

# **CU – Event Management System**

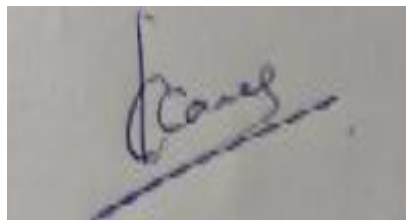
Submitted in partial fulfillment of the requirements for the award of degree of

## **BACHELOR OF ENGINEERING IN COMPUTER SCIENCE & ENGINEERING**



**Submitted to:  
Project Teacher**  
Anjuli Goel

**Submitted By:  
Student Group**  
1. Harshit Srivastava, 19BCS1142  
2. Tanmay Tripathi, 19BCS1120



**Mentor Signature**  
MS Charanpreet Kaur, 6227

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Chandigarh University, Gharuan**

**June 2021**

# **Final Progress Report**

## **Project Design**

This section explains in details how the project was approached. All software components that will be used will be explained in this section of the report.

### **1. Project Overview**

#### **1.1. The Research Stage**

The research stage was a critical stage that provided our team with the knowledge necessary to complete the other stages of our project. This stage was an ongoing process that our team had to return to many times during the development process to gain the knowledge needed to continue on with the project. Our research encompassed a wide range of sources, which included studies done at different universities and hobby enthusiast sources. Our research included topics like HTML, CSS, JavaScript, NodeJS and MongoDB.

#### **1.2. Building & Interfacing Stage**

This stage started when the ordered parts started arriving. During this stage we focused on verifying and testing each and every html, css and JS parts thoroughly. After each part was verified to be working correctly, we combined the components together and front-end part was done.

#### **1.3. The Programming Stage**

In this stage of our project we were to design three different things for our project. They were designing a HTML page, a CSS page and a Javascript page . Unfortunately we did not get very far in this stage and programming for Chatbot and back-end part has yet to be done.

## 2. Technology used

### Front-End :

- **HTML** : It describes the structure of a Web page.
- **CSS** : It describes how HTML elements are to be displayed on screen.
- **JS** : It adds interactivity to website.
- **Bootstrap** : It is the most popular HTML, CSS, and JavaScript framework for developing responsive websites.

### Backend :

- **Nodejs** : It is a JavaScript runtime environment that executes JavaScript code outside a web browser. It can create, open, read, write, delete, and close files on the server.
- **Nodemon** : It is a tool that helps develop node. js based applications by automatically restarting the node application when file changes occurs in the directory
- **Express** : It is a minimal and flexible Node.js web application framework that allows to dynamically render HTML Pages based on passing arguments to templates.
- **Mongoose** : It manages relationships between data, provides schema validation, and is used to translate between objects in code and the representation of those objects in MongoDB.
- **MongoDB** : It is a schema-less NoSQL document database.

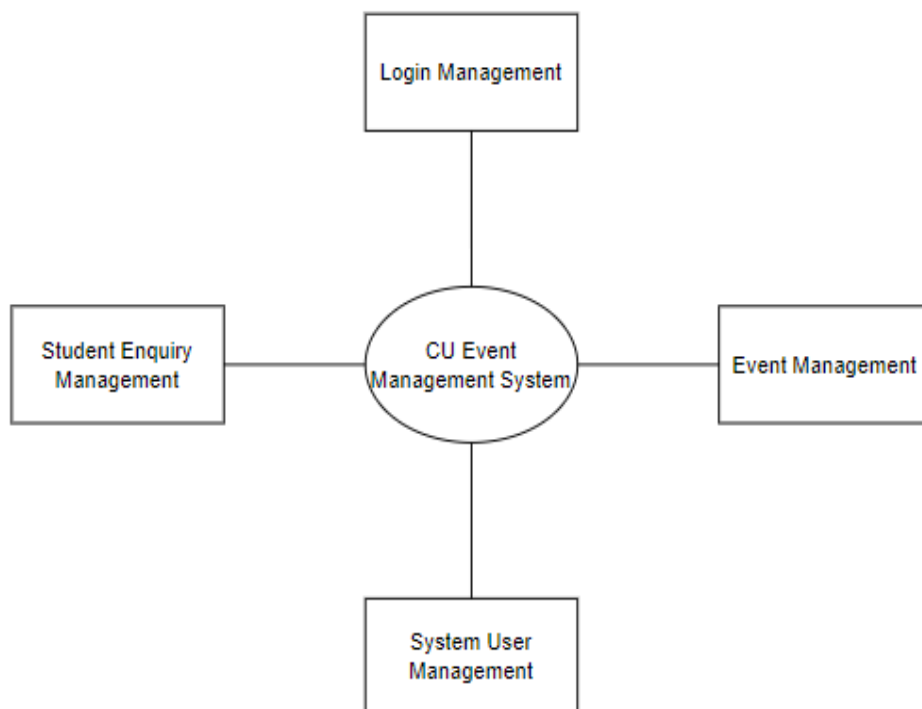
## Innovation in model/design/solution

### 1. Data Flow Diagram (DFD) for CU-EMS

A data-flow diagram is a way of representing a flow of data through a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself.

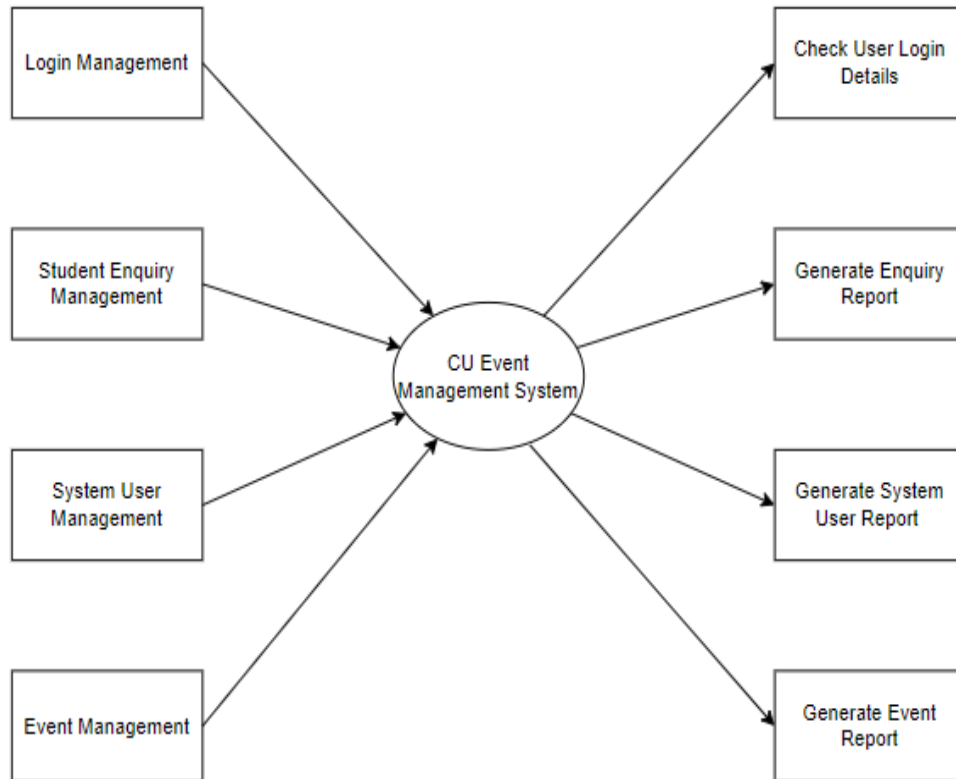
#### 1.1. Zero Level DFD for CU-EMS

This is the zero level DFD of event management system, where we have elaborated the high level process of Event management system. It's a basic overview of the whole event management system or process being analyzed or modeled.



