   4. Project deliverables…………………………………………………... 6
   5. Schedule summary………………………………………………….... 7
2. Project Context ………………………………………………….................... 7
   1. Proccess Model ………………………………………………........... 7
   2. Method, tools and technique …………………………………………7
   3. Product acceptance plan ……………………………………………. 8
3. Use Case Diagram ………………………………………………. 9
4. Use Case Description ……………………………………….…… 10
   1. Conclude Event ………………….………………… 10
   2. Register for Event ………………….……………….12
   3. Manage Events………………….………………..….14
   4. See Feed ………………….…………………………16
5. System Sequence Diagram……….……………………………….20
   1. Conclude Event ………………….………………… 20
   2. Register for Event ………………….……………….21
   3. Signup …….………………….……………………..21
   4. Manage Events………………….………………..….22
   5. Login …… ………………….……………….…..….23
   6. See Feed ………………….…………………………23
6. Supporting process plans…………………………………………………………….…..…1
   1. Risk management…………………….………………………………………………………….12
7. **List of figures**

Use case diagram  
System sequence diagram

1. **List of tables**
   1. Functional testing Non-functional testing
   2. Team roles and responsibilities Entity roles and responsibilities

# GITHUB LINK:

[**https://github.com/Talha-AppDev/Software-Construction\**](https://github.com/Talha-AppDev/Software-Construction\)

**1. Project Overview**

**1.1 Project Summary**

The department aims to create an Event Management System (EMS) named "Event Hub". This system will promote, manage, and document both departmental and external events by streamlining registration, event monitoring, and post-event feedback. The system is intended to be efficient and user-friendly, enhancing student participation and ensuring comprehensive record-keeping for future planning.

**1.2 Purpose, Scope, and Objectives**

**Purpose:**

To design an efficient, automated system that manages event registration, monitoring, and feedback, thereby ensuring a smooth and engaging experience for students, faculty, and event organizers.

**Scope:**

**Context:**

A standalone system managing event registration and the scheduling of student participation.

**Functional Requirements:**

- Event Management: Allows external organizers and departments to create, edit, and promote events.

- Student Registration: Enables eligible students to register for events.

- Post-Event Updates: Facilitates the upload of photos, news, and key outcomes after events are completed.

- Feedback and Rating: Provides mechanisms for students and teachers to rate and review events.

- User Administration: Implements role-based access control for admins, faculty, and students.

- Notification System: Issues automated alerts for upcoming events.

**Objectives:**

- Create a centralized system to track and manage departmental events.

- Allow students to easily register and access event details.

- Support secure access with appropriate permissions based on user roles.

- Facilitate continuous improvement through feedback and ratings from users.

- Ensure that the system is scalable and maintains data security during high concurrency periods.

**1.3 Assumptions and Constraints**

**Constraints:**

- The software is built using Java.

- User authentication is mandatory for registration and feedback submission.

- The system must handle high concurrent users during event registration.

- The project is to be completed within the semester timeline.

**Assumptions:**

- Access to the system is limited to departmental students and staff.

- Events have predefined eligibility criteria (e.g., department, year, batch).

- The EMS supports image uploads and post-event feedback.

**1.4 Project Deliverables**

- Event Management System (EMS): A fully functional web-based application.

- User Interface (UI): An intuitive, responsive interface for all user roles.

- Database Schema: A structured database to store event and user data.

- Post-Event Features: Capabilities for image uploads, event ratings, and feedback submission.

- Documentation: Comprehensive user manual, system design document, and test reports.

**1.5 Schedule Summary**

The project is divided into several phases:

1. Requirement Gathering & Planning: Completed

2. Analysis: 2 weeks

3. Coding: 4 weeks

4. Testing: 3 weeks

5. Deployment: 2 weeks

**2. Project Context and Methodology**

**2.1 Process Model**

An Agile methodology is employed to accommodate evolving requirements and ensure early delivery. Continuous feedback from students and faculty will be integrated into iterative development cycles. This approach ensures:

- Flexibility: Adapting to changes in requirements.

