Student Portal Use Cases

User Management Use Cases

UC-UM001: User Registration

Attribute	Details
Actor	Student/Faculty/Admin
Trigger	User initiates registration
Preconditions	User is not registered
Input	- Personal details (name, email) - Chosen role - Password - Verification information
Validation Steps	1. Validate email format 2. Check email uniqueness 3. Password strength validation 4. Verify role selection 5. Validate personal information completeness
Error Handling	- Invalid email format: Display specific error message - Email already exists: Prompt to login or use different email - Weak password: Show password strength requirements - Incomplete information: Highlight missing fields - System registration failure: Log error, provide retry option
Output	- User account creation confirmation - Verification email - User profile
Priority	High
Description	Create a new user account
Main Flow	1. User enters registration details 2. System validates information 3. Create user profile 4. Send verification email
Alternative Flow	If validation fails, show error messages
Post Conditions	User account created with appropriate role access

UC-UM002: Multi-Factor Authentication

Attribute	Details
Actor	Student/Faculty/Admin

Trigger	User attempts login
Preconditions	User has valid credentials
Input	- Username - Password - Second factor (SMS code/authenticator app)
Validation Steps	1. Check username existence 2. Validate password against stored hash 3. Verify second factor authentication 4. Check account status (active/locked) 5. Validate authentication attempt frequency
Error Handling	- Invalid username: Suggest username recovery - Incorrect password: Limit login attempts - Failed second factor: Provide alternative verification method - Account locked: Implement account recovery process - Suspicious login attempt: Trigger additional security checks - Multiple failed attempts: Temporary account lockout
Output	- Successful login token - Access to user dashboard - Error messages if authentication fails
Priority	Critical
Description	Secure login process
Main Flow	1. User enters username/password 2. System prompts for second authentication factor 3. Verify additional authentication method
Alternative Flow	If authentication fails, deny access
Post Conditions	Secure user login with multi-factor protection

Event Management Use Cases

UC-EM001: Create Event

Attribute	Details
Actor	Faculty/Admin
Trigger	User initiates event creation
Preconditions	User has event creation privileges
Input	- Event title - Description - Date and time - Location - Target audience

Validation Steps	1. Verify user's event creation permissions 2. Check event title length and content 3. Validate date and time format 4. Ensure location information is complete 5. Validate target audience specification
Error Handling	- Insufficient permissions: Block event creation - Invalid event title: Prompt for correction - Incorrect date/time format: Provide clear formatting guidelines - Missing location details: Require complete information - Undefined target audience: Request clarification
Output	- Event created in system - Event listing - Notification to potential participants
Priority	High
Description	Create a new academic event
Main Flow	1. Select 'Create Event' 2. Enter event details 3. Set event parameters 4. Save and publish event
Alternative Flow	Can save as draft
Post Conditions	Event is added to platform's event list

UC-EM002: Event RSVP

Attribute	Details
Actor	Student/Faculty
Trigger	User selects event
Preconditions	User is logged in
Input	- Event ID - Attendee details - Additional requirements
Validation Steps	1. Verify event availability 2. Check user eligibility to attend 3. Validate attendee information 4. Verify event capacity 5. Check for any event-specific requirements
Error Handling	- Event already full: Waitlist option - User not eligible: Explain restrictions - Incomplete attendee details: Prompt for missing information - Event registration closed: Provide clear reason - Conflicting event registrations: Show potential conflicts

Output	- RSVP confirmation - Updated event attendee list - Calendar sync notification
Priority	Medium
Description	Register for an academic event
Main Flow	1. Browse available events 2. Select desired event 3. Click RSVP 4. Confirm attendance
Alternative Flow	Can withdraw RSVP
Post Conditions	User's attendance is tracked and confirmed

Community Platform Use Cases

UC-CP001: Create Community

Attribute	Details
Actor	Student/Faculty
Trigger	User initiates community creation
Preconditions	User is logged in
Input	- Community name - Description - Category - Access rules - Initial members
Validation Steps	1. Check community name uniqueness 2. Validate description length 3. Verify category selection 4. Check initial member invitations 5. Validate access rule configuration
Error Handling	- Duplicate community name: Suggest alternative names - Insufficient description: Prompt for more details - Invalid category: Provide category selection guidance - Invalid member invitations: Highlight invitation errors - Inappropriate access rules: Provide configuration recommendations
Output	- New community created - Community invitation notifications - Community dashboard
Priority	Medium
Description	Establish a new interest-based community

Main Flow	1. Select 'Create Community' 2. Define community parameters 3. Set access rules 4. Invite members
Alternative Flow	Can set community as public/private
Post Conditions	New community added to platform

UC-CP002: Peer Assistance

Attribute	Details
Actor	Student
Trigger	User posts help request
Preconditions	User is part of a community
Input	- Problem description - Relevant course/subject - Attachment (optional)
Validation Steps	1. Verify community membership 2. Check problem description completeness 3. Validate attachment file type and size 4. Ensure appropriate content guidelines 5. Check user's assistance request history
Error Handling	- Not a community member: Prompt to join community - Insufficient problem description: Request more details - Invalid attachment: Provide file type and size guidelines - Inappropriate content: Block submission, provide guidelines - Excessive help requests: Implement request limitation
Output	- Community responses - Solution thread - Potential mentor notification
Priority	High
Description	Request help on academic tasks
Main Flow	1. Post help request 2. Community members respond 3. Collaborate on solution
Alternative Flow	Can escalate to mentor if needed
Post Conditions	Academic challenge addressed through peer support

Dashboard Use Cases

UC-DB001: Personalized Dashboard

Attribute	Details
Actor	Student
Trigger	User logs in or refreshes dashboard
Preconditions	User is logged in
Input	- User authentication - Current academic data - Recent activities
Validation Steps	1. Verify user authentication token 2. Check data retrieval permissions 3. Validate academic data completeness 4. Verify activity log accuracy 5. Ensure data privacy compliance
Error Handling	- Invalid authentication: Require re-login - Insufficient data retrieval permissions: Limit dashboard view - Incomplete academic data: Prompt for updates - Inconsistent activity log: Flag for review - Privacy rule violations: Redact sensitive information
Output	- Comprehensive dashboard view - Progress metrics - Recent notifications - Recommended actions
Priority	Critical
Description	View academic progress
Main Flow	1. Load user-specific dashboard 2. Display progress metrics 3. Show recent notifications
Alternative Flow	Can customize dashboard view
Post Conditions	User sees comprehensive academic overview

UC-DB002: Notification Management

Attribute	Details
Actor	Student/Faculty/Admin
Trigger	User modifies notification preferences
Preconditions	User account exists
Input	- Communication channels - Notification types - Frequency settings

Validation Steps	1. Verify communication channel availability 2. Check notification type compatibility 3. Validate frequency settings 4. Ensure user consent for notifications 5. Verify notification delivery mechanisms
Error Handling	- Unsupported communication channel: Provide alternative options - Invalid notification types: Offer selection guidance - Inappropriate frequency settings: Suggest optimal configurations - Lack of user consent: Explain notification policy - Delivery mechanism failures: Implement backup notification methods
Output	- Updated notification profile - Confirmation of changes - Test notification
Priority	Medium
Description	Manage platform notifications
Main Flow	1. Configure notification preferences 2. Select communication channels 3. Set notification types
Alternative Flow	Can mute specific notification types
Post Conditions	Personalized notification settings applied