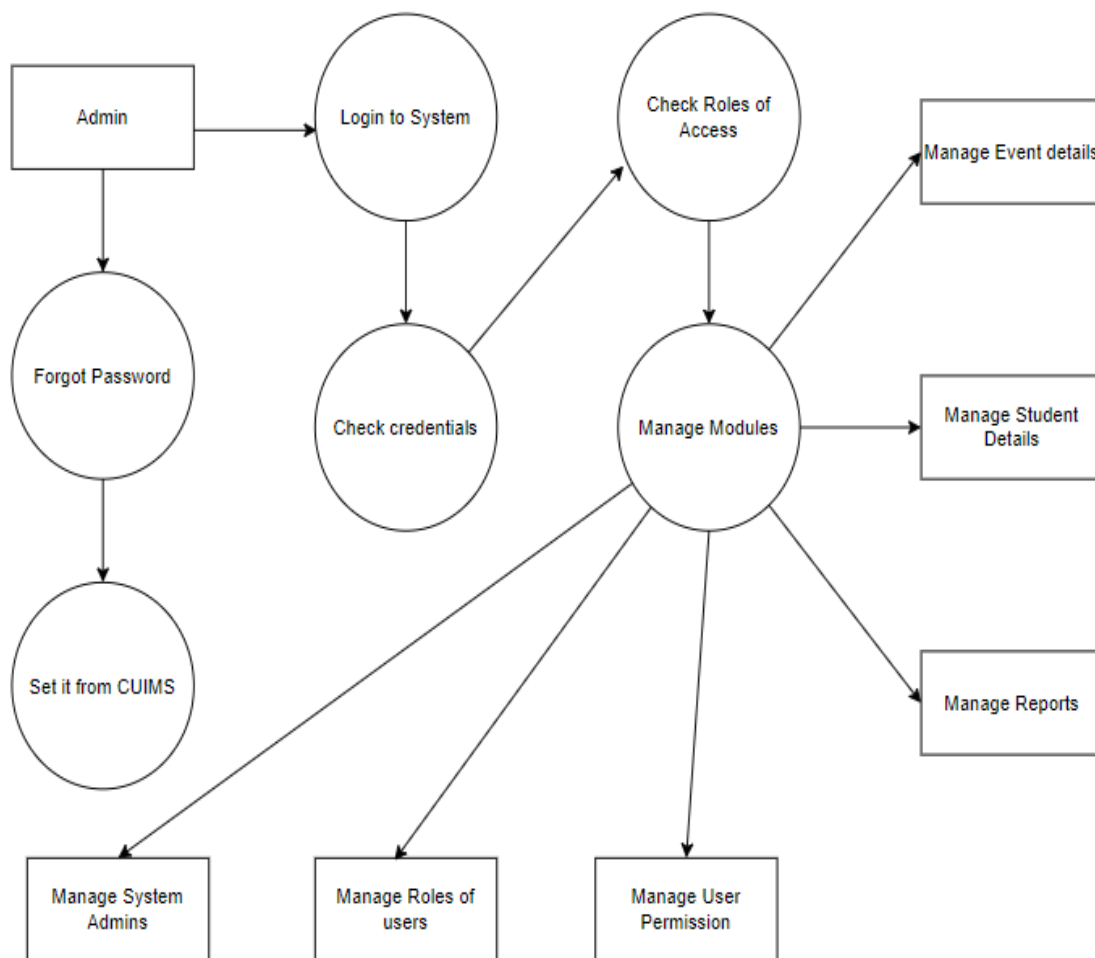
## 1.2. First Level DFD for CU-EMS

First Level DFD (1st Level) of Event Management System shows how the system is divided into subsystems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the Event Management System as a whole.



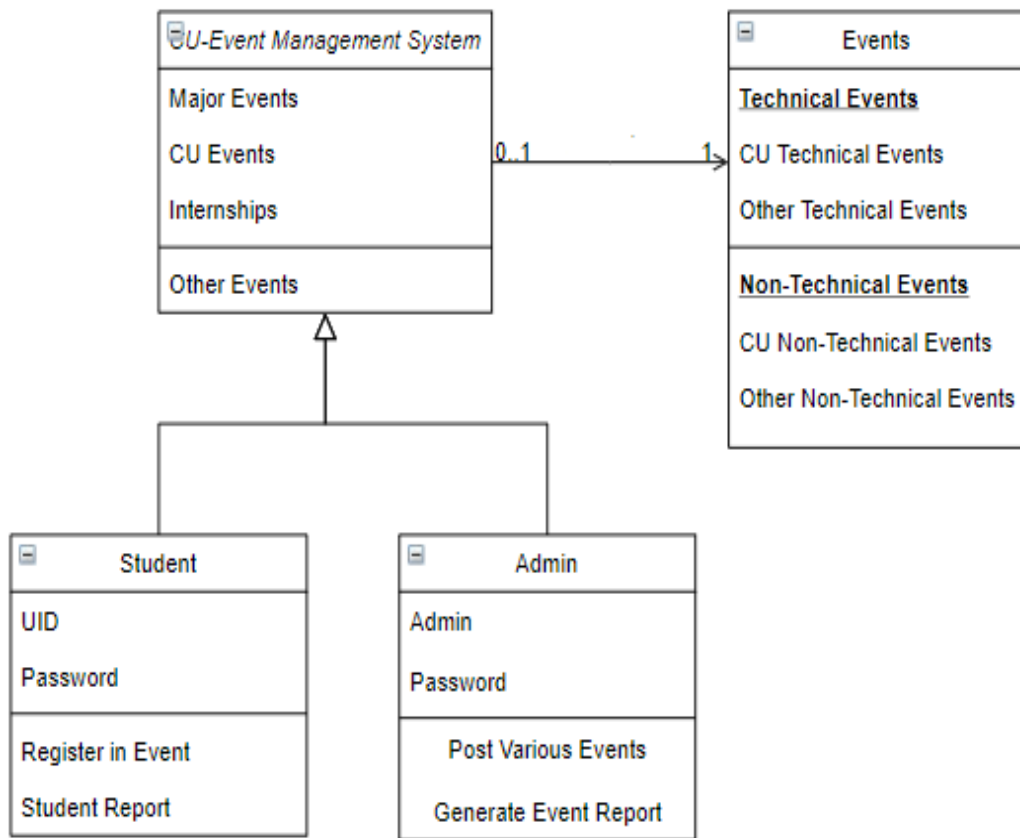
### 1.3. Second Level DFD for CU-EMS

DFD Level 2 then goes one step deeper into parts of Level 1 of Event Management. It may require more functionalities of Event Management to reach the necessary level of detail about the Event Management functioning. First Level DFD (1st Level) of Event Management System shows how the system is divided into sub-systems (processes). The 2nd Level DFD contains more details of First Level DFD Event Management System.



## 2. Class Diagram for CU-EMS

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the structure of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling.



# 1. Implementation

## Logo



Logo for CU event management system

### 1.1. Landing page :

- This is the landing page of Chandigarh University – Event Management System (CU-EMS).
- In this page, there are two buttons – one for student login and other is for admin login.

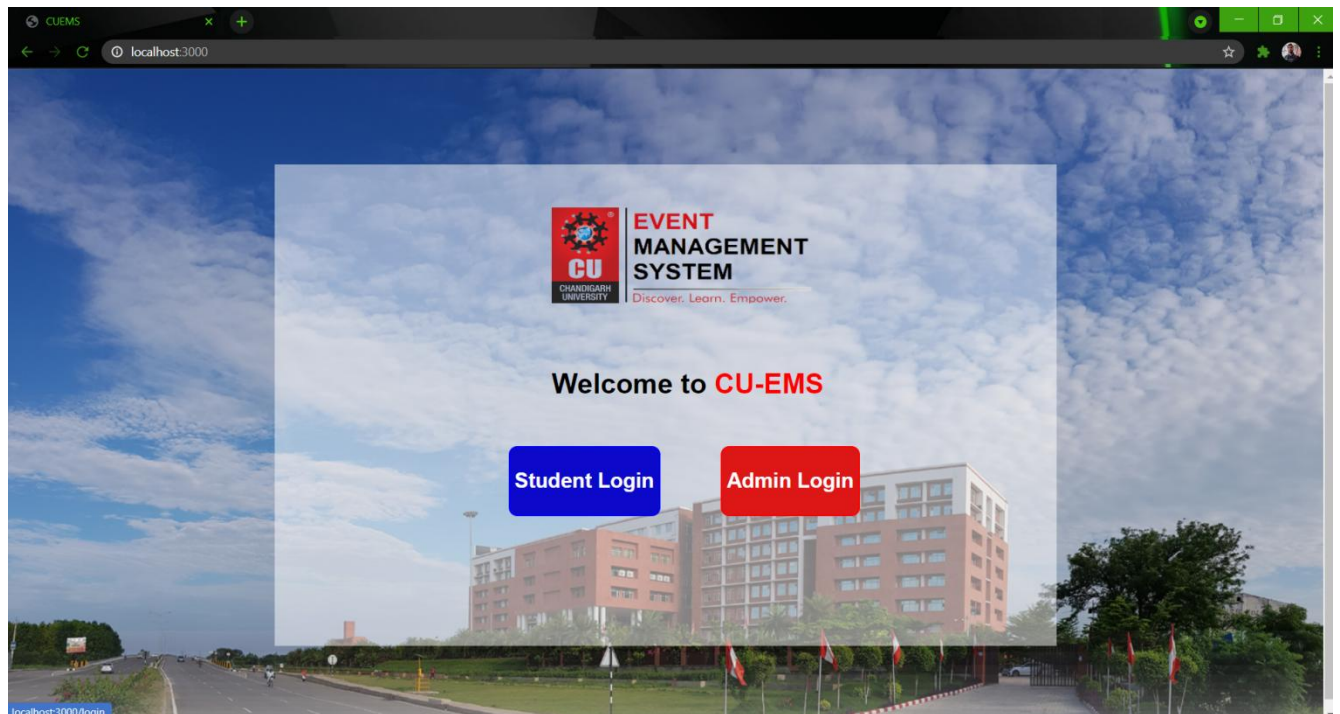
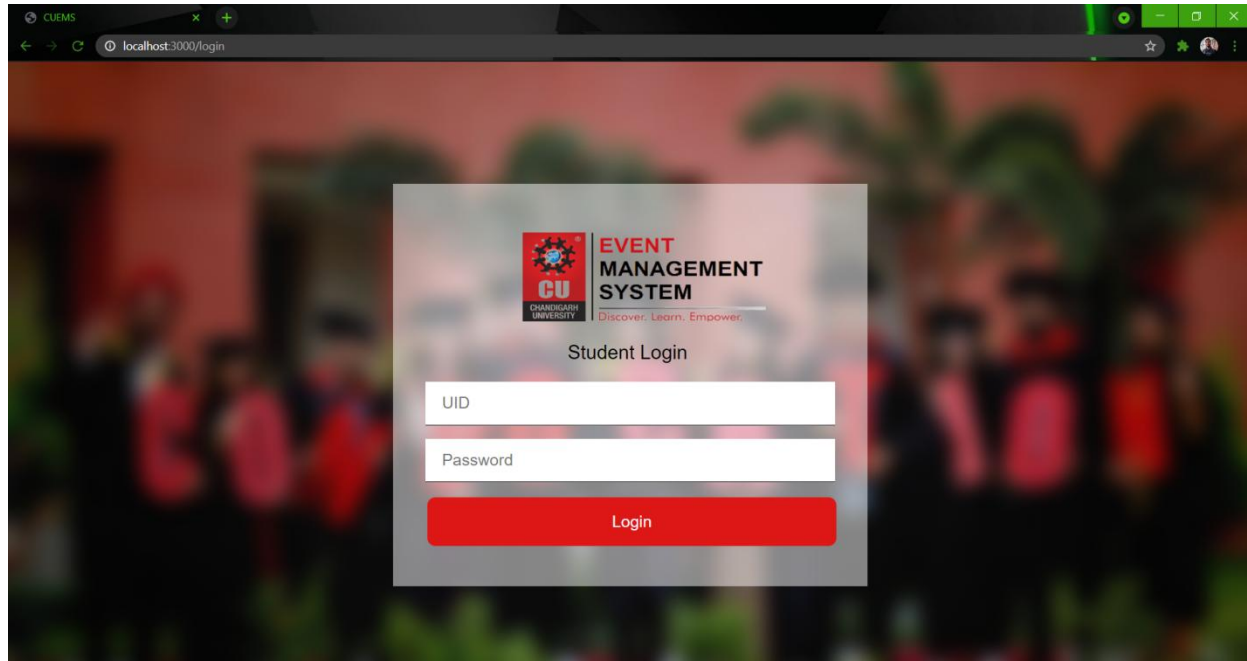


