A Web Based Database Application(Design Report)

This is a very efficient online portal which helps to view courses or edit the courses by professors of IITH. In this, firstly we have a Home Page on which the basic details of the project are given. There is a well-designed header which contains About-which gives a gist of this project. The header also contains

- Courses
- Register
- Login

The Courses are further divided into a drop down with – view courses offered and Time Table.

Register- It is a form which allows the professors to register themselves with their personal information. If they have already registered, they can simply login with their userid and password.

Login- If the user has already registered, they can login as mentioned above.

In our design, a course can be added only by the admin but it can offered by an professor at any time.

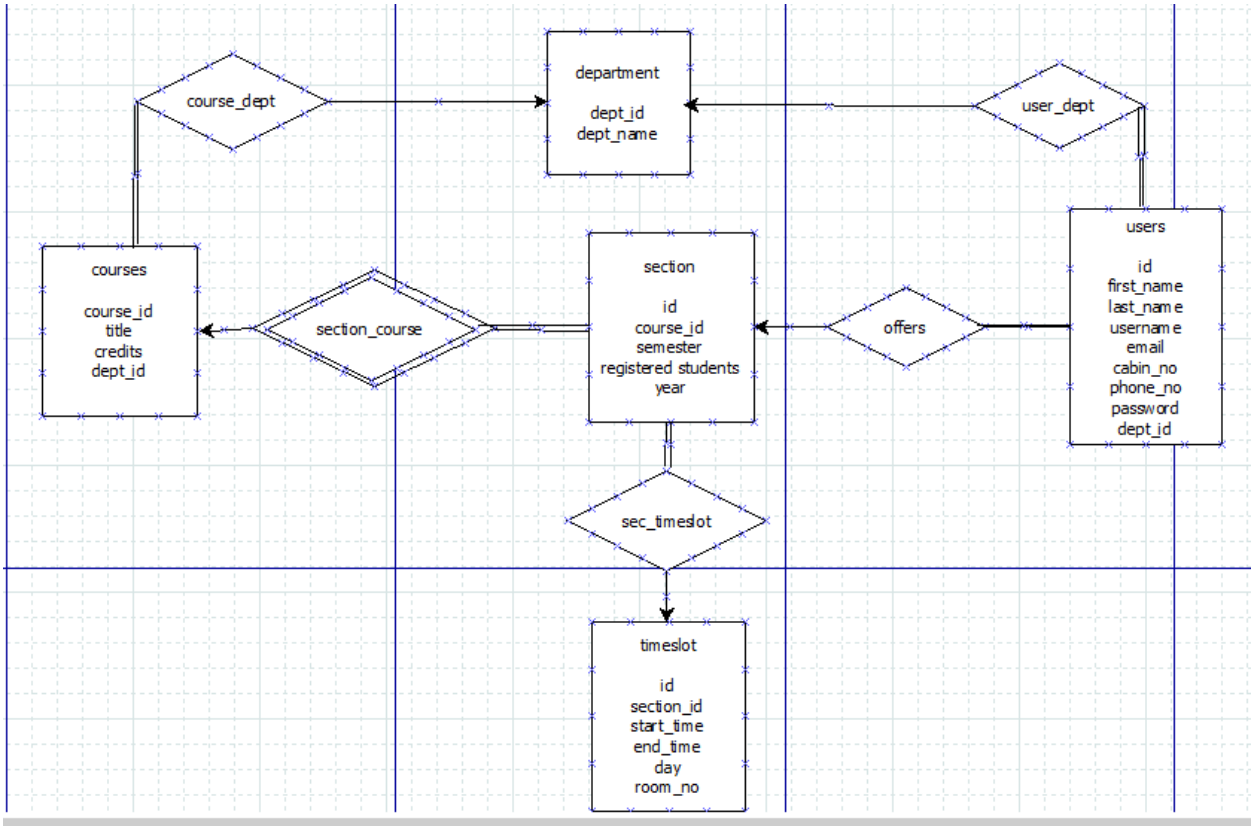
- ✓ **Courses**
  - This enables to view all courses irrespective of it is being offered by a professor or not.
  - There is a filter which enables a quick dynamic smart search.
  - As in, when in course cs is typed, the data shown below automatically keeps updating itself giving desired results.
  - There is no search button included-which is better as the results will be displayed dynamically and using ajax.
- ✓ **Register**
  - In this, a user can register as a professor.
- ✓ **Login**
  - In this, a user can login as a professor. After he logs in, a new header appears. Also a page with all his personal details appears. These details can be edited by him after he logs in.
  - There are two buttons at the bottom of his profile-View My courses and Offer Courses.
  - View My Courses helps him/her to view the courses ever offered by him/her till now.
  - Offer Course-helps him/her to offer a course which hasn't been offered by him in that semester
  - In the header we have
    - View Courses Offered- Helps to view respective courses
    - TimeTable- displays the timetable in a systematic table fashion
    - Offer Courses-helps to offer courses by inputing all required fields and also dynamically adding sections. There is a timeslot option which can be added dynamically.

