



PES UNIVERSITY, BANGALORE

Department of Computer Science and Engineering

## **Software Requirements Specification (SRS) for Student Event Management System**

Shreya Joshi PES2UG21CS501

Spoorthi Shivaprasad PES2UG21CS536

SECTION: I

SEM: V

### **Table Of Contents**

Sl.No.	Topic	Pg.No.
1.	Problem Statement	2
2.	Analysis Models 2.1 ER Diagram 2.2 Relational Schema	
3.	Implementation Details 3.1 User Interface 3.2 User Requirements	

### **1.Problem Statement:**

Clubs often struggle to manage their event databases effectively. This can be due to a number of factors, such as:

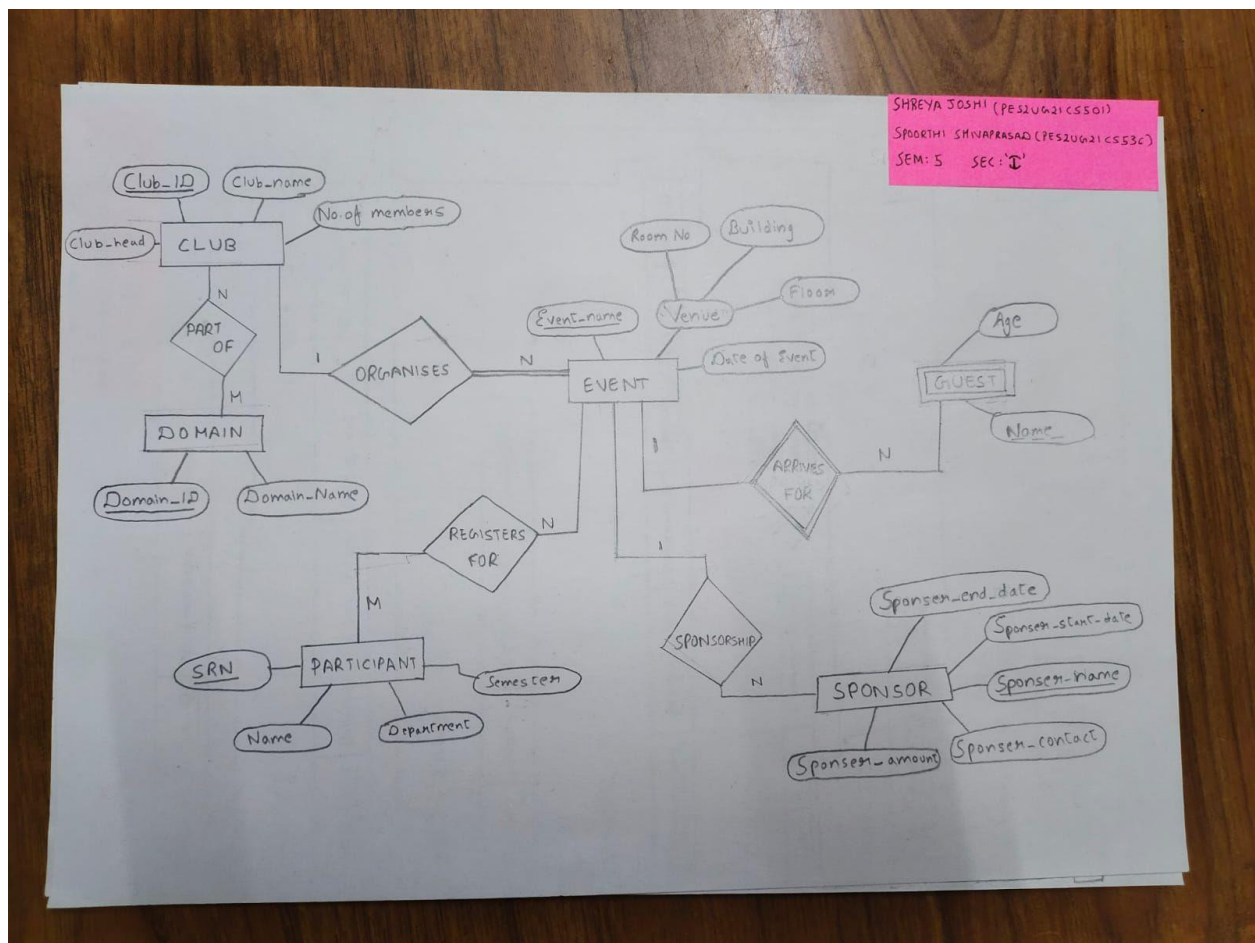
- Limited resources: Clubs often have limited resources, including time and money, to devote to managing their databases.
- Lack of expertise: Many club members may not have the necessary expertise to manage a database effectively.
- Complex requirements: Club databases can be complex, with a variety of different data types and relationships.