Figure 1 : CU-EMS Landing page



## 1.2. Student :

### 1.2.1. Student Login page :



**Figure 2 : CU-EMS Student Login page**

### 1.2.3. Student Home page :



**Figure 3 : CU-EMS Student Homepage**

### 1.3. Admin :

#### 1.3.1. Admin Login Page :

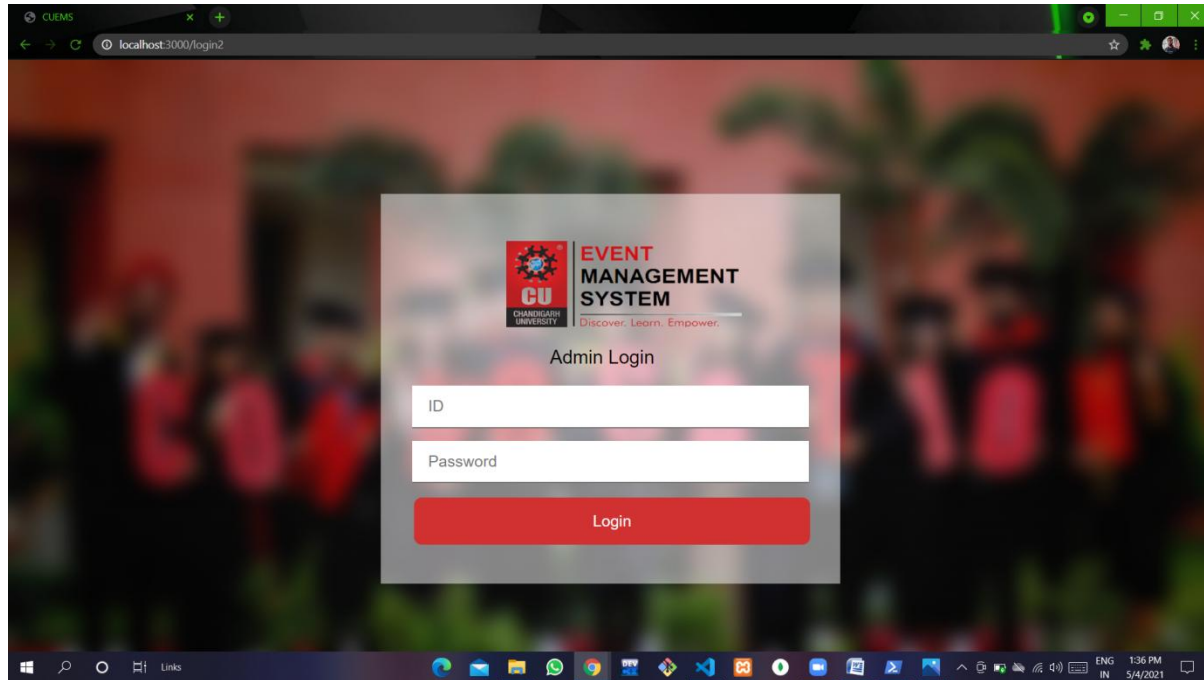


Figure 4 : CU-EMS Admin Login page

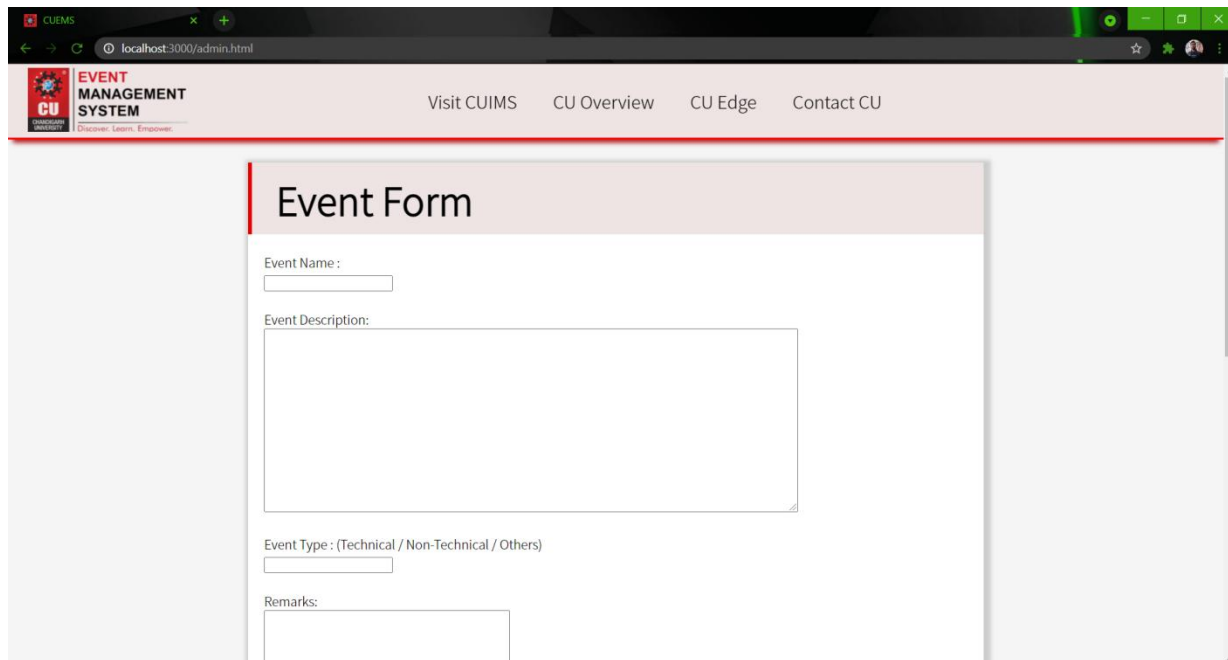
#### 1.3.2. Admin Home Page :



Figure 5 : CU-EMS Admin Homepage

### 1.3.3. Admin Event Form Page :

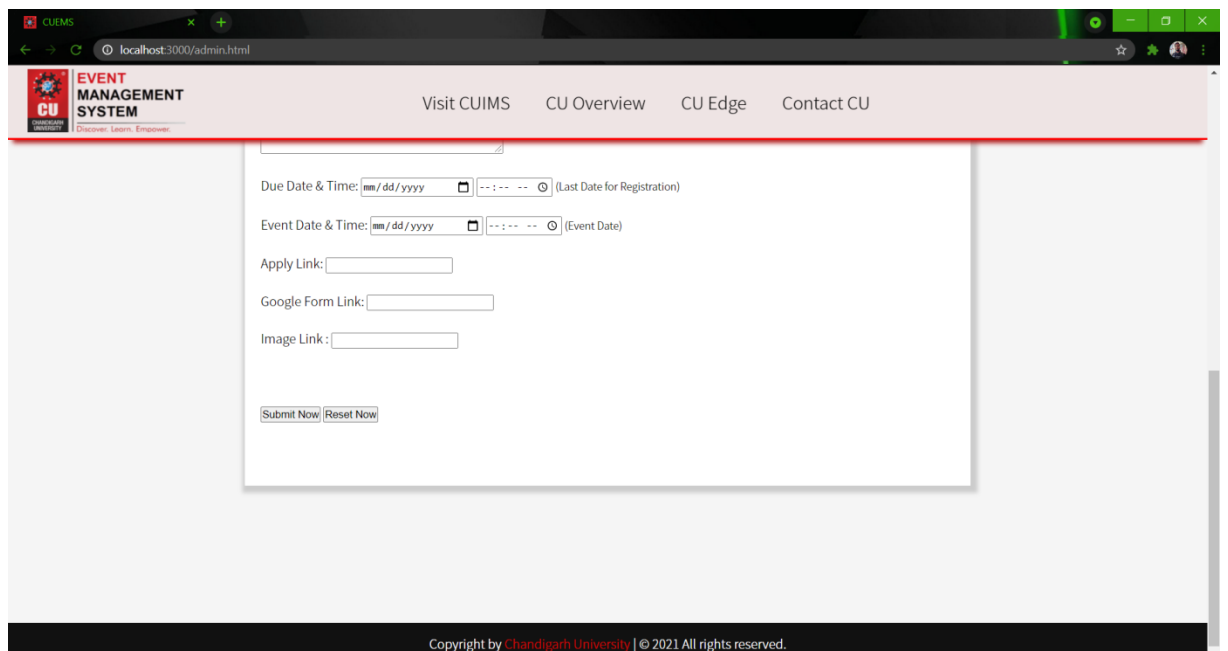
When admin clicks on the “Post” Button which is in the top-right side of the admin homepage, it redirects it to an Event form page where admin can post a new upcoming event.



The screenshot displays the 'Event Form' page in a web browser. The browser's address bar shows 'localhost:3000/admin.html'. The page header includes the 'EVENT MANAGEMENT SYSTEM' logo and navigation links: 'Visit CUIMS', 'CU Overview', 'CU Edge', and 'Contact CU'. The main content area is titled 'Event Form' and contains the following fields:

- Event Name :** A single-line text input field.
- Event Description:** A large multi-line text area.
- Event Type : (Technical / Non-Technical / Others)** A single-line text input field.
- Remarks:** A single-line text input field.

