Project Report

Classroom Management System

Naman Manish Trivedi (181CO156)

K V Sumanth Reddy (181CO225)

Contents

Abstract

Database Management System (DBMS) is one of the systems used to store data in a fashion that makes it easier for users to fetch, update and delete data from the database. Traditionally, Hierarchical File Systems were used for the above purpose. This created lots of problems in the management of databases. Some of the problems that are fixed by modern DBMS are:

- Data Redundancy - that is having the same information stored in the database over and over again.
- Data Sharing - sharing of an overtly complex data.
- Data Concurrency that is in the traditional database more than one person cannot fetch, update or delete data at a time.
- Data searching - it takes a huge amount of time to search for files through file systems this has reduced to a great extent using modern DBMS.
- Data security - access to certain parts of the data should not be open to all.
- Data integrity - some constraints are needed for the data to be accurate.
- System crashing was also greatly reduced.

DBMS is used in a variety of applications like Chat Apps, applications like Facebook and Instagram, Amazon, Flipkart, Google, Gmail, Cricbuzz, Yahoo, College Websites, Music apps, Notes app, etc. These websites have a backend system which relies completely on the DBMS to perform different operations on the data in the database. Previously also a few of these were existing but due to the increase in the use of the DBMS in the past decade the efficiency and scale of these has increased to a greater level.

DBMS is one of the modern marvels which allows us to do the above jobs with a high level of accuracy. This was tested out using our project Classroom management system. An institute or a teaching building will have classrooms, and they are an important factor in segregating students, so it is very important that information regarding the classrooms is stored in an orderly fashion. The people who would check the details of a classroom, would like to view that data with ease. So the website regarding it is created using a Django framework and Database SQLite3 is used to store the data.

There are numerous concepts of DBMS which were explored by doing this project they are as follows:

- Primary Key, Foreign Key.
- Query Optimizations.
- Transaction Concurrency Control
- Normalization.

Motivation

This project was done to study the different concepts related to DBMS and develop an understanding of how to design a database with maximum efficiency. There are a lot of concepts that were learned in this project some of which are listed below:

- Design of a database management system.
- Writing queries to fetch data.
- Normalizing the database to increase efficiency.
- Concurrency control using Django.
- Query optimizations.

There are other skills also learned through this project like website making using the Django framework, front end designing using HTML and CSS. The Backend part that was created on a DBMS. Since the Django framework is completely based on python a few concepts of programming in python like object oriented programming , regular expressions, etc. were also acquired through this.

Introduction to our project

Classroom management system is a software which is helpful for faculty and extra teaching staff who want to book a classroom for teaching purposes. In the general systems till date, all the activities are done manually. Our classroom management system deals with the various activities related to the faculty and Manager.

There are two types of user for our website:

1. Teaching staff (Faculty)
2. Manager (Head of the department/ Principal)

On this website we can register a teaching faculty as a teaching staff or a manager. A teacher can register as a user and can add, edit and delete his/her classroom booking status. The Manager can add, edit and delete the classroom available for booking and also decide the time slots which must be available for booking. All the users can see the available rooms and the time slots.

The main aim of the entire project is to automate the process of day to day activities of the classroom, like, room bookings, admission to new customers, etc.

Existing Approach

In most of the classroom management systems, all the work is done manually, which is time consuming and costly. The Manager has to do most of the work here, by collecting user information and writing them down without any conflicts. The manager is the only authority who can add, edit or delete anything present on the system. The user cannot add, edit or delete his booking unless they inform it to the manager beforehand.

A good analysis model should provide not only the mechanisms of problem understanding but also the framework of the solution. Thus, it should be studied thoroughly by collecting data about the system.

Proposed Approach

In our system, we have the provision for the teachers to register their own classrooms, at their own convenience. This reduces the burden on the manager. Also, teachers can edit their booking details when necessary. The following innovations were proposed to get an effective Classroom data management website:-

Query Optimizations to reduce Database Access

As the key to the performance of the database system was limiting access to the database, it was proposed utilise the same. As the scale of the project is much smaller compared to Branded Online Classroom Management Systems, caching was found to be quite unnecessary. It was, therefore, proposed to use query optimizations to reduce the access to the database to restrict costly processes of reading from and writing to the database.

