

Settings

<https://chat.openai.com/share/e43f4705-c493-4c23-aa52-f41a30503858>

Date : January 27, 2024

Context

An SRS depicts the requirements that a piece of software should satisfy. There are 4 key traits of the requirements in an SRS:

1. Unambiguity: A requirement is unambiguous if and only if it has only one possible interpretation.
2. Understandability: A requirement is understandable if all classes of SRS readers can easily comprehend its meaning with a minimum of explanation.
3. Correctness: A requirement is deemed correct when it accurately represents a required feature or function the system must possess.
4. Verifiability: A requirement is verifiable if finite, cost-effective methods exist for verifying that it is satisfied by the system as built.

Given is the following template of an SRS:

1. Problem Background
2. Stakeholders
3. Functional Requirements
4. Performance Requirements
5. Design constraints
6. External Interfaces
7. Security Requirements
8. Glossary of terms.

How would you like ChatGPT to respond

Give a detailed and thorough response

Prompt

Given is the following task description for a university club management portal:

``There are four main types of users - the administrator, the student council coordinator, club coordinators, and students. The administrator serves as the central authority of the system with complete control and visibility of activities assigned to all system users. They can create, edit, and delete clubs and appoint the student council coordinator and club coordinators from within the registered students. The student council coordinator manages and approves events for which each club puts forward proposals. The club coordinators for each club can add or edit the club's information, and schedule events and club activities.

They can also manage the club members and the visibility and access for club events to the student body. All other university students should be able to view the schedule of the club events and their details and register for the same.'''

The following is an SRS for the same:

'''

SRS for Student Clubs Event Management Platform

1. Problem Background:

The primary purpose of this platform is to enable club coordinators to post requests for scheduling events, which would be subject to approval by the Student Council's Clubs Coordinator. Post this approval, the event would be added to the platform for registration by the student body, and an email notification would be sent to all the students for the same. The platform would also have dedicated pages for the clubs, which the respective club's coordinator would manage.

2. Stakeholders / Users:

The primary users of the platform are (i) students of the university, (ii) the coordinators of various clubs, (iii) the Student Council's Clubs Coordinator and (iv) a representative from the Dean of Student Affairs office (Admin).

3. Functional Requirements:

3.1. All IIITD domain users should be able to log into the platform using their IIITD domain Gmail IDs. (all)

3.2. The platform should enable the club coordinators to schedule events for any given date and time. While applying for the same, they should be permitted to edit the event details and registration requirements. (Club Coordinators)

3.3. The platform would also have dedicated pages for the clubs, which the respective club's coordinator would manage. A club coordinator should be allowed to edit this page by editing club information, images from past events and point of contact information. (Club coordinators)

3.4. The Student Council's Clubs Coordinator should be given access to a page that lists all pending requests for scheduling the events while marking potential conflicts and should be able to approve or decline each one of them. (SC Clubs coordinator)

3.5. After the event's approval, it should be added to the platform for registration by the student body, and an email notification should be sent to all the students. Students should be able to post comments regarding the event on the platform. (all)

3.6. Students should be allowed to search for upcoming events based on their interests and view them. (all)

3.7. The platform's admin access should be given to a representative from the Student Affairs office(Admin) who can edit event details, Club Coordinator and respective clubs' and students' accounts information. (Admin)

3.8. There should be a report feature in the platform to enable everyone to report incidents and issues with the platform. (all)

3.9. Students should be able to request for the formation of a new club. (all).

3.10. The approval for new clubs will be a two-stage process involving the approval of the Student Council Clubs Coordinator and then the admin. (SC Clubs Coordinator and Admin)

4. Performance Requirements:

4.1. The platform should withstand considerable amounts of traffic and frequent shifts in load and traffic coming its way.

4.2. The platform should be responsive and have an average response time of 300 milliseconds or less under normal load.

5. Design Constraints:

5.1. The platform should be developed while keeping accessibility for all users in mind and should have features like screen readers, font adjusters, etc.