**Figure 6 : CU-EMS Admin Event Form Page (1/2)**



The screenshot displays the bottom section of the 'Event Form' page. It includes the following fields and controls:

- Due Date & Time:** A date and time picker with a calendar icon and a clock icon. The text '(Last Date for Registration)' is displayed to the right.
- Event Date & Time:** A date and time picker with a calendar icon and a clock icon. The text '(Event Date)' is displayed to the right.
- Apply Link:** A single-line text input field.
- Google Form Link:** A single-line text input field.
- Image Link :** A single-line text input field.
- Submit Now** and **Reset Now** buttons.

The footer of the page contains the text: 'Copyright by Chandigarh University | © 2021 All rights reserved.'

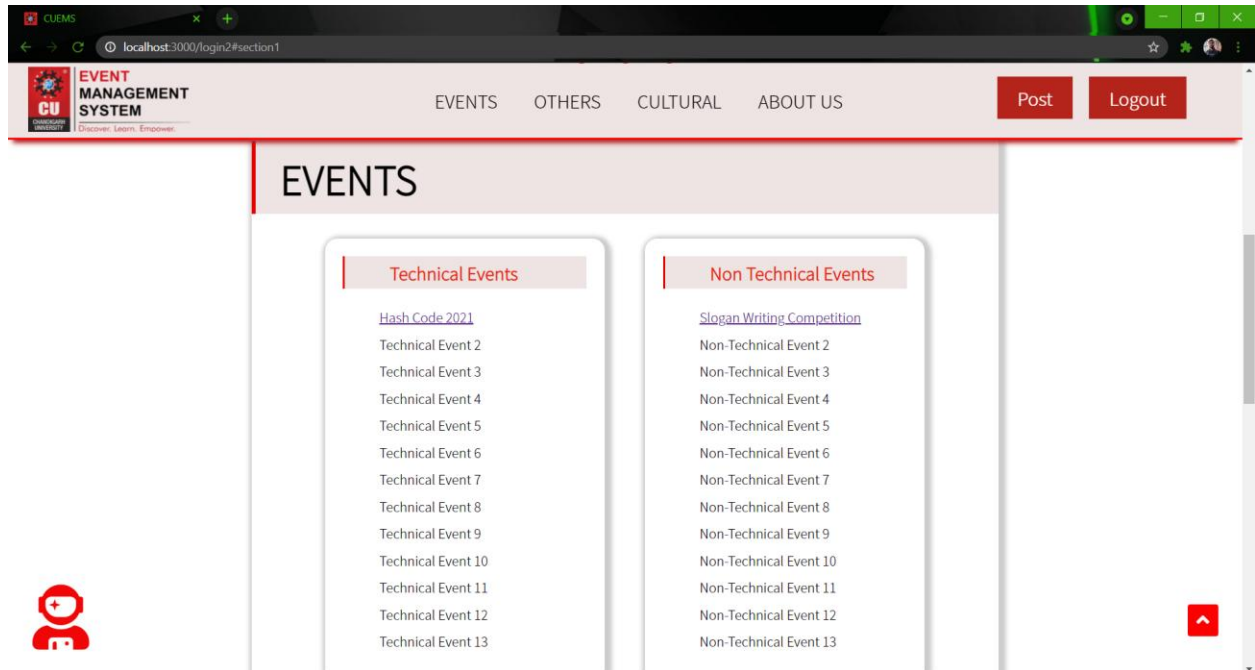
**Figure 7 : CU-EMS Admin Event Form Page (2/2)**

## 1.4. Common Homepage :

In the homepage be of admin or student there are main 3 menus in the header part – EVENTS, OTHERS, CULTURAL.

### 1.4.1. EVENTS :

When you click on the “EVENTS”, it redirects you to #section1 on the same page which can be seen below.



**Figure 8 : CU-EMS after clicking on EVENTS**

### 1.4.2. OTHERS :

When you click on the “OTHERS”, it redirects you to #section2 on the same page which can be seen below.

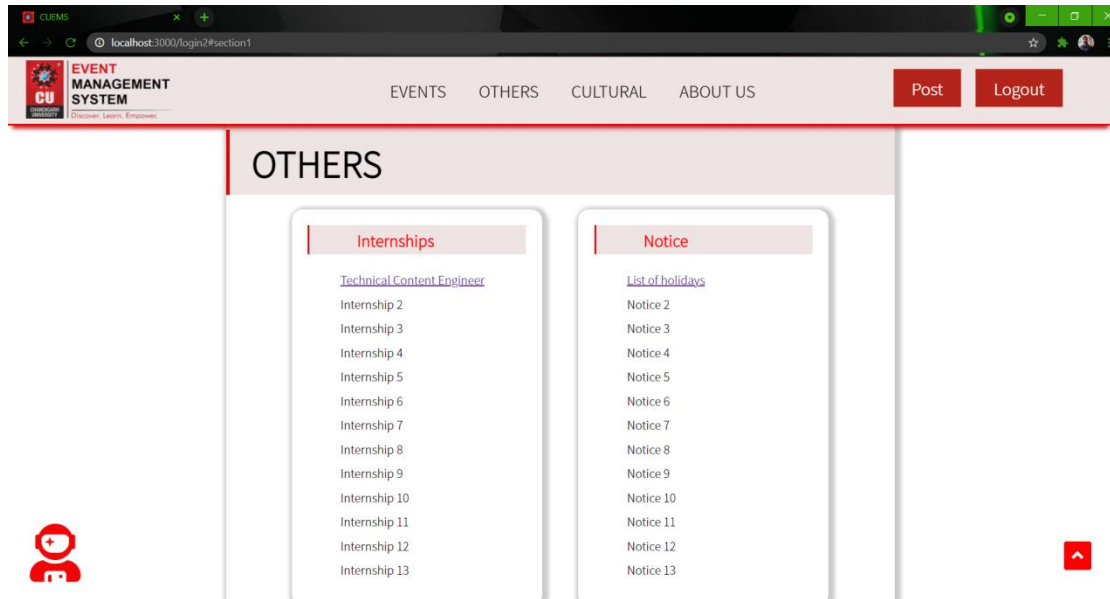


Figure 9 : CU-EMS after clicking on OTHERS

### 1.4.3. CULTURAL :

When you click on the “CULTURAL”, it redirects you to a page where all the events regarding cultural fests will be notified.

