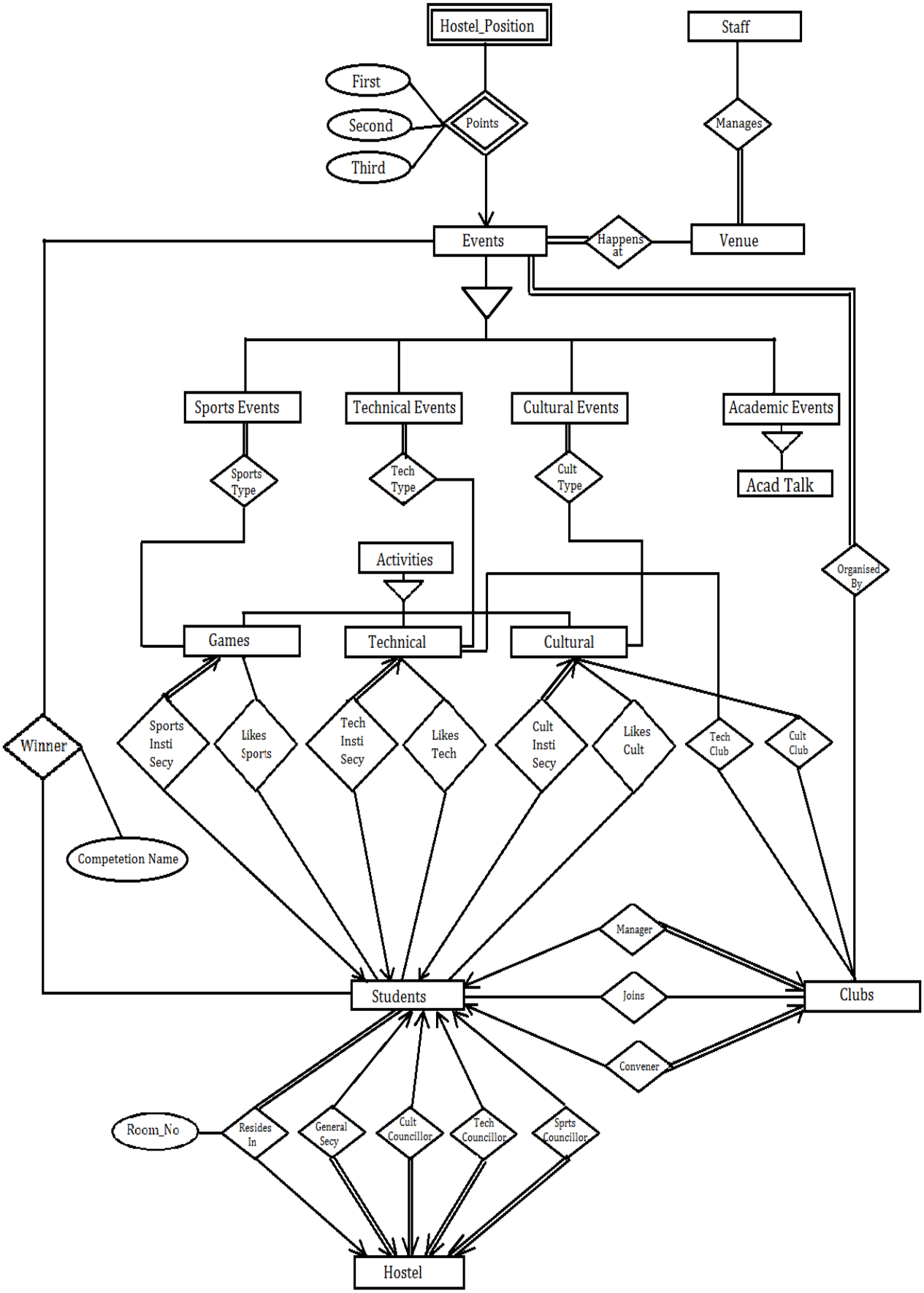
Student Activities Database Management System

ER Model



Relational Schema (using conventional conversion approach):

1) student (roll_no, student_name, phone_no, email_id, pasword)

   Functional Dependency: email_id →roll_no

2) hostel (hostel_no, room_no, roll_no, general_secretary_roll_no, cultural_councilor_roll_no, technical_councilor_roll_no, sports_councilor_roll_no)

   Deviation: Here we have merged the *resides_in* relationship (many to one from students to hostel) with hostel which should have been merged with students.

   Problems: The main problem asociated with this is the redundancy in information which will further lead to insertion and update anomalies.

   Here hostel_no and roll_no are the primary keys, so if there are 100 students in a hostel then there will be 100 entries in the table and the information like general secretary etc. will be repeated 100 times. This will introduce retundency in the schema.

   Foreign Key Constraints:

   roll_no REFERENCES student

   general_secretary_roll_no REFERENCES student

   cultural_councilor_roll_no REFERENCES student

   technical_councilor_roll_no REFERENCES student

   sports_councilor_roll_no REFERENCES student

   Functional Dependency:

   hostel_no →general_secretary_roll_no, cultural_councilor_roll_no, cultural_points, technical_points, technical_councilor_roll_no, sports_councilor_roll_no, sports_points,

   general_secretary_roll_no → hostel_no

   cultural_councilor_roll_no → hostel_no

   technical_councilor_roll_no → hostel_no

   sports_councilor_roll_no → hostel_no

3) club (club_name, genre, manager_roll_no, convener_roll_no)

   Foreign Key Constraints:

   manager_roll_no REFERENCES student

   convener_roll_no REFERENCES student

   Functional Dependency:

   manager_roll_no → club_name

   convener_roll_no → club_name

4) joins (roll_no, club_name)

Foreign Key Constraints:

roll_no REFERENCES student

club_name REFERENCES club

5) sports_activity (sports_activity_name, description, sports_institute_secretary_roll_no)

Foreign Key Constraints:

sports_institute_secretary_roll_no REFERENCES student

Functional Dependency:

sports_institute_secretary_roll_no → sports_activity_name

6) technical_activity (technical_activity_name, description, technical_institute_secretary_roll_no)

Foreign Key Constraints:

technical_institute_secretary_roll_no REFERENCES student

Functional Dependency:

technical_institute_secretary_roll_no → technical_activity_name

7) cultural_activity (cultural_activity_name, description, cultural_institute_secretary_roll_no)

Foreign Key Constraints:

cultural_institute_secretary_roll_no REFERENCES student

Functional Dependency:

cultural_institute_secretary_roll_no → cultural_activity_name

8) likes_cultural_activity (cultural_activity_name, roll_no)

Foreign Key Constraints:

roll_no REFERENCES student

cultural_activity_name REFERENCES cultural_activity

9) likes_technical_activity (technical_activity_name, roll_no)

Foreign Key Constraints:

roll_no REFERENCES student

technical_activity_name REFERENCES technical_activity

10) likes_sports_activity (sports_activity_name, roll_no)

Foreign Key Constraints:

roll_no REFERENCES student

sports_activity_name REFERENCES sports_activity

11) venue (venue_name, location, phone_no, capacity)

12) staff (staff_name, phone_no, email_id, venue_name)

Deviation: have merged the many-many relationship *manages* (from staff to venue) in staff.

Problems: There are 2 problems asociated with this merging:-

a) If a staff member manages many venues, his details will be stored many times, which will cause redundancy as well as insertion and update anamolies.
b) Since venue_name is a part of the primary key, therefore it cannot take NULL values, hence if a staff member does manage any venue, then his details cannot be stored in staff relation.

Foreign Key Constraints:

venue_name REFERENCES venue

13) technical_club (technical_activity_name, club_name)

Foreign Key Constraints:

technical_activity_name REFERENCES technical_activity

club_name REFERENCES club

14) cultural_club (cultural_activity_name, club_name)

Foreign Key Constraints:

cultural_activity_name REFERENCES cultural_activity

club_name REFERENCES club

15) events (event_name, start_date, end_date, start_time, duration)

16) sports_events (sports_event_name, start_date)

Foreign Key Constraint: sports_event_name, start_date REFERENCES events

17) technical_events (technical_event_name, start_date)

Foreign Key Constraint: technical_event_name, start_date REFERENCES events

18) cultural_events (cultural_event_name, start_date)

Foreign Key Constraint: cultural _event_name, start_date REFERENCES events

19) academic_events (academic_event_name, start_date)

Foreign Key Constraint: academic _event_name, start_date REFERENCES events

20) academic_talk (academic_event_name, start_date, speaker)

Foreign Key Constraint: academic _event_name, start_date REFERENCES academic_events

21) hostel_position (event_name, start_date, winner, first_runnerup, second_runnerup, first_point, second_point, third_point)

Foreign Key Constraint: event_name, start_date REFERENCES events

winner REFERENCES hostel

first_runnerup REFERENCES hostel

second_runnerup REFERENCES hostel

22) happens_at (event_name, start_date, venue_name)

