**BOOKMYSHOW CLONE**

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR INNOVATIVE WORK

UNDER

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

Submitted by:

**VIBHAV (2K20/CO/481)**

**VIVEK KUMAR THAKUR (2K20/CO/493)**

Under the supervision of:

**PROF. ROHIT BENIWAL**

Department of CSE, DTU



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi – 110042

APRIL 2022

**CANDIDATE’S DECLARATION**

We, Vibhav (2K20/CO/481) and Vivek Kumar Thakur (2K20/CO/493), students of B.TECH (COE) declare that the MTE Project Report titled “**BookMyShow Clone”** which is submitted by us to the Department of Computer Science and Engineering, Delhi Technological University, Delhi is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma, Fellowship, or other similar title or recognition.

**Place: DTU, Delhi VIBHAV – 2K20/CO/481**

**Date: 25th April 2022 VIVEK KUMAR THAKUR – 2K20/CO/493**

**CERTIFICATE**

I, hereby certify that the Project titled “**BookMyShow Clone**” submitted by Vibhav (2K20/CO/481) and Vivek Kumar Thakur (2K20/CO/493), to Department of Computer Science and Engineering, Delhi Technological University, Delhi, as part of Innovative Work is a record of project work carried out by the student under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

**Place: DTU, Delhi Prof. ROHIT BENIWAL**

**Date: 25th April 2022 (Assistant Professor, CSE, DTU)**

**ACKNOWLEDGEMENT**

We are very thankful to Prof. Rohit Beniwal (Assistant Professor, Computer Science Eng. Dept.) and all the faculty members of the Computer Science Engineering Dept. of DTU. They all provided immense support and guidance for the completion of the project undertaken by us. It is with their supervision that this work came into existence. We would also like to express our gratitude to the university for providing the laboratories, infrastructure, test facilities, and environment which allowed me to work without any obstructions. We would also like to appreciate the support provided by our lab assistants, seniors, and peer group who aided us with all the knowledge they had regarding various topics.

**VIBHAV(2K20/CO/481)**

**VIVEK KUMAR THAKUR (2K20/CO/493)**

**ABSTRACT**

BookMyShow is an online ticket booking platform for various events like concerts, movies, sporting events, etc. What it does is that it maintains an online server, a graphical user interface and a billing platform. With the help of GUI, a user can navigate their website and select an event, book his/her ticket by entering his/her details and then make payment. BookMyShow maintains a database of events, upcoming and past. Each event has its own details like when it is going to happen, at what location the event will take place, how many tickets can be booked, how many people have booked their tickets, etc. In this project we have tried to implement this model of maintaining a database, and providing a minimalistic GUI to user where he/she can enter his/her details. However, we have skipped the billing platform as it was too complex to implement.

**CONTENTS**

1. Introduction
2. Objectives
3. System Requirements and Software used
4. Concept of Database Management System used
5. Implementation and Execution
6. Limitations
7. References

**Introduction**

In this project we tried to implement a clone of the popular ticket booking site ‘BookMyShow’. The motive behind this project is to apply the concepts we have learnt in our DBMS course along with the new frontend development skills we have acquired in our college life. The project has been implemented with the help of php, javaScript, html, CSS and SQL server. The core idea is that individuals can book tickets for an event like a movie or a concert online without any hassle. However, the number of tickets is limited so they have to hurry and after the number of tickets booked reaches a limit, further tickets cannot be booked.

We have utilized the concepts of Relational Database Management Systems and store the data in forms of tables. A table has been created which contains the list of events and then subsequent tables have been created to store data the individuals who have booked a ticket for a particular event.

**Objectives**

* To create and maintain a database which stores the information about events with the help of SQL server and phpMyAdmin.
* To create a graphical user interface, where users can enter their details in order to book their tickets, with the help of html, javaScript and CSS.
* To generate SQL queries from the details the user have entered and then execute them to their data can be stored in the database. This ‘linking’ has been performed with the help of php.

**System Requirements and Software Used**

* The program runs on Windows 10 operating system.
* The program should run on 32-bit or 64-bit architecture system, 64-bit

system was used to create the program.

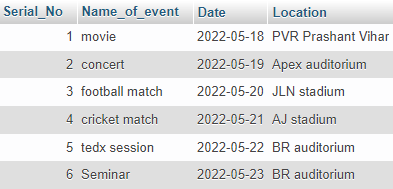
* At Least 2GB system RAM should be available and the Processor should be faster than 1.6 GHz.
* Visual Studio Code has been used for creating programs in php and html.
* XAMPP has been used for inserting the queries generated into the myphpAdmin’s SQL editor.
* Chrome has been used to display the webpages.