5.2. The platform should be dynamic and should be able to adjust to varying screen resolutions and devices (like laptops, cellphones, tablets, etc.).

5.3. The platform should be easy to maintain, and the code should be readable and well-documented.

5.4. The tech stack for the platform should be the following:

5.4.1. React JS for the front-end development.

5.4.2. Node JS for the back-end development with MongoDB database.

6. External Interfaces:

6.1. The platform should be compatible with all major browsers like Safari, Chrome, Firefox, Brave, Microsoft Edge, etc.

6.2. The platform would use Google sign-in API to allow users to log in using their IIIT Delhi domain email address. Upon login, the users would be redirected to Google API's interface.

6.3. The platform should be able to send email notifications to all students and other users.

7. Security Requirements:

7.1. The platform should be well protected from known attacks and penetration attempts to enable the protection of user information and data, like XSS (cross-site scripting) and SQL injection.

7.2. The platform should prevent packet interception and tampering with outgoing and incoming requests.

7.3. The platform should verify users using 2-factor authentication and IIIT Delhi domain login should be implemented using Google's API.

7.4. The user data should be stored and maintained as per the institute's policy.

8. Glossary of Terms:

8.1. Users: Those who will use the application.

8.2. Students: Users who study at IIIT Delhi and are not part of the management of the university.

8.3. General students: Users who aren't part of the management. They are the primary stakeholders of the application.

8.4. Club coordinators: Users who are part of the management, albeit at a lower level. They are responsible for managing the club activities and hosting events.

8.5. Student council club coordinator: They are student users responsible for managing and approving the events added by club coordinators/heads.

- 8.6. DoSA: Department of Student Affairs at the university.
- 8.7. FMS: Facilities Management Services at IIT Delhi.
- 8.8. Admin: The user can change logins and credentials, and manage the application.
- 8.9. Login: Sign into the application and let it know your details.
- 8.10. Explore: Browse through the events given.
- 8.11. Registering: Marking that the user 'may' be present at the event.
- 8.12. Participation: Participating in an event, showing interest, and attending when it is organised.
- 8.13. Credentials: information required for logging in; in this case, it is the Google account associated with the institute.
- 8.14. Request: It is for requesting allowance to conduct an event.

``

Your task is to determine if each requirement conforms to the 4 major qualities of a good requirement: unambiguousness, understandability, correctness, and verifiability. Evaluate each requirement on a scale of 1 to 5, with higher numbers indicating better conformance to the quality. Please do so for every requirement in

- Functional Requirements
- Performance Requirements
- Design Constraints
- External Interfaces
- Security Requirements

Outputs

Rating table – provided as a CSV

Chat

Given is the following task description for a university club management portal:

“There are four main types of users - the administrator, the student council coordinator, club coordinators, and students. The administrator serves as the central authority of the system with complete control and visibility of activities assigned to all system users. They can create, edit, and delete clubs and appoint the student council coordinator and club coordinators from within the registered students. The student council coordinator manages and approves events for which each club puts forward proposals. The club coordinators for each club can add or edit the club's information, and schedule events and club activities. They can also manage the club members and the visibility and access for club events to the student body. All other university students should be able to view the schedule of the club events and their details and register for the same.”

The following is an SRS for the same:

“

SRS for Student Clubs Event Management Platform

1. Problem Background:

The primary purpose of this platform is to enable club coordinators to post requests for scheduling events, which would be subject to approval by the Student Council's Clubs Coordinator. Post this approval, the event would be added to the platform for registration by the student body, and an email notification would be sent to all the students for the same. The platform would also have dedicated pages for the clubs, which the respective club's coordinator would manage.

2. Stakeholders / Users:

The primary users of the platform are (i) students of the university, (ii) the coordinators of various clubs, (iii) the Student Council's Clubs Coordinator and (iv) a representative from the Dean of Student Affairs office (Admin).