Foreign Key Constraint: event_name, start_date REFERENCES events

venue_name REFERENCES venue

23) organised_by (event_name, start_date, club_name)

Foreign Key Constraint: event_name, start_date REFERENCES events

club_name REFERENCES club

24) winner (event_name, start_date, roll_no, competition_name)

Foreign Key Constraint: event_name, start_date REFERENCES events

roll_no REFERENCES student

25) sports_type (sports_event_name, start_date, sports_activity_name)

Foreign Key Constraint: sports_event_name, start_date REFERENCES sports_events

sports_activity_name REFERENCES sports_activity

26) cultural_type (cultural_event_name, start_date, cultural_activity_name)

Foreign Key Constraint: cultural_event_name, start_date REFERENCES cultural_events

cultural_activity_name REFERENCES cultural _activity

27) technical_type (technical_event_name, start_date, technical_activity_name)

Foreign Key Constraint: technical_event_name, start_date REFERENCES technical_events

technical_activity_name REFERENCES technical_activity

Relational Schema (using object-oriented approach):

(*differs from conventional schema, starting from relation no. 16)

1) student (roll_no, student_name, phone_no, email_id, pasword)

   Functional Dependency: email_id →roll_no

2) hostel (hostel_no, room_no, roll_no, general_secretary_roll_no, cultural_councilor_roll_no, technical_councilor_roll_no, sports_councilor_roll_no)

   Deviation: Here we have merged the *resides_in* relationship (many to one from students to hostel) with hostel which should have been merged with students.

   Problems: The main problem asociated with this is the redundancy in information which will further lead to insertion and update anomalies.

   Here hostel_no and roll_no are the primary keys, so if there are 100 students in a hostel then there will be 100 entries in the table and the information like general secretary etc. will be repeated 100 times. This will introduce retundency in the schema.

   Foreign Key Constraints:

   roll_no REFERENCES student

   general_secretary_roll_no REFERENCES student

   cult_councilor_roll_no REFERENCES student

   tech_councilor_roll_no REFERENCES student

   sports_councilor_roll_no REFERENCES student

   Functional Dependency:

   hostel_no → general_secretary_roll_no, cultural_councilor_roll_no, cultural_points, technical_points, technical_councilor_roll_no, sports_councilor_roll_no, sports_points,

   general_secretary_roll_no → hostel_no

   cultural_councilor_roll_no → hostel_no

   technical_councilor_roll_no → hostel_no

   sports_councilor_roll_no → hostel_no

3) club (club_name, genre, manager_roll_no, convener_roll_no)

   Foreign Key Constraints:

   manager_roll_no REFERENCES student

   convener_roll_no REFERENCES student

   Functional Dependency:

   manager_roll_no → club_name

   convener_roll_no → club_name

4) joins (roll_no, club_name)

Foreign Key Constraints:

roll_no REFERENCES student

club_name REFERENCES club

5) sports_activity (sports_activity_name, description, sports_institute_secretary_roll_no)

Foreign Key Constraints:

sports_institute_secretary_roll_no REFERENCES student

Functional Dependency:

sports_institute_secretary_roll_no → sports_activity_name

6) technical_activity (technical_activity_name, description, technical_institute_secretary_roll_no)

Foreign Key Constraints:

technical_institute_secretary_roll_no REFERENCES student

Functional Dependency:

technical_institute_secretary_roll_no → technical_activity_name

7) cultural_activity (cultural_activity_name, description, cultural_institute_secretary_roll_no)

Foreign Key Constraints:

cultural_institute_secretary_roll_no REFERENCES student

Functional Dependency:

cultural_institute_secretary_roll_no → cultural_activity_name

8) likes_cultural_activity (cultural_activity_name, roll_no)

Foreign Key Constraints:

roll_no REFERENCES student

cultural_activity_name REFERENCES cultural_activity

9) likes_technical_activity (technical_activity_name, roll_no)

Foreign Key Constraints:

roll_no REFERENCES student

technical_activity_name REFERENCES technical_activity

10) likes_sports_activity (sports_activity_name, roll_no)

Foreign Key Constraints:

roll_no REFERENCES student

sports_activity_name REFERENCES sports_activity

11) venue (venue_name, location, phone_no, capacity)

12) staff (staff_name, phone_no, email_id, venue_name)

Deviation: have merged the many-many relationship *manages* (from staff to venue) in staff.

Problems: There are 2 problems asociated with this merging:-

A) If a staff member manages many venues, his details will be stored many times, which will cause redundancy as well as insertion and update anamolies.
B) Since venue_name is a part of the primary key, therefore it cannot take NULL values, hence if a staff member does manage any venue, then his details cannot be stored in staff relation.

Foreign Key Constraints:

venue_name REFERENCES venue

13) technical_club (technical_activity_name, club_name)

Foreign Key Constraints:

technical_activity_name REFERENCES technical_activity

club_name REFERENCES club

14) cultural_club (cultural_activity_name, club_name)

Foreign Key Constraints:

cultural_activity_name REFERENCES cultural_activity

club_name REFERENCES club

15) events (event_name, start_date, end_date, start_time, duration)
16) sports_events (sports_event_name, start_date, end_date, start_time, duration)
17) technical_events (technical_event_name, start_date, end_date, start_time, duration)
18) cultural_events (cultural_event_name, start_date, end_date, start_time, duration)
19) academic_events (academic_event_name, start_date, end_date, start_time, duration)
20) academic_talk (academic_event_name, start_date, end_date, start_time, duration, speaker)
21) winner_event (event_name, start_date, roll_no, competition_name)

Foreign Key Constraints:

event_name REFERENCES events

start_date REFERENCES events

roll_no REFERENCES student

22) winner_sports_event (sports_event_name, start_date, roll_no, competition_name)

Foreign Key Constraints:

sports_event_name REFERENCES sports_events

start_date REFERENCES sports_events

roll_no REFERENCES student

23) winner_technical_event (technical_event_name, start_date, roll_no, competition_name)

Foreign Key Constraints:

technical_event_name REFERENCES technical_events

start_date REFERENCES technical_events

roll_no REFERENCES student

24) winner_cultural_event (cultural_event_name, start_date, roll_no, competition_name)

Foreign Key Constraints:

cultural_event_name REFERENCES cultural_events

start_date REFERENCES cultural_events

roll_no REFERENCES student

25) event_happens_at (event_name, start_date, venue_name)

Foreign Key Constraints:

event_name REFERENCES events

start_date REFERENCES events

venue_name REFERENCES venue

26) sports_event_happens_at (sports_event_name, start_date, venue_name)

Foreign Key Constraints:

sports_event_name REFERENCES sports_events

start_date REFERENCES events

venue_name REFERENCES venue

27) technical_event_happens_at (technical_event_name, start_date, venue_name)

Foreign Key Constraints:

technical_event_name REFERENCES technical_events

start_date REFERENCES events

venue_name REFERENCES venue

28) cultural_event_happens_at (cultural_event_name, start_date, cultural_venue_name)

Foreign Key Constraints:

cultural_event_name REFERENCES cultural_events

start_date REFERENCES events

venue_name REFERENCES venue

29) academic_event_happens_at (academic_event_name, start_date, venue_name)

Foreign Key Constraints:

academic_event_name, start_date REFERENCES events

venue_name REFERENCES venue

30) sports_type(sports_event_name, start_date, sports_activity_name)

Foreign Key Constraints:

sports_event_name, start_date REFERENCES sports_events

sports_activity_name REFERENCES sports_activity

31) technical_type(technical_event_name, start_date, technical_activity_name)

Foreign Key Constraints:

technical_event_name, start_date REFERENCES technical_events

technical_activity_name REFERENCES technical_activity

32) cultural_type(cultural_event_name, start_date, cultural_activity_name)

Foreign Key Constraints:

cultural_event_name, start_date REFERENCES cultural_events

cultural_activity_name REFERENCES cultural_activity

33) hostel_positon_event(event_name, start_date, winner, first_runnerup, second_runnerup, first_points, second_points, third_points)

Foreign Key Constraints:

event_name , start_date REFERENCES events

winner REFERENCES hostel

first_runnerup REFERENCES hostel

second_runnerup REFERENCES hostel

34) hostel_positon_sports_event(sports_event_name, start_date, winner, first_runnerup, second_runnerup, first_points, second_points, third_points)

Foreign Key Constraints:

sports_event_name , start_date REFERENCES sports_events

winner REFERENCES hostel

first_runnerup REFERENCES hostel

second_runnerup REFERENCES hostel

35) hostel_positon_technical_event (technical_event_name, start_date, winner, first_runnerup, second_runnerup, first_points, second_points, third_points)

