# Part A

# Smart Sphere - Project Team Information

# **1. Team Introduction**

* **Team Name:** SmartSphere
* **Team Logo:**



* **Group Picture:**   
  

# **2. Team Lead**

* **Team Lead:** Haider Zia

# **3. Team Members & Biographies**

### **Ibtisam Shahzad**

* **GitHub:** [ibtisam313](https://github.com/ibtisam313)

### 

### 

### 

### 

**Biography:** Ibtisam Shahzad is a Computer Science student at FAST-NUCES Islamabad, currently in his 6th semester. Skilled in C++, Java, and Python, he has worked on projects like a chess game and a desktop application. Passionate about software development, he continuously improves his problem-solving skills through hands-on projects.

### **Ibrahim Asim**

* **GitHub:** [**https://github.com/ibby-exe**](https://github.com/ibby-exe)



**Biography:** I am currently pursuing a Bachelor's degree in Computer Science at FAST NUCES Islamabad, where I am in my 6th semester. Throughout my academic journey, I have developed projects in C++, C, Assembly, and Java, including games like Tetris, Pacman, and Space Shooter, as well as systems like Smart Traffic Management and University Portal. I have also gained foundational knowledge in Python and JavaScript. Prior to my university education, I completed my O Levels at Garrison Academy Multan and Intermediate studies at Nishat College of Science Multan.

### **Haider Zia**

* **GitHub:** [RGRIDER.github.io](https://github.com/RGRIDER/RGRIDER.github.io)
* 
* **Biography:** I am Haider Zia. I have done my FSC from Cadet College Palandri. Currently I am at 6th semester doing bachelors in Computer Science. I have a grip on programming languages like python, C++, java, html, CSS. I have done several projects involving game development and management systems.

## **4. Team Roles**

|  |  |
| --- | --- |
| **Name** | **Role** |
| **Ibtisam Shahzad** | Requirement Analyst/Architect |
| **Ibrahim Asim** | Developer |
| **Haider Zia** | Scrum Master |

## **5. Team Agreement**

### **Methods of Communication**

* **Primary Communication:** WhatsApp group chat
* **Email:** For formal documentation and assignments
* **Phone Calls:** For urgent discussions
* **Trello:** For task tracking
* **GitHub Discussions:** For version control-related discussions

### **Communication Response Times**

* **WhatsApp:** Response within **2 hours** (9 AM - 10 PM)
* **Email:** Response within **12 hours**
* **Phone Calls:** Return within **2 hours**
* **Trello Updates:** At least **once every two days**

### **Meeting Attendance**

* **Face-to-Face Meetings:** Scheduled **Monday to Thursday** on campus
* **Online Meetings:** Can be held at any time based on availability
* **Sprint Meetings:** Mandatory at the start and end of each sprint
* **Emergency Meetings:** 24-hour prior notice required

### **Running Meetings**

* **Format:** Online via Zoom/Google Meet or in-person on campus
* **Agenda:** Shared **6 hours before** each meeting
* **Roles:**
  + **Facilitator:** Rotates weekly
  + **Minutes Recorder:** Documents key points
  + **Timekeeper:** Keeps the meeting on schedule

**Meeting Preparation**

* Review the agenda and assigned tasks before meetings
* Share discussion points **6 hours before** meetings
* Go through previous meeting notes

### **Version Control**

* **Repository:** GitHub for version control
* **Commit Guidelines:**
  + Clear and descriptive commit messages
  + Use branches for different features or bug fixes
  + No direct commits to **main branch**, changes go through pull requests
* **What to Commit:** Code files, documentation
* **What Not to Commit:** Large media files, compiled binaries, sensitive data

### **Division of Work**

* Work is divided based on **skill sets and interest**
* Tasks assigned through Trello
* Disagreements resolved by **majority vote**
* **Stakeholders:**
  + **Team Lead:** Oversees progress
  + **Developers:** Implement features
  + **Tester:** Ensures quality
  + **UI Designer:** Handles user experience