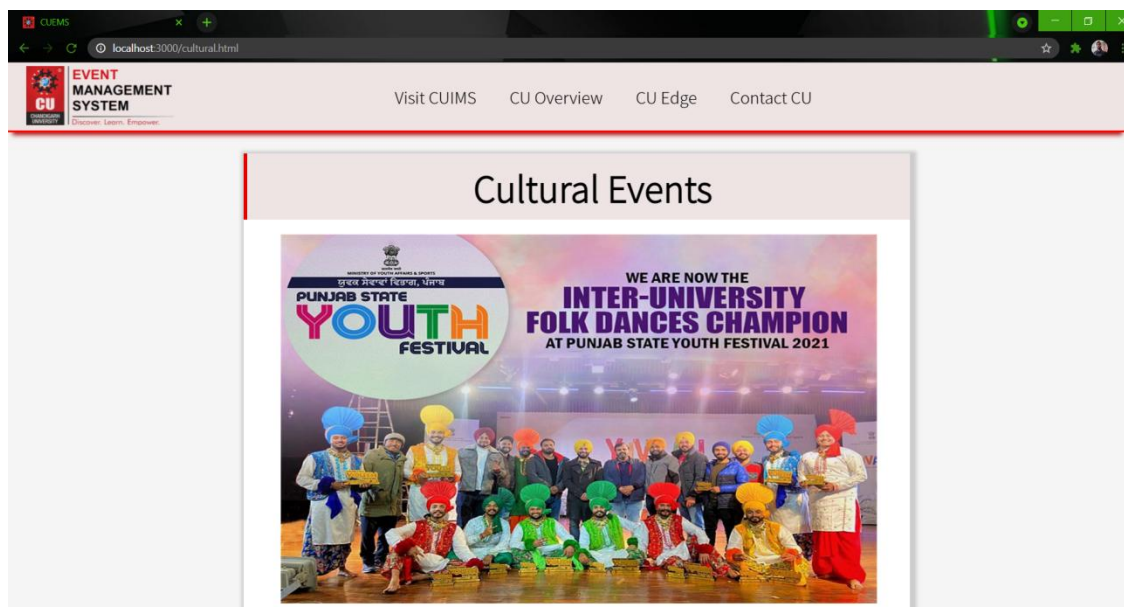
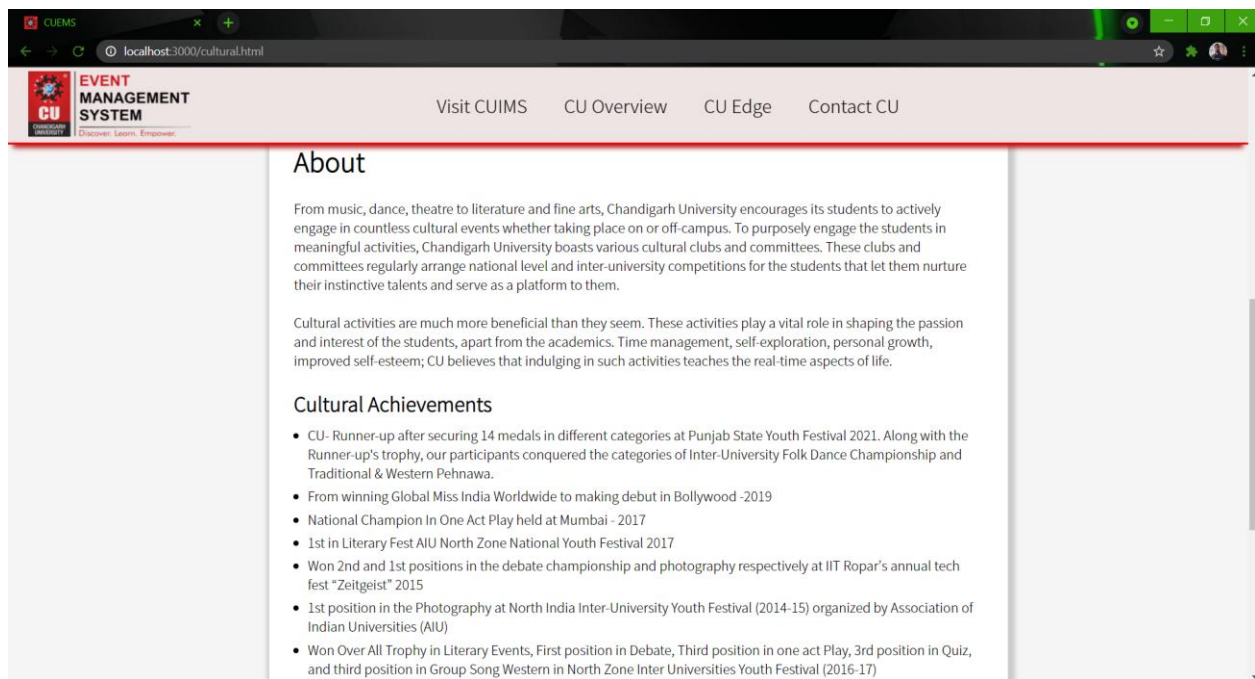
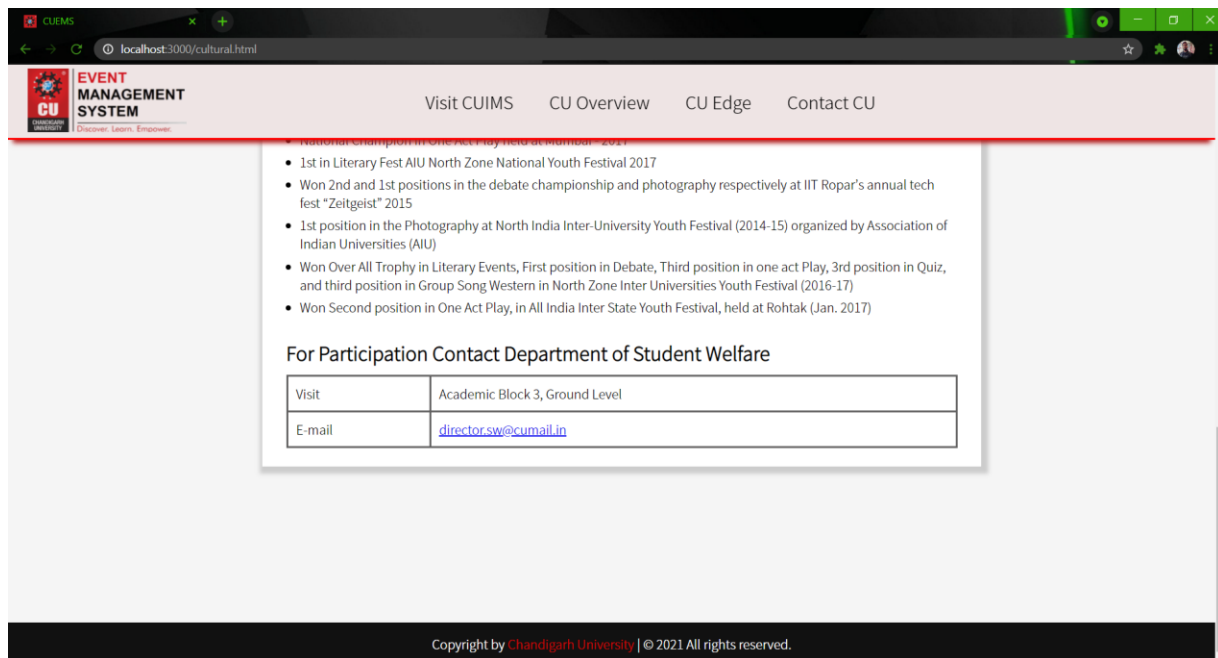


Figure 10 : CU-EMS Cultural Page (1/3)



**Figure 11 : CU-EMS Cultural Page (2/3)**



**Figure 12 : CU-EMS Cultural Page (3/3)**



## 1.5. Types of Events :

### 1.5.1. Technical Events :

Below is the sample page of a Technical Event.

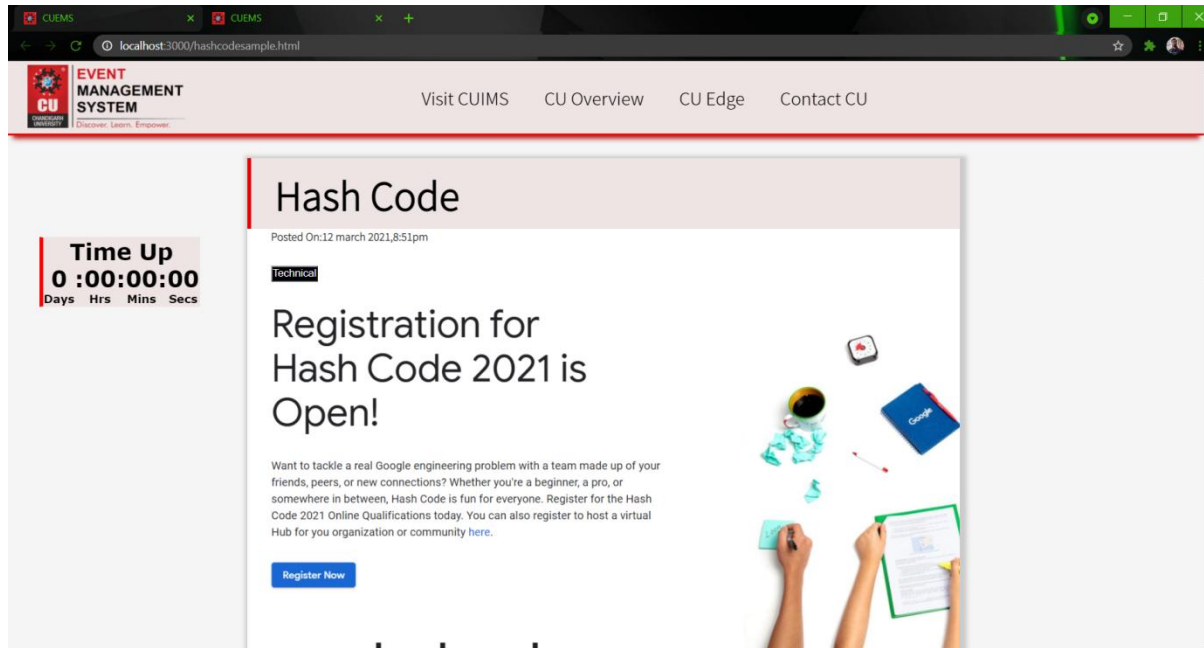


Figure 13 : CU-EMS sample technical event page (1/3)