Foreign Key Constraints:

technical_event_name , start_date REFERENCES technical_events

winner REFERENCES hostel

first_runnerup REFERENCES hostel

second_runnerup REFERENCES hostel

36) hostel_positon_cultural_event(cultural_event_name, start_date, winner, first_runnerup, second_runnerup, first_points, second_points, third_points)

Foreign Key Constraints:

cultural_event_name , start_date REFERENCES cultural_events

winner REFERENCES hostel

first_runnerup REFERENCES hostel

second_runnerup REFERENCES hostel

37) hostel_positon_academic_event(academic_event_name, start_date, winner, first_runnerup, second_runnerup, first_points, second_points, third_points)

Foreign Key Constraints:

academic_event_name, start_date REFERENCES academic_events

winner REFERENCES hostel

first_runnerup REFERENCES hostel

second_runnerup REFERENCES hostel

Assertions:

- A student can take only one Positon of Respnsibility at a time in the institute i.e. no student should be repeated in hostel general secretary, hostel cultural councillor, hostel sports councillor, hostel technical councillor, club manager, club convener, sports institute secretary, cultural institute secretary and technical institute secretary relation schemas.
- No two events can happen at the same venue at same time.
- Winner, first-runnerup and second runner-up must be different for an event. In other words, one hostel can not be the winner and first-runnerup in an event simultaneously.
- No two students can have the same email-id.
- A hostel can have exactly one General Secretary, Cultural Councillor, Techinical Councillor and Sports Councillor
- A club can have exactly one Manager and Convener.

Screen Design

Note: You need to be connected to the internet to view the google calendar interface. You can access the home interface at http://www.cse.iitb.ac.in/~abhishekgupta/cs387_project/home.html

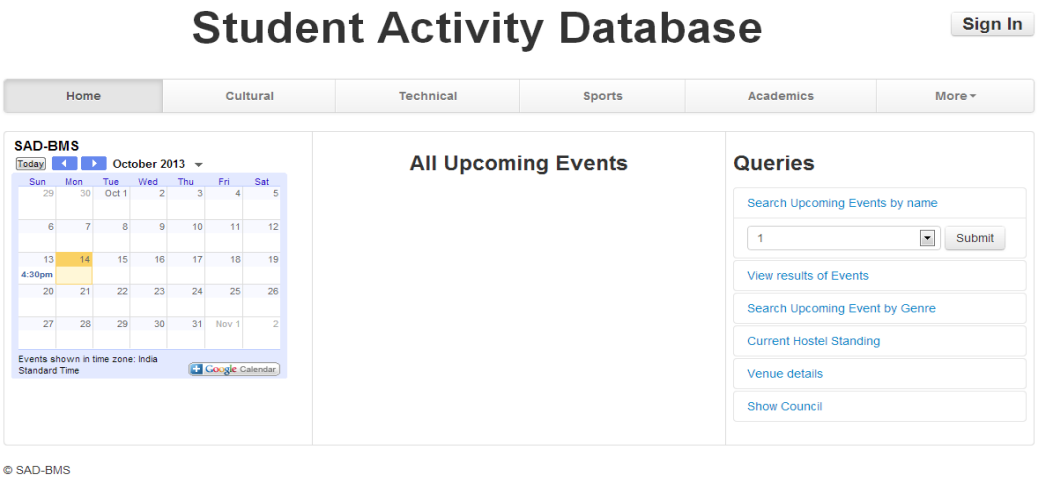
To access Student interface: http://www.cse.iitb.ac.in/~abhishekgupta/cs387_project/student.html

To access Admin Interface: http://www.cse.iitb.ac.in/~abhishekgupta/cs387_project/admin.html

Our Screen Design has following three interfaces:

1. Home Interface

1) The Home interface can be opened by home.html file. This will be the main interface which is visible to the user on first time access. User does not need to login to view this interface.



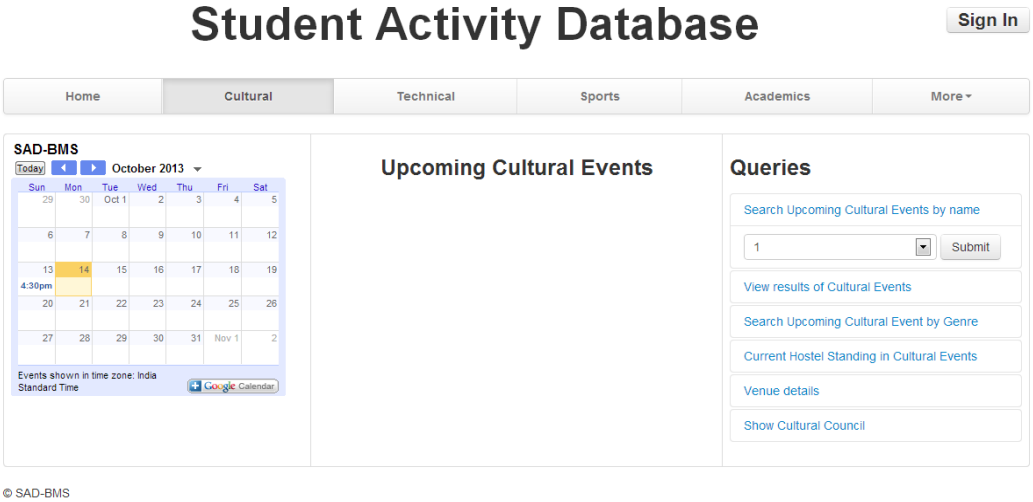
2) The left hand side contains a google calender which shows all the scheduled events and past events. On clicking the events in the calender more details about the events will be shown in the central part of screen (titled Upcoming events currently).

3) The right hand side contains the interface to run the queries.

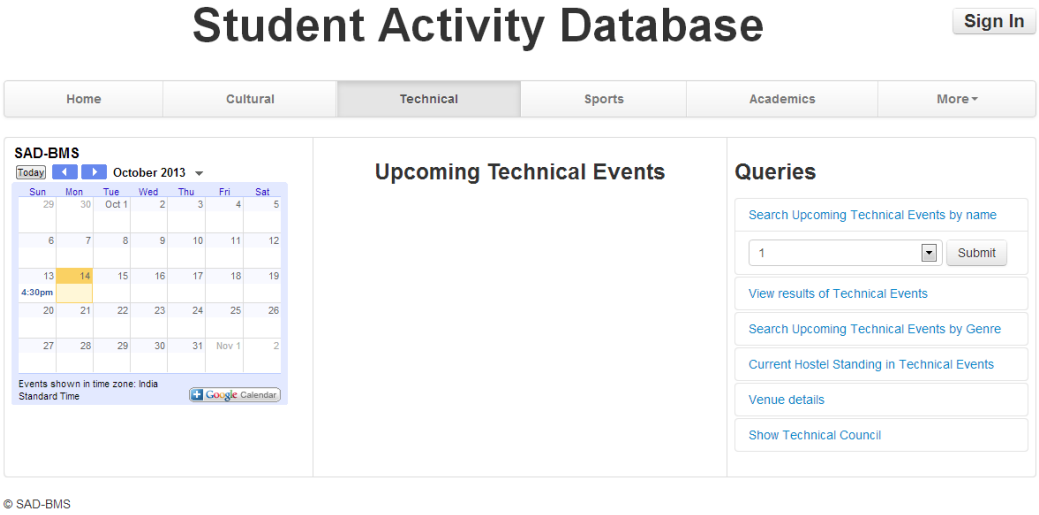
4) The central part of screen will show the upcoming events in the next few days. This part is also used to print the results of all the queries.

5) The user can switch tabs to specific categorey: Cultural, Sports, Technical, Academic. These tabs have the same interface as Home but show everything(queries/results/events/calender) corresponding to that categorey. These interfaces will look as below:

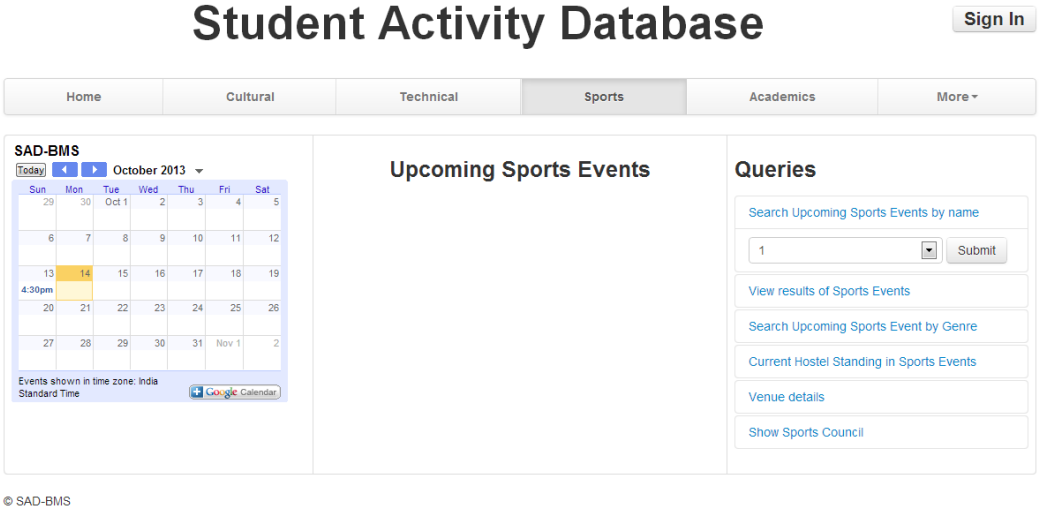
Cultural:



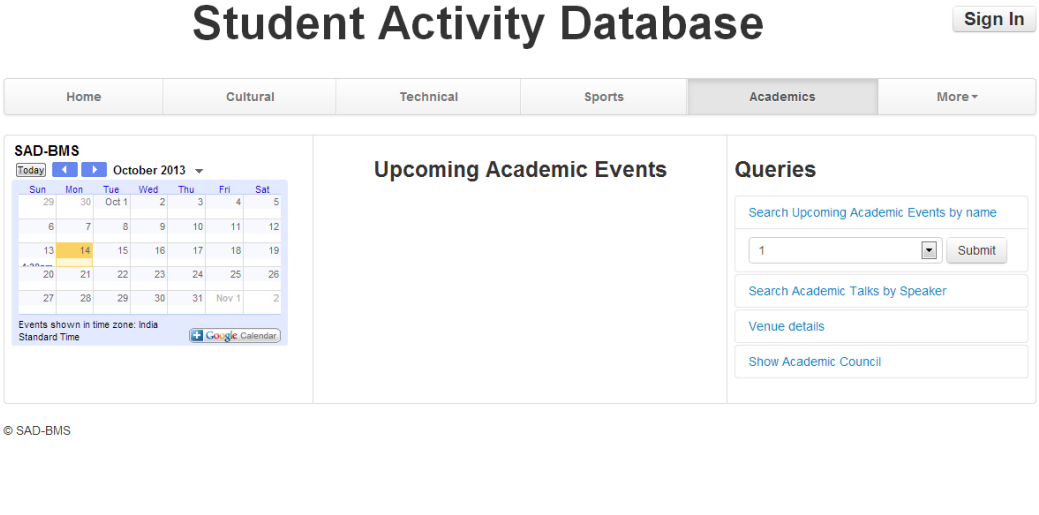
Technical:



Sports:

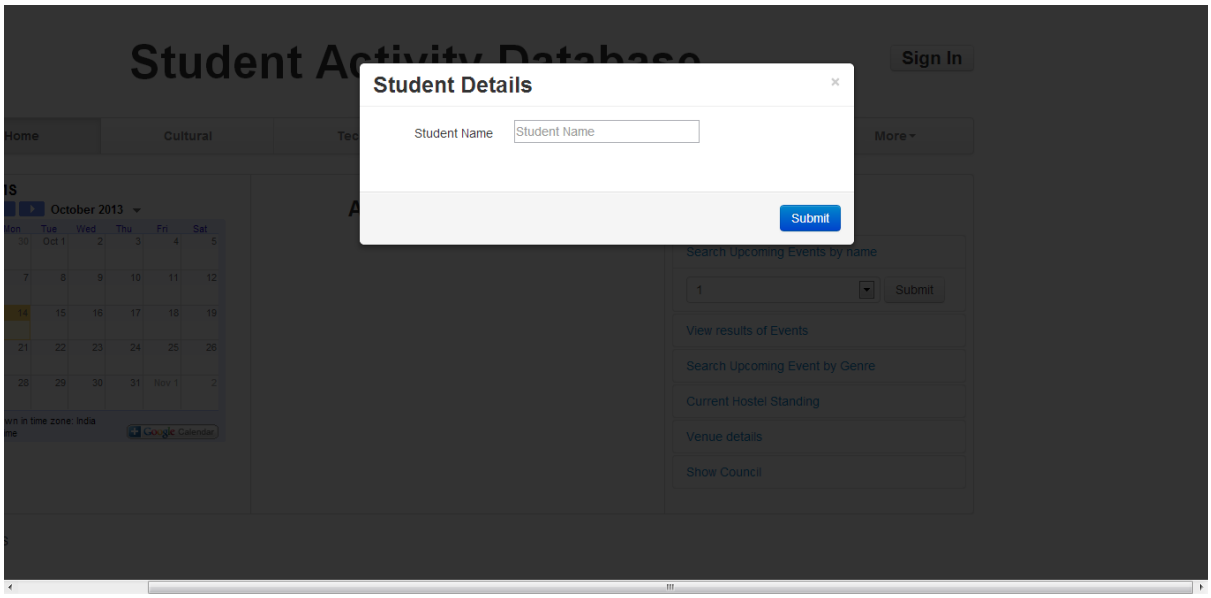


Academics:

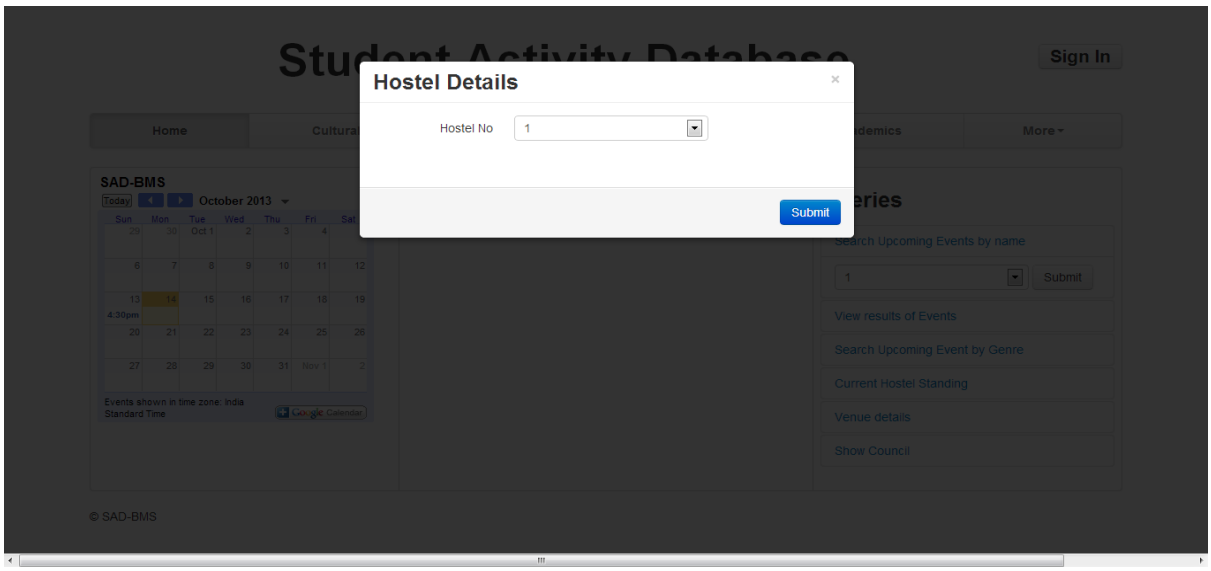


6) Apart from all this, user can also run some advanced queries listed under 'More' namely:

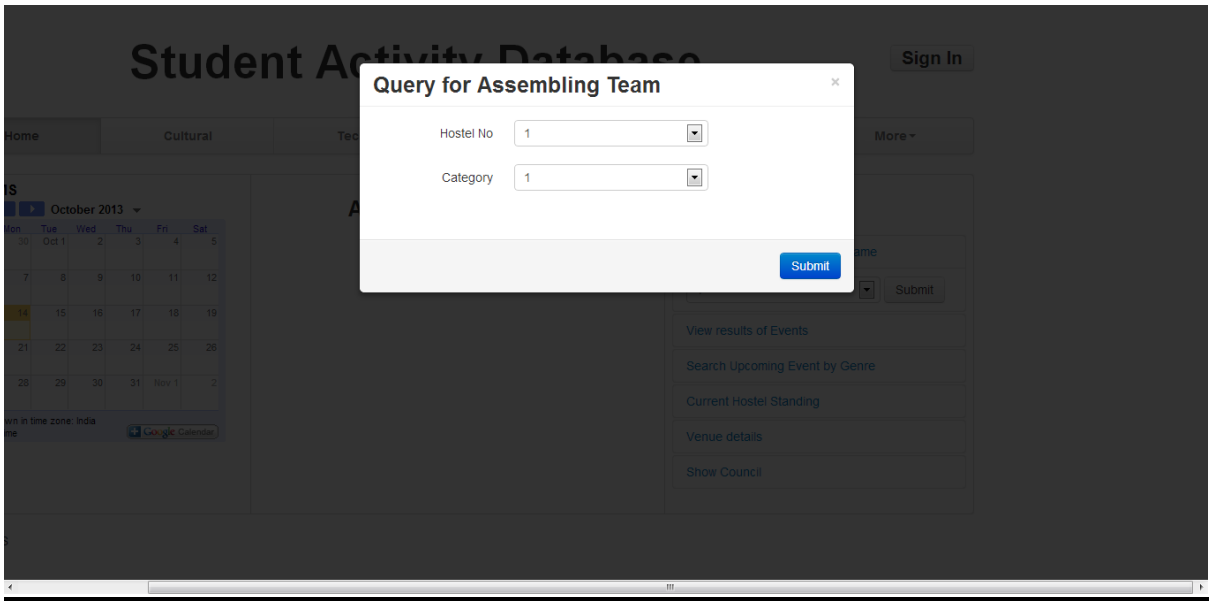
(i) Search for a particular student's details (interests/ email-ID/ clubs joined)



