# Use Cases for Event Enrolment

|  |
|---|
| **UC. 1 Student Login** |
| **Goal in context**<br>create an account to purchase tickets to events This use case allows potential students of university of Wisconsin Milwaukee to sign up and create an account to purchase tickets to events |
| **Scope**<br>• university of Wisconsin Milwaukee student event enrolment. |
| **Level**<br>• User goal |
| **Actors**<br>• Student<br>• Stakeholders and interest<br>• Student – Creation of account in simple and secure manner<br>• Event Management- Expand student base, Guarantees complete and accurate data collection |
| **Preconditions**<br>• Architecture up and running<br>• Request is not from existing or prior terminated student |
| **End Condition Success**<br>• Account created for student<br>• Email conformation for account creation is sent to student |
| **End condition failure**<br>• Existing account- account not formed<br>• Terminated account – need to regulate what will be done in this case • Student does not accept terms – account not created<br>• Infrastructure failure – Student tries at a later point in time |
| **Main success scenario**<br>1. Student authentication using PAWS<br>2. Capture student data from PAWS<br>3. Capture credit card data<br>4. Validate credit card data<br>5. Verify credit card data<br>6. Create account in the application<br>7. Send email confirmation for account creation |

**Extensions**

a. Infrastructure failure

- Student retries at later time

2.a Capturing data from PAWS

- Any missing data notify the student

2.b Capture of incorrect/ invalid data

- Highlight the data that is incorrect and request for updating of data in PAWS

4.a Existing account

- Notify student about existing account and terminate process

4.b Previous terminated account found

- Notify student about previous terminated account and terminate

process 9.c Student decides not to create account

- Discard data and terminate process

9.a Incomplete data

- Highlight missing data and prompt for re-entry

9.b Invalid credit card data

- Highlight invalid/ incorrect data and prompt for re-entry

9.c Credit cards expired

- Notify student and prompt for alternative credit card

9.d Credit number does not match type

- for re-entry Indicate error to student and prompt

9.e Credit card number not associated with the student

- Place hold or card and notify student about possible fraudulent use of credit card

9.f Credit card reported lost or stolen

- Decline cards and notify student about possible fraudulent use of credit card

11.a Email not deliverable

- Flag account as inactive

**Special Requirements, including performance**

- Account creation response
- Student should belong to University of Wisconsin Milwaukee
- Event held on campus

**Technology and data variants**

- Should be accessible via Android and IOS

**Frequency**

- 50 tickets to be sold per event
- No transactions limit per event per day

**Open Issues**

- What should be the correct action for closed accounts?
- How do we recognize existing or terminated account?
- What credit cards types will support?
- Do we need to capture card holder name and credit cards billing address as part of this use case?
- What are our obligations for cards reported lost or stolen?

**Related use cases**

- UC. 2 Student account update
- UC. 3 Student account closure
- UC. 12 Terminate account

**Non-functionality requirements**

- Usability – Simple navigation and language, a forgiving interface, and ease of use
- Performance time – Account creation response time should be under 5 seconds

- Security – No access to private customer and their account information • Supportability – Accessible via Android and IOS

**UC.2 Student Account Update**

This use case provides students with the option to modify and update their personal data in the app. The changes can include preferred name, contact number and email, demographics, credit card details where verification and validation apply accordingly.

**UC. 3 Student Account Closure**

The student account closure use case is to let students close their account for any reasons provided or they could provide their own reasons if it’s not listed. They can also deactivate if they want to.

**UC. 4 Terminate Account**

This use case permits students and the student support team can terminate the student account. This is different from closing or deactivation of an account as the data is not archived in any database and is completely erased from the system. Student can re-register if they want to get back to using the application.

**UC. 5 Selection of Event**

**Goal in Context**

This use case allows students to select event by browsing through events available post  successful login.

**Scope**

- Browsing through all the events available.
- Searching for specific event through filters available.
- Selecting the event of their choice.

**Actors**

- Students

**Stakeholders and Interests**

- Students – Select the event of their choice.
- Student Support

**Preconditions**

- List of events happening at University of Wisconsin Milwaukee are available on the application.
- Students successfully login into the application to access the events. • Student support have all the data about all the events happening.