Concurrency Control

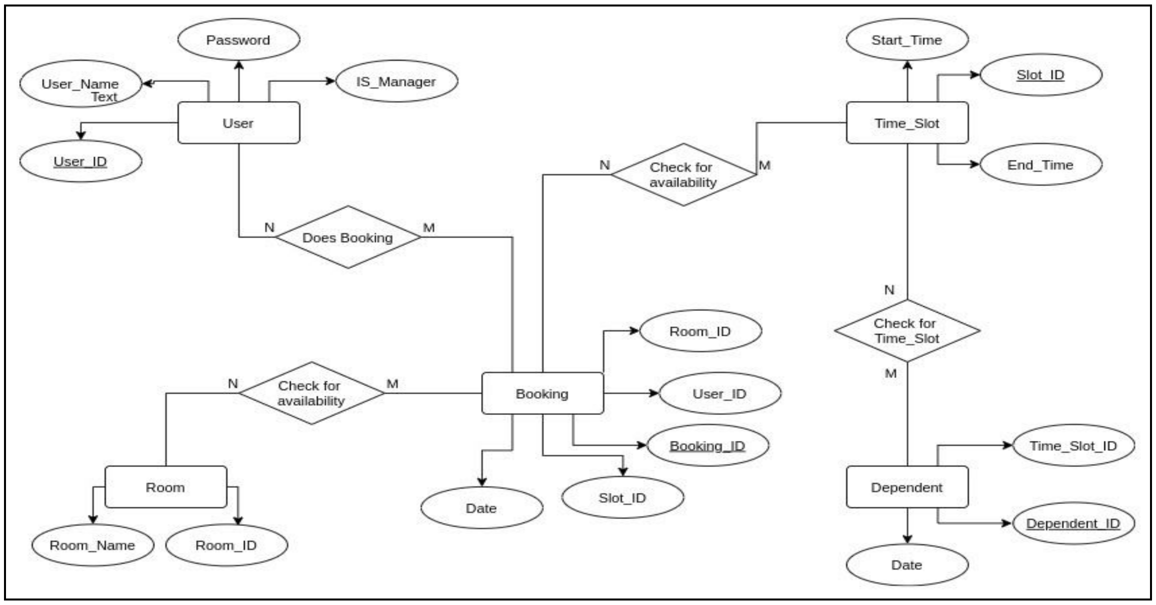
As the system will be handling multiple requests to access and modify data, the fear of concurrent transactions leading the database into inconsistent states is always present. But keeping in mind the nature of this data, it is understood that concurrent modifications are not so common, as compared to a database for bank transactions for example. Therefore it was proposed to use Optimistic Concurrency Control to handle the transactions. The basic idea will be to introduce a variable that changes its value upon every commit to the database. While modifying data, the value of this variable shall be checked, if it is the same as the one initially read, then it is understood that no other transaction has committed a modification to the particular data in question, and the data is committed, if the value is different, then it is understood that a different transaction has updated the value of the data in question, and the transaction is aborted.

Database Design

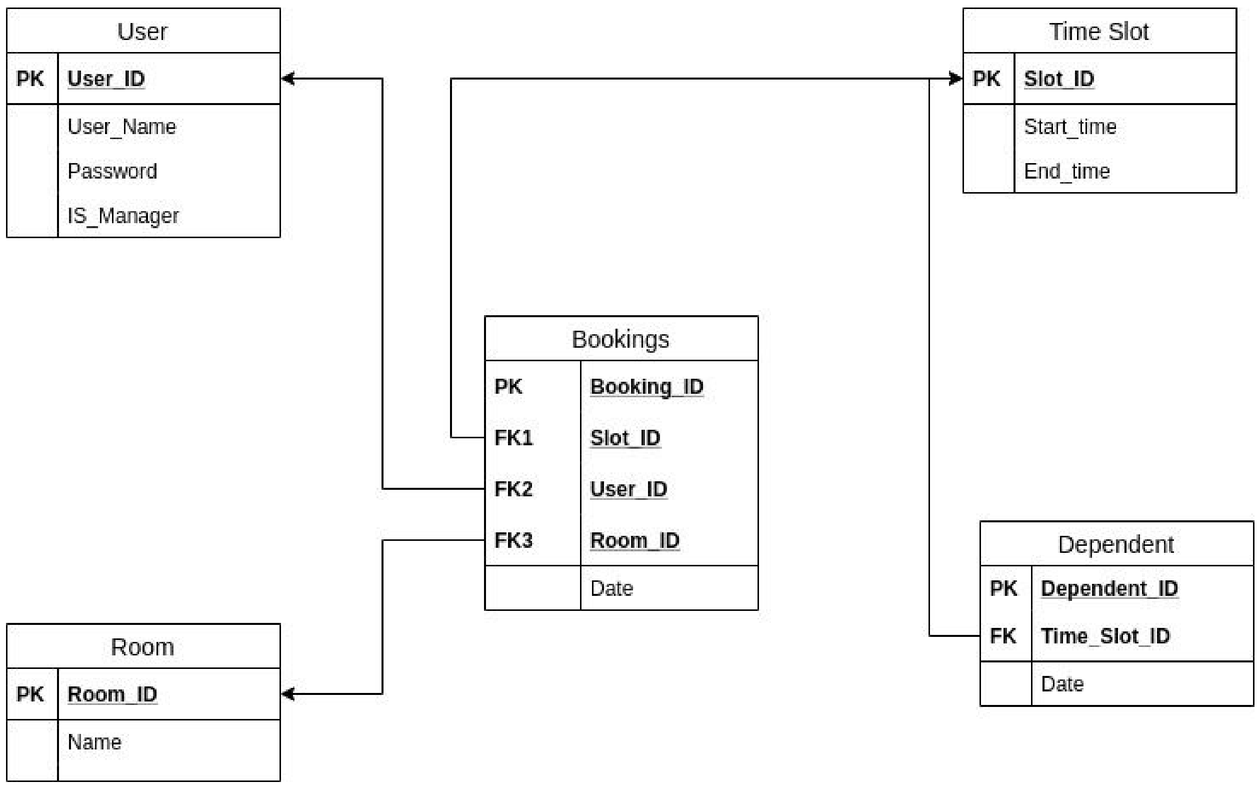
The database system was made as a website using the Django framework which uses python as the server side language and the database to which it is connected is Sqlite3.