As a result of these challenges, many clubs end up with inaccurate or incomplete event databases. This can make it difficult for clubs to keep track of their events, communicate with their members, and manage their resources effectively.

A solution to this problem is to develop a user-friendly event database management system that is specifically designed for clubs to manage events. The system should be easy to use and require minimal expertise to manage and it should be flexible enough to meet the needs of a variety of different clubs and their events.

## 2. Analysis Models:

### 2.1 ER Diagram



Formalized paragraph for documentation:

Our database schema for events and clubs consists of six entities:

Strong entities:

- Club: club\_id (primary key), club\_name, no\_of\_members, club\_head
- Domain: domain\_id (primary key), domain\_name
- Event: event\_name, date\_of\_event, venue (composite attribute: room\_no, building, floor)
- Sponsor: sponsor\_name (primary key), sponsor\_start\_date, sponsor\_end\_date, sponsor\_contact
- Participant: srn (primary key), name, department, semester

Weak entity:

- Guest: name(partial key),age

The following are the relationships between the entities in the database schema for events and clubs:

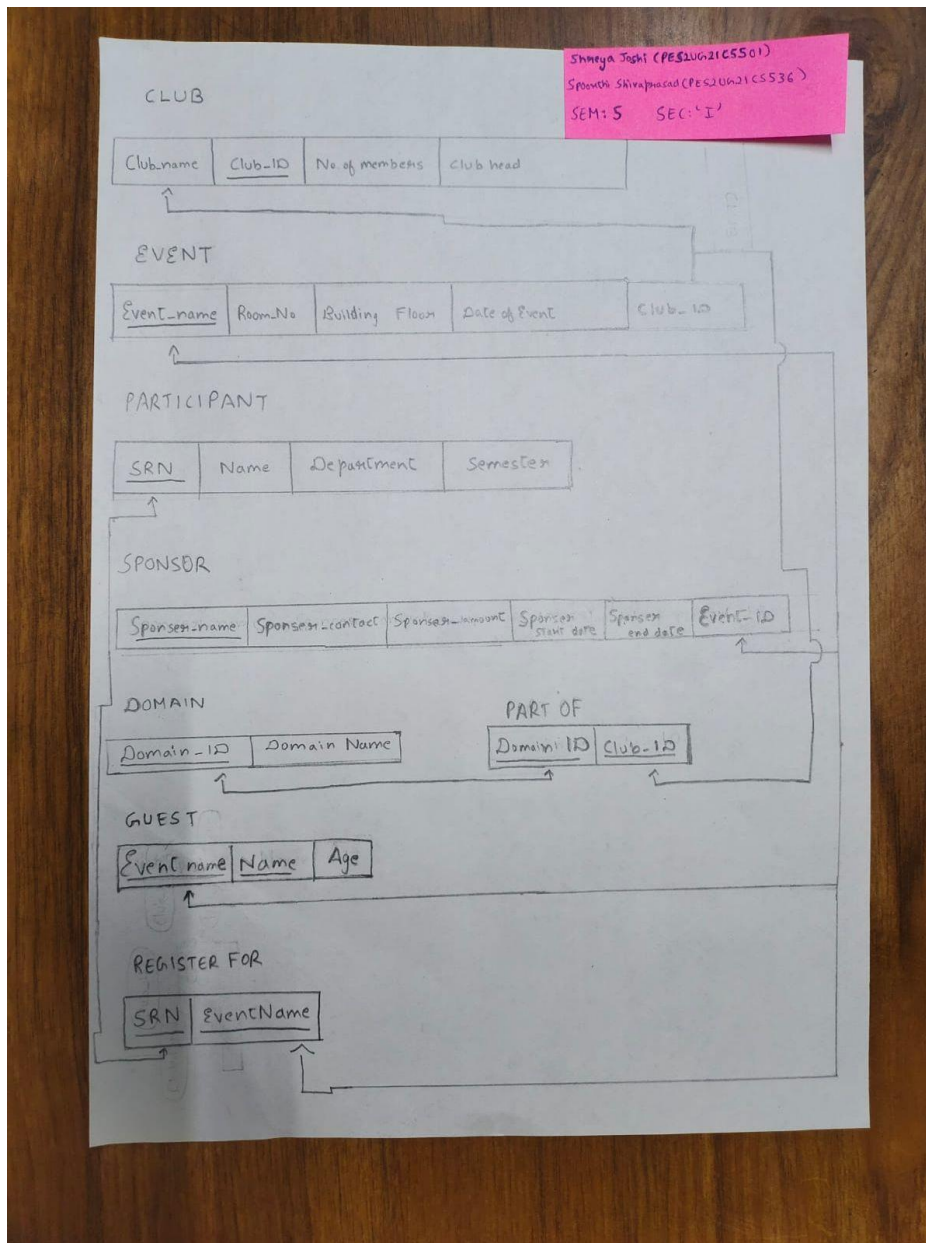
- **Club and Domain:** Many-to-many relationship. A club can belong to many domains, and a domain can have many clubs.
- **Club and Event:** One-to-many relationship. A club can organize many events, but an event can only be organized by one club hence its participation in the relationship is total.
- **Event and Participant:** Many-to-many relationship. An event can have many participants, and a participant can attend many events.
- **Event and Sponsor:** Many-to-many relationship. An event can have many sponsors, and a sponsor can sponsor many events.