**Concepts of Database Management System used:**

1. **Tables in RDBMS**

The data in an RDBMS is stored in database objects which are called as **tables**. This table is basically a collection of related data entries and it consists of numerous columns and rows. A table is the most common and simplest form of data storage in a relational database.

Following is an example of ‘events’ table:



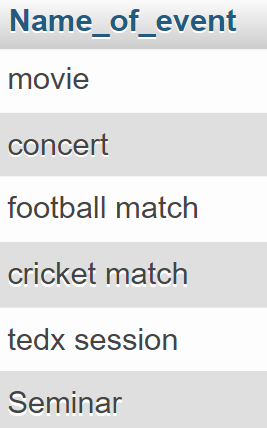
Every table is broken up into smaller entities called attributes or fields. The fields in the ‘events’ table consist of Serial\_No, Name\_of\_event, Date and Location.



A record is also called as a **row** of data is each individual entry that exists in a table.



A **column** is a vertical entity in a table that contains all information associated with a specific field in a table.



1. **Constraints in SQL**

Constraints are the rules enforced on data columns on a table. These are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database.

Constraints can either be column level or table level. Column level constraints are applied only to one column whereas, table level constraints are applied to the entire table.

Following are some of the most commonly used constraints available in SQL −

* NOT NULL Constraint − Ensures that a column cannot have a NULL value.
* DEFAULT Constraint − Provides a default value for a column when none is specified.
* UNIQUE Constraint − Ensures that all the values in a column are different.
* PRIMARY Key − Uniquely identifies each row/record in a database table.
* FOREIGN Key − Uniquely identifies a row/record in any another database table.
* CHECK Constraint − The CHECK constraint ensures that all values in a column satisfy certain conditions.
* INDEX − Used to create and retrieve data from the database very quickly.

1. **Important keywords of SQL used frequently in the project**

* The CREATE DATABASE statement is used to create a new SQL database.
* The CREATE TABLE statement is used to create a new table in a database.
* The CHECK constraint is used to limit the value range that can be placed in a column.

If you define a CHECK constraint on a column it will allow only certain values for this column.

If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.

* The SELECT statement is used to select data from a database. The data returned is stored in a result table, called the result-set.
* The WHERE clause is used to filter records. It is used to extract only those records that fulfill a specified condition.

**Implementation and Execution**

1. First of all, ‘project\_dbms’ database was created.

create database project\_dbms;

1. Then ‘events’ table was created

create table events

(

Serial\_No int(3) primary key not null,

Name\_of\_event text,

Date date,

Location text

);

1. To insert the details of each event, a query is generated.

insert into events VALUES (1,'movie','2022-05-18','PVR Prashant Vihar');

insert into events VALUES (2,'concert','2022-05-19','Apex auditorium');

insert into events VALUES (3,'football match','2022-05-20','JLN stadium');

insert into events VALUES (4,'cricket match','2022-05-21','AJ stadium');

insert into events VALUES (5,'tedx session','2022-05-22','BR auditorium');

insert into events VALUES (6,'Seminar','2022-05-23','BR auditorium');

1. Now we create the table for our first event ‘concert’.

CREATE TABLE `project\_dbms`.`concert`

( `Serial No.` INT(3) NOT NULL AUTO\_INCREMENT ,

`name` TEXT NOT NULL , `age` INT(3) NOT NULL ,

`gender` VARCHAR(8) NOT NULL ,

`email` VARCHAR(22) NOT NULL ,

`phone` VARCHAR(10) NOT NULL ,

`desc` TEXT NOT NULL ,

`date` DATETIME NOT NULL DEFAULT CURRENT\_TIMESTAMP ,

PRIMARY KEY (`Serial No.`)) ENGINE = InnoDB;

Now we will insert two constraints on the table ‘concert’.