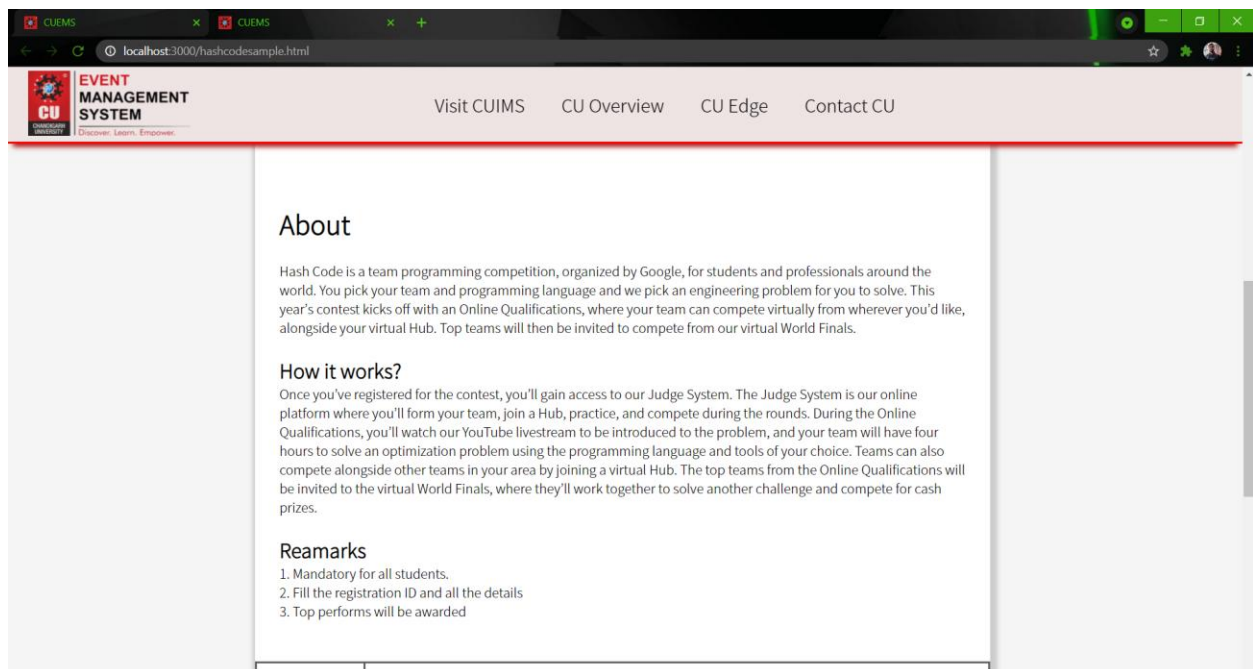
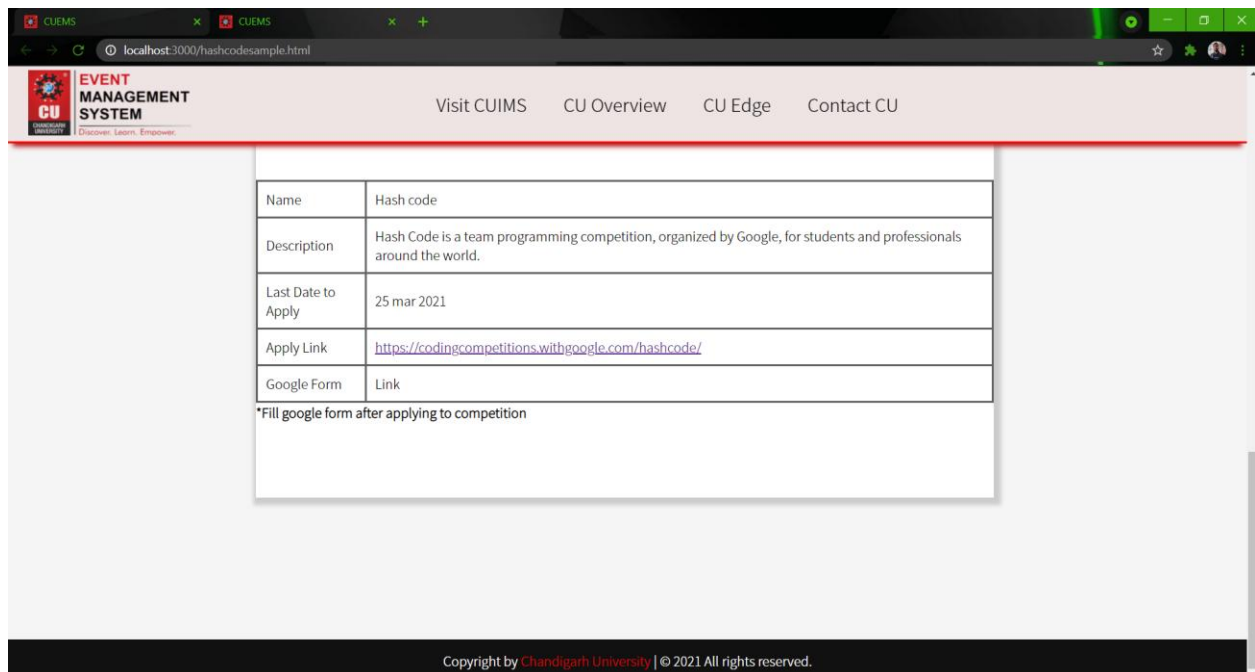


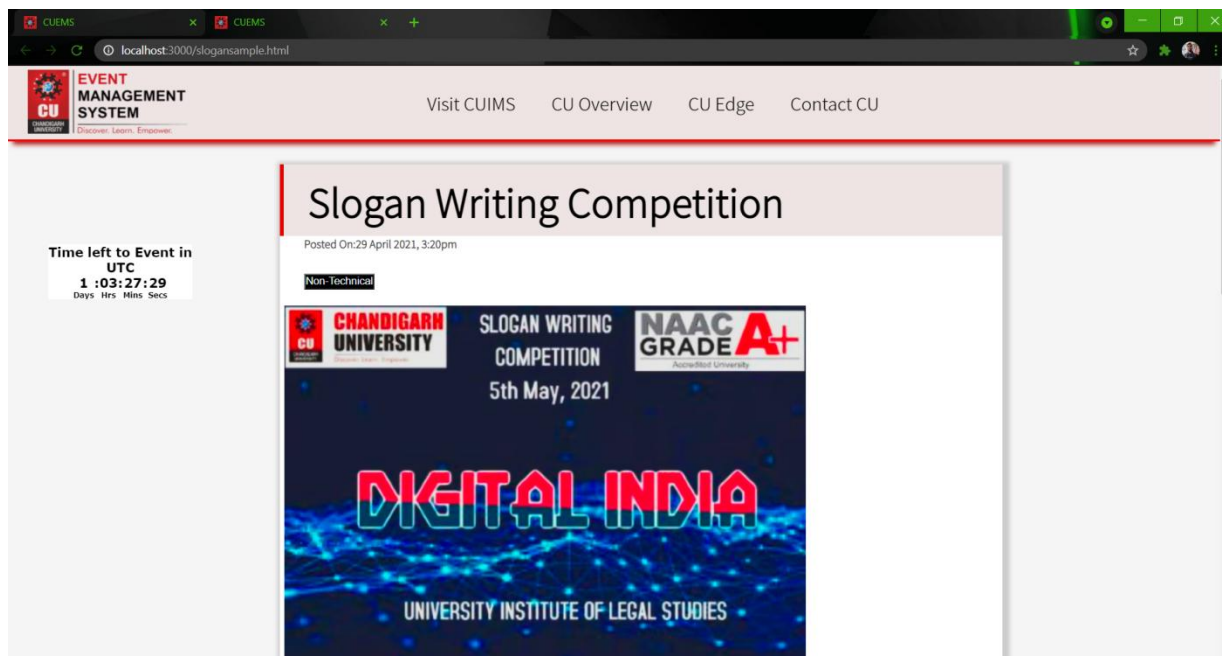
Figure 14 : CU-EMS sample technical event page (2/3)



**Figure 15 : CU-EMS sample technical event page (3/3)**

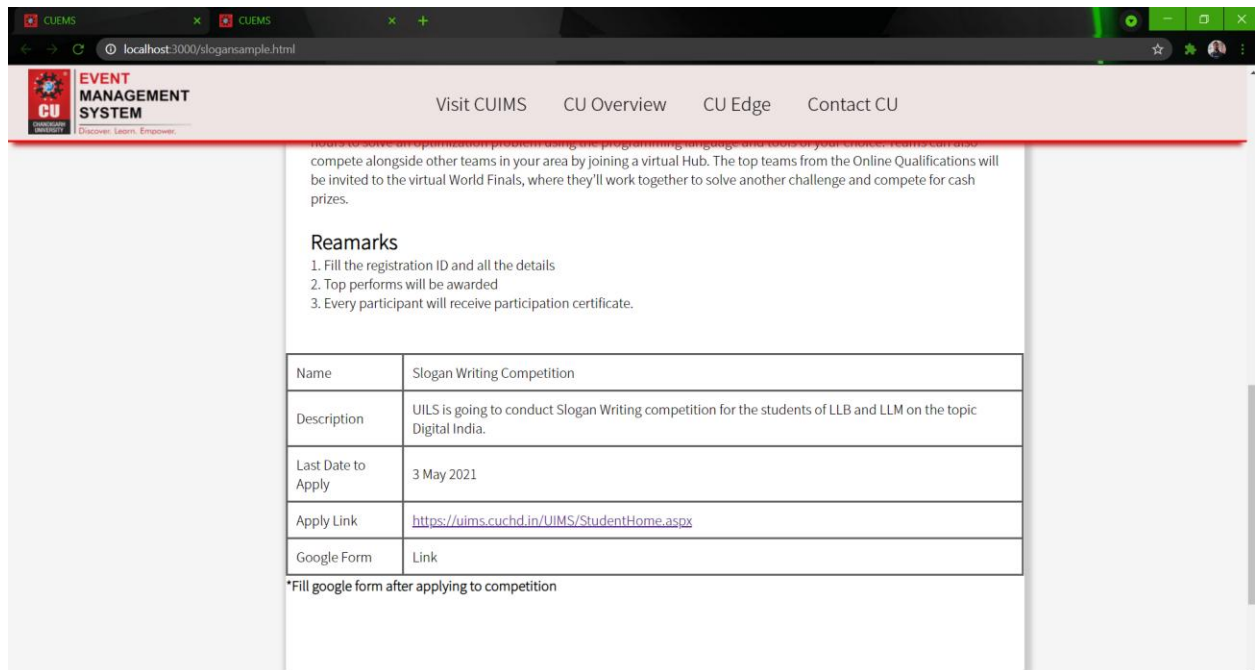
## 1.5.2. Non-Technical Events :

Below is the sample page of a Non-Technical Event.