### **Submitting Assignments**

* **Deadline:** 24 hours before the official deadline
* **Review Process:**
  + One member compiles the final document
  + Another member reviews for errors before submission
* **Submission Responsibility:** Assigned **team lead** submits the final version

### **Contingency Planning**

* **If a Team Member Drops Out:** Work redistributed; instructor notified
* **If a Member Misses Meetings:**
  + Warning after **two absences**
  + Escalation to the instructor if continued
* **If a Team Member is Academically Dishonest:**
  + Exclude their work
  + Report incident to instructor
  + Implement preventive measures

# Part C:

# **Deliverable 1 - SmartSphere Event Management System**

## **1. Introduction**

The **SmartSphere Event Management System** is designed to streamline the organization and execution of NASCON events. This system aims to enhance user experience for participants, organizers, and administrators by automating key event management processes.

## 

## **2. Project Vision**

Our vision is to provide an intuitive and robust platform that simplifies event registration, scheduling, communication, and management. By digitalizing these processes, we aim to eliminate manual inefficiencies and ensure seamless execution of NASCON events.

## **3. Intended Use of the System**

### **Who Will Use the System?**

* **Participants:** To register, enroll, and receive updates on events.
* **Organizers:** To create, modify, and manage event logistics.
* **Admins:** To oversee the platform, approve enrollments, and generate reports.

### **Identified Stakeholders & Their Needs**

|  |  |
| --- | --- |
| **Stakeholder** | **Needs** |
| Participants | Easy registration, enrollment, notifications, event schedule visibility |
| Organizers | Event creation, participant management, real-time updates |
| Admins | System oversight, user authentication, reporting, and analytics |

## **4. Features & Overall Functionality**

### **System Capabilities**

* User authentication and role-based access control.
* Event creation, modification, and deletion by organizers.
* Participant registration and event enrollment.
* Secure payment processing and invoice generation.
* Venue and schedule management.
* Email and SMS notifications.
* Reporting and analytics dashboard for admins.

### 

### **How It Helps Users**

* **Participants:** Ensures smooth event registration and communication.
* **Organizers:** Reduces manual workload and improves event logistics.
* **Admins:** Enables efficient monitoring and management of NASCON events.

## **5. User Stories**

#### **User Stories for Sprint 1**

##### **User Story 1: User Registration**

* **As a** new user **I want** to create an account **So that** I can enroll in events and manage my profile.  
  **Sub-User Stories:**
  + Validate user input (email, password, name).
  + Store user credentials securely.
  + Send a confirmation email upon successful registration.

##### **User Story 2: User Login**

* **As a** registered user **I want** to log in securely **So that** I can access my account and enroll in events.  
  **Sub-User Stories:**
  + Implement secure authentication with encrypted passwords.
  + Provide a "Forgot Password" feature for account recovery.
  + Maintain session management for security.

##### **User Story 3: Event Enrollment**

* **As a** registered user **I want** to enroll in events **So that** I can participate in NASCON activities.  
  **Sub-User Stories:**
  + Display a list of available events with details.
  + Implement event selection and confirmation process.
  + Send an email notification after successful enrollment.

##### **User Story 4: Admin Approval for Enrollment**

##### **As an** admin, **I want** to review and approve enrollments **So that** I can ensure only eligible participants join the events. **Sub-User Stories:**

* + Display pending enrollments in an admin panel.
  + Provide approve/reject functionality.
  + Notify users of approval or rejection via email.

## **6. Structured Specifications**

##### **1. User Registration Specification**

* **Input:** Name, email, password
* **Process:** Validate input, store securely, send confirmation email
* **Output:** User receives a confirmation and can log in

##### **2. User Login Specification**

* **Input:** Registered user enters email and password
* **Process:** Authenticate credentials, create session, redirect to dashboard
* **Output:** User is logged in successfully

##### **3. Event Enrollment Specification**

* **Input:** User selects an event and confirms enrollment
* **Process:** Validate eligibility, update database, send confirmation email
* **Output:** User receives event enrollment confirmation

