# Software Requirements Specification (SRS)

# Software Requirements Specification (SRS)

## Project Title: **Virtual Event / Conference Manager**

Prepared for: [Your College Name / Tutor Name]

Prepared by: [Your Team Name / Members]

Date: [Insert Date]

## 1. Introduction

### 1.1 Purpose

The purpose of this project is to design and implement a **database system** for managing virtual events or conferences. This system will enable organizers to create events, schedule sessions, manage attendees, assign speakers, and process ticket sales. The deliverables include:

- Entity-Relationship Diagram (ERD)
- Normalized database schema
- SQL scripts for creating and populating the database
- Basic C# interface to interact with the database

### 1.2 Scope

The Virtual Event / Conference Manager system will support:

- Event and session management by organizers
- Registration and ticket purchase by attendees
- Speaker assignments for sessions
- Real-time attendee-session tracking

The system is focused on the **database layer** and **basic UI interaction** in C#. No advanced features like video streaming or payment processing are included in this version.

### 1.3 Definitions, Acronyms, and Abbreviations

- **ERD** - Entity Relationship Diagram
- **SQL** - Structured Query Language

- **ORM** - Object-Relational Mapping

- **CRUD** - Create, Read, Update, Delete

- **C#** - Programming language used for the front-end interface

---

## 2. Overall Description

### 2.1 Product Perspective

This is a stand-alone database-backed desktop application or small-scale web app. The C# interface connects to a MySQL or SQL Server database, providing CRUD functionality.

### 2.2 User Classes and Characteristics

- **Organizer:** Creates and manages events and sessions

- **Attendee:** Registers for sessions, purchases tickets

- **Speaker:** Assigned to sessions but does not interact directly with the system

- **Admin (optional):** Manages users and general settings

### 2.3 Assumptions and Dependencies

- Database system is MySQL or SQL Server

- C# application uses ADO.NET or Entity Framework

- Authentication and user roles are not fully implemented; forms are manually operated

---

## 3. Specific Requirements

### 3.1 Functional Requirements

#### FR1: Organizer Management

- FR1.1: Create, update, and delete organizer profiles

- FR1.2: Organizers can create and manage multiple events

#### FR2: Event and Session Management

- FR2.1: Each event can have multiple sessions
- FR2.2: Sessions are associated with a specific event and have time slots
- FR2.3: Assign speakers to sessions

#### FR3: Attendee Management

- FR3.1: Add new attendees
- FR3.2: Allow attendees to register for sessions
- FR3.3: Allow attendees to purchase tickets

#### FR4: Ticketing

- FR4.1: Events have multiple ticket types (e.g., General, VIP)
- FR4.2: Store purchase history including date and quantity

#### FR5: C# Interface

- FR5.1: Form to create and view events/sessions
- FR5.2: Form to register attendees and assign them to sessions
- FR5.3: Form to simulate ticket purchases

### 3.2 Non-Functional Requirements

- **NFR1: Usability** - Interface should be simple, form-based, and intuitive
- **NFR2: Maintainability** - Database should be normalized (3NF)
- **NFR3: Security** - Basic input validation on forms (e.g., required fields)
- **NFR4: Performance** - Efficient queries; minimal joins due to proper indexing
- **NFR5: Compatibility** - C# application must connect to the selected RDBMS

Software Requirements Specification (SRS)

## 4. Data Requirements

The system must store and retrieve the following types of data:

- Event info: title, description, date, organizer, location
- Session info: title, time, description, associated event
- Attendee info: name, contact, registered sessions
- Speaker info: name, contact, biography
- Ticket info: type, price, quantity sold

All data will be stored in relational tables as defined in the schema and will follow foreign key constraints to ensure integrity.

## 5. ER Diagram and Schema

- ER Diagram includes:
  `Organizer -< Event -< Session -< SpeakerSession >- Speaker`
  `Attendee -< AttendeeSession >- Session`
  `Event -< TicketType -< TicketSale >- Attendee`
- Schema and relationships are described in the accompanying SQL script.

## 6. Sample Interface Descriptions (C#)

- **EventForm.cs:** Form to create/edit event details
- **SessionForm.cs:** Select event, then add/edit sessions
- **AttendeeForm.cs:** Register attendees and assign sessions
- **TicketForm.cs:** View and purchase tickets

## 7. Future Enhancements (Optional)

- Authentication system for attendees and organizers
- Email notifications for session reminders
- Integration with Zoom/Teams APIs
- Payment gateway integration for paid tickets