3. Functional Requirements:

3.1. All IIITD domain users should be able to log into the platform using their IIITD domain Gmail IDs. (all)

3.2. The platform should enable the club coordinators to schedule events for any given date and time. While applying for the same, they should be permitted to edit the event details and registration requirements. (Club Coordinators)

3.3. The platform would also have dedicated pages for the clubs, which the respective club's coordinator would manage. A club coordinator should be allowed to edit this page by editing club information, images from past events and point of contact information. (Club coordinators)

3.4. The Student Council's Clubs Coordinator should be given access to a page that lists all pending requests for scheduling the events while marking potential conflicts and should be able to approve or decline each one of them. (SC Clubs coordinator)

3.5. After the event's approval, it should be added to the platform for registration by the student body, and an email notification should be sent to all the students.

Students should be able to post comments regarding the event on the platform. (all)

- 3.6. Students should be allowed to search for upcoming events based on their interests and view them. (all)
- 3.7. The platform's admin access should be given to a representative from the Student Affairs office(Admin) who can edit event details, Club Coordinator and respective clubs' and students' accounts information. (Admin)
- 3.8. There should be a report feature in the platform to enable everyone to report incidents and issues with the platform. (all)
- 3.9. Students should be able to request for the formation of a new club. (all).
- 3.10. The approval for new clubs will be a two-stage process involving the approval of the Student Council Clubs Coordinator and then the admin. (SC Clubs Coordinator and Admin)

4. Performance Requirements:

- 4.1. The platform should withstand considerable amounts of traffic and frequent shifts in load and traffic coming its way.
- 4.2. The platform should be responsive and have an average response time of 300 milliseconds or less under normal load.

5. Design Constraints:

- 5.1. The platform should be developed while keeping accessibility for all users in mind and should have features like screen readers, font adjusters, etc.
- 5.2. The platform should be dynamic and should be able to adjust to varying screen resolutions and devices(like laptops, cellphones, tablets, etc.).
- 5.3. The platform should be easy to maintain, and the code should be readable and well-documented.
- 5.4. The tech stack for the platform should be the following:
 - 5.4.1. React JS for the front-end development.
 - 5.4.2. Node JS for the back-end development with MongoDB database.

6. External Interfaces:

- 6.1. The platform should be compatible with all major browsers like Safari, Chrome, Firefox, Brave, Microsoft Edge, etc.
- 6.2. The platform would use Google sign-in API to allow users to log in using their IIT Delhi domain email address. Upon login, the users would be redirected to Google API's interface.
- 6.3. The platform should be able to send email notifications to all students and other users.

7. Security Requirements:

- 7.1. The platform should be well protected from known attacks and penetration attempts to enable the protection of user information and data, like XSS(cross-site scripting) and SQL injection.
- 7.2. The platform should prevent packet interception and tampering with outgoing and incoming requests.
- 7.3. The platform should verify users using 2-factor authentication and IIT Delhi domain login should be implemented using Google's API.
- 7.4. The user data should be stored and maintained as per the institute's policy.

8. Glossary of Terms:

- 8.1. Users: Those who will use the application.
- 8.2. Students: Users who study at IIT Delhi and are not part of the management of the university.
- 8.3. General students: Users who aren't part of the management. They are the

primary stakeholders of the application.

8.4. Club coordinators: Users who are part of the management, albeit at a lower level. They are responsible for managing the club activities and hosting events.

8.5. Student council club coordinator: They are student users responsible for managing and approving the events added by club coordinators/heads.

8.6. DoSA: Department of Student Affairs at the university.

8.7. FMS: Facilities Management Services at IIIT Delhi.

8.8. Admin: The user can change logins and credentials, and manage the application.

8.9. Login: Sign into the application and let it know your details.

8.10. Explore: Browse through the events given.

8.11. Registering: Marking that the user 'may' be present at the event.

8.12. Participation: Participating in an event, showing interest, and attending when it is organised.

8.13. Credentials: information required for logging in; in this case, it is the Google account associated with the institute.

