Student Actor

Student Registers on System

Use Case Name:

- Name: Student Registration

Summary: This use case involves students creating an account within the esports verification program system. After entering their details, students must verify their registration using their school email addresses ending with ".edu."
Priority: High
Source: User feedback and system requirements document.
Dependency: None
Actors:

- Primary Actor: Student

Preconditions:

- The esports verification program website is accessible.
- The student does not have an existing account in the system.
- The student possesses a valid school email address ending with ".edu."

Flow of Events/Description of Main Sequence:

- Basic Flow:
    - Student navigates to the registration page.
    - Student enters personal details (name, student ID, school email, etc.).
    - Student creates a unique username and a secure password.
    - Student confirms the entered information.
    - System validates the data provided.
    - System generates an email verification link and sends it to the student's provided school email address.
    - Student receives the verification email and clicks the link to activate the account.
    - System confirms successful verification and displays a confirmation message.
- Alternative Flow: Existing Account:
    - If the entered email or student ID matches an existing account:

- System displays an error message indicating the need for a unique email and student ID.
- Student revises the entered information.

Exceptions:

- System encounters technical issues and fails to send the verification email.
- Student exits the registration process before clicking the verification link.

- **Security:** User registration data must be securely stored and encrypted. Email verification link should be valid for a limited time.

- **Usability:** The registration interface should be user-friendly and accessible on various devices.

Post Conditions:

- Student's account is created with the provided details and is verified via the email verification process.
- The student can log in to the esports program system using the created username and password.

Student Submits Academic Credentials

Use Case Name:

- Name: Student Submits Academic Credentials

Summary: This use case involves students submitting their academic credentials, such as GPA and full-time status, to verify their eligibility for the esports program.
Priority: High
Source: User feedback and academic eligibility criteria.
Dependency:

- This use case depends on the successful completion of the "Student Registration" use case.

Actors:

- Primary Actor: Student

Preconditions:

- The student is logged into the esports program system.
- The student has completed the registration process.
- The academic eligibility section is accessible.

Flow of Events/Description of Main Sequence:

- Basic Flow:
  - Student navigates to the academic eligibility section.
  - Student enters academic details, including GPA and full-time enrollment status.
  - Student uploads necessary documents, such as transcripts or enrollment certificates.
  - System validates the provided data and documents.
  - System updates the student's eligibility status based on the submitted credentials.
- Alternative Flow: Invalid Credentials:
  - If the provided academic credentials do not meet the eligibility criteria:
    - System displays an error message indicating the need for valid academic credentials.
    - Student revises the entered information and resubmits the documents.

Exceptions:

- Student submits incomplete or invalid documents.
- System encounters technical issues while processing the documents.

Nonfunctional Requirements:

- Accuracy: The system must accurately validate the submitted academic credentials.
- Performance: The system should process documents efficiently, especially during peak submission times.
- Usability: The interface for document submission should be intuitive and user-friendly.

Post Conditions:

- Student's academic credentials are verified, and their eligibility status is updated in the system.

Student Logs Into System

Use Case Name: Student Logs In

Summary: This use case involves students logging into the esports program system using their registered username and password.

Priority: High

Source: User requirements and system design specifications.

Dependency: This use case depends on the successful completion of the "Student Registration" use case.

Actors:

- Primary Actor: Student

Preconditions:

- The student has successfully registered an account in the system.

Flow of Events/Description of Main Sequence:

Basic Flow:

- Student navigates to the login page.
- Student enters their username and password.
- System verifies the entered credentials.
- System grants access to the student, and the student is redirected to the system's main dashboard.

Alternative Flow: Invalid Credentials:

- If the entered username or password is incorrect:
    - System displays an error message indicating the need for valid credentials.
    - Student revises the entered information and resubmits.

Exceptions:

- Student forgets their password and initiates the password recovery process.

Nonfunctional Requirements:

- Security: Passwords must be securely stored and protected from unauthorized access.
- Usability: The login interface should be user-friendly and responsive.

Post Conditions:

- The student gains access to the esports program system upon successful login.

Student Views Tournament Information

Use Case Name: Student Views Tournament Information

Summary: This use case involves students accessing information about upcoming esports tournaments, including dates, game titles, requirements, prizing, and registration details.

Priority: High

Source: User requirements and esports tournament organizers' announcements.

Dependency: The student must be logged into the system to access tournament information.

Actors:

- Primary Actor: Student

Preconditions:

- The student is logged into the esports program system.
- Tournament information is available and up-to-date in the system.

Flow of Events/Description of Main Sequence:

Basic Flow:

- Student navigates to the "Tournaments" or "Events" section of the system.

- System displays a list of upcoming tournaments, including details such as dates, game titles, venues, and registration deadlines.
- Student selects a specific tournament to view more detailed information.
- System shows additional details about the selected tournament, such as rules, prizes, and registration instructions.

