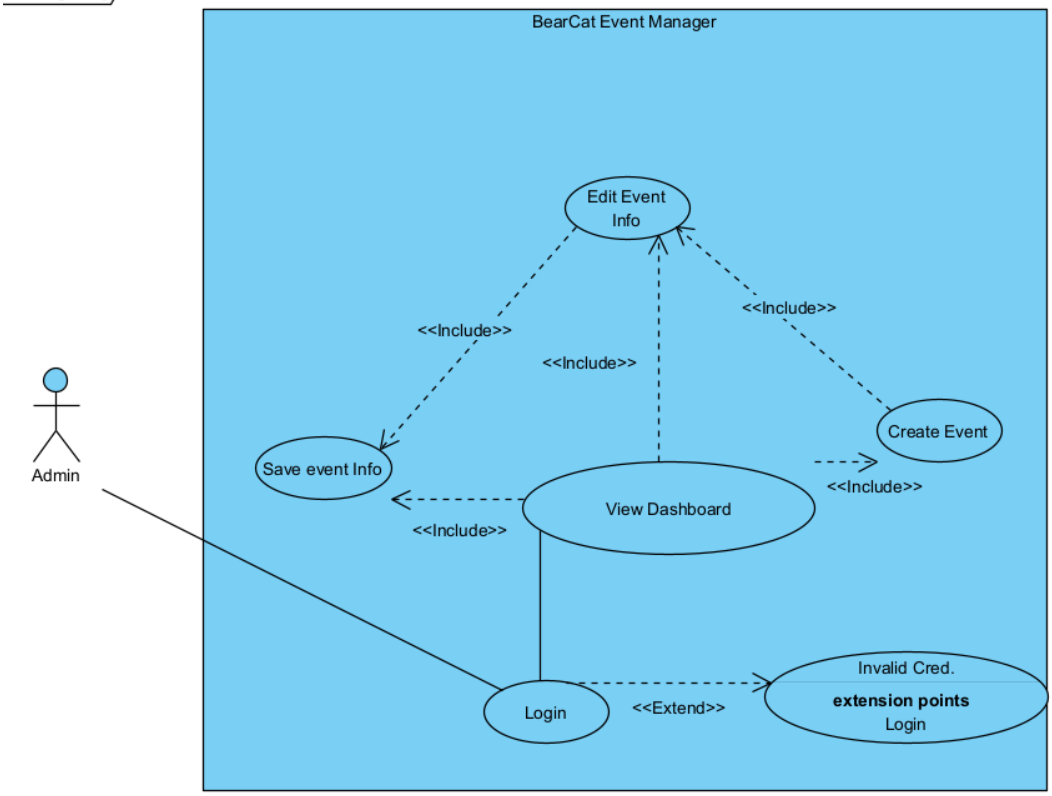
Use Cases for Bearcat Manager

Link to the wiki: https://github.com/djgamekid/GDP-Group-I-bearcatmanager/wiki/Use-Cases-(Iteration-1)

Use Case 1: Creating Events for Students & Staff

Use Case Diagram:



Primary Actor: Admin *(Administrative Director)*

Scope: Admin

Brief: The Admin inserts information regarding an event idea that details the idea itself, time, location, and date. This event is then monitored with analytics and a list of accepted invites from students and staff.

Stakeholders: Admin, University Administration

Postconditions

Minimal Guarantees:

- The Admin will have a monitorable event with notifications sent to students and staff after one signs up.

Success Guarantees:

- An event card that is viewable by Users and updatable by the Admin.

Preconditions

- The system lists event spaces available to the Admin.

Triggers:

- The Admin clicks on an interface button to "Create a new Event."

Basic flow:

1. The system will present a dashboard of analytics regarding any events previously created or a message to create one if none are created.
2. The system will have a button interface that links to a form page for the Admin to insert information.
3. The system will list a series of fillable tabs along with a generated list of locations to select from.
4. The Admin inputs the necessary information and clicks save, which will send the information to be stored and viewed.
5. The system returns the Admin to the dashboard with the viewable card that presents the information in a styled manner.