- **Event and Guest:** One-to-many relationship. An event can have many guests, but a guest can only attend an event if it exists and hence it is a weak entity dependent on the existence of the event entity.

The relationship between `Event` and `Participant` is total participation on the part of `Event`. This means that every event must have at least one participant.

The relationships between `Club` and `Domain`, `Event` and `Sponsor`, and `Event` and `Guest` are all many-to-many relationships. This means that a club can belong to many domains, an event can have many sponsors, and an event can have many guests.

## 2.2

### Relational Schema:



The relational schema for events and clubs consists of eight relations:

Strong entities:

- `Club` (`club_id` (primary key), `no_of_members`, `club_name`, `club_head`)
- `Domain` (`domain_id` (primary key), `domain_name`)
- `Event` (`event_name` (primary key), `room_no`, `building`, `floor`, `date_of_event`, `club_id`(foreign key to the `Club` table))
- `Participant` (`srn`(primary key), `name`, `department`, `semester`)
- `Sponsor` (`sponsor_name` (primary key), `sponsor_contact`, `sponsor_start_date`, `sponsor_end_date`, `sponsor_amount`, `event_id`(foreign key to the `Event` table))

Weak entity:

- `Guest` (`name`(primary key), `age`, `event_id`(primary key))

Many-to-many relationships:

- `Register_for` (`srn`(primary key), `event_name`(primary key))
- `Part_of` (`domain_id`(primary key), `club_id`(primary key))

The `Event` entity is the N-side of a 1:N relationship with the `Club` entity. This means that an event can only be organized by one club, but a club can organize many events and hence the `club_id` foreign key in the `Event` table points to the `club_id` primary key in the `Club` table. This ensures that each event has a valid club associated with it.