Alternative Flow: No Upcoming Tournaments:

- If there are no upcoming tournaments available:
  - System displays a message indicating that there are currently no tournaments scheduled.
  - Student can choose to receive notifications when new tournaments are announced.

Exceptions:

- Student encounters technical issues while accessing tournament information.
- Tournament details are updated in real-time, and the system might be temporarily unavailable during updates.

Nonfunctional Requirements:

- Performance: The system should load tournament information quickly, even during peak usage times.
- Availability: The tournament information should be accessible 24/7, with minimal downtime for updates.

Post Conditions:

- The student has access to up-to-date information about upcoming esports tournaments.

School Administrator Actor

Administrator Registers for an Account

Use Case Name: Administrator Registers for an Account

Summary: This use case involves a school administrator registering for an account within the esports program system. The administrator provides necessary details to create an account, including school information and personal contact details.

Priority: High

Source: User requirements and system registration policies.

Dependency: None

Actors:

- Primary Actor: School Administrator

Preconditions:

- The administrator has access to a valid school email address.
- The esports program website is accessible.

Flow of Events/Description of Main Sequence:

Basic Flow:

- Administrator navigates to the registration page.
- Administrator enters school details, including school name, address, and official school email address ending with ".edu."
- Administrator creates a unique username and a secure password.
- Administrator confirms the entered information.
- System validates the data provided.
- System generates a confirmation message, indicating successful registration.

Alternative Flow: Existing Account:

- If the entered email or school name matches an existing account:
    - System displays an error message indicating the need for a unique email and school name.
    - Administrator revises the entered information.

Exceptions:

- System encounters technical issues and fails to validate the registration data.
- Administrator exits the registration process before completing all required fields.

Nonfunctional Requirements:

- Security: User registration data must be securely stored and encrypted.
- Usability: The registration interface should be user-friendly and accessible on various devices.

Post Conditions:

- Administrator's account is created with the provided details.
- The administrator can log in to the esports program system using the created username and password.

Administrator Adds Verified School

Use Case Name: Administrator Adds Verified School

Summary: This use case involves the school administrator, after registering for an account, adding their school to the verified list of schools within the esports program system. Verification ensures that the school is eligible to participate in esports activities.

Priority: High

Source: Administrator's verification request and school verification policies.

Dependency: The administrator must be registered and logged into the system.

Actors:

- Primary Actor: School Administrator

Preconditions:

- The administrator is logged into the esports program system.

- The administrator's account is successfully registered in the system.

Flow of Events/Description of Main Sequence:

Basic Flow:

- Administrator navigates to the "Add Verified School" section in the system.
- Administrator enters school details, including school name, address, official school email address, and proof of eligibility.
- Administrator submits the verification request.
- System validates the submitted information.
- Upon successful verification, the school is added to the verified list of schools.

Alternative Flow: Rejected Verification:

- If the submitted information does not meet the verification criteria:
  - System notifies the administrator about the rejection and provides specific reasons.
  - Administrator can revise the submitted information and resubmit the verification request.

Exceptions:

- System encounters technical issues while processing the verification request.
- Administrator exits the verification process before completing all required fields.

Nonfunctional Requirements:

- Accuracy: The verification process must accurately validate the submitted school information.
- Timeliness: The system should process verification requests in a timely manner to minimize delays for eligible schools.

Post Conditions:

- The school is added to the verified list of schools within the esports program system.

School Administrator Manages Student Accounts

Use Case Name: School Administrator Manages Student Accounts

Summary: This use case involves the school administrator managing student accounts within the esports program system, including adding new students, updating student information, and deactivating student accounts when necessary. The administrator must first register for an account and verify their school's eligibility before managing student accounts.

Priority: High

Source: User requirements and school administrative policies.

Dependency: The school administrator must be registered, logged into the system, and their school must be verified.

Actors:

- Primary Actor: School Administrator

Preconditions:

- The administrator is logged into the esports program system.
- The administrator's account is successfully registered and verified in the system.
- Student information is up-to-date in the system.

Flow of Events/Description of Main Sequence:

Basic Flow:

- School administrator accesses the "Manage Students" or similar section in the system.
- System displays a list of registered students with relevant details.
- School administrator can add a new student by entering necessary information.
- School administrator can update existing student details such as name, student ID, or academic status.
- School administrator can deactivate a student account if the student leaves the school or is no longer eligible for the esports program.

Alternative Flow: Bulk Student Upload:

- If the school administrator needs to add multiple students at once:
    - School administrator uploads a CSV or Excel file containing student details.
    - System processes the file and adds new student accounts accordingly.

Exceptions:

- School administrator encounters technical issues while managing student accounts.
- Attempting to update a student account that does not exist in the system.
- Attempting to deactivate an account that is already deactivated.

Nonfunctional Requirements:

- Usability: The interface for managing student accounts should be intuitive and user-friendly.
- Data Integrity: The system should prevent duplicate student accounts and ensure data accuracy.

Post Conditions:

- Student accounts are managed and updated as per the school administrator's actions.