Extensions:

1–2.

a. **Welcome Message:**

1. The system has a simple greeting after an Admin login.

b. **Editable from Dashboard:**

1. The system displays a button for each created event.
2. The system does not display the button if the time has passed the set time.

4-5.

a. **Alert Confirming Input:**

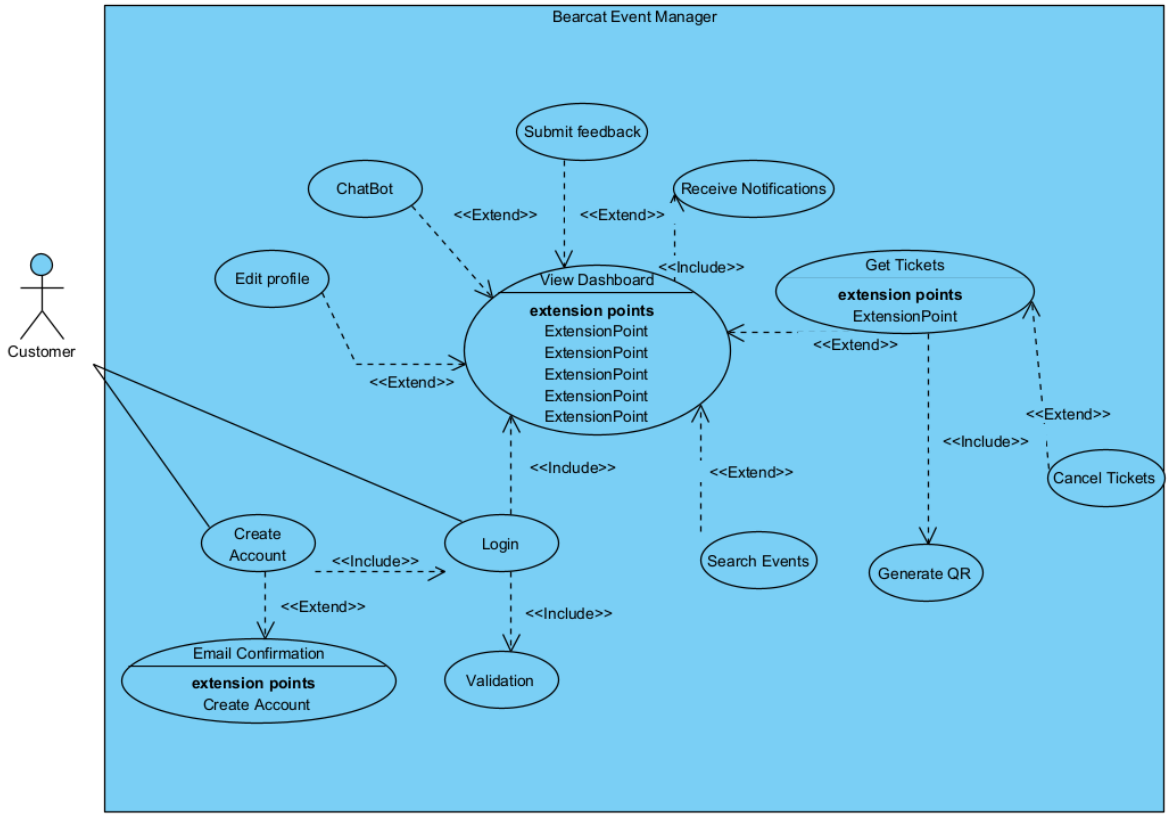
1. The system provides an alert pop-up ensuring the data entered is correct.

b. **Event Notification of Success:**

1. The system sends a notification to the Admin of a successful creation.

Use Case 2: A User (Student or Staff) Registers for an Event

Use Case Diagram:



Primary Actor: User (Student or Staff)

Scope: Event Registration

Brief: A User views a list of upcoming events, selects an event they are interested in, and completes the registration process. The system then updates the event's attendee list and sends a confirmation notification to the User.