##### **4. Admin Approval Specification**

* **Input:** Admin views pending enrollments
* **Process:** Admin approves/rejects applications, system updates status, email notification sent
* **Output:** Users receive confirmation of approval or rejection

## 

## 

## 

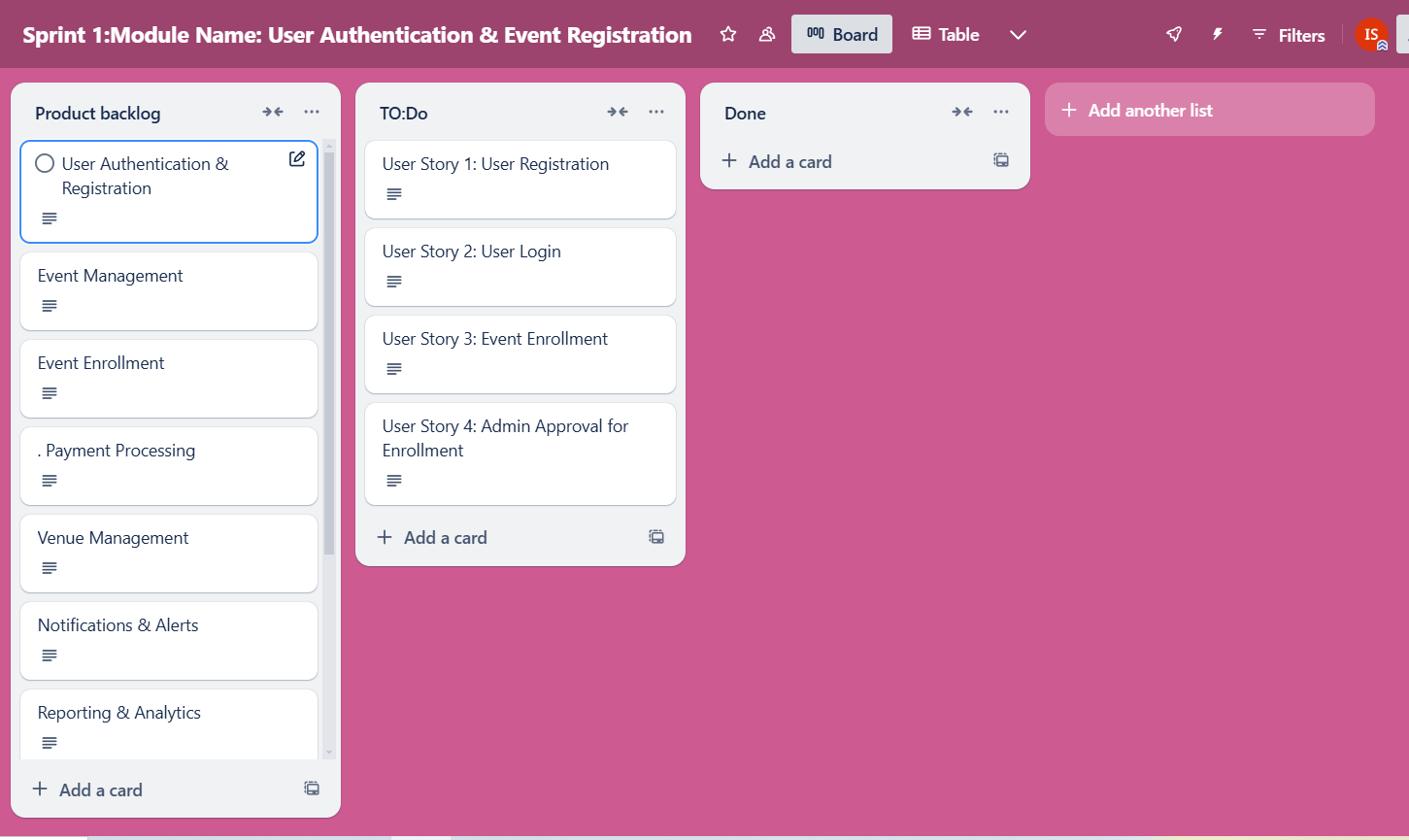