Tournament Organizer Actor

Tournament Organizer Registers for an Account

Use Case Name: Tournament Organizer Registers for an Account

Summary: This use case involves a tournament organizer registering for an account within the esports verification program system. The organizer provides necessary details to create an account, enabling them to create and manage esports tournaments for participating schools and students.

Priority: High

Source: User requirements and system registration policies.

Dependency: None

Actors:

- Primary Actor: Tournament Organizer

Preconditions:

- The tournament organizer has access to a valid email address.
- The esports program website is accessible.

Flow of Events/Description of Main Sequence:

Basic Flow:

- Organizer navigates to the registration page.
- Organizer enters personal details, including name, email address, and a secure password.
- Organizer confirms the entered information.
- System validates the data provided.
- System generates a confirmation message, indicating successful registration.

Alternative Flow: Existing Account:

- If the entered email matches an existing account:
  - System displays an error message indicating the need for a unique email.
  - Organizer revises the entered information.

Exceptions:

- System encounters technical issues and fails to validate the registration data.
- Organizer exits the registration process before completing all required fields.

Nonfunctional Requirements:

- Security: User registration data must be securely stored and encrypted.
- Usability: The registration interface should be user-friendly and accessible on various devices.

Post Conditions:

- Organizer's account is created with the provided details.
- The organizer can log in to the esports program system using the created username and password.

Tournament Organizer Creates a Tournament

Use Case Name: Tournament Organizer Creates a Tournament

Summary: This use case involves a tournament organizer creating a new esports tournament within the system. The organizer provides all necessary details for the tournament, including game title, date, venue, rules, and registration requirements. Once created, the tournament is made available to participating schools and students.

Priority: High

Source: User requirements and tournament organization policies.

Dependency: The tournament organizer must be registered and logged into the system.

Actors:

- Primary Actor: Tournament Organizer

Preconditions:

- The tournament organizer is logged into the esports program system.
- The organizer's account is successfully registered in the system.
- Tournament management section is accessible and up-to-date.

Flow of Events/Description of Main Sequence:

Basic Flow:

- Organizer navigates to the "Create Tournament" or similar section in the system.

- Organizer enters tournament details, including tournament name, game title, date, venue, rules, registration requirements, and maximum number of participants.
- Organizer sets tournament-specific rules, such as match format, scoring, and prize information.
- System validates the entered information.
- Organizer submits the tournament for approval and publication.

Alternative Flow: Editing Tournament Details:

- If the organizer needs to edit tournament details after submission:
  - Organizer accesses the list of created tournaments.
  - Organizer selects the desired tournament to edit and modifies the necessary information.
  - Organizer saves the changes, and the system updates the tournament details.

Exceptions:

- Organizer encounters technical issues while creating or editing the tournament.
- Attempting to create a tournament without filling out mandatory fields.
- Attempting to edit a tournament that is already in progress.

Nonfunctional Requirements:

- Performance: The system should handle a large number of tournaments efficiently, especially during peak usage times.
- Usability: The interface for creating and editing tournaments should be intuitive and user-friendly.

Post Conditions:

- The esports tournament is created and made available to participating schools and students.
- Participants can view and register for the tournament through the system.

Tournament Organizer Manages Tournament Participants

Use Case Name: Tournament Organizer Manages Tournament Participants

Summary: This use case involves the tournament organizer managing participants registered for an esports tournament. The organizer can view the list of registered teams or individual players, approve or reject registrations, and communicate important tournament-related information to the participants.

Priority: High

Source: User requirements and tournament organization policies.

Dependency: The tournament organizer must be registered, logged into the system, and have created or been assigned as the organizer for a tournament.

Actors:

- Primary Actor: Tournament Organizer

Preconditions:

- The tournament organizer is logged into the esports program system.
- The organizer's account is successfully registered in the system.
- A tournament is created or assigned to the organizer.

Flow of Events/Description of Main Sequence:

Basic Flow:

- Organizer accesses the management dashboard for the specific tournament.
- System displays a list of registered participants (teams or individual players) with their details.
- Organizer can review registration information, including team names, player names, and contact information.
- Organizer can approve registrations, marking them as eligible for participation.
- Organizer can reject registrations, providing reasons for rejection if necessary.
- Organizer can send notifications to approved participants, informing them about tournament details, schedules, and rules.

Alternative Flow: Communication with Participants:

- Organizer composes and sends messages to individual participants or all registered teams/players.

- Participants receive notifications in their accounts or via registered email addresses.

Exceptions:

- Organizer encounters technical issues while managing participant registrations.
- Attempting to approve or reject a participant without valid reasons.
- Attempting to send notifications to participants with invalid email addresses.

Nonfunctional Requirements:

- Usability: The interface for managing participants should be intuitive and provide easy access to participant details.
- Communication: The notification system should be reliable and deliver messages promptly to participants.

Post Conditions:

- Participant registrations are managed as per the organizer's actions.
- Participants receive necessary information about the tournament and their eligibility status.