

n-gpu-layers

25

n_ctx

16384

Context length. Try lowering this if you run out of memory while loading the model.

threads

0

threads_batch

0

n_batch

512

alpha_value

1

Positional embeddings alpha factor for NTK RoPE scaling. Recommended values (NTKv1): 1.75 for 1.5x context, 2.5 for 2x context. Use either this or compress_pos_emb, not both.

rope_freq_base

1000000

If greater than 0, will be used instead of alpha_value. Those two are related by $\text{rope_freq_base} = 10000 * \alpha_value ^ (64 / 63)$

compress_pos_emb

1

Positional embeddings compression factor. Should be set to $(\text{context length}) / (\text{model's original context length})$. Equal to $1/\text{rope_freq_scale}$.

Use llama-cpp-python compiled with tensor cores support. This increases performance on RTX cards. NVIDIA only.

☒ tensorcores

Do not offload the K, Q, V to the GPU. This saves VRAM but reduces the performance.

☐ no_offload_kqv

Disable the mulmat kernels.

☐ no_mul_mat_q

☐ no-mmap

☐ mlock

NUMA support can help on some systems with non-uniform memory access.

☐ numa

☐ cpu

tensor_split

Split the model across multiple GPUs, comma-separated list of proportions, e.g. 18,17

simple-1

max_new_tokens

4096

temperature

0.7

top_p

0.9

min_p

0

top_k

20

repetition_penalty

1.15

presence_penalty

0

frequency_penalty

0

typical_p

1

tfs

1

mirostat_mode

0

mirostat mode 1 is for llama.cpp only.

mirostat_tau

5

mirostat_eta

0.1

Seed (-1 for random)

-1

Other parameters

Truncate the prompt up to this length

16384

The leftmost tokens are removed if the prompt exceeds this length. Most models require this to be at most 2048.

Maximum tokens/second

0

To make text readable in real time.

Maximum UI updates/second

0

Set this if you experience lag in the UI during streaming.

Custom stopping strings

"", "You:"

In addition to the defaults. Written between "" and separated by commas.

Custom token bans

Specific token IDs to ban from generating, comma-separated. The IDs can be found in the Default or Notebook tab.

Forces the model to never end the generation prematurely.

☐ Ban the eos_token

☒ Activate text streaming

Load grammar from file (.gbnf)

None

Grammar

Learn more

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.

Context [Full]

```
{%- set ns = namespace(found=false) -%}
{%- for message in messages -%}
    {%- if message['role'] == 'system' -%}
        {%- set ns.found = true -%}
    {%- endif -%}
{%- endfor -%}
{%- if not ns.found -%}
    {{- " " + 'An SRS depicts the requirements that a piece of software should satisfy. There are
4 key traits of the requirements in an SRS:\n1. Unambiguity: A requirement is unambiguous
if and only if it has only one possible interpretation.\n2. Understandability: A requirement is
understandable if all classes of SRS readers can easily comprehend its meaning with a
minimum of explanation.\n3. Correctness: A requirement is deemed correct when it
accurately represents a required feature or function the system must possess.\n4.
Verifiability: A requirement is verifiable if finite, cost-effective methods exist for verifying that
it is satisfied by the system as built.\nGiven is the following template of an SRS:\n1.
Problem Background\n2. Stakeholders\n3. Functional Requirements\n4. Performance
Requirements\n5. Design constraints\n6. External Interfaces\n7. Security Requirements\n8.
Glossary of terms.
' + '\n\n' -}}
{%- endif -%}
{%- for message in messages -%}
    {%- if message['role'] == 'system' -%}
        {{- " " + message['content'] + '\n\n' -}}
    {%- else -%}
        {%- if message['role'] == 'user' -%}
            {{- '#### Instruction:\n' + message['content'] + '\n\n' -}}
        {%- else -%}
            {{- '#### Response:\n' + message['content'] + '\n\n' -}}
        {%- endif -%}
    {%- endif -%}
{%- endfor -%}
{%- if add_generation_prompt -%}
    {{- '#### Response:\n' -}}
{%- endif -%}
```

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 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 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

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 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 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

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)

- Unambiguousness: 5
- Understandability: 4
- Correctness: 5
- Verifiability: 3

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)

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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)

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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)

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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)

- Unambiguousness: 4
- Understandability: 3
- Correctness: 2
- Verifiability: 1

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

- Unambiguousness: 5
- Understandability: 4

- Correctness: 3
- Verifiability: 2

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)

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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)

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

4. Performance Requirements:

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

- Unambiguousness: 5
- Understandability: 5
- Correctness: 3
- Verifiability: 2

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

- Unambiguousness: 4
- Understandability: 5
- Correctness: 3
- Verifiability: 1

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.

- Unambiguousness: 4
- Understandability: 5
- Correctness: 3
- Verifiability: 2

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

- Unambiguousness: 4
- Understandability: 5
- Correctness: 3

- Verifiability: 2

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

- Unambiguousness: 4
- Understandability: 5
- Correctness: 3
- Verifiability: 1

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

- React JS for the front-end development.
- Node JS for the back-end development with MongoDB database.
- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

6. External Interfaces:

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

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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.