## 

## 

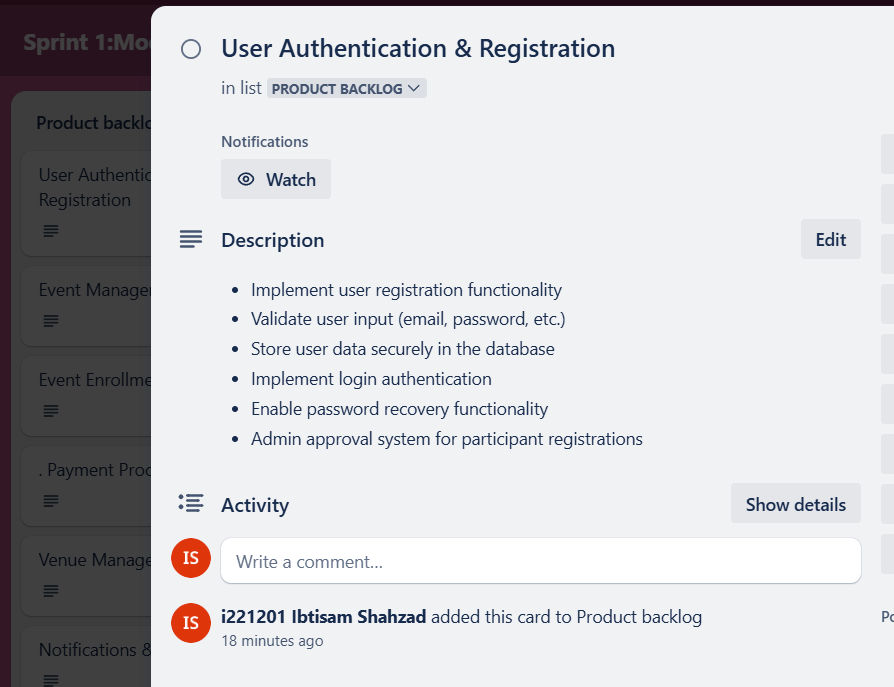
## **7. Scrum Board**

* **Snapshot 1:**

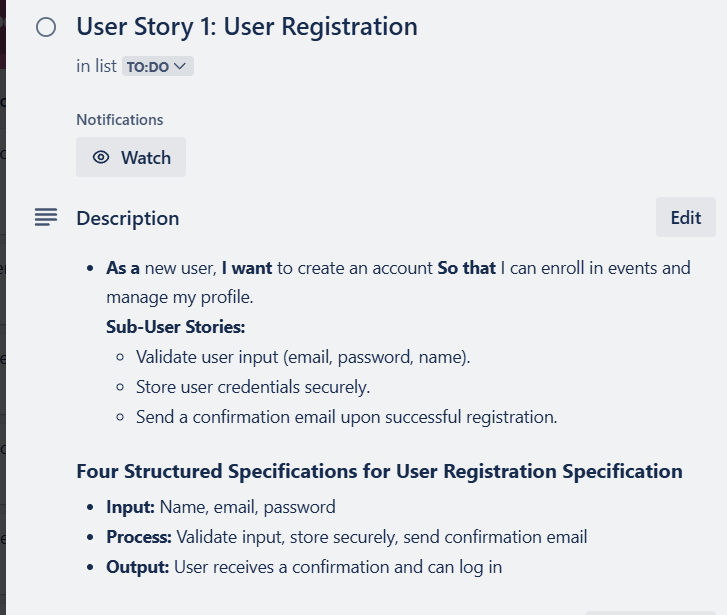
**Overall scrum Board:**



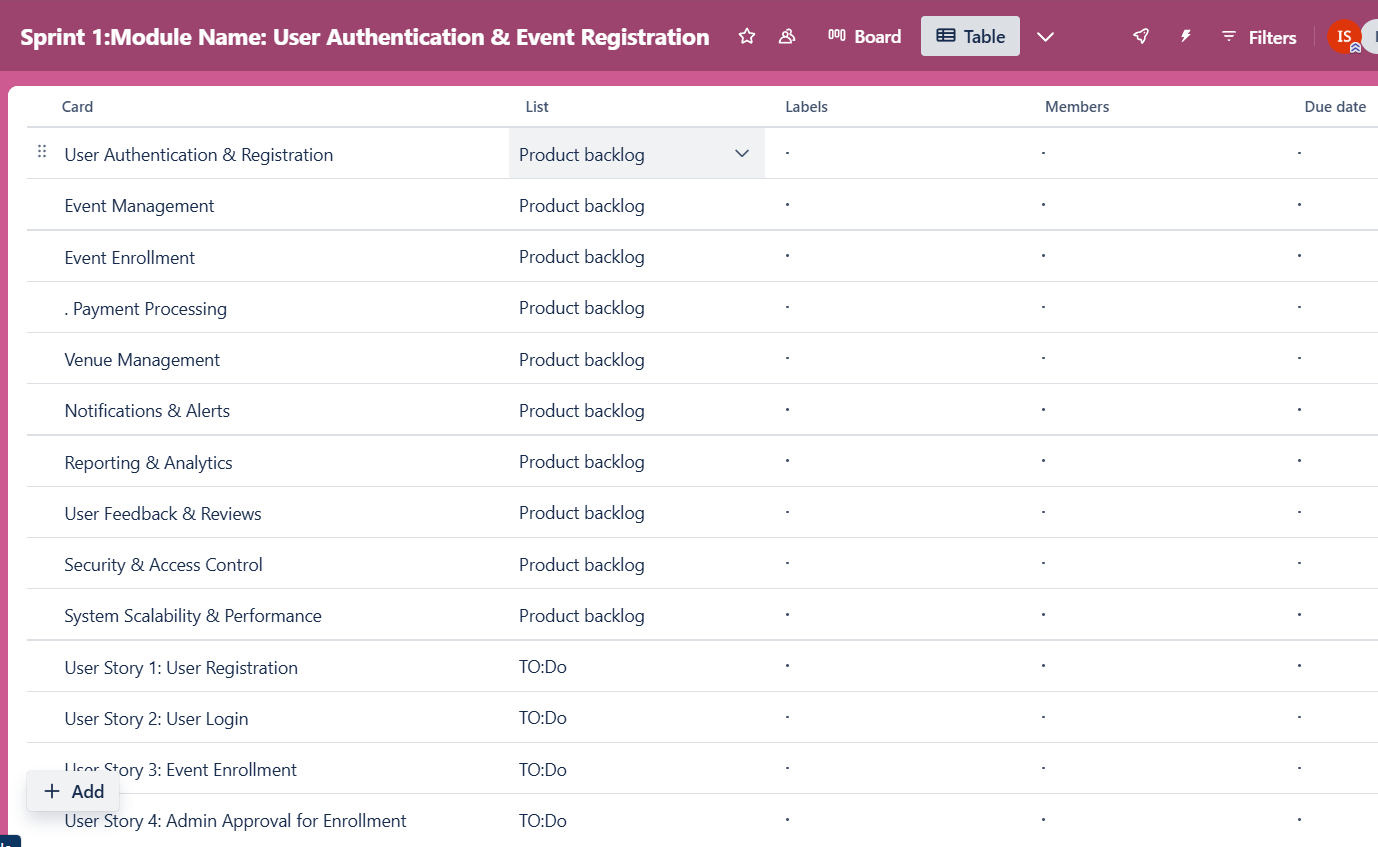
**Product backlog:**



**User Story:**



**Sprint Table:**



* **Snapshot 2:** Mid-sprint progress not done
* **Snapshot 3:** Completed tasks not done any

## **8. Non-Functional Requirements (NFRs)**

* **Performance:** System should handle concurrent users without lag.
* **Scalability:** Supports multiple events and large participant base.
* **Security:** Ensures encrypted data storage and secure authentication.
* **Usability:** Intuitive UI for easy navigation by all user types.