- Early Delivery: Deploying usable features quickly.

- User Involvement: Refining the system through ongoing feedback.

- Higher Quality: Reducing errors through frequent testing.

**2.2 Methods, Tools, and Techniques**

- Programming Language: Java

- Documentation: MS Word

- Project Management: Project Libre for scheduling and tracking progress

- Version Control: Git and GitHub

- Database: Firebase for managing event and user data

- Server-Side Deployment: Apache

**2.3 Product Acceptance Plan**

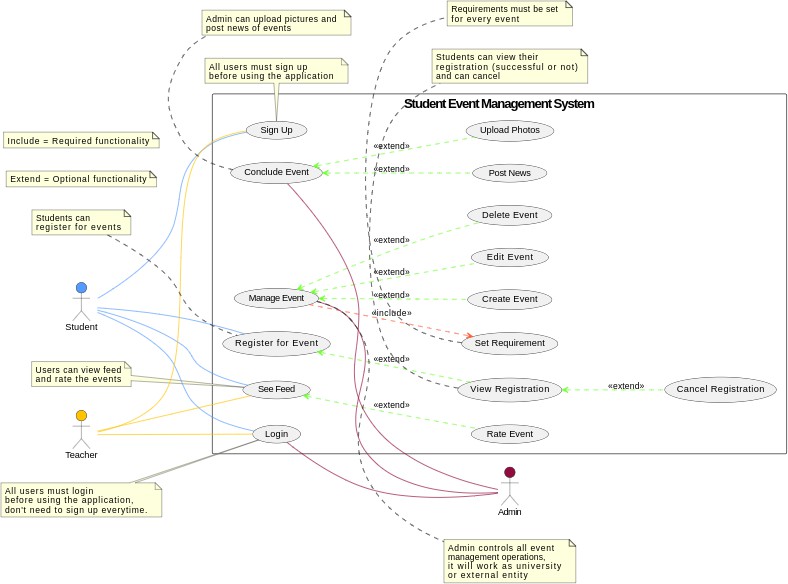
The EMS will be accepted when it:

- Allows creation, modification, and deletion of events by authorized users.

- Enables secure registration with confirmation for both students and faculty.

- Supports post-event updates (photos, news, ratings, and feedback).

- Implements a secure, role-based login system.

**Use Case Diagram**

**3. Functional Requirements and Use Cases**

***BY: M. Waleed***

**3.1 Use Case 1: Conclude Event**

**Primary Actor:**

* Admin (Event Organizer)

**Stakeholders and Interests:**

* **Student:**
  + Wants to see event outcomes, news, and photos, and provide feedback.
* **Department:**
  + Wants a record of completed events for documentation and future planning.
* **Event Organizer (Admin):**
  + Wants to officially mark the event as completed and share updates.

**Preconditions:**

* The event must have taken place.
* The admin must be logged into the system.

**Success Guarantee (Postconditions):**

* The event status is updated to "Completed."
* Photos and news are uploaded for students and teachers to view.
* The system allows students and teachers to provide ratings and feedback.

**Inputs:**

* Event ID
* Post-event summary
* Key highlights
* Photos/images
* Event news updates

**Outputs:**

* Event marked as "Completed."
* Uploaded images and news displayed to students and teachers.
* Rating and feedback section enabled for users.

**Main Success Scenario (Basic Flow):**

1. The admin selects the event to mark as completed.
2. The system prompts for post-event details (summary, key highlights, photos).
3. The admin uploads event news, images, and key outcomes.
4. The system updates the event status to "Completed."
5. The system enables the rating and feedback feature for students and teachers.

**Alternate Scenario (Extensions):**

* **a. System Failure at Any Time:**
  + The software saves existing data automatically.
  + When the system is reopened, it resumes from the last saved step.
* **3a. The admin does not have all post-event details:**
  + The system allows saving progress and updating later.
* **4a. Upload fails due to file size or format issues:**
  + The system notifies the admin and prompts for a compatible file.