8.14. Request: It is for requesting allowance to conduct an event.

'''

Your task is to determine if each requirement conforms to the 4 major qualities of a good requirement: unambiguousness, understandability, correctness, and verifiability. Evaluate each requirement on a scale of 1 to 5, with higher numbers indicating better conformance to the quality. Please do so for every requirement in

- Functional Requirements
- Performance Requirements
- Design Constraints
- External Interfaces
- Security Requirements

To evaluate the requirements in the given SRS for the Student Clubs Event Management Platform, I will assess each requirement against the four key qualities of a good requirement: unambiguity, understandability, correctness, and verifiability. I will rate each quality on a scale from 1 to 5 for each requirement, where 1 indicates poor conformance and 5 indicates excellent conformance.

Functional Requirements

3.1 All IIITD domain users should be able to log into the platform using their IIITD domain Gmail IDs.

- Unambiguity: 5 (Clearly states the login method)
- Understandability: 5 (Easily comprehensible)
- Correctness: 5 (Accurately reflects the intended functionality)
- Verifiability: 5 (Easily testable by attempting a login)

3.2 The platform should enable club coordinators to schedule events for any given date and time.

- Unambiguity: 4 (Specific but lacks details about time constraints or limitations)
- Understandability: 5 (Clear and straightforward)
- Correctness: 5 (Correctly represents required functionality)
- Verifiability: 4 (Verifiable but requires detailed criteria for scheduling limitations)

3.3 The platform would have dedicated pages for the clubs, managed by club coordinators.

- Unambiguity: 4 (Mentions dedicated pages but lacks detail on page content)
- Understandability: 5 (Easily understood)
- Correctness: 5 (Correct representation of required features)
- Verifiability: 4 (Verifiable, though specifics of 'management' need clarity)

3.4 The Student Council's Clubs Coordinator should approve or decline event scheduling requests.

- Unambiguity: 5 (Clearly defined role and action)
- Understandability: 5 (Straightforward)
- Correctness: 5 (Accurate representation of the process)
- Verifiability: 5 (Easily testable through the approval process)

3.5 After approval, events should be added to the platform for registration, with email notifications to students.

- Unambiguity: 5 (Specific sequence of actions is clear)
- Understandability: 5 (Easily comprehensible)
- Correctness: 5 (Correctly outlines the post-approval process)
- Verifiability: 5 (Can be tested by tracking the event lifecycle)

3.6 Students should be able to search for and view upcoming events based on interests.

- Unambiguity: 4 (Specific but lacks details on how 'interests' are determined)
- Understandability: 5 (Straightforward concept)
- Correctness: 5 (Aligned with the system's goals)
- Verifiability: 4 (Verifiable, though 'interests' need clearer definition)

3.7 The platform's admin access should be with the Student Affairs office, who can edit various details.

- Unambiguity: 4 (Clear but lacks specifics on editable details)
- Understandability: 5 (Easily understood)
- Correctness: 5 (Correctly assigns admin role)

- Verifiability: 4 (Verifiable, requiring a detailed list of editable items)

3.8 There should be a report feature for incident and issue reporting.

- Unambiguity: 5 (Clearly stated functionality)
- Understandability: 5 (Simple and clear)
- Correctness: 5 (Appropriate for the system)
- Verifiability: 5 (Easily testable)

3.9 Students should be able to request the formation of new clubs.

- Unambiguity: 5 (Specific functionality)
- Understandability: 5 (Straightforward)
- Correctness: 5 (Aligned with the platform's purpose)
- Verifiability: 5 (Testable through the request process)

3.10 The approval for new clubs is a two-stage process.

- Unambiguity: 5 (Clear steps defined)
- Understandability: 5 (Easily understood)
- Correctness: 5 (Accurate representation of the approval process)
- Verifiability: 5 (Clearly testable in stages)