**End conditions – Success**

- Student selecting the event of their choice.
- Reaching the payment page.

**End conditions – Failure**

- Not able to login due to various reasons which include termination, closure.

- If the total number of available seats is zero, then selection of event cannot be  done.

| |
|---|
| **Main Success Scenario**<br>1. Student successfully login by authentication using PAWS account.<br>2. Student data is validated.<br>3. Student credit card details are captured, validated and verified.<br>4. Browse through available events or search for the event of their choice. 5. Number of seats available for the selected event is greater than zero. 6. Student selects the event and select the seat they wish to.<br>7. Once the selection is done, students are directed to the payment page. |
| **Extensions**<br>a. Infrastructure failure<br>• Student tries again at later time.<br>• Student would not be<br>2a. Unsuccessful login or authentication using PAWS account<br>• Notify student of error and prompt for re-entry, if less than 3 times. 3a. If student account is terminated or closed.<br>• Appropriate message is displayed and process is terminated<br>5a. If a particular event is fully filled up.<br>• Inform student that the event is full and ask them to select some other event. 7a. If an event gets cancelled.<br>• Student is informed regarding the cancellation of event and any updates of the event.<br>12a. If student needs any information or more updates of the event. • Student can contact the student support team and get the related information.<br>14a. If a student searches for the event, it is based on the events shown on the app. • Student get the information about each event and all the events data get updated on the event report. |
| **Special Requirements, including Performance**<br>• Any event can be browsed and the response time should be less than 10 secs, 95% of time. |
| **Technology and Data Variants**<br>• Should be accessible via IOS and Android mobile platforms. |
| **Frequency**<br>• Based on the number of events happening, 100 searches per event. |
| **Open Issues**<br>• Sudden cancellation of event upon selection.<br>• Sudden change of availability of the event.<br>• When the event is fully signed up.<br>• How many number of seats can be selected per event?<br>• How to select multiple events in one go?<br>• How to show the number of seats available in a particular event? |

**Related Use Cases**

- UC.1 Student login
- UC.6 Update of event
- UC.7 Cancellation of event

- UC.8 Purchase of tickets
- UC.12 Inquiries
- UC.14 Event Report

**Non-Functional Requirements**

• Usability – Easy to use, simple search and browsing through events • Reliability – 99% uptime, down time not to exceed 4 hours/month. • Performance – availability checking response time should be under 20 seconds 95% of time.
• Security – No one should be able to get into our app with administrator privileges. • Supportability – Is fully accessible via Android and IOS operating system.

**UC.6 Update of event**

This use case allows event organizers to make changes to the schedule of events, update or modify an event. Changes are confined to modifying the number of seats, changing the event date, event venue. After update of event, the updated details are captured and everything is verified again with respect to availability of venue on the date mentioned to see if there is any overlap with other events. If any changes are made after a student register for the event, updated details are provided via emails.

**UC. 7 Deletion of event**

This use case allows event organizers to cancel events in case of any unexpected situations and allows student support to communicate this to students. If any event is cancelled after registrations, the money will be refunded to students and email communications are sent for the inconvenience caused.

**UC. 8 Purchase of Tickets**

**Goal in Content**

This use case allows students with PAWS account to purchase tickets to any event.

**Scope**

- University of Wisconsin Milwaukee

**Level**

- User goal

**Actors**

- Students

**Stakeholders & Interests**

• Students – purchase of tickets without difficulty, and in a secure
way • University of Wisconsin Milwaukee – increase sales of tickets

**Preconditions**

- Architecture up and running
- Student should have a valid account

**End conditions – success**

- Tickets are sold to students
- Seat inventory reduced for every subsequent purchase
- Email confirmation sent to students

**End Conditions – failure**

- Payment failure – seek alternative credit card or terminate process

- Enough seats not available – offer alternative option to students
- Students elects not to purchase tickets – terminate process
- Infrastructure failure – Student tries at a later point

**Main success scenario**

1. Student authentication using PAWS
2. Capture student data from PAWS
3. Patron specifies event name, date, time,
4. System checks availability of seats