* First constraint is added on ‘Serial No.’ which limits the count of ‘Serial No.’ up to 100. This is done because all the events has limited number of seats which in our case is 100 for all the events. Once the count of ‘Serial No.’ reaches 100, no further entries can be made i.e. all the seats have been booked.

alter table concert

add constraint check\_seat check(`Serial No.`<=100);

* Second constraint is added on ‘date’ which doesn’t allow the user to book a ticket after on a date after the event has already occurred.

alter table concert

add constraint check\_date check(date<events.Date where events.Name\_of\_event like 'concert')

1. Similarly, tables for ‘movie’, ‘cricket match’, ‘football match’, ‘seminar’ and ‘tedx session’ has been generated. Their queries can be seen in the base file.
2. Here is the entity – relationship diagram



1. Now the GUI has been implemented with the help of php, html, javaScript and CSS. The code is as follows:

**php and html code:**

<?php

//if(isset($\_POST['name']))

    $con = mysqli\_connect("localhost:3308" , "root" , "", "project\_dbms");

    if(!$con)

    {

        die("conection to this database failed due to".mysqli\_connect\_error());

    }

    echo "Success connecting to the db";

    $name = $\_POST['name'];

    $gender = $\_POST['gender'];

    $age = $\_POST['age'];

    $email = $\_POST['email'];

    $phone = $\_POST['phone'];

    $desc = $\_POST['desc'];

$sql = "INSERT INTO `project\_dbms`.`concert` ( `name`, `age`, `gender`, `email`, `phone`, `desc`, `date`) VALUES ( '$name', '$age', '$gender', '$email', '$phone', '$desc', current\_timestamp());";

    echo $sql;

    if($con->query($sql) == true){

        echo "Successfully inserted";}

    else{

        echo "ERROR : $sql <br> $con->error";

    }

    $con->close();

?>

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Welcome to BookMyShow</title>

    <link rel = "stylesheet" href="style.css">

</head>

<body>

    <div class="container">

        <h3>BookMyShow</h3>

        <p>Enter your details to book your ticket for the concert <br> HURRY, LIMITED TICKETS ARE AVAILABLE</p>

        <form action+"index.php" method = "post">

            <input type="text" name="name" id="name" placeholder="Enter your name">

            <input type="text" name="age" id="age" placeholder="Enter your Age">

            <input type = "text" name ="gender" id="gender" placeholder="Enter your gender">

            <input type="email" name="email" id="email" placeholder="Enter your email">

            <input type="phone" name="phone" id="phone" placeholder="Enter your phone">

            <textarea name ="desc" id=""desc" cols= "30" rows ="10" placeholder="Enter any other inforation here"></textarea>

            <button class = "btn"> Submit</button>

            <button class = "btn"> Reset</button>

        </form>

    </div>

    <script src = "index.js"></script>

</body>

</html>

**CSS code:**

.container

{

*max-width*: 80%;

*background-color*: rgb(254, 6, 2);

*margin*: auto;

*padding*: 4px;

}

.container h3,p{

*text-align*: center;

}

.btn{

*color*: white;

*background*: rgb(253, 0, 0);

*padding*: 8px 12px;

*font-size*: 20px;

*border*: 2px solid white;

*border-radius*: 14px;

*cursor*: pointer;

}

input,textarea{

*font-size*: 25px;

*width*: 80%;

*margin*: 11px;