Performance Requirements

4.1 The platform should withstand considerable amounts of traffic.

- Unambiguity: 3 (Lacks a specific definition of 'considerable amounts')
- Understandability: 4 (Generally clear but could be more specific)
- Correctness: 5 (Relevant for the platform)
- Verifiability: 3 (Requires specific metrics for traffic amounts)

4.2 The platform should be responsive with an average response time of 300 milliseconds or less.

- Unambiguity: 5 (Very specific)
- Understandability: 5 (Clear and precise)
- Correctness: 5 (Directly related to performance)
- Verifiability: 5 (Easily measurable)

Design Constraints

5.1 The platform should be developed with accessibility features.

- Unambiguity: 3 (Broad, lacks specific features)
- Understandability: 4 (Generally understandable)

- Correctness: 5 (Important for user inclusivity)
- Verifiability: 3 (Requires a clear list of accessibility features)

5.2 The platform should adjust to varying screen resolutions and devices.

- Unambiguity: 5 (Clear requirement)
- Understandability: 5 (Easily comprehensible)
- Correctness: 5 (Relevant for modern web platforms)
- Verifiability: 5 (Testable across devices)

5.3 The platform should be easy to maintain with readable and well-documented code.

- Unambiguity: 3 (Subjective terms like 'easy' and 'readable' need clarification)
- Understandability: 4 (Generally clear)
- Correctness: 5 (Important for long-term sustainability)
- Verifiability: 3 (Requires specific criteria for code quality)

5.4 Specific tech stack requirements.

- Unambiguity: 5 (Very specific technologies listed)
- Understandability: 5 (Clear to those familiar with the technologies)
- Correctness: 5 (Appropriate for the platform)
- Verifiability: 5 (Easily verifiable through code review)

External Interfaces

6.1 Compatibility with major browsers.

- Unambiguity: 4 (Specific but lacks version details)
- Understandability: 5 (Easily understood)
- Correctness: 5 (Relevant for user accessibility)
- Verifiability: 4 (Testable, though exhaustive testing across all versions is challenging)

6.2 Use of Google sign-in API.

- Unambiguity: 5 (Very specific)
- Understandability: 5 (Clear to those familiar with Google API)
- Correctness: 5 (Aligned with login requirements)
- Verifiability: 5 (Testable through implementation)

6.3 Ability to send email notifications.

- Unambiguity: 5 (Specific functionality)
- Understandability: 5 (Straightforward)
- Correctness: 5 (Relevant for communication)

- Verifiability: 5 (Easily testable)

Security Requirements

7.1 Protection from known attacks.

- Unambiguity: 3 (Broad, lacks specifics of attacks)
- Understandability: 4 (Somewhat clear but could be more specific)
- Correctness: 5 (Essential for platform security)
- Verifiability: 3 (Challenging without a specific list of attacks)

7.2 Prevent packet interception and tampering.

- Unambiguity: 4 (Specific but lacks technical details)
- Understandability: 4 (Clear to those with technical knowledge)
- Correctness: 5 (Critical for security)
- Verifiability: 4 (Testable through security audits)

7.3 Two-factor authentication and domain login verification.

- Unambiguity: 5 (Specific security measures)
- Understandability: 5 (Clear to those familiar with 2FA)
- Correctness: 5 (Important for secure logins)
- Verifiability: 5 (Easily testable)

7.4 User data management as per institute's policy.

- Unambiguity: 3 (Depends on the institute's policy, which may vary)
- Understandability: 4 (Clear but policy details are needed)
- Correctness: 5 (Essential for data protection)
- Verifiability: 3 (Requires knowledge of the institute's policy)

This assessment provides a comprehensive view of how well the requirements in the SRS align with the key qualities of unambiguity, understandability, correctness, and verifiability. It's evident that most requirements are well-defined and align closely with these qualities, indicating a well-structured SRS. However, some areas, particularly in the specifics of security and performance requirements, could benefit from further detail to enhance clarity and verifiability.