Stakeholders: Users (Students and Staff), Admin, University Administration

Postconditions

Minimal Guarantees:

- The system will update the list of attendees for the event.
- The User will receive a confirmation notification of their registration.

Success Guarantees:

- The User successfully registers for the event.
- The User receives all necessary information about the event (e.g., date, time, location) in the confirmation notification.

Preconditions:

- The system displays a list of upcoming events available for registration.
- The User is logged into the system.

Triggers:

- The User clicks on an event they are interested in from the list of upcoming events.

Basic flow:

1. The system presents the User with a dashboard or list of upcoming events.
2. The User selects an event they are interested in by clicking on it.
3. The system displays the event details, including date, time, location, and description.
4. The User clicks the "Register" button for the event.
5. The system prompts the User to confirm their registration.
6. The User confirms their registration.
7. The system updates the event's attendee list with the User's information.
8. The system sends a confirmation notification to the User, containing the event details and a confirmation message.
9. The system returns the User to the dashboard or list of events, where they can see their registered events.

Extensions:

2–3.

a. **Event Filtering:**

1. The system allows the User to filter events by categories such as date, type, or location.
2. The system displays the filtered list of events based on the User's selection.

5-6.

a. **Registration Confirmation Alert:**

1. The system provides a pop-up alert to confirm the User's intent to register for the event.
2. The User has the option to cancel or confirm the registration in the pop-up alert.

7-8.

a. **Notification Preferences:**

1. The system checks the User's notification preferences (email, SMS, in-app notification) and sends the confirmation accordingly.
2. The system logs the notification sent to the User for auditing purposes.

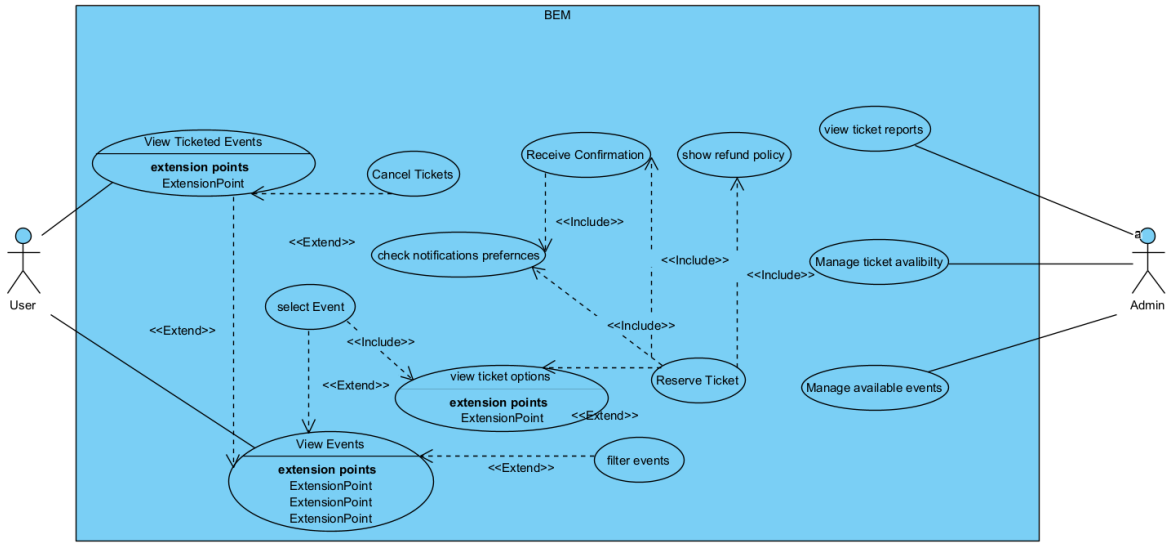
9-10.

a. **View Registered Events:**

1. The system provides a section or filter on the dashboard where Users can view the events they have registered for.
2. The system allows the User to cancel their registration if they no longer wish to attend.