The Database Schema and the Relational Diagram :-

ER Diagram



Relational Diagram



Normalization

- 1NF: as all values in each attribute will be atomic
- 2NF: id is the only prime key and all attributes functionally depend on this attribute, therefore there is no Partial Dependency
- 3NF: id is the only prime key and all attributes functionally depend on this attribute, therefore there is no Transitive Dependency
- BCNF: id is the only prime key and all attributes functionally depend on this attribute, therefore there is no prime key that is functionally dependent on the non-prime key.

  In this schema as everything is determined by pk hence non-prime attribute isn't determining anything thus we can say that this schema is in BCNF.

Query Optimization

As we are entering the world of big data the amount of data to be searched for processing the query is always going to increase and thus developing efficient methods for query execution is very important

<u>Basic rules</u> :-

- Only fetch the necessary attributes of the needed relation

  ORM:

  ```
  Table.objects.only('attr1','attr2')
  ```

- Do not use for loops in server side language to establish connection to the db every time

  ORM:

  Instead of looping through the list use :

  ```
  Table.objects.filter(attr__in = [list])
  ```

  Instead of looping and creating join query every time while searching through Foreign key use prefetch_related or select_related

  ```
  Tabler.objects.select_related('OneToOne').prefetch_related('ManyToMany').filter()
  ```

  This essentially creates a temporary table which is formed by joining the tables needed for the query and than searching on it instead of joining table every time we loop through it

  In general we shouldn't use loops and rather use such queries as they also create separate threads and run parallely and with loops this isn't possible they have to be executed serialy.

- If you just want to update and not read anything do not bring the contains of db in execution than update and store back to db rather just send query to update the db

  ```
  from django.db.models import F
  
  obj.attr = F('attr')  + update
  ```

- If you have to change multiple values change all and commit all of them together

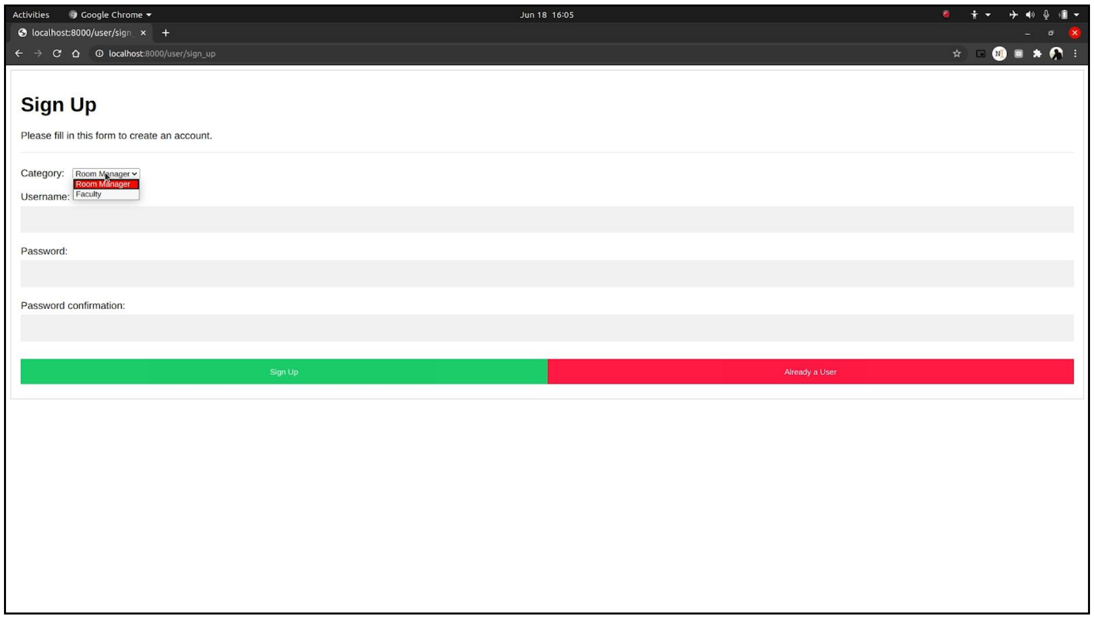
  ```
  with transaction.atomic():
  
          //executable code
  ```

