Room-Booking-Portal

Names	Roll No.	CGPA so far
1. Suryavanshi Virendrasingh	B16037	8.1
2. Lakshay Arora	B16060	8.3
3. Aman Khandelwal	B16007	8.5

A. Project Statement

Without an online booking system, booking rooms for any event or task is tedious. So this project aims to make a Hostel and Room booking Portal to book the hostel rooms and academic blocks rooms through the online portal in as easy way as possible.

This online system would require managing following data:

Room: Manage information of every room in academic blocks and other buildings. Information required to identify rooms would be RoomID, RoomName and on which day of week there are available and on which days they are not.

Hostel: Manages Information of every hostel room. Information required to identify hostel rooms would be HostelID, HostelName, NumFloor and NumRoom.

Facilities to be provided to the end user:

User will have the option to search for room information (mentioned in the previous heading) based on the RoomID or HostelID.

Room Availability: Whether room is available for given date D. If available display the details of each room available. You can cancel booking of rooms and hostel rooms anytime.

Functionality will be provided to **book tickets** too. For that user must have an account. Sign up option will be provided to create a new account.

Sign Up: In order to book room user must have an account. To register for a new account user will have to fill a form. Fields consists of E-mail, user name and password.Once the user has successfully setup his/her account then he/she can book room, can see the past bookings, cancel the booking.

Room Booking: User can book the room (if available) by simply selecting it from the availabilities shown in portal.

Booking Cancellation: Sometime it is required to cancel the booking booked in the past. So it is obvious to have a functionality to cancel the booking booked in the past.

Summary: First step for the user is to create an account. Once the account will be created then they can access any room information, can check whether room is available or not for given date, if available they can book room and can see all the past booking information, and can even cancel the booking.

B. Requirement Analysis

(i) Features and Functions:

In order to do something user must have an account. So he/she can **signup** if he/she don't have one and he/she is good to go. Now he/she have an account so after successful **login** he/she can

1. *know about available rooms:* **User** need to select **room number** as **input**. Room information will be displayed to the user such as **room availability**.
2. *book a room:* Booking a room is very simple. User just need to select the room correctly.

(ii) Operating System:

All operating systems with web browsers are supported.

(iii) Hardware:

A PC or Laptop and good internet connection to access the platform to practice on it.

(iv) Software:

The Room_Booking_Portal can run on any recent version of Windows 7 or later, also on Linux, such as Ubuntu, Debian, Fedora Core, Redhat Enterprise, etc.

It uses:

1. PHP 7.1
2. MySQL 5.6

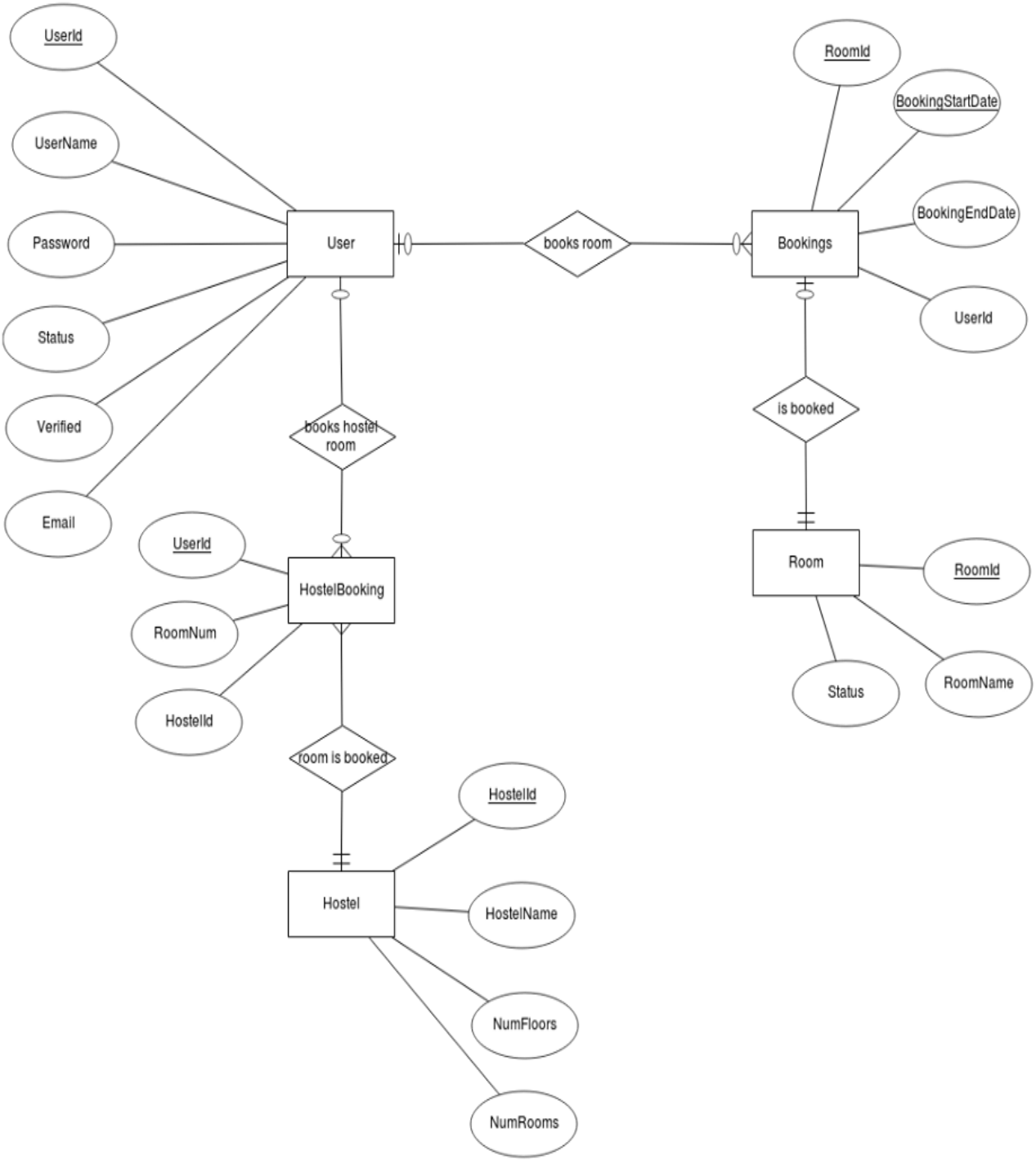
It requires:

1. Apache2 2.4 or Later
2. MySQL 5.6 or Later
3. PHP 5.6 or Later

The Room_Booking_Portal user-interface works with any of the following graphical browsers on any hardware and OS:

1. Firefox 5.0
2. Internet Explorer 7.2
3. Chrome 2.0
4. Safari 3.6
5. Opera 2.3
6. Microsoft Edge and Higher versions of these browsers are likely to work but cannot be guaranteed.

C. Conceptual design: ER Model



D. Special Features

Triggers:

We have created one trigger that will automatically send E-mail to the user who books the room.

Indexing:

We have here used B-TREE indexing.

Benefits:

- So our all nodes are shorted and tree is balanced.
- Fast Traversal and Quick Search
- No Overflow pages

Disadvantage over B+ TREE indexing:

The drawback of B-tree used for indexing over B+ tree indexing is that it stores the data pointer (a pointer to the disk file block containing the key value), corresponding to a particular key value, along with that key value in the node of a B-tree. This technique, greatly reduces the number of entries that can be packed into a node of a B-tree, thereby contributing to the increase in the number of levels in the B-tree, hence increasing the search time of a record.

Query Optimization:

We used inbuilt query optimizer of phpMyAdmin to optimize our database.

Your SQL query has been executed successfully.

```
OPTIMIZE TABLE `wp_commentmeta`, `wp_comments`, `wp_links`, `wp_options`, `wp_postmeta`, `wp_posts`, `
```

+ Options

Table	Op	Msg_type	Msg_text
sgtutori_db1.wp_commentmeta	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_commentmeta	optimize	status	OK
sgtutori_db1.wp_comments	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_comments	optimize	status	OK
sgtutori_db1.wp_links	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_links	optimize	status	OK
sgtutori_db1.wp_options	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_options	optimize	status	OK
sgtutori_db1.wp_postmeta	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_postmeta	optimize	status	OK
sgtutori_db1.wp_posts	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_posts	optimize	status	OK
sgtutori_db1.wp_termmeta	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_termmeta	optimize	status	OK
sgtutori_db1.wp_terms	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_terms	optimize	status	OK
sgtutori_db1.wp_term_relationships	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_term_relationships	optimize	status	OK
sgtutori_db1.wp_term_taxonomy	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_term_taxonomy	optimize	status	OK
sgtutori_db1.wp_usermeta	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_usermeta	optimize	status	OK
sgtutori_db1.wp_users	optimize	note	Table does not support optimize, doing recreate + ...
sgtutori_db1.wp_users	optimize	status	OK

Normalization:

The Normalization form we have in our relation is BCNF because all of the functional dependencies X -> Y involved are such that X is a superkey.

Security:

To secure our application from several attacks we have taken following security measures:

1. To prevent CSRF (Cross site request forgery) attacks, we are using SESSION TOKENS. The idea is to assign a token to a session which only our app knows and every request that the application makes to its pages, it sends the token along with it. Any request without a token will be rejected. So the app can always distinguish itself from cross site requests.

2. To prevent SQL injection attacks, we are using SQL prepare statements where we first send a template statement to mysql and then send the data, so it will never misinterpret incoming data as a query.

3. We are encrypting user password as a one way hashed string. If anyone gets access to the data, they will never know the password.

4. All the inputs from the user are first sanitized, to prevent script injections, and ambiguous data.