(ii) Search for hostel details (GC points/ General secretary/ hostel councillors)

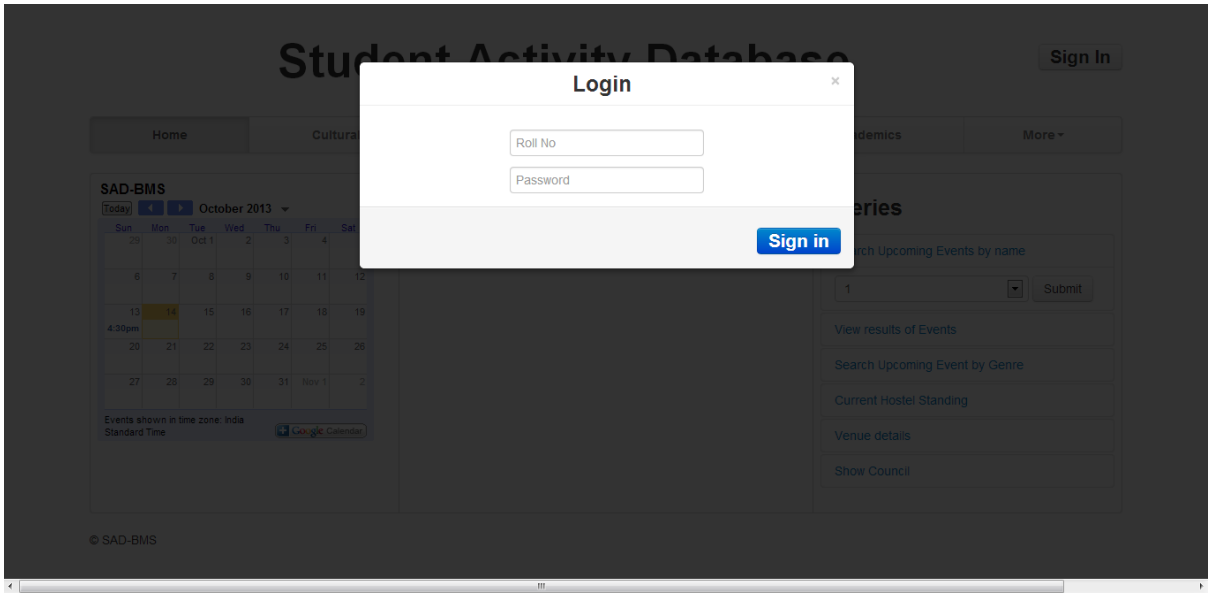


(iii) Assemble team from a hostel



2. Student Interface:

1) Once home interface has been opend user can enter this interface by logging in with his Roll No and Password. (Button given on top right corner of screen)



2) This has all the features of Home interface + additional features(under 'Interests') to

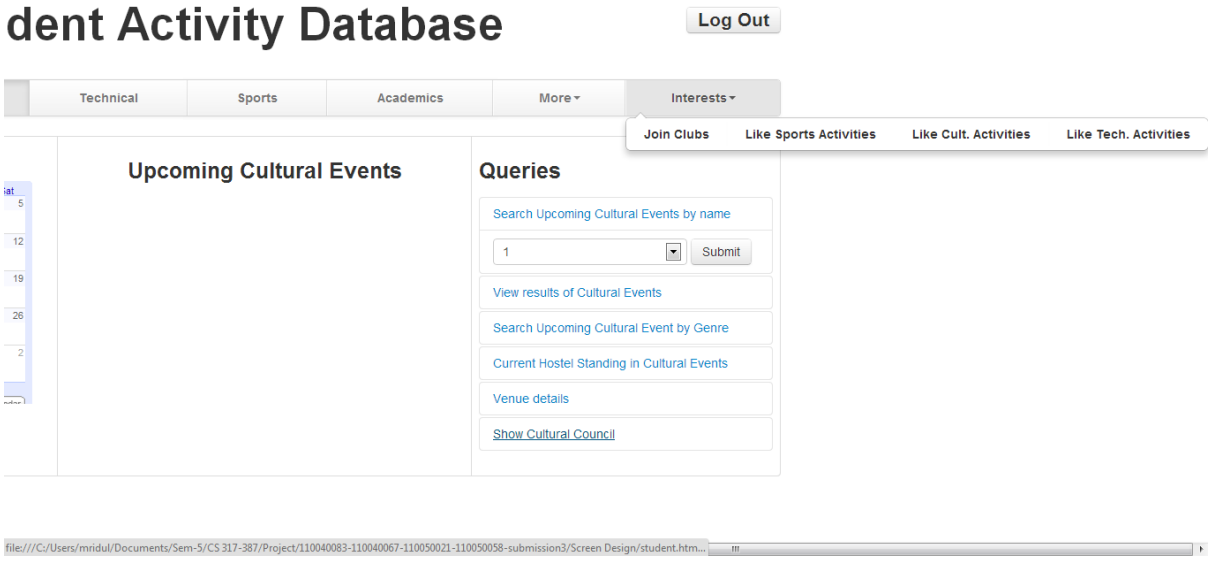
(i) Join clubs

(ii) Like Sports

(iii) Like Cultural activities

(iv) Like Technical activities

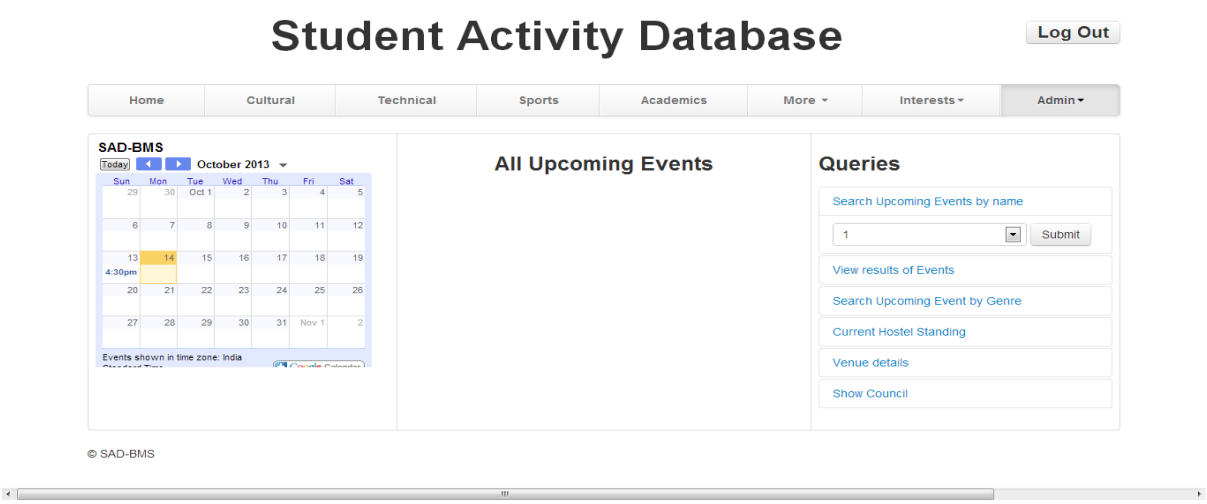
These options will appear as a dropdown as shown below:



3) A logout button on left top corner will take the user back to the Home interface and user is logged out.