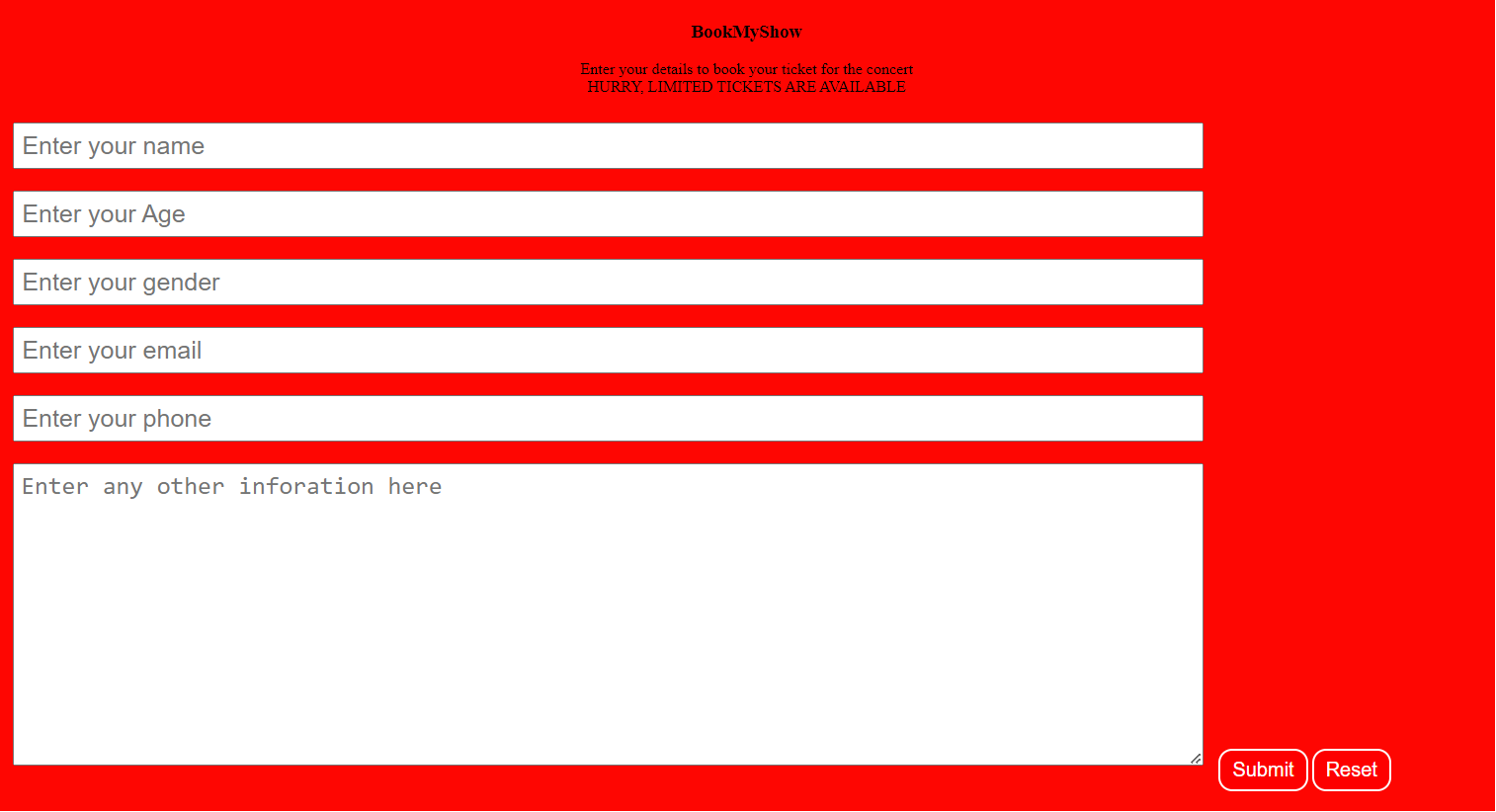
*padding*: 7px;

}

After executing the above code, we have to access

<http://localhost/new_fol/>

The following webpage appears which happens to be the interface where our user can enter his/her details for a particular event.