Could you formulate it in a tabular format with the section, requirement, rating and reasons for those for all 4 parameters being the columns? So something like

Section|Requirement No.|Requirement|Unambiguity|Reason for Unambiguity|Understandability|Reason|Correctness|Reason|Verifiability|Reason|?

Certainly, here's the evaluation of the SRS in a tabular format. The table includes the section, requirement number, the requirement itself, and the ratings for unambiguity, understandability, correctness, and verifiability, along with the reasons for each rating.

Section	Requirement No.	Requirement	
Unambiguity	Reason for Unambiguity		
Understandability	Reason for Understandability		
Correctness	Reason for Correctness		
Verifiability	Reason for Verifiability		
Functional Requirements	3.1	All IIITD domain users should be able to log into the platform using their IIITD domain Gmail IDs.	5
		login method	5
		self-explanatory	5
		addresses a key functionality	5
		functionality is easily testable	5
Functional Requirements	3.2	Club coordinators to schedule events for any date and time.	4
		constraints	5
		Aligns with intended system use	5
		Testable, needs clear criteria for limitations	4
Functional Requirements	3.3	Dedicated pages for clubs managed by club coordinators.	4
		content specifics	5
		Correctly represents required feature	5
		Content management is verifiable, specifics needed	4
Functional Requirements	3.4	Student Council's Clubs Coordinator approves or declines event scheduling requests.	5
		and action	5
		Accurate process representation	5
		Approval process is testable	5
Functional Requirements	3.5	Events added post-approval for registration, with email notifications to students.	5
		Easily comprehensible	5
		Outlines post-approval process correctly	5
		Lifecycle of event is testable	5
Functional Requirements	3.6	Students search and view upcoming events based on interests.	4
		undefined	5
		Matches system's purpose	5
		Verifiable, but 'interests' need definition	4
Functional Requirements	3.7	Admin access with Student Affairs office, can edit event details and user accounts.	4
		on editables	5
		clear	5
		capabilities	4
		detailing for testing	5
Functional Requirements	3.8	Report feature for incident and issue reporting.	5
		Clearly states the functionality	5

Simple and clear	5	
Appropriate for system requirements	5	
Functionality is easily testable		
Functional Requirements 3.9	Students can request new club formation.	
5	Specifies a particular functionality	5
Straightforward and clear	5	
Aligns with platform's purpose	5	
Request process is testable		
Functional Requirements 3.10	Two-stage approval process for new clubs.	
5	Clear steps defined	5
Easily understood	5	
Accurately represents approval process		5
Staged testing is possible		
Performance Requirements 4.1	Platform should withstand considerable	
amounts of traffic.	3	'Considerable amounts' is
vague	4	Generally clear, needs
specifics	5	Relevant for the platform
3	Specific traffic metrics needed	
Performance Requirements 4.2	Responsive platform with an average response	
time of 300 milliseconds or less.	5	Very specific target
5	Clear and precise	5
Directly relevant to performance		
Response time is measurable		
Design Constraints 5.1	Develop with accessibility features like screen	
readers, font adjusters, etc.	3	Broad, lacks specific
features	4	Understandable but needs
feature details	5	Important for inclusivity
3	Specific accessibility features need clarification	
Design Constraints 5.2	Adjust to varying screen resolutions and devices.	
5	Clear requirement	5
Comprehensible across devices		5
Relevant for modern web platforms		5
Testable across different devices		
Design Constraints 5.3	Easy to maintain with readable and well-documented	
code.	3	'Easy' and 'readable' are subjective
4	Generally clear, could be more specific	
5	Important for sustainability	3
Needs specific criteria for code quality		
Design Constraints 5.4	Specific tech stack: React JS (front-end), Node JS	
(back-end), MongoDB (database).	5	Very specific technology
requirements	5	Clear to those familiar with
the tech stack	5	Appropriate for the
platform	5	Verifiable through
code review		
External Interfaces 6.1	Compatibility with major browsers.	
4	Lacks version specifics	5
Easily understood		5
Relevant for accessibility	4	
Exhaustive testing across versions is challenging		
External Interfaces 6.2	Use of Google sign-in API.	
5	Very specific integration	5