**Special Requirements:**

* The application should be developed in Java.
* Student registration should be easily visible, using a readable font size (e.g., 12pt).

**Frequency of Occurrence:**

* Nearly continuous.

**Open Issues:**

* Open for feedback to improve the use case text.
* Consider additional input and output details for better usability.

***By M. Ahmad Hassan***

**3.2 Use Case 2: Student Registration**

**Primary Actor:**

* Student

**Stakeholders and Interests:**

* **Student:**
  + Wants to register easily for eligible events and view registration details.
* **Department:**
  + Wants an organized system to manage student participation.
* **Event Organizer:**
  + Wants to monitor student registrations and ensure participant limits are maintained.

**Preconditions:**

* The student must be logged into the system.
* The event must be open for registration.

**Success Guarantee (Postconditions):**

* The student is successfully registered for the event.
* The system updates the event's participant list.

**Inputs:**

* Event ID
* Student ID
* Eligibility verification data
* Registration confirmation request

**Outputs:**

* Confirmation message of successful registration
* Updated participant list
* Error message (if registration fails due to eligibility or full capacity)

**Main Success Scenario (Basic Flow):**

1. The student navigates the events section and selects an event.
2. The system displays event details and eligibility criteria.
3. The student confirms eligibility and proceeds with registration.
4. The system registers the student, updates the participant list, and sends a confirmation.
5. The student can view or cancel their registration.
6. If canceled, the system updates the participant list and confirms cancellation.

**Alternate Scenario (Extensions):**

* **a. System Failure at Any Time:**
  + The software saves existing registration data automatically.
  + When the system is reopened, it resumes from the last saved step.
* **3a. Student is not eligible for the event:**
  + The system displays an error message and does not allow registration.
* **4a. Event registration limit is reached:**
  + The system notifies the student that registration is full and prevents further registration.

**Special Requirements:**

* The application should be developed in Java.
* Student registration should be easily visible, using a readable font size (e.g., 12pt).

**Frequency of Occurrence:**

* Nearly continuous.

**Open Issues:**

* Open for feedback to improve the use case text.
* Consider additional input and output details for better usability.

# BY: AHMAD MASOOD

## Use Case 3: Manage Event

**Primary Actor:**

* Admin

**Stakeholders and Interests:**

* **Admin:**
  + Needs to create, update, delete events, and set event requirements efficiently.

**Inputs:**

* Event details (name, description, location, date, time, participant limit)
* Edited event details
* Confirmation for deletion
* Requirement details (prerequisites, maximum participants)

**Outputs:**

* Confirmation of event creation
* Updated event details
* Deletion confirmation
* Error messages if required fields are missing or unauthorized access is attempted

**Preconditions:**

* The event management system must be available and functional.
* Only Admins can create, edit, delete, and set requirements for events.

**Postconditions:**

* The event is successfully created, updated, or deleted.
* Event requirements are correctly configured and stored.

**Main Success Scenario (Basic Flow):**

1. The admin navigates to the "Manage Event" section.
2. The system displays options for managing events.
3. The admin selects "Create Event."
4. The system prompts for event details (name, description, location, date, time, participant limit, etc.).
5. The admin enters the required details and submits the form.
6. The system validates the input and saves the event.
7. The system confirms event creation and adds it to the event list.
8. The admin selects an existing event to “Edit.”
9. The system displays the event’s details.
10. The admin modifies the necessary fields and submits changes.
11. The system validates and updates the event information.
12. The admin selects an event to “Delete.”
13. The system prompts for confirmation.
14. Upon confirmation, the system removes the event.
15. The admin selects "Set Event Requirements."
16. The system displays available requirement fields (e.g., prerequisites, maximum participants, category).