5. System computes total amount due for tickets including miscellaneous and
other changes

6. Students agrees to purchase tickets
7. System process payment: include UC: 9 Process Payment
8. System marks seats are sold
9. System emails confirmation of the purchase to the students

1. 10. System initiates ticket delivery

**Extensions**

a. Infrastructure failure

- Student tries again later

2.a PAWS Authentication Failure

- Notify student regarding the failure of Authentication

4.a Student account terminated/ closed

- Display appropriate message and terminate process

8.a If tickets are not available

- Notify student about unavailability and suggest any other event

8.b Student plans not to proceed

- Remove any selected ticket and terminate process

9.a Failure in payment process

- Notify student of reason for failure and prompt for other credit card/ terminate process

11.a Email to student bounces

- Issue in sending email to the student

**Special Requirements, including performance**

- Tickets purchase response time should be under 15 seconds 95% of the time

**Technology and Data Variants**

- Should be accessible via Android and IOS clients

**Frequency**

- Depends on the number of events that are held and 100 transactions per event per day

**Open Issues**

- How do we handle concurrent requests?
- How long should tickets be held for before they are released to general availability?
- How can we prevent automated purchase?
- What should the limit be for number of tickets purchased?
- How do we recognize multiple purchases from the same student or account? • When should this use be expanded to deal with multiple events in one purchase?

- Should the system provide information on sold out events, or should they be filtered out?
- Should mobile ticket delivery be restricted to the smartphone on file with this account?

| |
|---|
| **Related Use Cases**<br>UC. 9 – Process Payment<br>UC. 1 - Student Login<br>UC. 5 - Selection of Event<br>UC. 11 - Deliver Ticket |
| **Non-functional requirements**<br>• Usability – easy to use, simple to navigate, forgiving interface desired • Reliability – 99% uptime, downtime does not exceed 8 hours/ month • Performance – availability checking response time should be under 15 seconds  95% of the time<br>• Security – no eavesdropping, or access to private customer or account information • Supportability – accessible via Android and IOS. |

| **UC. 9 – Process Payment** |
|---|
| **Goal**<br>• This use case processes a credit card payment for the purchase of the tickets. |
| **Scope**<br>• UWM event ticket sales |
| **Level**<br>• Sub- function |
| **Actors**<br>• Students- primary (through UC. 8 purchase tickets) |
| **Stakeholders & interests**<br>• Student -secure, secure, secure<br>• Payment processor -accurat6e and timely processing |
| **Preconditions**<br>• Link to payment processing is up and functioning<br>• Payment amount is available |
| **End conditions – success**<br>• Student’s credit is charged correctly for the amount due |
| **End conditions - failure**<br>• Infrastructure failure -System flags as the transaction is incomplete<br>• Invalid credit data- Transaction flagged as failed<br>• Charge request denied -Transaction flagged as failed |

**Main success scenario**

1.Systems initiates payment request with the amount due and requests credit card information form the student
2.Student selects stored credit card
3.Student provides credit verification value

4. System verifies credit card data
5. System issues charge request to payment processor using details provided 6.Payment processor approves charge
7.System returns approval authorization to calling use case for appropriate processing

**Extensions**

a. Infrastructure failure
- IT personals resets connection and tries again

9a. CVV missing
- Flag missing data and prompt for re-entry

9b. Credit card expired
- Indicate error to student and prompt for the new credit card

9c. Unable to connect to payment processor
- Retry after a short timeout

9d. No response from payment processor
- After a pre-set timeout, reissue the charge request

9e. Payment processor denies charge
- Capture reasons for denial to relay to student in calling use case

**Special requirements**

Response time should be about 10 seconds when called from UC.8 (purchase Tickets)

**Technology and Data variants**

None

**Frequency**

- IT depends on the number of events that are held and 100 transactions per event per day

**Open issues**

- How much security is required for this use case?
- How soon should the transaction expire if the credit processor doesn't respond?
- What types of failure notices may we expect from the credit processor, and what are our obligations for each one?

