DBIS Project Scope Report
Generic Event Manager (GEM)

Tejas Iyer	< *tejas@cse* >	98005004
Satyen Kale	< *satyen@cse* >	98005009
B. Aditya	< *baditya@cse* >	98005033
Vijay Victor D'Silva	< *vjvictor@cse* >	98005102

Overview:

This event manager will provide an easy way to manage, supervise and access a host of large scale events like cultural/technical festivals, sports meets, and the like, which occur at a single location.

The inspiration for such a project is the need to easily access details of, register for and organize various such events that are held in our institute every year through a centralized database in the institute.

Aims:

1. Maintain a database of current as well as upcoming events, and possibly even archives of past events. For eg. this database would store a list of events in IIT Bombay, such as Mood-Indigo, Techfest, Swimming Gala, Inter-IIT meets.

2. A single event is organised by a single event orgainser. He shall be assisted by a number of sub-event organisers, who manage a single sub-event. Examples of sub-events are Hardware Competiton (TechFest) and 50 mtr freestyle relay (Swimming Gala). These organisers shall specify the fields in the database and may modify the initial template to suit their own customised needs.

3. Due to the online management of the events, an organiser may remotely administer his event. He also has the option of deciding whether the information in his database should be made private/public. He shall also decide whether registration of team can be done as a single entity. However, this feature will have to be common for that sub-event.

4. A participant who wishes to note the sub-events of an event, or other such details may do so by logging into the database anonymously and viewing them through the interface only. If the event allows it, he can view information about the particpants in a sub-event.

5. The participant may register for an event through such an anonymous login. On doing so, he gets a user-id (preferably his email-id) and password.

It is with this user-id that he shall login thereafter and be able to modify his profile and registration details.

6. A single participant may be participating in several sub-events, which could be possibly spread across events themselves. His basic profile however shall be common, and hence upon logging into the database, he shall be shown a list of events he has registered for, and the list of events he can register for if he wishes to do so. He can now edit his profile, and other event-specific details.

7. It is also possible for a user with a single user-id to register a team of members for a single event. Thus the team contingent leader can register a team of people participating in a sub-event, if the corresponding property for that event exists. For eg. in a relay swimming event, the team manager shall register the details of his team (four names, etc).

8. Both the event manager and the participant shall be able to view event-static details such as schedules, venues, results, etc. The participant could also be informed of a schedule clash in two or more events he has registered for.

9. A suitable external security mechanism shall be employed for the authentication and authorisation of a user of the database.

10. Since it is not reasonable to have such a large number of users as registered users of the database, we shall allocate this task of role/user specification to the Java-middle-tier application itself.

11. We can have a separate log of changes made to each event database, be it in that of its structure or a new registration.

User Hierarchy:

Our software envisages four levels of users with their corresponding permissions and duties.

1. Dbase administrator :
   (a) Administers entire database of events.
   (b) Creates/Removes Events.
   (c) Allocates access permissions.
   (d) Can set constraints on structure/size of event.

2. Event Manager/Organizer :
   (a) Designs the structure of the event.

(b) Creates/inserts/removes sub-events for her event.

(c) Sets and modifies overall attributes of the event.

(d) Can set access permissions for sub-event orgainsers and participants of his own event.

(e) Can set constraints like number of participants, registration deadline.

(f) May authorise participation of any person wishing to participate.

(g) Decides whether event is public/private (open registration or not).

3. Sub-event Manager :
Essentially the same tasks of a manager, but permissions are restricted to that of his own sub-event.

4. Participant :

   (a) Can register for event

   (b) Can view details of events, schedules

   (c) Can modify/delete her own entries

   (d) Can be an individual or a group

Notable Features:

1. The event and sub-event managers shall have distinct user-id's such as MI-org, MI-FC-org, and hence a manager of two separate events shall have two distinct user-id's. On the other hand, it is cumbersome to have a separate user-id for each of several events a participant can and will participate in. For example, in a swimming gala it is seen that a single good swimmer would participate in many events. It is easier for him to login with a single id and modify/view his entries for all such events together.

2. Corresponding to each event, links to related information could be given. For eg. a link to a FAQ, or even websites related to the topic could be referenced !!

3. For each event, there will two sets of attributes:

   (a) Sub-event level attributes: These will be general attributes which pertain to the whole sub-event. Like, date, times, judges for competitions, results of competitions, available seats, etc.

   (b) Participant level attributes: These will participant specific attributes: say, like name, id, email-id, address, phone no., etc. This info will be used to create the participant registration forms for this sub-event.

4. The event manager can set participation restrictions (open registration, allow individual to register a group etc). He can alter general properties of the event. Event properties include: total duration, list of sub-events, available venues etc. This will be used to generate default forms for sub-events. Extra sub-event specific properties may be added as required.

Interface:

The interface for each class of users shall be different. For eg, the user shall view a form for modifying his own profile, while the event-organiser should be able to view the whole list of participants and their attributes in a spreadsheet-like manner. The system-administrator shall view the list of events in the database and shall navigate to one of the events from where he shall continue on as the organiser of that event.

The basic scheme for entering data and creating registrations will be in the form of a wizard which is invoked when a new event is created and helps an event manager in designing his event specific details. A similar wizard will also appear when, say, a participant wishes to register for any event. This will have the standard format of filling information in steps with navigability between pages.

□□□