**Alternate Scenario:**

* **a. System Failure at Any Time:**
  + The software saves existing data automatically.
  + When the system is reopened, it resumes from the last step saved.
* **5a. If required fields are missing, the system prompts the admin to complete them.**
* **10a. If required fields are missing, the system prompts the admin to complete them.**

**Special Requirements:**

* The application should be developed in Java.
* Event details should be stored securely.

**Frequency of Occurrence:**

* Nearly continuous.

**Open Issues:**

* Open for feedback to improve the use case text.
* Consider additional input and output details for better usability.

***By Talha Arif Wains***

**3.4 Use Case 4: See Feed and Event Rating**

**Primary Actors:**

* **Student**
* **Teacher**

**Stakeholders & Interests:**

* **Students & Teachers:**
  + Want to see event updates, news, and photos.
  + Provide feedback by rating events.
* **Admin:**
  + Upload event-related news and images.
  + Ensure the event feed is updated with relevant content.

**Preconditions:**

* The user is authenticated (logged in).
* The admin has successfully uploaded event updates (news and/or photos).

**Success Guarantee (Postconditions):**

* The user successfully views the event feed, including photos and news.
* Any rating submitted by the user is stored, and the event’s overall rating is updated accordingly.

**Inputs:**

* Event selection by the user
* Rating submitted by the user (e.g., 1 to 5 stars)
* Feedback comments (optional)

**Outputs:**

* Display of event feed, including images and news updates
* Updated event rating
* Feedback comment (if provided)

**Main Success Scenario (Basic Flow):**

1. The student/teacher logs into the system.
2. The user navigates to the "Event Feed" section.
3. The system displays a list of completed events with images, news, and details.
4. The user selects an event to view.
5. The system displays detailed event information, including images and news updates.
6. The user has the option to rate the event by selecting a star rating (1-5).
7. If the user provides a rating, the system updates the event’s overall rating.
8. The user can also leave a feedback comment (optional).
9. The system stores the feedback and associates it with the event.
10. The event’s updated rating and feedback become visible to other users.

**Alternate Scenarios (Extensions):**

* **a. System Failsreure at Any Time:**
  + The system automatically saves event ratings and comments.
  + Users can reattempt their actions after the system recovers.
* **3a. No events available to view:**
  + The system displays a message: "No events available."
* **6a. User attempts to submit a rating but is not logged in:**
  + The system prompts the user to log in first.
* **8a. User enters inappropriate content in the feedback:**
  + The system detects inappropriate language and prompts the user to modify the comment.

**Special Requirements:**

* The application should be developed in Java.
* Event details and feedback should be stored securely.
* User ratings should be aggregated to show the overall event rating.

**Frequency of Occurrence:**

* Frequently after events are completed.

**Open Issues:**

* Consider adding a reporting system for inappropriate feedback.
* Implement an option to sort events based on ratings.

**4. Supporting Process Plans**

**4.1 Risk Management**

The project includes proactive risk management through the following measures:

- **Risk Identification:** Regular team sessions, stakeholder meetings, and historical data reviews identify potential technological, scheduling, personnel, and complexity risks.

- **Risk Analysis:** Risks are assessed using a risk matrix and prioritized as high, medium, or low.

- **Mitigation Strategies:**

\* Technological: Regular code reviews, automated testing, and robust security measures.

\* Scheduling/Budget: Use of Project Libre to track milestones and trigger resource reallocation if delays occur.

\* Personnel: Cross-training team members and thorough documentation to mitigate knowledge loss.

\* Complexity: Modular development and phased implementation to break down complex tasks.

- **Monitoring:** Maintaining a risk register, conducting weekly review meetings, and communicating updates to stakeholders.