**Related use cases**

- UC.8 -Purchase Tickets

**Non-functional requirements**

• Usability – Simple navigation and language, a forgiving interface, and ease of use
• Performance time – Account creation response time should be under 5 seconds
• Security – No access to private customer and their account information • Supportability – None

**UC. 10 Email payment confirmation**

This use case permits payment processors to send confirmation to students upon  successful payment for the event. An invoice is sent to them stating the payment details,  with amount paid, student details and mode of payment.

**UC. 11 E-Ticket delivery through email**

This use case allows the payment processor to send the ticket which is an electronic ticket  to their email for the event for which the student has registered and made payment. The  email is sent with a ticket consisting of event details and student details.

**UC.12 Inquiries**

This use case allows students, event organizers, student support to enquire or search data  related to student account, list of events sold out, seats available for the event, ticket  prices for the given event.

**UC. 13 Ticket Sales Report**

This use case allows student support to access the report showing the ticket sales made.  This report shows sales made for every week. Weekly reports are generated, and the  support team accesses them to work on improving services.

**UC. 14 Event Report**

This use case allows event organizers and student support to access an event report  which generates a number of events conducted in a week.

| **UC. 15 Creation of Event** |
|---|
| **Goal in Context**<br>Create an event form for event organizers to fill their personal and also the event details they want to host. This use case allows organizers from different departments of University of Wisconsin Milwaukee to enter all required information about event and later sent for approval. |

**Scope**

- University of Wisconsin Milwaukee event creation

**Level**

- User goal

**Actors**

- Event Organizer
- Event Support
- Event Management

**Stakeholders and Interests**

- Organizers – Fill the form and update information regarding the event
- Event Support

**Preconditions**

- Event organizer should be a member of any department of UWM

**End conditions – Success**

- Information of the event submission to event support team.
- Email conformation for event creation is sent to organizer along with approval waiting time.

**End conditions – Failure**

- Event not created
- Event form abruptly ended
- organizer does not accept terms –event not created
- Infrastructure failure – server lost connection

**Main Success Scenario**

1. Capture event data through form
2. Capture event requirements (logistics and Advertising) data
3. Validate event and event organizer data
4. Check for availability of venue on the date mentioned in the form
5. Capture new date if the venue is not available on previous mentioned date
6. Validate event data
7. Verify event and event organizer data
8. Create event in the application
9. Send email confirmation for event creation
10. Send email stating event waiting period for approval

**Extensions**

a. Infrastructure failure
- Organizer retries at later time

2a. Capturing data using event form.
- Any missing data notify the event organizer

2b. Capture of incorrect/ invalid data.
• Highlight the data that is incorrect and request for updating of data in Email through link, after approval of event if necessary

5a. Event organizer decides not to create event
- Discard data and terminate process.

5b. Incomplete data.
- Highlight missing data and prompt for re-entry.

6a. Email not deliverable.

- Flag account as inactive.

**Special Requirements, including Performance**

- Organizer should belong to University of Wisconsin Milwaukee
- Event held on campus.
- Event creation response

**Technology and Data Variants**

- Should be accessible via IOS and Android mobile platforms.

**Frequency**

- 50 event forms can be created in a second

**Open Issues**

- How to avoid server failures?
- What needs to be considered for approval of event?
- What are our obligations for events from being created?

**Related Use Cases**

- UC. Publishing of event
- UC Event Management account

**Non-Functional Requirements**

- Usability – Simple navigation and language, a forgiving interface, and ease of use.
- Performance – availability checking response time should be under 20 seconds 95% of time.
- Security – No one should be able to get into our app with administrator privileges.
- Supportability – Is fully accessible via Android and IOS operating system.

**UC. 16 Event Maintenance**

**Goal in Context**

To keep track of events published and conducted throughout the semester, this use case provides archiving of events that have been already conducted and shall not repeat during the semester and keeps future events on the app.

**Scope**

- University of Wisconsin Milwaukee event management and maintenance.

**Level**

- Event support goal

**Actors**

- Event Organizer
- Event Support
- Event Management

**Stakeholders and Interests**

- Organizers – Fill the form and update information regarding the event
- Event Support

**Preconditions**

- Event organizer should be a member of any department of UWM
- Event should be in approved status
- Event start date and end date should be provided.