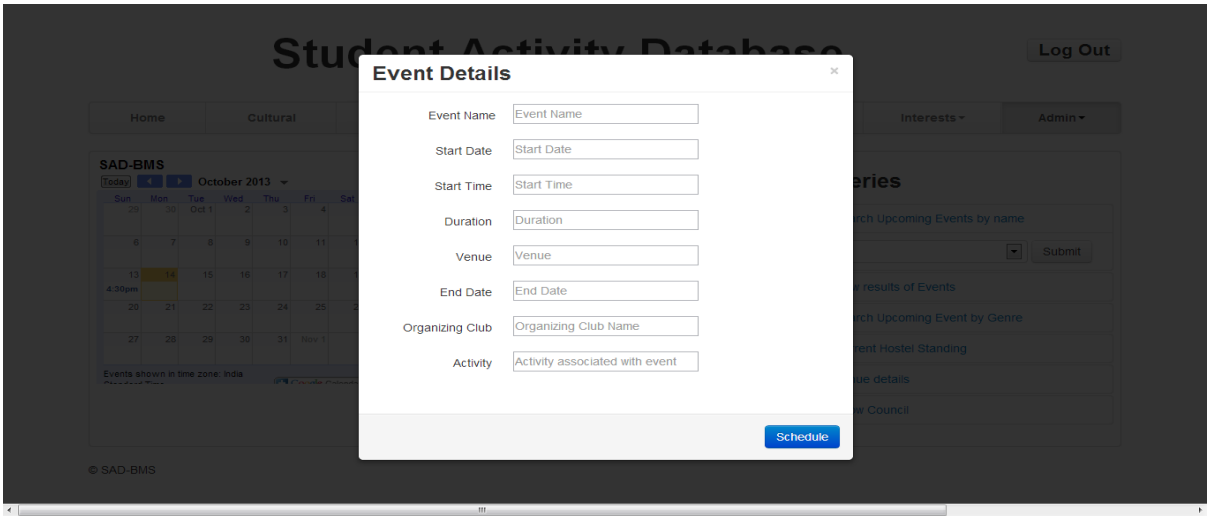
3. Admin Interface

1) As previously mentionedwe will give some special privileges to some users (institute secretaries, club managers and conveners) for scheduling events, sending reminder mails etc. When such user would login they will be shown this admin interface:

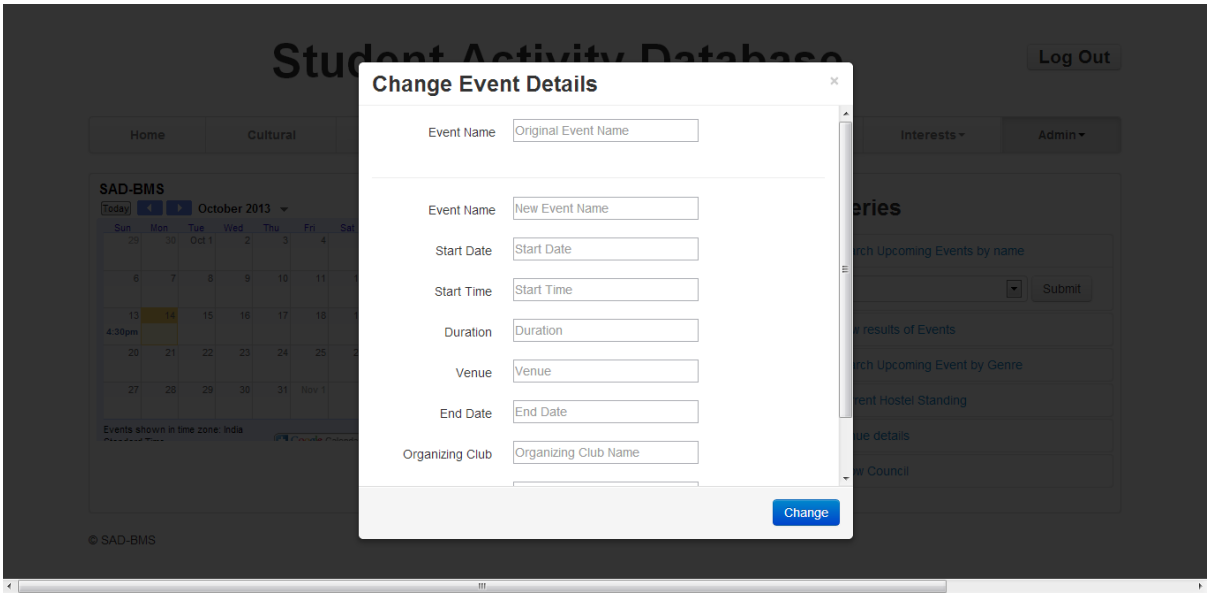


This has all the features of Student interface + additional features(under 'Admin') to

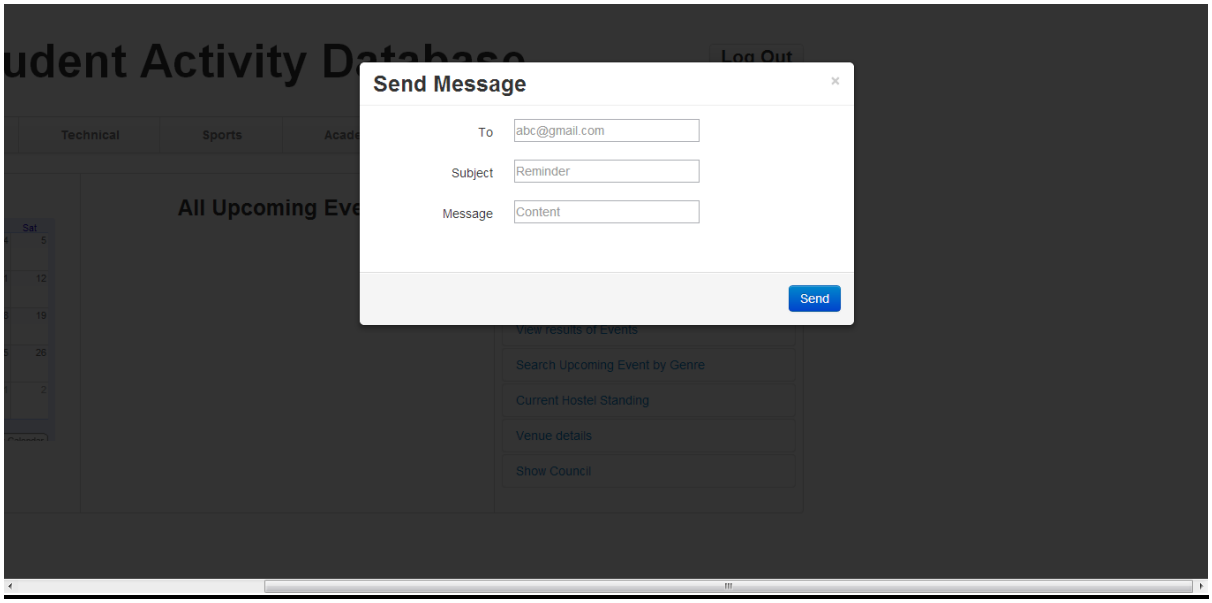
(i) Schedule event



(ii) Change details of an event

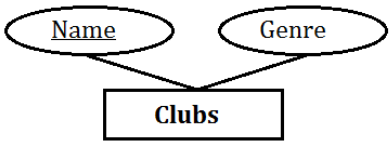


(iii) Send reminder mails

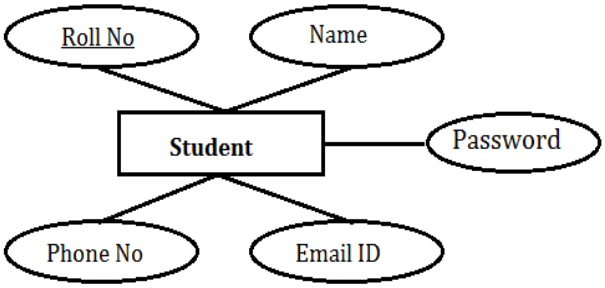


Entity Sets

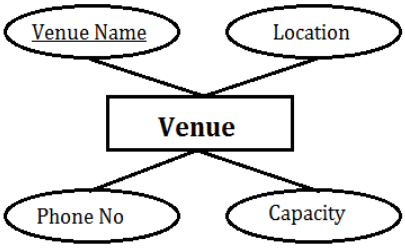
1. **Clubs**: consists of various student clubs belonging to different genres. For example Fourthwall (*Name*) is a Dramatics (*genre*) club. Note that it is does not specify the type of club i.e. We do not model that Fourthwall is a cultural club. This information is indeed modelled by *CultClub* relationship set. We will have clubs of all types viz cultural, technical, sports. Various student councils are also modelled as Clubs with genre as Council.



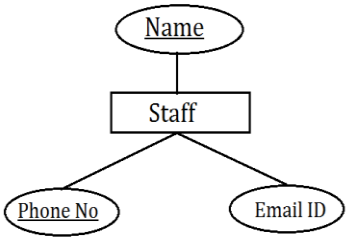
2. **Student**: consist of all student details-*Name, Roll No, Phone No, Email ID, Login ID* and *Password*. We will be making a sign-in interface for students providing options to like/unlike clubs, modify their profile (Phone No, Email ID) and login details (Password) etc.