The `Sponsor` entity is the N-side of a 1:N relationship with the `Event` entity. This means that a sponsor can sponsor many events, but an event can only be sponsored by one sponsor. The `event_id` foreign key in the `Sponsor` table points to the `event_id` primary key in the `Event` table. This ensures that each sponsorship has a valid event associated with it.

The `Guest` entity is a weak entity that depends on the `Event` entity. The partial key of the `Guest` entity, `name`, is combined with the primary key of the `Event` entity, `event_id`, to form the primary key of the `Guest` relation. In other words, a guest can only attend one event, and an event can have many guests. The `name` attribute uniquely identifies a guest within a particular event, but it may not be unique across all guests in the database. The `event_id` attribute uniquely identifies an event, and it is combined with the `name` attribute to uniquely identify a guest within the `Guest` relation.

The `Register_for` relation represents the many-to-many relationship between `Participant` and `Event`. A participant can register for many events, and an event can have many participants registered for it.

The `Register_for` relation has the following primary keys:

- `srn` (foreign key to the `Participant` table)
- `event_name` (foreign key to the `Event` table)

The `Part_of` relation represents the many-to-many relationship between `Domain` and `Club`. A domain can have many clubs, and a club can belong to many domains.

The `Part_of` relation has the following primary keys:

- `domain_id` (foreign key to the `Domain` table)
- `club_id` (foreign key to the `Club` table)

### **3. Implementation details:**

In this project, we will be developing a website for student event management. The website's homepage will have buttons to manage events, the sponsoring companies and manage users.

#### **3.1 User Interface:**

##### 1. XAMPP Control Panel:

- The XAMPP Control Panel is the starting point for managing the server environment. It has the options to start/stop Apache and MySQL services.

- The components of the XAMPP Control Panel include Start/Stop buttons for Apache and MySQL, and the server status indicators.

## 2. Apache Server Interface:

- Apache server configuration interface will allow users to set up virtual hosts, manage security settings, and configure server modules.
- The components for the Apache Server Interface include virtual host setup, security settings and configuration of server modules.

## 3. MySQL Database Management Interface:

- The MySQL interface will have tools for managing databases, tables, users, and privileges.
- The components include database creation, table management, user privileges, query editor.
- The interface consists of the following functions: Save, apply, reset, help.

## 4. Student Event Management System Web Application:

- This is the main interface that will be accessed through the web URL. It allows users to create and manage events, and view event details and sponsoring companies.
- It will include the event creation form, event list, event details page.
- The buttons/functions include create event, edit event, delete event and view event details.
- Error messages will be displayed using pop-up alerts.

### **3.2 User Requirements:**

#### **User Requirement 1: Event Creation**

**Description and Priority:** This feature will allow organizers to create new events, edit event details, and manage event participants. The priority of this system feature is High.



### Stimulus/Response Sequences:

- Organizer selects "Create New Event."
- Organizer enters event details (name, date, location, description)
- System will validate the input and create the event
- Organizers can view event details, edit information, or manage participants

### Functional Requirements:

- REQ-1: Organizer will fill a form to input event details
- REQ-2: Event details include name, date, location, organizing club, sponsoring company and description
- REQ-3: The system validates the input and stores the event information in the database

### User Requirement 2: Edit Event

Description: This feature will allow the user to edit the event details.

### Functional Requirements:

- REQ-1: The system displays the list of events for user to choose from
- REQ-2: The user chooses the event to be edited
- REQ-3: The user enters the new details for date, location, organizing club, sponsoring company and description
- REQ-4: The updated records are updated in the database

### User Requirement 3: View Events

Description: This feature allows the user to view the event details.

### Functional Requirements:

- REQ-1: User clicks on 'View Events' button

-REQ-2: A list of all the events and its details will be displayed

#### User Requirement 4: Delete Event

Description: This feature allows the user to delete the event details.

#### Functional Requirements:

-REQ-1: User clicks on 'Delete Event' button

-REQ-2: A list of all the events and its details will be displayed

-REQ-3: User selects the event to be deleted

-REQ-4: The selected event is removed from the database