**End conditions – Success**

- Event end date greater than or equals to current date.

**End conditions – Failure**

- Event end date less than current date, archive the event.
- Infrastructure failure – server lost connection.

**Main Success Scenario**

1. Event published is verified with the dates
2. Old events are archived
3. Upcoming and current events are kept published

**Extensions**

a. Infrastructure failure

- Organizer retries at later time.

2a. Start date and end date verified

- Any missing data notify the event organize.

**Special Requirements, including Performance**

- Event published should have start and end dates within current date and in future
- Past events should be archived and not displayed on application
- Organizer should belong to University of Wisconsin Milwaukee
- Event held on campus

**Technology and Data Variants**

- Should be accessible via IOS and Android mobile platforms.

**Frequency**

- 50 event forms can be created in a second

**Open Issues**

- What should be the correct criteria apart from start and end dates, to archive events?
- Do we need waiting time or inform organizers before archiving any event?
- What are our obligations for an event to be archived?

**Related Use Cases**

- UC. Publishing of event
- UC Event creation

**Non-Functional Requirements**

- Usability – Simple navigation and language, a forgiving interface, and ease of use.
- Performance – Archive response within 5 secs.
- Security – No access to private customer, who doesn't belong to UWM or any of its departments
- Supportability – Is fully accessible via Android and IOS operating system.

## UC. 17 Approval of Event

**Goal in Context**

After event creation is successful, once the event is approved after all criteria is met and after the event that has been approved by event support team the organizers are sent an email with a link to update more details about event logistics and advertising.

**Scope**

- University of Wisconsin Milwaukee event publishing.

**Level**

- User support goal

**Actors**

- Event Organizer
- Event Support
- Event Management

**Stakeholders and Interests**

- Organizers – Fill the form and update information regarding the event
- Event Support

**Preconditions**

- Event organizer should be a member of any department of UWM.
- Event should be in approved status.

**End conditions – Success**

- Event approved for organizer
- Email confirmation for event approval is sent to organizer along with a link to update event details such as logistics, advertising details etc.

**End conditions – Failure**

- Event not approved
- Event approved yet link to update details not sent
- organizer does not accept terms and conditions –event not published
- Infrastructure failure – server lost connection

**Main Success Scenario**

1. The event information is received by the management team
2. The management team manually verifies all the event data provided by the organizers
3. If any information is to be updated, an email is sent to the organizers to update
4. After update of the new event data, it is again manually verified
5. Send email confirmation for event approval
6. Capture additional event data (logistics and advertising) through link provided in email
7. Capture and validate event logistics data
8. Verify event logistics data
9. Capture and validate event advertising data
10. Verify event advertising data
11. Send email
12. Publish the event in the application
13. Send email confirmation for event publishing

**Extensions**

b. Infrastructure failure
- Organizer retries at later time

2a. Capturing logistics data using external url.
- Any missing data notify the event organizer

5a. Capturing advertising data using external url.
- Any missing data notify the event organizer

6a. Event support team decides not to publish event during verification.

- Discard data and terminate process.

6b. Incomplete data.

- Highlight missing data and prompt for re-entry.

9a. Email not deliverable.

- Flag account as inactive.

**Special Requirements, including Performance**

- Event publish response time within span of details provided
- Organizer should belong to University of Wisconsin Milwaukee
- Event held on campus

**Technology and Data Variants**

- Should be accessible via IOS and Android mobile platforms.

**Frequency**

- 50 events can be published in a second

**Open Issues**

- What should be the correct criteria for capturing of logistics and advertising data?
- What type of events can be published and supported?
- Do we need waiting time after approval till event published?
- What are our obligations for an event to be approved?

**Related Use Cases**

- UC Event Creation
- UC Event Management

**Non-Functional Requirements**

- Usability – Simple navigation and language, a forgiving interface, and ease of use.
- Performance – Event approval response time should be 5 to 7 days
- Security – No access to private customer, who doesn’t belong to UWM or any of its departments.
- Supportability – Is fully accessible via Android and IOS operating system.