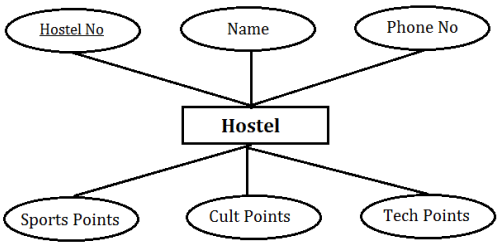
3. **Venue:** consist of the details of all venues in the institute where different events are conducted. The details are *Venue Name*, *Location, Phone No* and *capacity.*



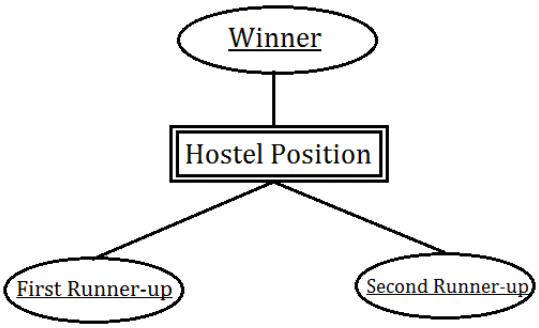
4. **Staff:** consists of the details- *Name*, *Phone No, Email ID*-of people who manage the venues. This will be required for booking the venues while scheduling an event.



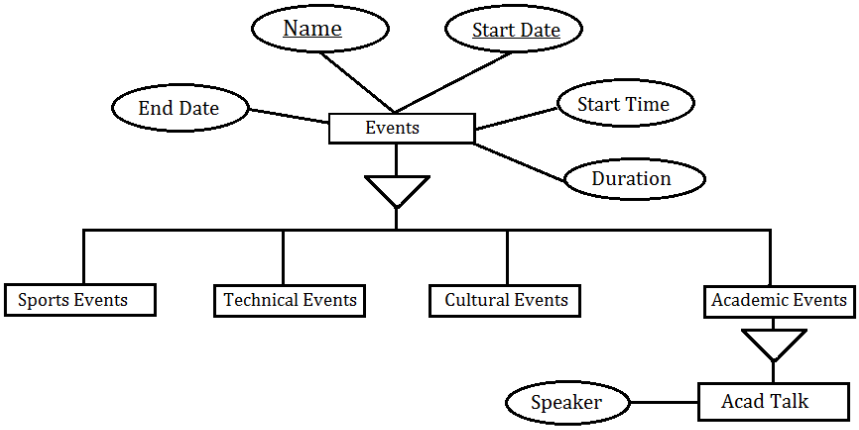
5. **Hostel:** is an entity set that describes a Hostel. It has attributes *Hostel No*, *Hostel Name, Phone No, Sports Points, Cult Points and Tech Points.*



6. **Hostel Position:** It is a weak entity set that models which hostel won the GC event, first runner-up and second runner-up. Since hostels get different points in different GCs, these points are modelled as attributes of weak relationship set between **Hostel Position** and **Events.**



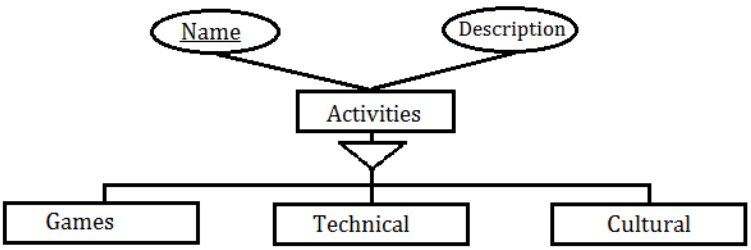
7. **Event:** An event is a planned occasion that occurs within the institute, for example Academic talks, Orientations, General Championships etc. The events are therefore classified into four categories which *are* ***Cultural Events, Technical Events, Sports Events*** *and* ***Academics Events***. The events which have a winner, such as GC, store the winner in another relationship set *Winner* between Events and Students.



8. **Acad Talk**: Since most of the academic events are academic talks (which have a speaker), the academic events are further classified into this category which contains *speaker* as an attribute.

9. **Activities**: This entity set models all the activities along with their description. Activities can be classified as *Games, Cultural* and *Technical*.
    - **Games** include sports activities such as Cricket, Football, Athletics and Swimming etc.
    - **Cultural** models all the cultural activities. For example Music, Dramatics, Literary Arts, Film/Media etc.

- **Technical** models various technical activities such as Web & Coding, Robotics, Electronics, Maths & Physics, Aero modelling etc.



Relationship sets

1. **Winner**: establishes the relationship between an Event and its winner Student/team members. The name of the *competition* is stored as an attribute of relationship set.

2. **Points**: Identifying relationship set between the weak entity set *Hostel Position* and strong entity set *Event*. This is used to specify which hostel secured which position in an event.
3. The *points* for this are stored as an attribute of this relationship set.

4. **Manages**: is used to model the *Venue* and its *manager*, where manager is a Staff member.

5. **Happens at**: relationship between an *Event* and its *Venue*.

6. **Organised by:** Establishes the relationship between an *event* and its organising *club*.

7. **Sports Type**: Sports event are related to a type of game which is modelled by this relationship set.
8. **Tech Type**: Any technical event pertains to a particular type of technical activity. This is taken care of by *Tech Type* relationship set.

9. **Cult Type**: A cultural event is similarly a type of cultural activity which is modelled by this relationship set.

10. **Tech Club**: All technical events are organised by a Technical *Club*.

11. **Cult Club**: All cultural activities are organised by a Cultural *Club*.

12. **Likes Sports**: This is a relationship set between *student* and the *Games* he likes.

13. **Likes Tech**: This is a relationship set between a student and the technical activities he likes.

14. **Likes Cult**: relationship between a student and the cultural activities he likes.

15. **Sports Insti Secy**: Every game has an Institute level Secretary who is a student.

16. **Tech Insti Secy**: Every technical activity has an Institute level Secretary who is a student.

17. **Cult Insti Secy**: Every cultural activity has an Institute level Secretary who is a student.

18. **Manager:** Every club has a Manager who is a student.

19. **Convener:** Every club has a Convener who is a student.

20. **Joins:** A student can join various clubs and student bodies.

21. **Resides In:** This relationship set models a student's address by mapping a student to his hostel. The *room number* is stored as an attribute of this relationship set.

22. **General Secy:** Every hostel has a general secretary.

23. **Cult Councillor:** Every hostel has a cultural councillor.

24. **Tech Councillor:** Every hostel has a technical councillor.

25. **Sports Councillor:** Every hostel has a sports councillor.

Functional Specifications

- Calendar Features: A calendar will be displayed in a similar manner as a Google calendar is displayed. On the calendar, people will select a date and all the events and their details will be provided. If the date is in the future, the events scheduled on that day along with their details will be displayed. If the date is in the past, the events conducted on that day will be displayed along with its details and its results. The details will include Event name, Venue of the event, details about the organizers and the start time of the event.

- Hostel Tally: For each overall General Championships (G.C.) (cultural, technical or sports) a table will be displayed, which shows the hostel's points for each event of the overall G.C., if any. In addition to that, if a user wishes to see any particular G.C. event's winners (the top three performing hostels and the first coming student, (if not a team event)), he can check it directly on the calendar. If the event is not a G.C. event (e.g. open competitions), then only individual winner's name will be displayed in the calendar. If the event is not a competition (e.g. a talk or a hands-on session), nothing will be stored.

- Scheduling of Events: This is an administrator (admin) only privilege. The admin can book a venue for an event for a particular time frame starting from a specific time. Thus all the admins can plan their events such that there is no venue clash. Note that there can be a time clash, which is not a problem as long as they are on different venues. Venues will be dispensed on a first come first serve basis. The scheduler can change the details of the event, such as time or venue whenever he wishes, prior to the event. If the admin is unaware that a particular venue has been already booked and he wishes to book the same venue which would lead to a clash, he/she will notified then that it leads to a clash and hence to reschedule the event. Once an event is booked or some changes are made to the event's schedule, the concerned persons will be notified. The admin can also check on a particular time on a date which venues are not booked. The admin can also check the dates and time durations during which the venue is not booked.

- Event Details: Any user can request details of an event. Upon his request, a page will display all the details about the particular event. Details include the start time, time duration, venue, organizers and rules of the event, if any. If the user just wishes for the start time and venue, he can just look it up directly in the calendar. The user can search for upcoming events for a particular genre or by a particular club. E.g. a user can look up all the upcoming events by Web and Coding Club.

- Venue Details: A user can check details about a venue. Its details contain the name of the venue, its location, the phone number at the venue and its capacity. It also contains the name of the staff who maintains the venue along with his phone number and email address. It will help an admin to plan for an event and in the execution of the event with the help of the staff. E.g. If a talk is scheduled at F.C.Kohli auditorium, the organizer will know its capacity and will also know whom to contact in case of microphone failure.