**5. Documentation and Final Deliverables**

The final deliverables for the Event Hub project include:

- The fully functional Event Management System (EMS)

- A web-based user interface

- A well-structured database schema

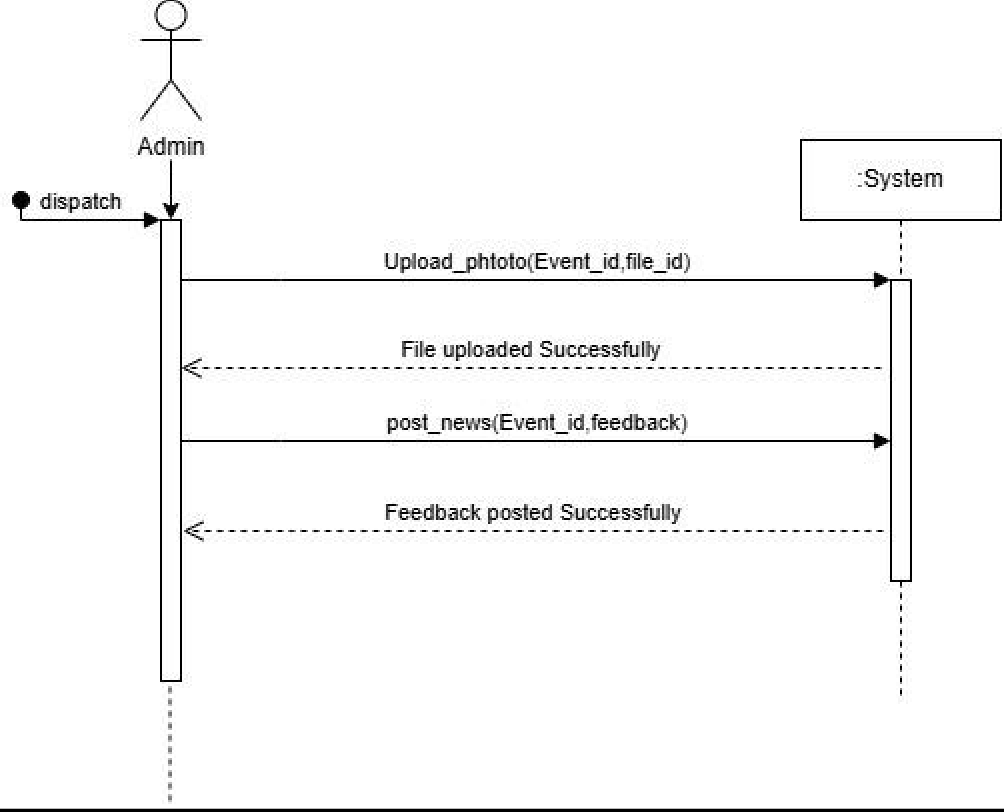
- Post-event features (image uploads, ratings, feedback)

- Comprehensive documentation including the user manual, design documents, and test reports

**Class System Sequence Diagram:**

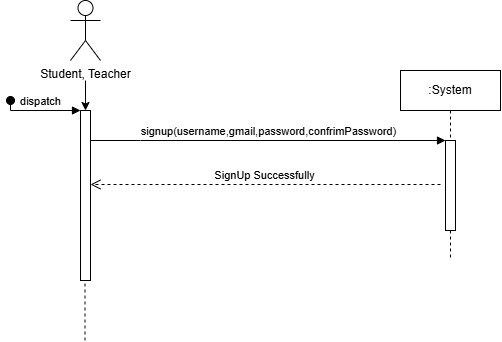
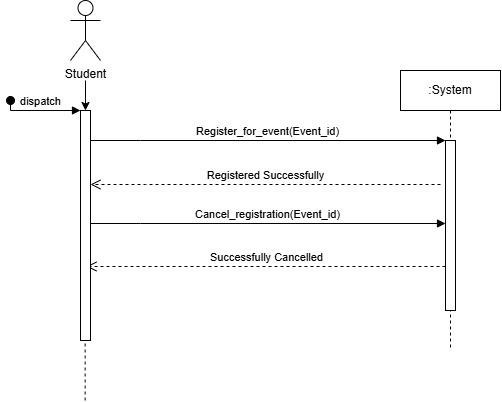
**BY: M. Waleed**