Clear to those familiar with Google API	5	
Aligns with login requirements	5	
Testable through implementation		
External Interfaces 6.3	Send email notifications to users.	
5	Specific functionality	5
Straightforward functionality	5	
Relevant for user communication	5	
Functionality is testable		
Security Requirements 7.1	Protection from known attacks like XSS and SQL injection.	
	3	Broad, more specifics of attacks needed
	4	Somewhat clear, could be more specific
	5	Essential for security
3	Challenging without specific attack list	
Security Requirements 7.2	Prevent packet interception and tampering.	
4	Specific but lacks technical details	4
Clear to those with technical knowledge		5
Critical for security	4	
Testable through security audits		
Security Requirements 7.3	Two-factor authentication and domain login verification using Google's API.	
	5	Specifies security measures clearly
	5	Clear and understandable
	5	Important for secure logins
	5	Easily testable through 2FA and domain verification
Security Requirements 7.4	User data management as per institute's policy.	
3	Depends on institute's policy, which may vary	4
Clear but requires policy details		5
Essential for data protection	3	
Verification relies on policy knowledge		

This table provides a detailed and structured analysis of each requirement in the SRS, assessing their conformity to the key traits of good requirements in software development.

Could you fix the requirements that score anything less than 5 in any criteria and rewrite those in a table with the format:

[Section|Requirement No|Old requirement|New Requirement]

Certainly! I'll revise the requirements that scored less than 5 in any of the criteria and present them in the requested table format.

Section	Requirement No.	Old Requirement
New Requirement		
----- ----- -----		
----- ----- -----		
----- ----- -----		
Functional Requirements 3.2	Club coordinators to schedule events for any date and time.	Club coordinators can schedule events specifying date, time, duration, and maximum participant capacity.
Functional Requirements 3.3	Dedicated pages for clubs managed by club	

coordinators.	Club coordinators manage dedicated club pages, including sections for club description, upcoming events, past event galleries, and contact information.
Functional Requirements 3.6	Students search and view upcoming events based on interests.
	Students can search for upcoming events using filters such as date, club name, event type, and keywords related to their interests.
Functional Requirements 3.7	Admin access with Student Affairs office, can edit event details and user accounts.
	Admins in the Student Affairs office have privileges to edit event details, user accounts, club coordinator assignments, and event approval statuses.
Performance Requirements 4.1	Platform should withstand considerable amounts of traffic.
	The platform must support up to 10,000 concurrent users and maintain functionality during peak usage times, such as event registration openings and result announcements.
Design Constraints 5.1	Develop with accessibility features like screen readers, font adjusters, etc.
	The platform will include accessibility features such as screen readers compatibility, font size adjusters, high contrast mode, and keyboard navigation support.
Design Constraints 5.3	Easy to maintain with readable and well-documented code.
	The platform's codebase shall adhere to industry-standard practices for readability, including clear naming conventions, modular design, and comprehensive documentation.
External Interfaces 6.1	Compatibility with major browsers.
	The platform must be compatible with the latest two versions of major browsers including Chrome, Firefox, Safari, Edge, and Brave.
Security Requirements 7.1	Protection from known attacks like XSS and SQL injection.
	The platform shall implement security measures against specific attacks including XSS, SQL injection, CSRF, and DDoS attacks, with regular updates based on emerging threats.
Security Requirements 7.4	User data management as per institute's policy.
	User data shall be managed in compliance with IIIT Delhi's Data Protection Policy, including provisions for data encryption, regular audits, and adherence to privacy regulations.

These revised requirements aim to address the deficiencies in unambiguity, understandability, correctness, and verifiability, thereby enhancing the overall quality and clarity of the SRS.