- Hostel councilors can use it for assembling hostel teams for the events: The councilor has to send participating teams into G.C. A huge headache for him is to find interested and able persons, who can make up a winning team or individually win a G.C. and get the hostel crucial points. Since all users can put up their interests in their profile and these will be stored in the database. Now, he can search up interested persons in a particular event or

genre within a hostel and make a team out of them, who are more likely to be the best combination to win the G.C.

- Determine technical, cultural and sports council: A user might not know the concerned persons and their contact details for a particular genre. He might query a particular club/genre's council which might include the institute secretary of a genre, the manager of a club, its conveners or coordinators. In case he wishes for the overall council, he can also ask for it which will include the General Secretary and institute nominees along with the aforementioned. This will help the user in asking for an event's organizers as the details of the events will also include the organizing committee.

- Academic talks: Many talks are given in our institute by professors and industry professionals. We receive a lot of mails in this regard and most of them are forgotten in due time. We will include all these talks in our calendar so that anyone accessing it will also know about the details (title, speaker, time and venue) on such an event. On request, further details (abstract of the talk and speaker credentials) will also be provided to the user. A user can also check for all details of all the talks given or are to be given by a particular speaker within an academic year.

- Determine events organized by a particular club or genre: Suppose a user is interested in only a few particular genre of cultural or technical or a particular sport, he can just directly check for all the events being organized by the responsible body for that particular sport or genre. He can also check out events to be organized by a particular club.

- Sending out reminder mails: Users with administrative privileges will be able to send reminder mails to all the relevant persons.

- Find details about a particular student or a hostel: Any user can check out details of any student of our institute. The details will include name, room no, roll no and his interests. If the student who details are being searched is an admin, basically convener or manager or secretary, his phone number will also be provided. Thus we are protecting the privacy of a student, except if he is an organizer, in which case he has to anyway provide his phone number.

Definitions:-

Event – An event is a planned occasion that occurs within the institute.

Venue – Is any place within the institute where events can take place

Administrator – We will be giving some special privileges to some users (institute secretaries, club managers and conveners)

Concerned persons/Relevant persons – The meaning of this term depends on the context it is used at, for example in case of a cultural event, it refers to all the people who likes the club which is organizing this event or likes that particular genre.

Model Requirements

- ✓ Store information about students, i.e. their Roll No, Name, Phone no, email id's etc.
- ✓ Each student will have user-name and password, which they will use for logging in.
- ✓ Students reside in hostels. There are various hostels. Each hostel has a unique no, which represents that hostel. Along with this each hostel has a name and a phone for communication.
- ✓ Each hostel has different councilors(Cult councilor, Tech councilor, Sports councilor) and a General Secretary. The councilors are students of that hostel itself. A student cannot take 2 posts in the hostel
- ✓ There are various student clubs which organizes different events in the institute. The clubs have unique names and belong to different genres.
- ✓ The clubs have manager and convener which are again students.
- ✓ Students can join various clubs.
- ✓ There are various student activities in the institute like sports, cultural and technical activities.
- ✓ Each activity (like a game or a particular cultural genre) will have secretaries which will be students. No particular student can work as secretary for 2 activities.
- ✓ Students like different games, take part in different genres in cultural and technical activities.
- ✓ There are various types of events that happen in the institute like cultural, technical, sports and academic events. The events are scheduled at a fixed venue, have fixed starting and ending dates and a fixed starting time. Each event will have a name associated with it.
- ✓ The events are organized by the respective clubs. The Sports events and Academic events are organized by sports council and academic council respectively, which has to be modeled as a club. The cultural and technical activities are organized by cultural clubs(like FourthWall or SilverScreen) and Technical clubs(like WnCC or Electronics club) respectively.
- ✓ Some of these events have winners, points associated with it(like General Championships). Some of these don't have these features, like academic talks.
- ✓ Some events will be hostel-wise and hence will have hostel-teams as winner. In those cases, hostels will get points in that event type(cultural, academic or technical). Those points need to be stored and hostel wise positions in different genre needs to be stored.
- ✓ The winner of the event is a student itself.
- ✓ The event is scheduled at a venue, which is managed by a staff. The venue has address associated with it, and has fixed capacity. The staff's name, contact no and email id has to be stored in order for communication.

Test Plan

1) Scheduling of Events(For scheduling an event, user should have administrative privileges)

INPUT	CONDITIONS	EXPECTED OUTPUT
Event Name and details	Date is in future and no clashes	Event scheduled, venue booked
Event Name and details	Date is in future and venue clash	Event will not be scheduled, user will be told about the clash
Event Name and details	Date is in future and another event is at that time but at different venue	User will be shown this, and on confirmation event will be scheduled
Event Name and details	Date is in past	An error msg will be thrown to the user stating that date is in the past
Event Name and details	Date is invalid	An error msg will be shown to the user
Venue	Venue is valid	All the upcoming days on which venue is free will be shown
Venue and Date	Valid date, venue and date is in future	The time during which the venue will be free during that day, and what all events are organized on that day on this venue and at what time
Venue and Date	anything is invalid	An error msg will be shown to the user
Date	Valid date and date is in future	all venues that are free on that day will be shown along with the time at which they will be free
Date	invalid date	an error msg will be shown to the user

2) Changing schedule of an event(for changing schedule of an event, user should have administrative privileges)

INPUT	CONDITIONS	EXPECTED OUTPUT
Event Name and new details	valid details(i.e. Valid date, no clash)	Event details changed
Event Name and new details	invalid details	an error msg will be shown to the user

3) Querying for an Event

INPUT	CONDITIONS	EXPECTED OUTPUT
Date	Date is in future and another event is at that time but at different venue	All The events scheduled on that day will be shown along with the details. The details will include event name, venue of the event, start time, details about organisers
Date	Date is in past	All the events which were scheduled on that day will be shown along with their details and results
Name	if valid event name, event is yet to happen	All the event details along with the details of the organizers and venue will be shown to the user
Name	if valid event name, event has already happened	All the event details along with the details of the organizers and venue will be shown to the user, if the event had a winner, that will also be displayed
Name	invalid name	an error msg will be shown to the user
Genre name(for cult, tech)	valid genre name(else error msg will be shown)	All the upcoming events of that genre will be displayed along with their details
Sports name	Valid sports name	All the upcoming events of that game will be displayed along with their details

4) Queries for Hostel Tallies

INPUT	CONDITIONS	EXPECTED OUTPUT
Cultural	-	A table will be displayed which will show how much points hostels have got for all the cultural events
Technical	-	A table will be displayed which will show how much points hostels have got for all the technical events
Sports	-	A table will be displayed which will show how much points hostels have got for all the sports events

5) Some specific queries regarding Points

INPUT	CONDITIONS	EXPECTED OUTPUT
Event Name, Hostel Name	The event had some points associated with it	Points scored by that hostel in that event
Event Name, Hostel Name	The event does not have some points associated with it	User will be displayed a msg stating that the event did not have any points associated with it
Event Name	The event had some points associated with it	The top 3 hostels in that event along with the points they got will be shown

6) Queries for venues

INPUT	CONDITIONS	EXPECTED OUTPUT
Venue Name	valid venue name	All the details about the venue will be shown along with the managing staff, also all the upcoming events scheduled on that venue will be shown
Venue Name	invalid venue name	an error msg will be shown to the user

7) Queries for Assembling team

INPUT	CONDITIONS	EXPECTED OUTPUT
Hostel No, Genre/Game	valid details	All the students of that particular hostel who likes that genre or plays that game along with their details
Hostel No, Genre/Game	Invalid details	an error msg will be shown to the user

8) Queries for determining council

INPUT	CONDITIONS	EXPECTED OUTPUT
Type(sports/cult/tech)	-	The whole council along with the details of the members(secies, G.S.) will be displayed
club name	valid club name	all the details about the club and its managers, conveners are shown
club name	invalid club name	an error msg will be shown to the user

9) Queries about academic talks

INPUT	CONDITIONS	EXPECTED OUTPUT
Speaker Name	Valid speaker name	All the upcoming talks scheduled for that speaker will be displayed along with their details(On request further details like abstract about the talk can also be shown)
Speaker Name	Valid speaker name	The no of talks given by that professor in this year will be shown(details about a particular talk can be shown on request)

10) Sending out reminder mails

INPUT	CONDITIONS	EXPECTED OUTPUT
User with admin privileges can send a reminder mail to all the relevant persons	-	Mail will be send to all the relevant persons

11) Finding details of a particular hostel or student

INPUT	CONDITIONS	EXPECTED OUTPUT
Student Name	should be present in the database	All the details of the student will be shown to the user
Student Name	not present in the database	an error msg will be shown to the user
Hostel No	should be present in the database	All the details of the particular hostel will be shown to the user
Hostel No	not present in the database	an error msg will be shown to the user