  By this it will hold all commits till the loop has completed execution

Working Screenshots

(Screenshots captured from demo-video)

Sign-Up Page



Sign Up

Please fill in this form to create an account.

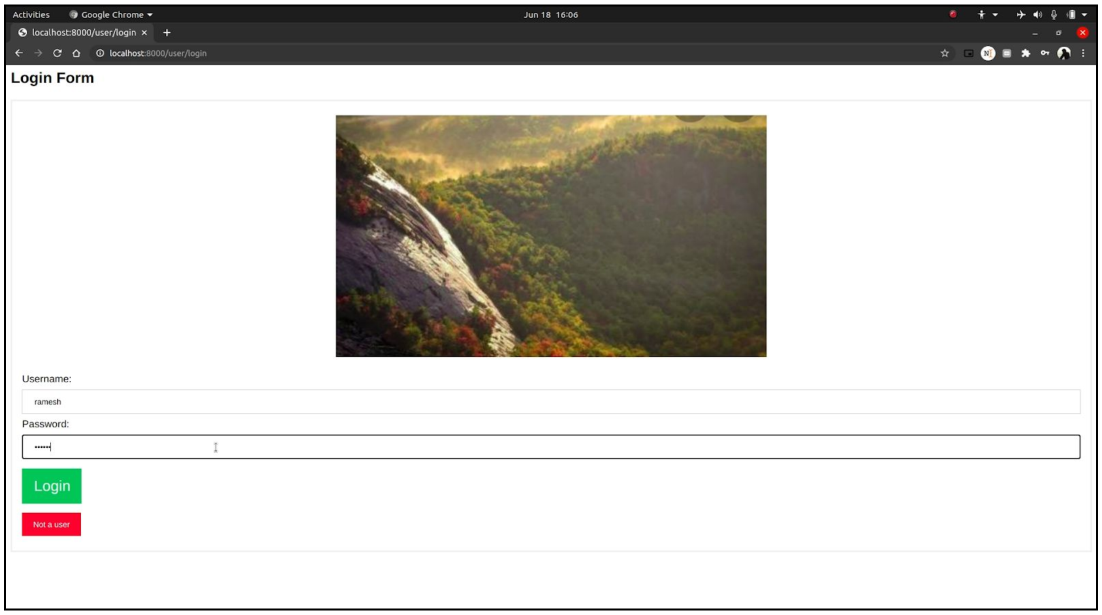
Category: Room Manager

Username:

Password:

Password confirmation:

Login Form

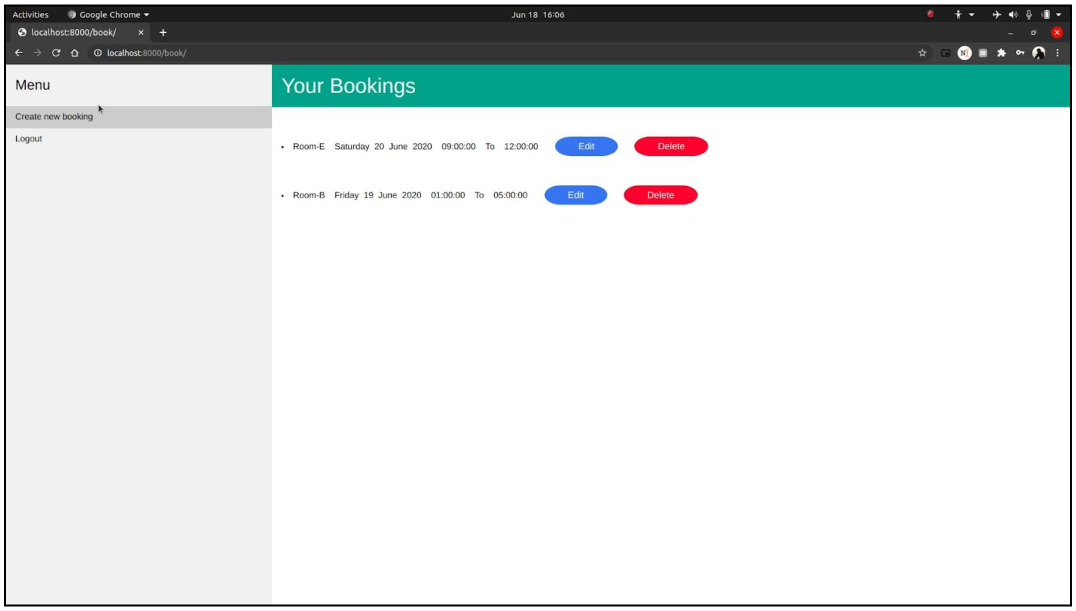


Login Form

Username: ramesh

Password: ••••••

Faculty Interface



Menu

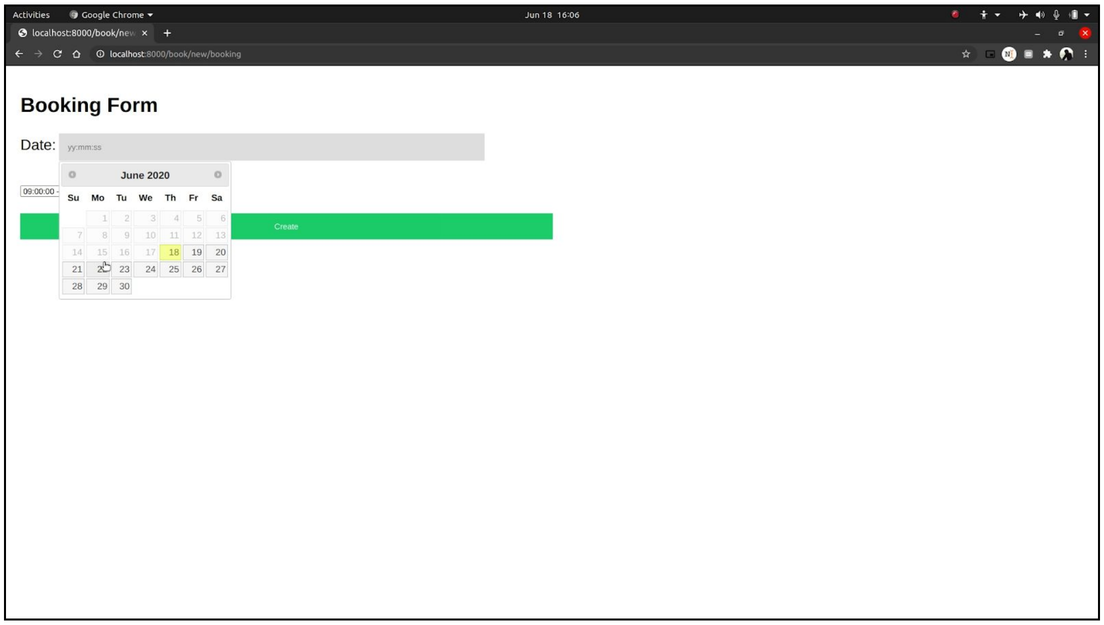
Create new booking

Logout

Your Bookings

- Room-E Saturday 20 June 2020 09:00:00 To 12:00:00
- Room-B Friday 19 June 2020 01:00:00 To 05:00:00