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 1

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.

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 1

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

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 1

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

- Unambiguousness: 5
- Understandability: 4
- Correctness: 3
- Verifiability: 2

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 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.

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|?

Output a single table with functional requirements, performance requirements, design constraints, external interfaces, and security requirements.

Section	Requirement No.	Requirement	Unambiguity	Understandability	Correctness	Verifiability
---------	-----------------	-------------	-------------	-------------------	-------------	---------------

---	---	---	---	---	---	---
-----	-----	-----	-----	-----	-----	-----

Functional Requirements	3.1	All IIITD domain users should be able to log into the platform using their IIITD domain Gmail IDs. (all)	5	4	5	3
-------------------------	-----	--	---	---	---	---

Login is a common functionality, therefore it is easy to understand and has only one interpretation, making it unambiguous

Functional Requirements	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)	5	4	3	2
-------------------------	-----	---	---	---	---	---

This requirement is specific to club coordinators and has only one interpretation, making it unambiguous

Functional Requirements	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)|5|4|3|2|This requirement is specific to club coordinators and has only one interpretation, making it unambiguous

Functional Requirements|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)|5|4|3|2|This requirement is specific to SC clubs coordinators and has only one interpretation, making it unambiguous

Functional Requirements|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)|4|3|2|1|The requirement is not specific to any user group and has multiple interpretations, making it ambiguous

Functional Requirements|3.6|Students should be allowed to search for upcoming events based on their interests and view them. (all)|5|4|3|2|This requirement is general and has only one interpretation, making it unambiguous

Functional Requirements|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)|5|4|3|2|This requirement is specific to Admin and has only one interpretation, making it unambiguous

Functional Requirements|3.8|There should be a report feature in the platform to enable everyone to report incidents and issues with the platform. (all)|5|4|3|2|This requirement is general and has only one interpretation, making it unambiguous

Functional Requirements|3.9|Students should be able to request for the formation of a new club. (all).|5|4|3|2|The requirement is general and has only one interpretation, making it unambiguous

Functional Requirements|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)|5|4|3|2|This requirement is specific to SC club coordinators and admins, and has only one interpretation, making it unambiguous

Performance Requirements|4.1|The platform should withstand considerable amounts of traffic and frequent shifts in load and traffic coming its way.|5|5|3|2|This requirement is general and has only one interpretation, making it unambiguous

Performance Requirements|4.2|The platform should be responsive and have an average response time of 300 milliseconds or less under normal load.|4|5|3|1|The requirement is specific to the users' experience with the platform, but has only one interpretation, making it unambiguous

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.|4|5|3|2|This requirement is general and has only one interpretation, making it unambiguous

Design Constraints|5.2|The platform should be dynamic and should be able to adjust to varying screen resolutions and devices(like laptops, cellphones, tablets, etc.).|4|5|3|2|This

requirement is general and has only one interpretation, making it unambiguous

Design Constraints|5.3|The platform should be easy to maintain, and the code should be readable and well-documented.|4|5|3|1|The requirement is specific to the developers' experience with the platform but has only one interpretation, making it unambiguous

External Interfaces|6.1|The platform should be compatible with all major browsers like Safari, Chrome, Firefox, Brave, Microsoft Edge, etc.|5|4|3|2|This requirement is general and has only one interpretation, making it unambiguous

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.|5|4|3|2|This requirement is general and has only one interpretation, making it unambiguous

Security Requirements|7.2|The platform should prevent packet interception and tampering with outgoing and incoming requests.|5|4|3|1|This requirement is specific to the security of the data being transmitted by the platform but has only one interpretation, making it unambiguous

Security Requirements|7.3|The platform should verify users using 2-factor authentication and IIIT Delhi domain login should be implemented using Google's API.|5|4|3|1|This requirement is specific to user verification but has only one interpretation, making it unambiguous

Security Requirements|7.4|The user data should be stored and maintained as per the institute's policy.|5|4|3|2|This requirement is general and has only one interpretation, making it unambiguous

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|

Section	Requirement No.	Old Requirement	New Requirement
Functional Requirements 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)	The approved events should appear on the platform with a description of the same along with the necessary details about the club that is organizing it. An email would also be sent to the entire student body notifying them of the event and its registration link. Users will now be able to post comments regarding the event after viewing the information for the same.	

Performance Requirements 4.2	The platform should be responsive and have an average response time of 300 milliseconds or less under normal load.	The platform should respond within a maximum of 500 ms under normal traffic conditions, while ensuring that it maintains the same level of performance for all users.
------------------------------	--	---

External Interfaces 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.	The platform will enable Google authentication and allow users to access the application directly without being required to go through the Google API's interface. This feature enhances security by eliminating the need for users to log into a
-------------------------	--	---

third-party service before accessing IIIT Delhi's student management portal.

Security Requirements7.2The platform should prevent packet interception and tampering with outgoing and incoming requests. All data transmitted between the client and server will be encrypted using SSL/TLS to prevent eavesdropping, man-in-the-middle attacks, and other forms of unauthorized access. The platform will also have a mechanism in place that prevents tampering with outgoing and incoming requests by verifying their authenticity using digital signatures.