Use Case 3: Ticketing System for Event Management

Use Case Diagram:



Primary Actor: User (Student or Staff)

Scope: Event Ticketing

Brief: A User (Student or Staff) can purchase or reserve tickets for an event through the system. The system manages ticket availability, processes payments if applicable, and sends a confirmation and digital ticket to the User.

Stakeholders: Users (Students and Staff), Admin, University Administration, Payment Processing Service

Postconditions

Minimal Guarantees:

- The system will update the ticket availability for the event.
- The User will receive a confirmation notification and a digital ticket.

Success Guarantees:

- The User successfully purchases or reserves a ticket for the event.
- The User receives a digital ticket with all necessary event information.

Preconditions:

- The system displays a list of events with available tickets.
- The User is logged into the system.
- The User has a valid payment method.

Triggers:

- The User clicks on an event they are interested in and selects the option to purchase or reserve a ticket.

Basic flow:

1. The system presents the User with a list of events that have available tickets.
2. The User selects an event and clicks the "Buy Ticket" or "Reserve Ticket" button.
3. The system displays the event details and ticket options (e.g., general admission, VIP).
4. The User selects the desired ticket option and quantity.
5. The system prompts the User to confirm their selection and proceed to payment (if applicable).
6. The User provides payment information (if applicable) and confirms the purchase or reservation.
7. The system processes the payment (if applicable) and updates the ticket availability.
8. The system sends a confirmation notification and a digital ticket to the User.
9. The system returns the User to the list of events, where they can see their ticketed events.

Extensions:

2–3.

a. **Event Filtering and Sorting:**

1. The system allows the User to filter and sort events by categories such as date, type, or location.
2. The system displays the filtered and sorted list of events based on the User's selection.

4-5.

a. **Ticket Option Details:**

1. The system displays detailed information about each ticket option, including benefits and pricing.
2. The User can view and compare different ticket options before making a selection.

6-7.

a. **Payment Processing:**

1. The system integrates with a secure payment processing service to handle payments.
2. The system displays a payment confirmation page after successful payment processing.

8-9.

a. **Notification Preferences:**

1. The system checks the User's notification preferences (email, SMS, in-app notification) and sends the confirmation and digital ticket accordingly.
2. The system logs the notification and ticket sent to the User for auditing purposes.

10-11.