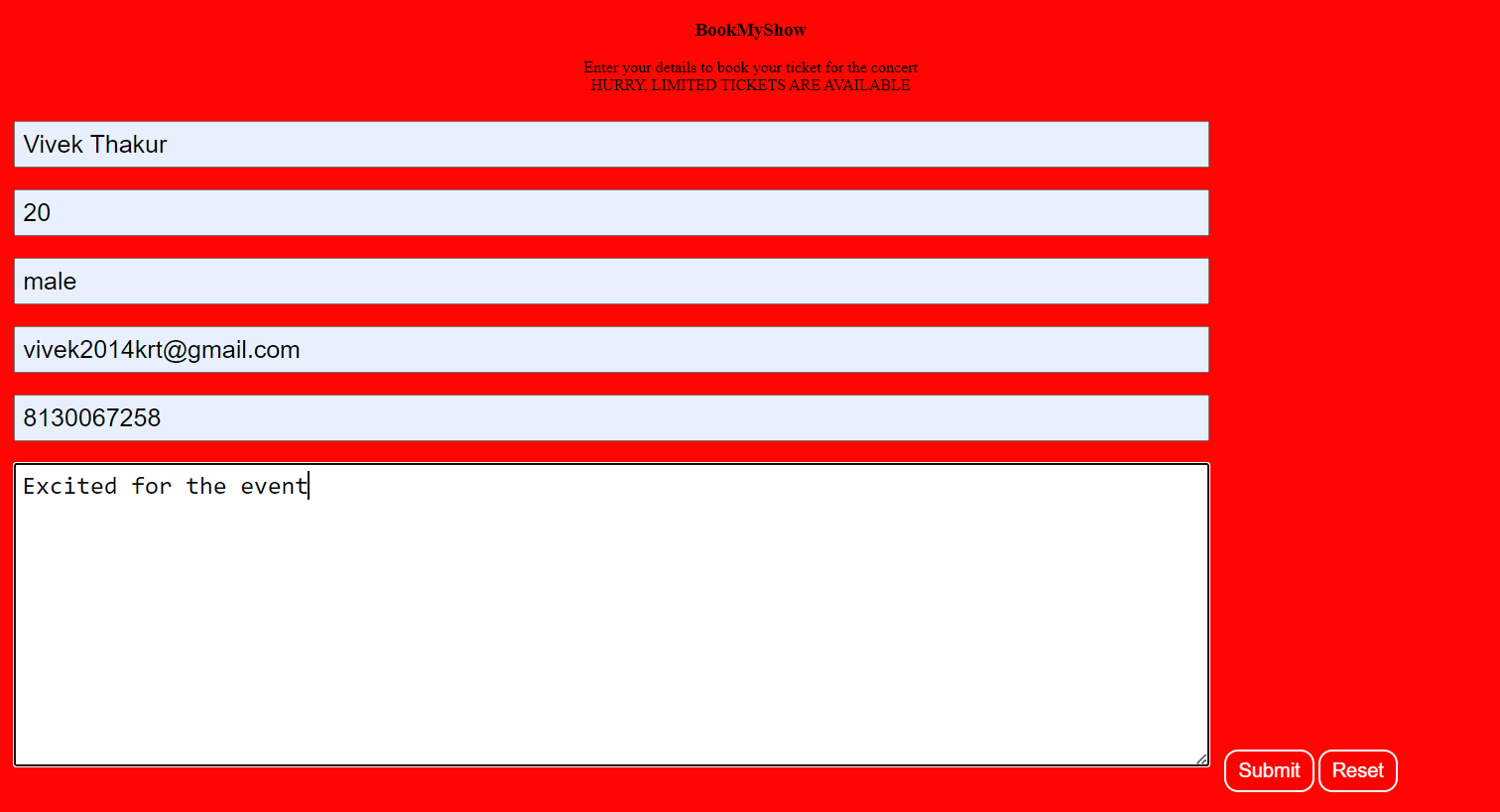
This is the interface where users can enter their details. For simplicity, for the time being the model is made in such a way that at a given time, users can only book tickets for a one event only. To allow the booking of tickets for a different event, appropriate changes need to be made on backend.

$sql = "INSERT INTO `project\_dbms`.`concert` ( `name`, `age`, `gender`, `email`, `phone`, `desc`, `date`) VALUES ( '$name', '$age', '$gender', '$email', '$phone', '$desc', current\_timestamp());";

It can be seen that currently the tickets are being booked for ‘concert’, in order to change the event from ‘concert’ to ‘movie’, the syntax needs to be modified as follows:

$sql = "INSERT INTO `project\_dbms`.`movie` ( `name`, `age`, `gender`, `email`, `phone`, `desc`, `date`) VALUES ( '$name', '$age', '$gender', '$email', '$phone', '$desc', current\_timestamp());";

Once a user enters his details and hit submit, an SQL query is generated and executed which marks his booking.

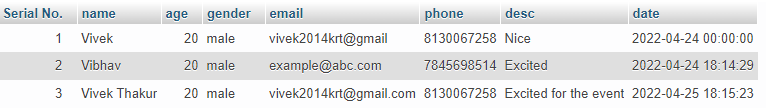


Once the user hits submit, following query is generated.

INSERT INTO `project\_dbms`.`concert` ( `name`, `age`, `gender`, `email`, `phone`, `desc`, `date`) VALUES ( 'Vivek Thakur', '20', 'male', 'vivek2014krt@gmail.com', '8130067258', 'Excited for the event', current\_timestamp());

The same can also be seen on the top of the webpage.

We can see that data has been successfully inserted in our ‘concert’ table



Similarly, entries can be made for different events.

**Limitations**

* There is no billing platform.
* User cannot select an event himself, he can only book a ticket for the event i.e. currently permitted by the admin.
* The GUI doesn’t look that intuitive or innovative.
* There is no way a user can cancel his booking on his, it can only be done by the admin.
* The SQL queries that are being generated sometimes don’t give the desired output.

**References**

<https://www.tutorialspoint.com/sql/sql-rdbms-concepts.htm#:~:text=What%20is%20a%20table%3F,of%20numerous%20columns%20and%20rows>

<https://www.javatpoint.com/xampp>

<https://classroom.google.com/u/2/c/NDUwODI5MjQzNzM2/m/NDUxNTkyMTEzNTky/details>

<https://www.w3schools.com/sql/>