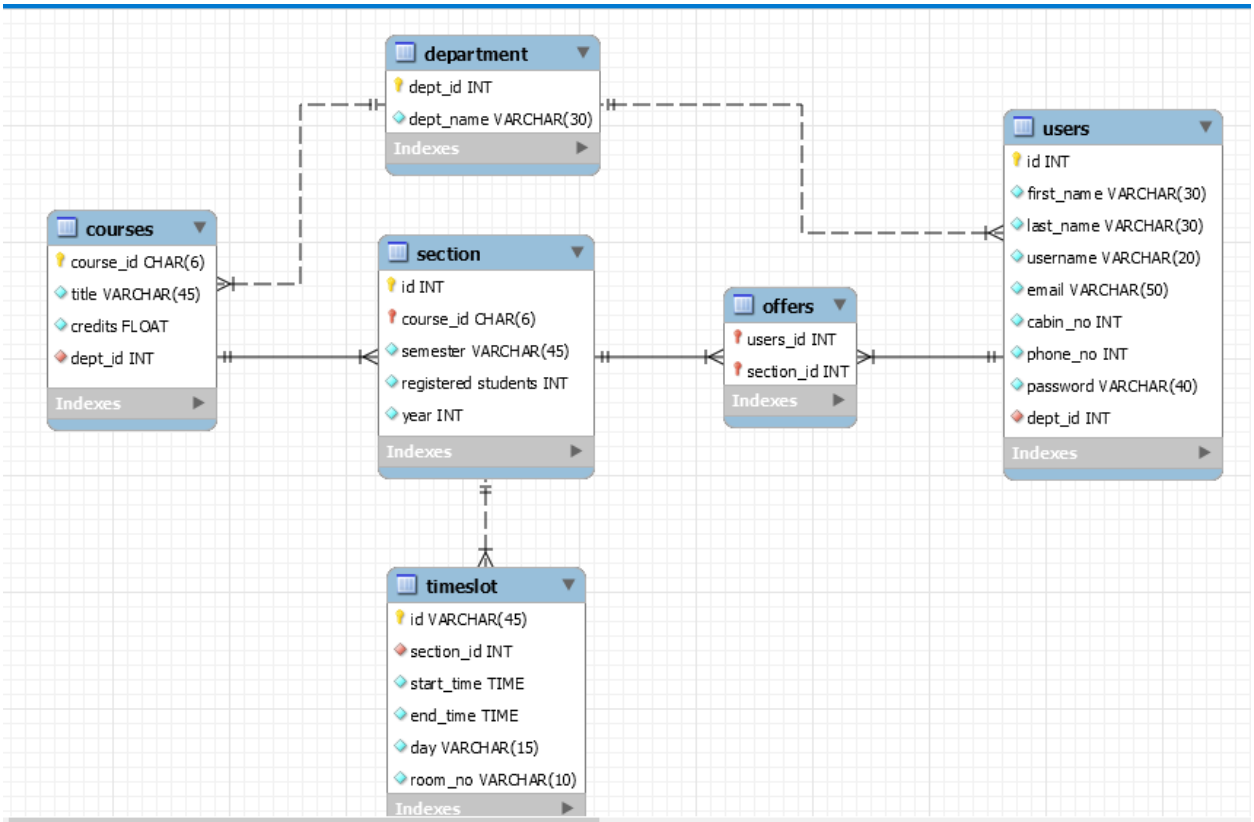
- My Courses- all the courses offered can be viewed with options to delete or edit a specific course. Delete will remove the course from the list and edit helps to change the room number or timings.
- All courses -gives a list of all courses approved by the admin with a filter in order to make search simple, it is very efficient and user-friendly with ample dynamic features. Also it has a plus button which helps the professor to add that course to be offered by him/her.
- Logout- option to get out of the current account and go to home page.

Special Features in this website:

- Dynamic data search using ajax
- Taken care of FCCs
- Ample number of filters
- Precise forms to register or edit data
- Encrypted Passwords in database for more security
- SQL Injection

E-R Diagram :





E-R Diagram Explanation:

In this there are 5 important entities. The tables department and courses are one to many relationship with total participation from courses. The department and users have also a one to many relationship with every user being in a specific department but every department having many users. Now coming to users-section relationship, we assume that many users can be mapped to a section but each section has at a particular time can be taught by only one professor/user. The relationship between section and courses is a week entity as section depends on course to complete itself. Lastly, for timeslot-section relationship, we know that every timeslot can contain only a particular section at a time as only one course can be taught at one time in a section.

Table	Primary Key
Department	dept_id
Users	Id
Courses	course_id
Section	id
Timeslot	Id

Table	Foreign Key	Referenced Table
Department	-	-
Users	dept_id	Department(dept_id)
Courses	dept_id	Department(dept_id)
Section	course_id,user_id	Courses(course_id),users(id)
Timeslot	section_id	Section(id)

Normal Forms Explanation:

In this, we normalize the data in order to eliminate redundant data and ensure data dependencies make sense. It's tough to handle databases without facing data loss else.

Here, we have used **BCNF** form.

Explanation of BCNF or 3.5NF :

It is a higher version of 3NF and deals with certain anomalies which are not handled well by 3NF as well. The 3NF table which doesn't allow multiple overlapping candidate keys is in BCNF. There is no redundancy based on functional dependency although other types might exist.

A relational schema R is in Boyce–Codd normal form if and only if for every one of its dependencies $X \rightarrow Y$, at least one of the following conditions hold:

- $X \rightarrow Y$ is a trivial functional dependency ($Y \subseteq X$)
- X is a super key for schema R (as quoted in Wikipedia)

In the tables we used, let's go one by one and check for their dependencies and redundancy.

- In the table department, we have two attributes- dept_id and dept_name which are trivial functional dependencies.
- In the next table users and courses, there are multiple(9) attributes which imply superkey relations.
- In the latter tables as well, we observe that all functional dependencies which are either primary keys or superkey, henceforth leading to BCNF form only.