a. **View Ticketed Events:**

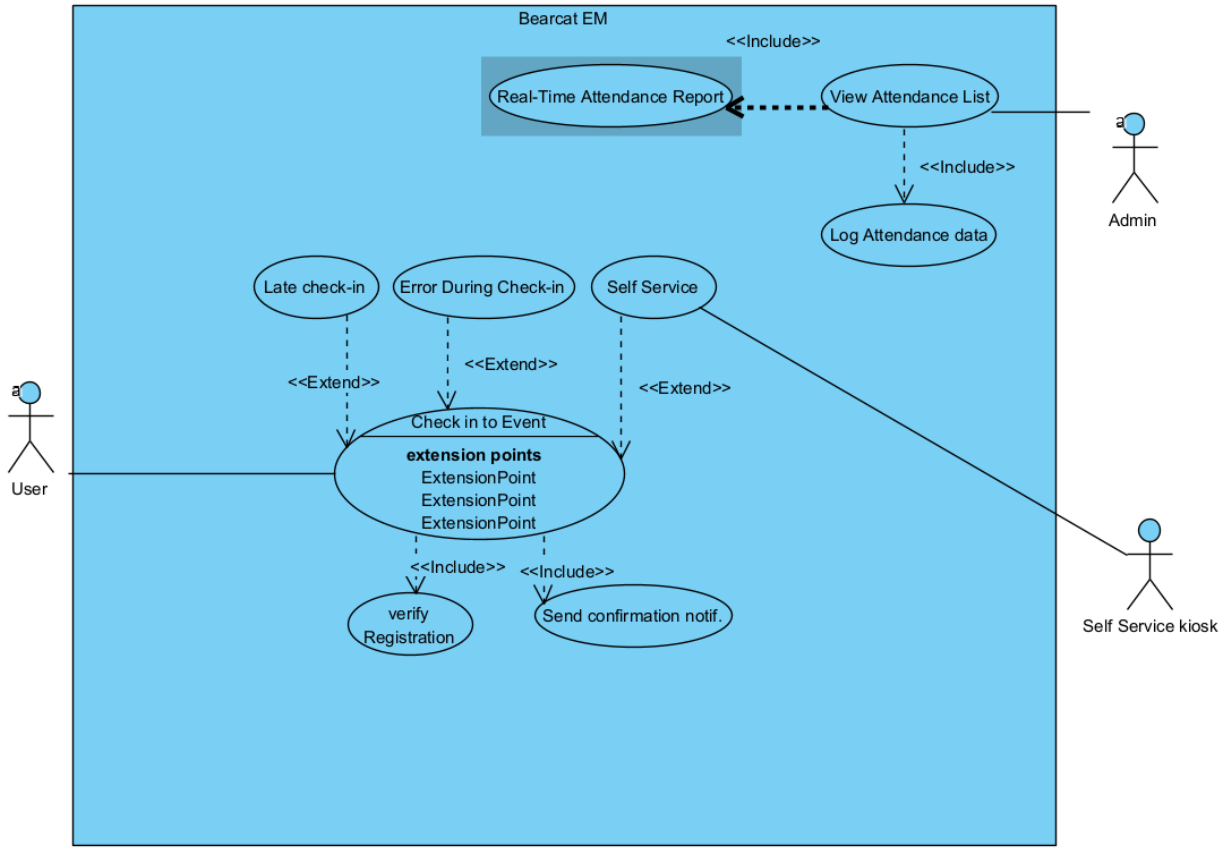
1. The system provides a section or filter on the dashboard where Users can view their ticketed events.
2. The system allows the User to cancel their ticket if they can no longer attend (subject to event policy).

b. **Refund and Cancellation Policy:**

1. The system displays the refund and cancellation policy to the User before confirming the purchase or reservation.
2. The User can request a refund or cancel their ticket based on the event's policy.

Use Case 4: Event Check-In for Registered Users

Use Case Diagram:



Primary Actor: User (Student or Staff)

Scope: Event Check-In

Brief: A registered User checks into an event on the day of the event. The system verifies the User's registration, marks them as "checked in," and updates the attendance list. The system can also allow the User to check in via a mobile app or through a Self-service System at the event venue.

Stakeholders: Users (Students and Staff), Admin, Event Organizers

Postconditions

Minimal Guarantees:

- The system will mark the User as "checked in" for the event.
- The system provides a real-time attendance report for event organizers, showing all checked-in attendees.

Success Guarantees:

- The User is successfully checked into the event and their attendance is recorded.
- The system logs the attendance data for further analytics and reporting.

Preconditions:

- The User is registered for the event.
- The event is active (ongoing or about to start).
- The system has check-in options available (via app or Self-service System).

Triggers:

- The User arrives at the event venue and clicks "Check-In" from the mobile app or checks in using a self-service system at the event location.

Basic flow:

1. The system displays a "Check-In" button for the event on the User's dashboard or in the mobile app.
2. The User selects the event they want to check into and clicks "Check-In."
3. The system verifies the User's registration status for the event.
4. Upon successful verification, the system marks the User as "checked in."
5. The system updates the event's attendance list to reflect the User's check-in.
6. The system sends a confirmation notification to the User, confirming the successful check-in.
7. The system updates the event organizer's dashboard to show the User's attendance.

Extensions:

2–3.

a. **Self-service Check-In:**

1. The User scans their event QR code or taps their ID card at the event.
2. The self-service system displays a confirmation screen, marking the User as "checked in."

4-5.

a. **Error During Check-In:**

1. If the User is not registered, the system displays an error message and prompts them to contact event support.
2. The system logs the error for auditing purposes.

6-7.

a. **Late Check-In:**

1. The system allows for a grace period after the event start time for late check-ins.
2. If the User checks in after the grace period, the system sends a notification indicating they are late.