**Figure 16 : CU-EMS sample non-technical event page (1/2)**



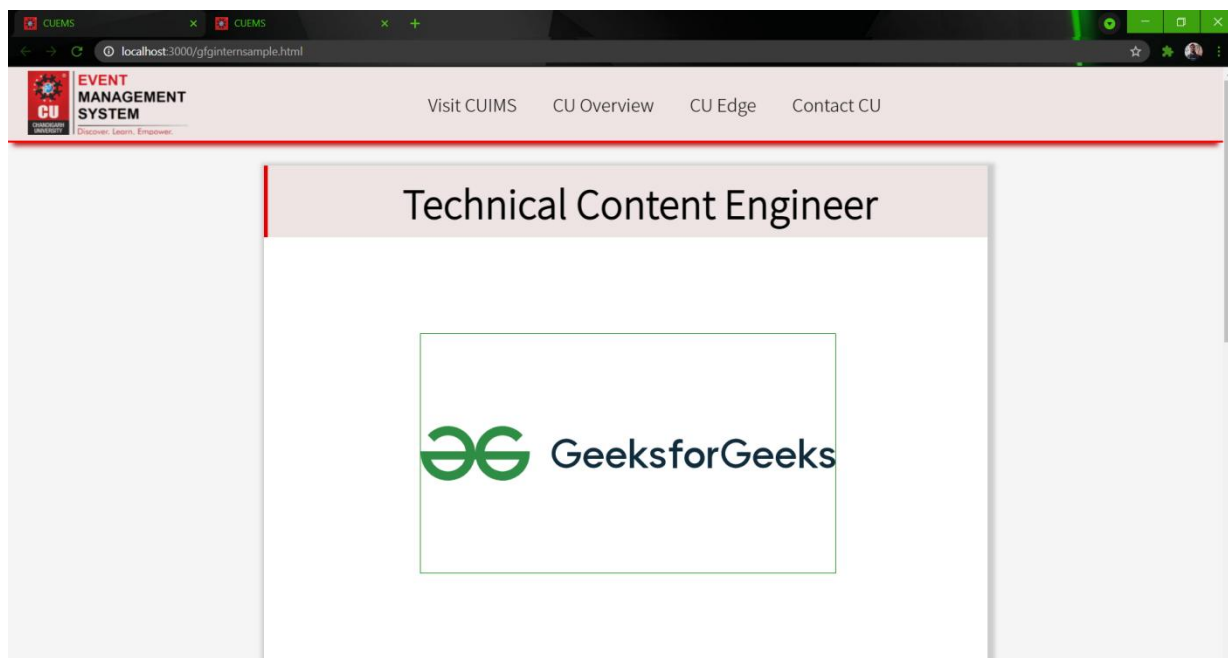


**Figure 17 : CU-EMS sample non-technical event page (2/2)**

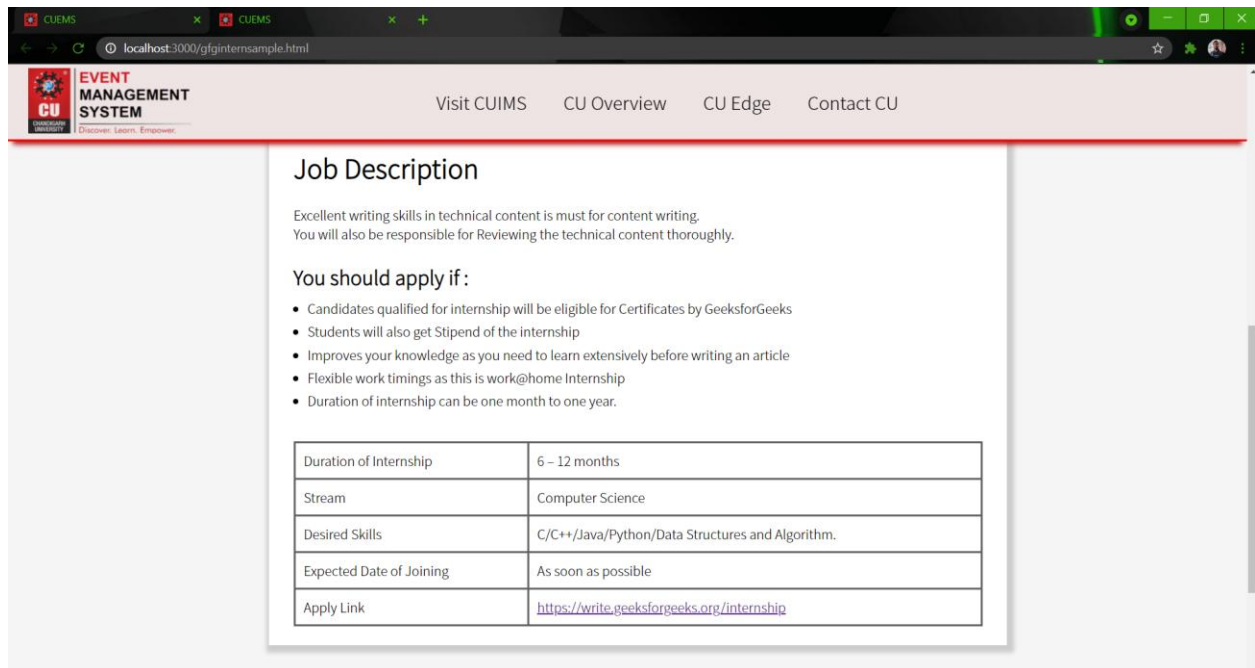
## 1.6. Others :

### 1.6.1. Internships :

Below is the sample page of Internship.



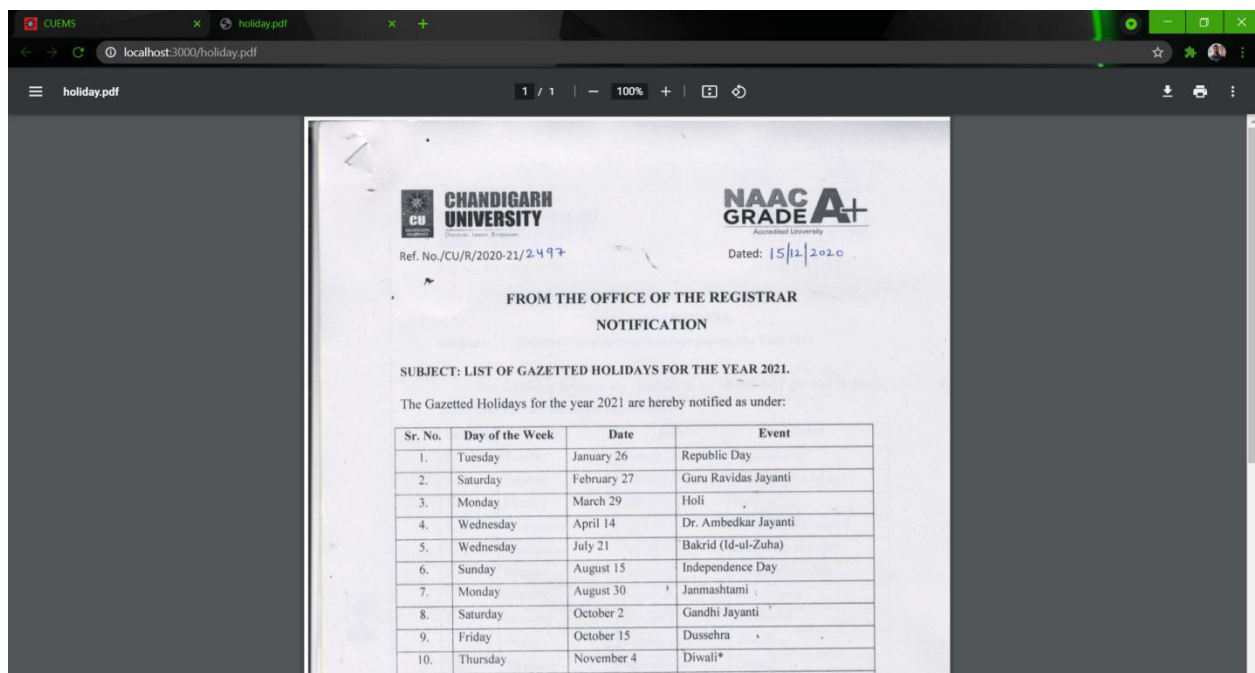
**Figure 18 : CU-EMS sample internship page (1/2)**



**Figure 19 : CU-EMS sample internship page (2/2)**

## 1.6.2. Notices :

Below is the sample of a notice. It redirects to another page which contains the direct pdf file of notice.