**Use Case: Conclude Event**

****

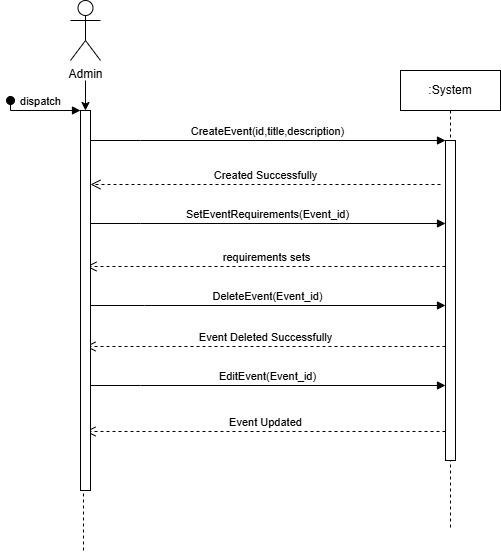
# BY: M. AHMAD HASSAN

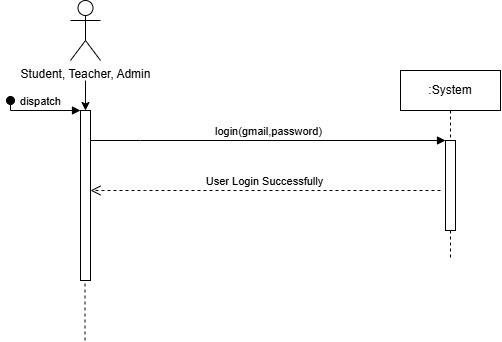
**Use Case: Register for Event**

**Use Case: Signup**

# BY: AHMAD MASOOD

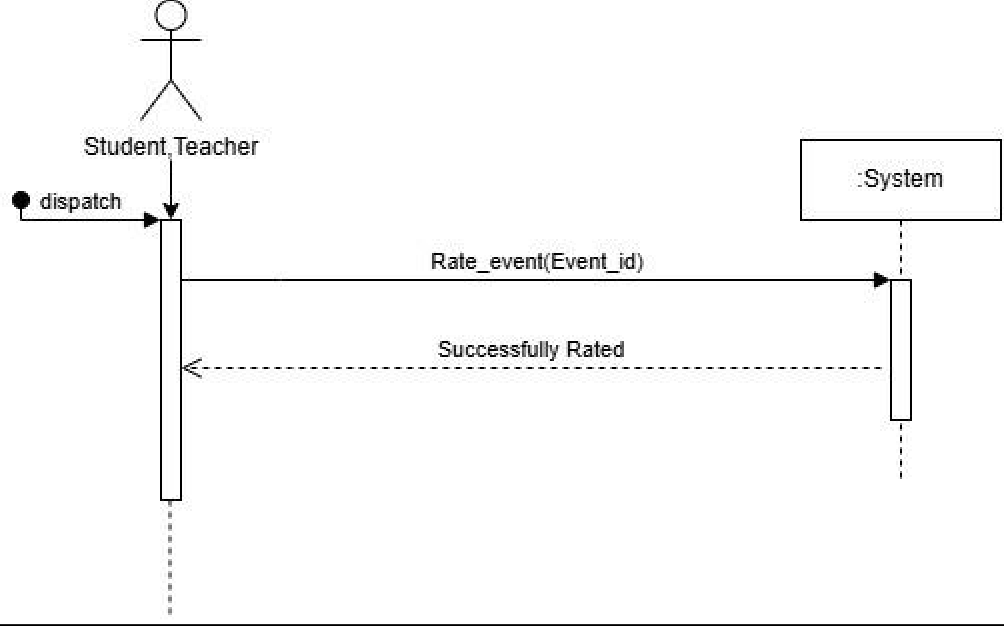
**Use Case: Manage Event**



**Use Case: Login**

# BY: TALHA WAINS

## Use Case: See Feed

****

**Domain Model:**