Booking Interface



Booking Form

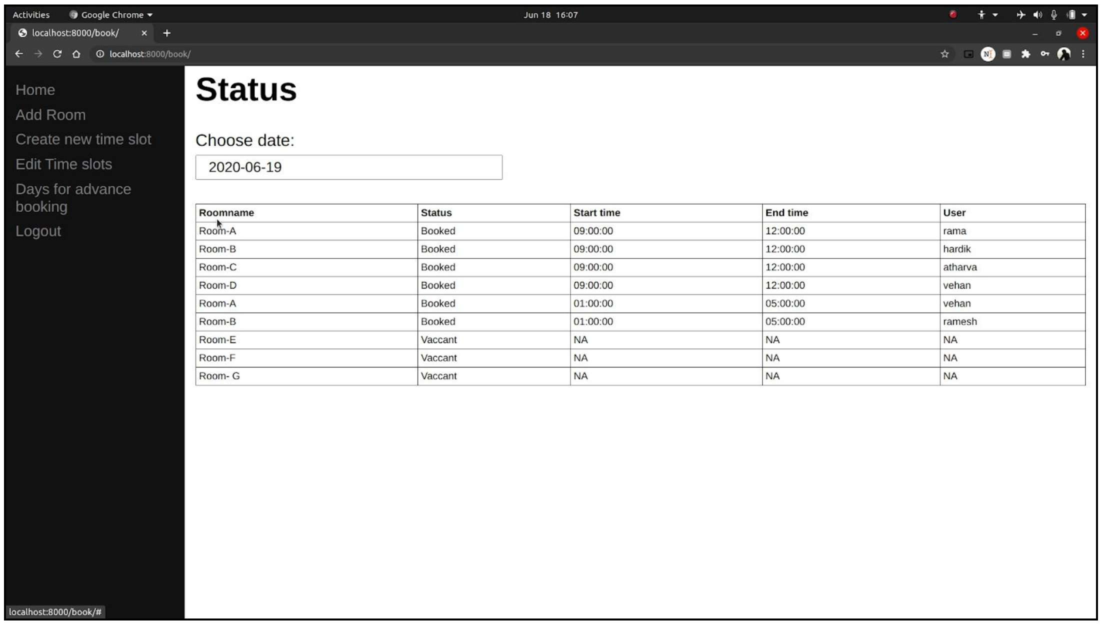
Date: yy:mm:ss

09:00:00

June 2020

Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Manager Interface

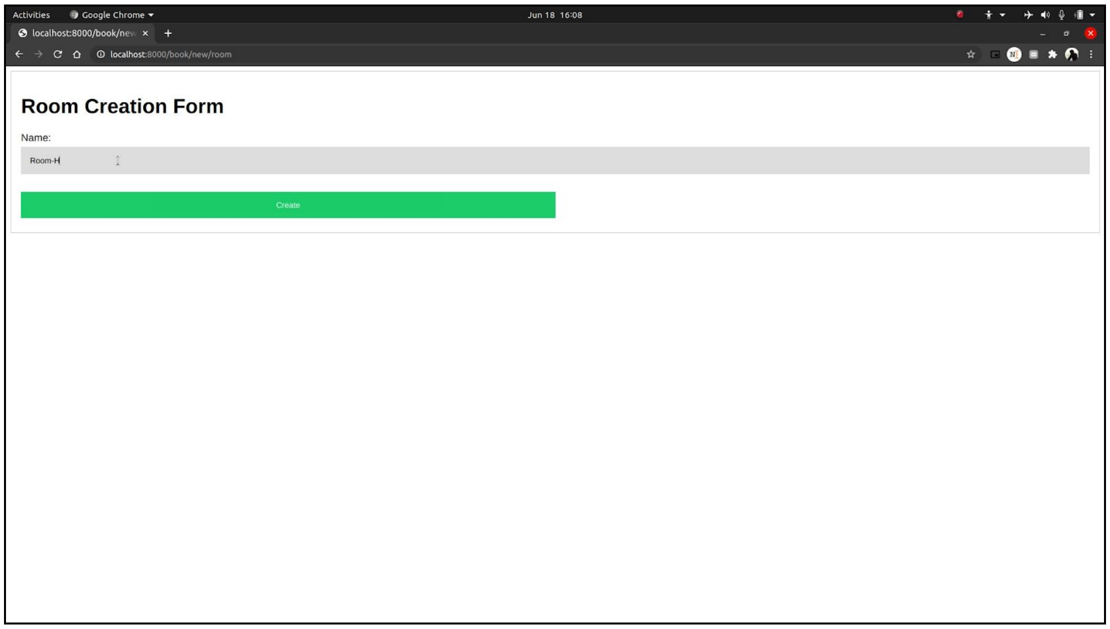


Status

Choose date:

2020-06-19

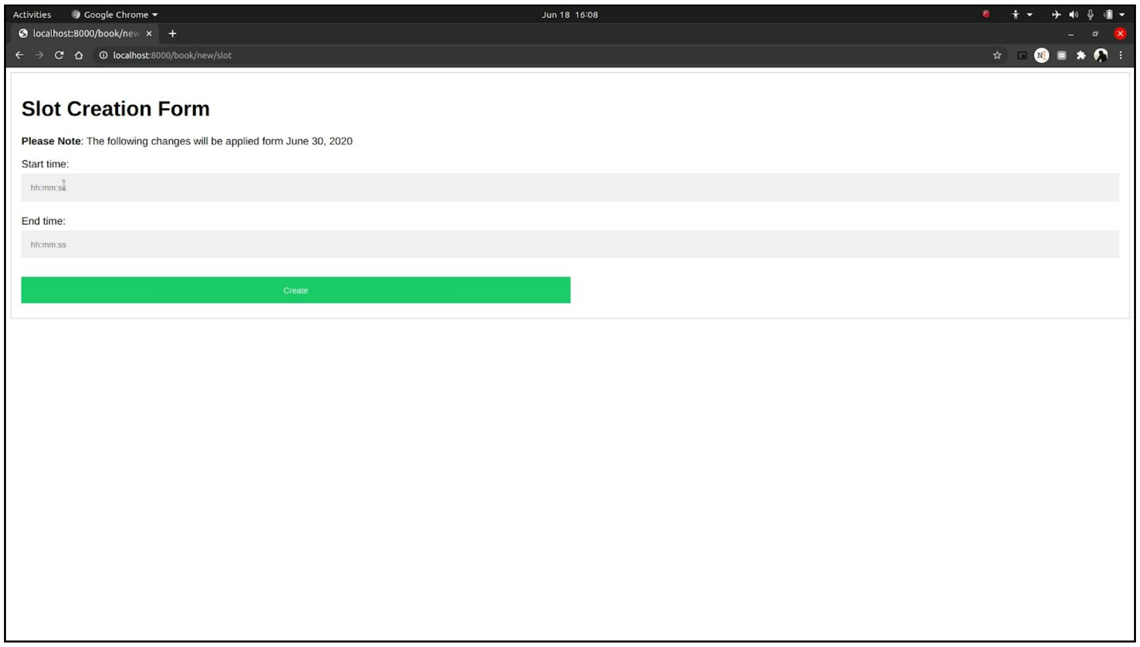
Roomname	Status	Start time	End time	User
Room-A	Booked	09:00:00	12:00:00	rama
Room-B	Booked	09:00:00	12:00:00	hardik
Room-C	Booked	09:00:00	12:00:00	atharva
Room-D	Booked	09:00:00	12:00:00	vehan
Room-A	Booked	01:00:00	05:00:00	vehan
Room-B	Booked	01:00:00	05:00:00	ramesh
Room-E	Vaccant	NA	NA	NA
Room-F	Vaccant	NA	NA	NA
Room- G	Vaccant	NA	NA	NA



Room Creation Form

Name:

Room-H



Slot Creation Form

Please Note: The following changes will be applied form June 30, 2020

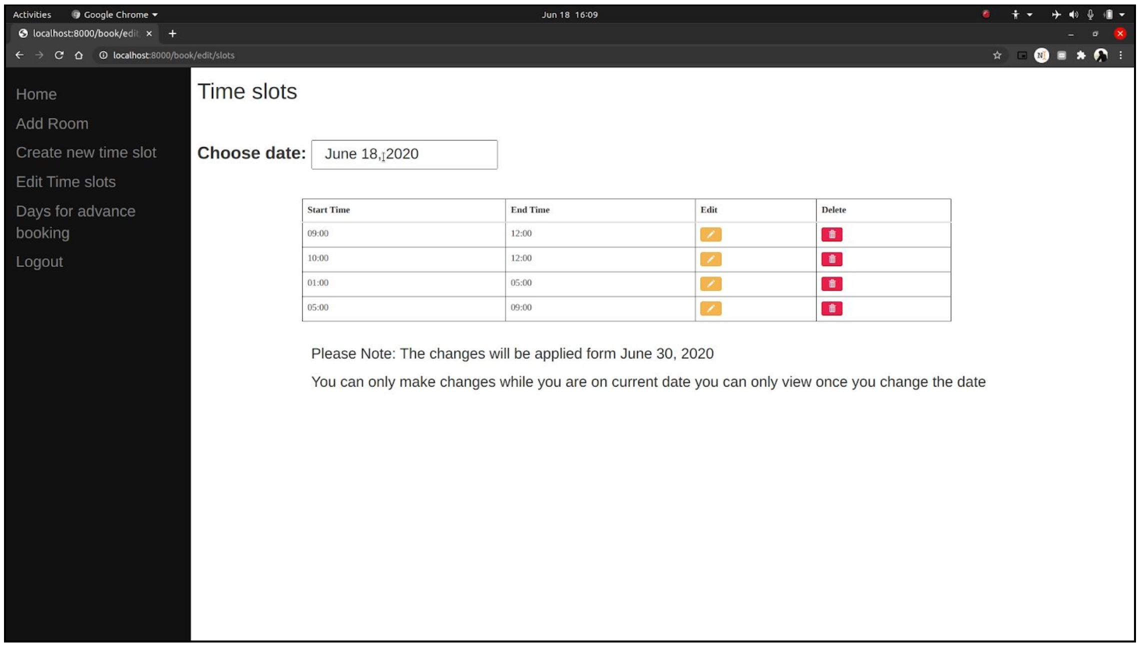
Start time:

hh:mm:ss

End time:

hh:mm:ss

Create



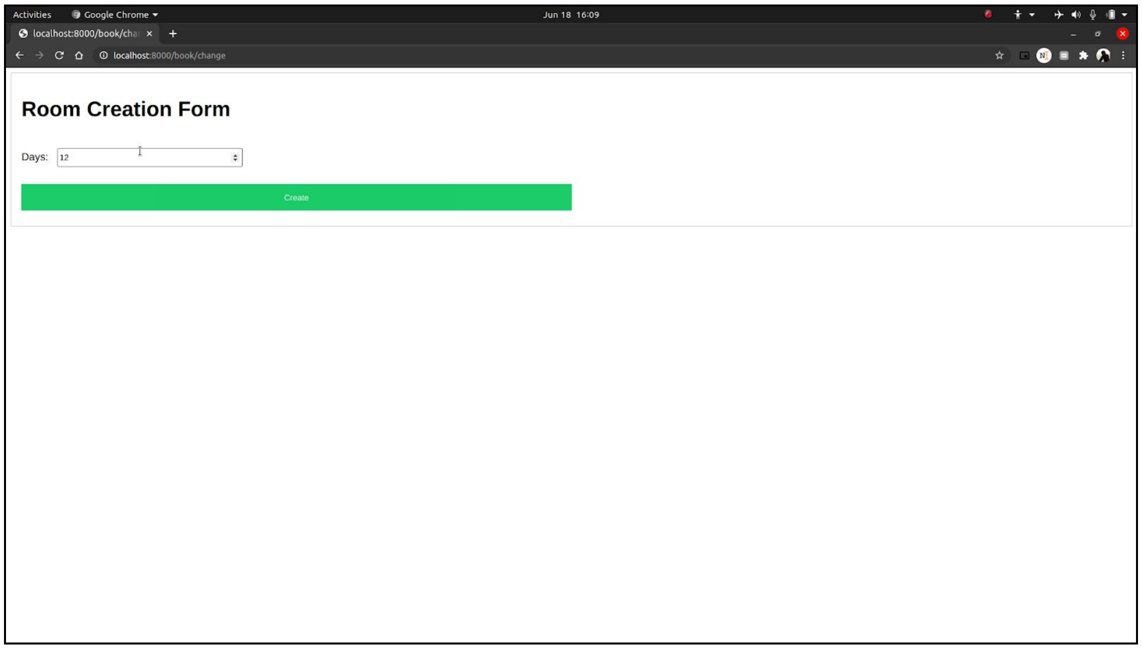
Time slots

Choose date: June 18, 2020

Start Time	End Time	Edit	Delete
09:00	12:00	✎	🗑
10:00	12:00	✎	🗑
01:00	05:00	✎	🗑
05:00	09:00	✎	🗑

Please Note: The changes will be applied form June 30, 2020

You can only make changes while you are on current date you can only view once you change the date



Room Creation Form

Days: 12

REST API

Nowadays, all the applications are using front end frameworks, so we essentially just need to build endpoints where JS can query and thus REST APIs are really helpful.
More so, if any third party application wants to integrate to our application, it will also require endpoints.
It is always easy to change the course of execution in an API governed organization, because changing one doesn't affect the other.

Scope of future Applications

This project can be used in colleges and schools after adding some more useful modules in the project for which classrooms are providing services.

Utmost care and back-up procedures must be established to ensure 100% successful implementation of the computerized Classroom system. In case of system failure, the organization should be in a position to process the transaction with another organization or if the worst comes to the worst, it should be in a position to complete it manually.

Conclusion

The website was successfully created and all the errors that came while making the website were resolved. Different concepts of DBMS were successfully tested after making the website. The database was optimized to a very high level using the concepts and efficiency was increased. There are a few more approaches which were looked upon by us but implemented these can be further integrated into the websites so that it can handle larger traffic.