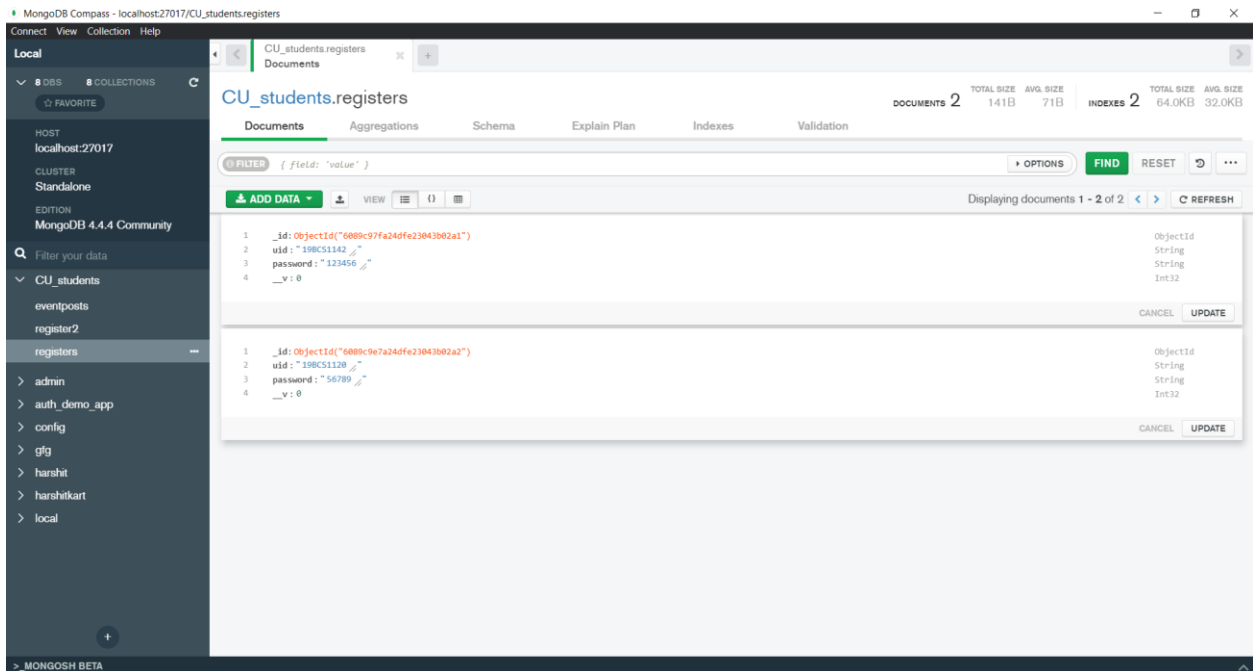


**Figure 20 : CU-EMS sample notice redirecting page**

## 2. Output validation and comparison

### 2.1. Student Login page validation with backend

#### 2.1.1. Student's sample database of login credentials



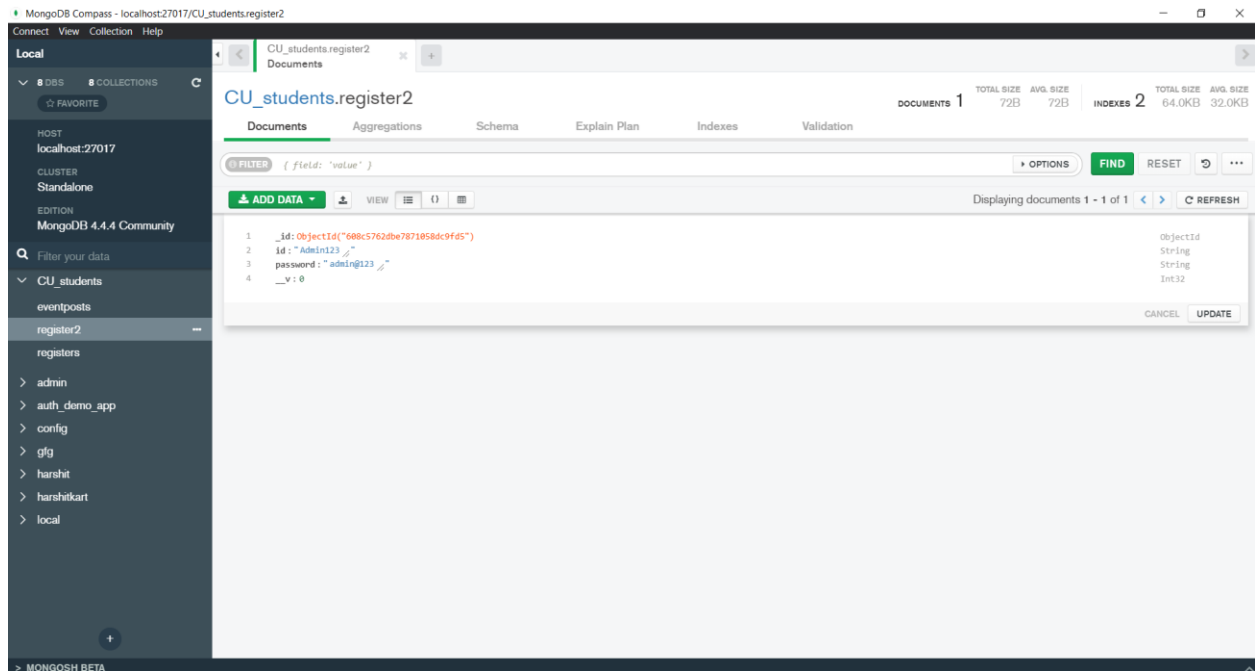
**Figure 21 : CU-EMS Student Database**

#### 2.1.2. Validation/Verification of the above login credentials

```
app.post("/login", async (req,res) => {
  try {
    const uid = req.body.uid;
    const password = req.body.password;
    const studentuid = await Register.findOne({uid:uid});
    if(studentuid.password === password){
      res.status(201).render("index2");
    }else{
      res.send("Invalid password");
    }
  } catch (error) {
    res.status(400).send("Invalid UID");
  }
});
```

## 2.2. Admin Login page validation with backend :

### 2.2.1. Admin's sample database of login credentials :



**Figure 22 : CU-EMS Admin Database**

### 2.2.2. Validation/Verification of the above login credentials :

```
app.post("/login2", async (req,res) => {
  try {
    const id = req.body.id;
    const password = req.body.password;
    const adminid = await Register2.findOne({id:id});
    if(adminid.password === password){
      res.status(201).render("index3");
    }else{
      res.send("Invalid password");
    }
  } catch (error) {
    res.status(400).send("Invalid ID");
  }
});
```

## 2.3. Validation with backend for posting a new Event :

### 2.3.1. Saving the event posted by admin in the database :

**Input :**

*eventpost.js*

```
const mongoose = require("mongoose");
const studentSchema = new mongoose.Schema({

  myName : {
    type:String,
    required:true
  },
  myEvent : {
    type:String,
    required:true
  },
  eventType :{
    type:String,
    required:true
  },
  myRemark :{
    type:String,
    required:true
  },
  myDate :{
    type:String,
    required:true
  },
  appt :{
    type:String,
    required:true
  },
  myDate1 :{
    type:String,
    required:true
  },
  appt1 :{
    type:String,
    required:true
  },
  myLink :{
    type:String,
    required:true
  },
  myLink1 :{
    type:String,
```

```

        required:true
    },
    img :{
        type:String,
        required:true
    }
})

const eventpost = new mongoose.model("eventpost", studentSchema);
module.exports=eventpost;

```

The below line must(compulsory) in the html code of the event form :

```
<form action="/eventpost" method="POST">
```

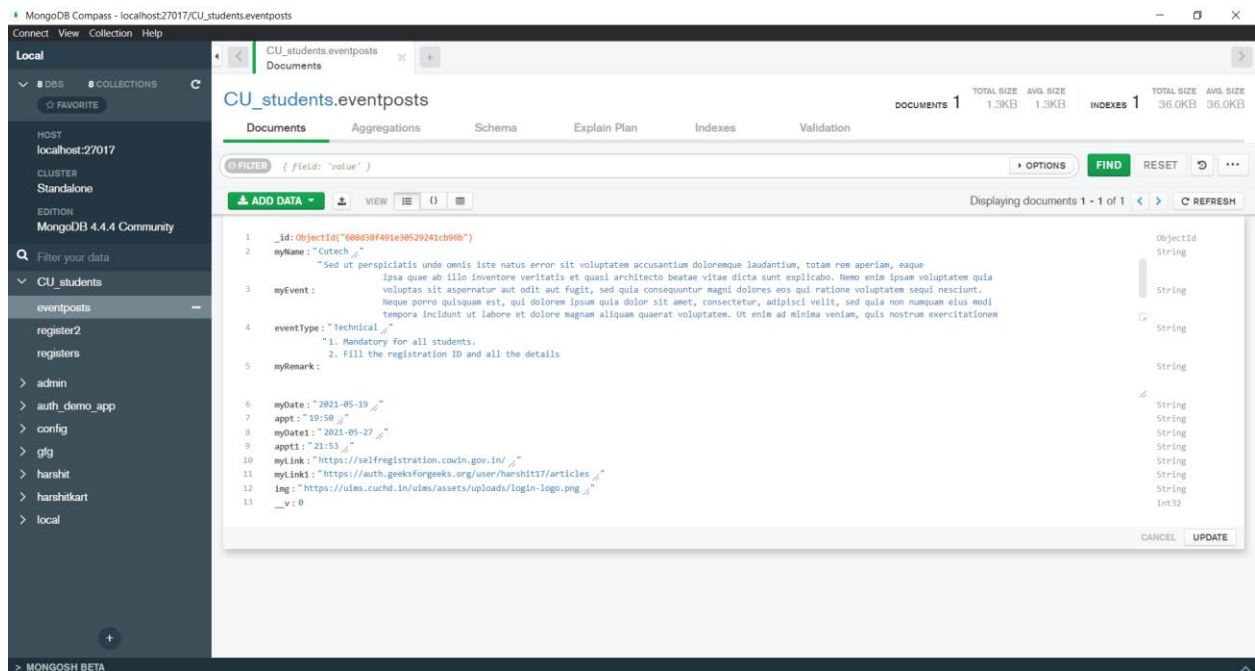
### 2.3.2. Validation of the event form

```

app.post("/eventpost", async (req,res) => {
    try {
        const registerevent = new eventpost({
            myName : req.body.myName,
            myEvent : req.body.myEvent,
            eventType :req.body.eventType,
            myRemark :req.body.myRemark,
            myDate :req.body.myDate,
            appt :req.body.appt,
            myDate1 :req.body.myDate1,
            appt1 :req.body.appt1,
            myLink :req.body.myLink,
            myLink1 :req.body.myLink1,
            img :req.body.img
        })
        const registeredevent = await registerevent.save();
        res.status(201).render("index3");
    } catch (error) {
        res.status(400).send(error);
    }
});

```

### 2.3.3. Verification of the event form :



**Figure 23 : CU-EMS Database of upcoming events made by admin**

The event gets saved in the database and after the event's details are verified, we will be posting in the site.

## **Team Work :**

<b>Name</b>	<b>UID</b>	<b>Class/Sec</b>	<b>Tasks Done</b>
Harshit Srivastava	19BCS1142	CSE 2A	<ul style="list-style-type: none"><li>• Planning, designing and implementation of validation and verification part of the project</li><li>• Front End part using HTML, CSS, JS and Bootstrap.</li><li>• Backend part of the website using Nodejs, Nodemon, Express, Mongoose and MongoDB</li><li>• Documentation</li></ul>
Tanmay Tripathi	19BCS1120	CSE 1C	<ul style="list-style-type: none"><li>• Planning, designing and leading project to end.</li><li>• Designing and development of the front end pages using HTML, CSS, and JS</li><li>• Worked on things like tags, keywords, logo and timers using complex JS libraries.</li><li>• Documentation</li></ul>

## **Conclusion**

Chandigarh University Event Management System is a web page initially for students at Chandigarh University for getting them notified about all competitions and events held in college and outside college which are technical and non-technical. Most of you may be familiar with site like sarkariresult.com which I think most of us had used, basically it is like it.

Main aim of this project to link it with college official information management site, and then on that page teachers, organizations and admins can post events and competition on that page by which it become easier for student to get notified and for admins it become easier to manage records.