CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 GENREAL OVERVIEW:

A Database Management System (DBMS) is system software for creating and managing databases. A database is a collection of information that is organized so that it can be easily accessed managed and updated. The DMBS provides users and programmers with a systematic way to create, retrieve, update and manage data. Most of giant IT farms are investing billions on business information from a raw data manage them in files and document which may sometimes even suffer loss hence no security or backup. This project makes a decent effort overcome this problem.

1.2 PURPOSE:

Hostel Room Allocation System is a web application which aims at computerization of the current procedure of allocating hostel rooms.
Currently, the process involves students filling up the forms and submitting them in respective hostel offices which involves a lot of paperwork, hence less efficient.

1.3 OBJECTIVES:

- To deal with Hostel Management System in an easy and anefficient manner.
- Create strong and secrete database that allows for any connection in a secret way, to prevent any outside or inside attacks.

1.4 SCOPE

- Hostel Management System is designed for Hostel (like schools, Universities).
- There will be predefined criteria for the Reservation to the hostels.
- He/She checks the attested application forms of the students obtained.
- If the students are found eligible then they are allotted to the hostel Room.

The Web Application has two main parts:

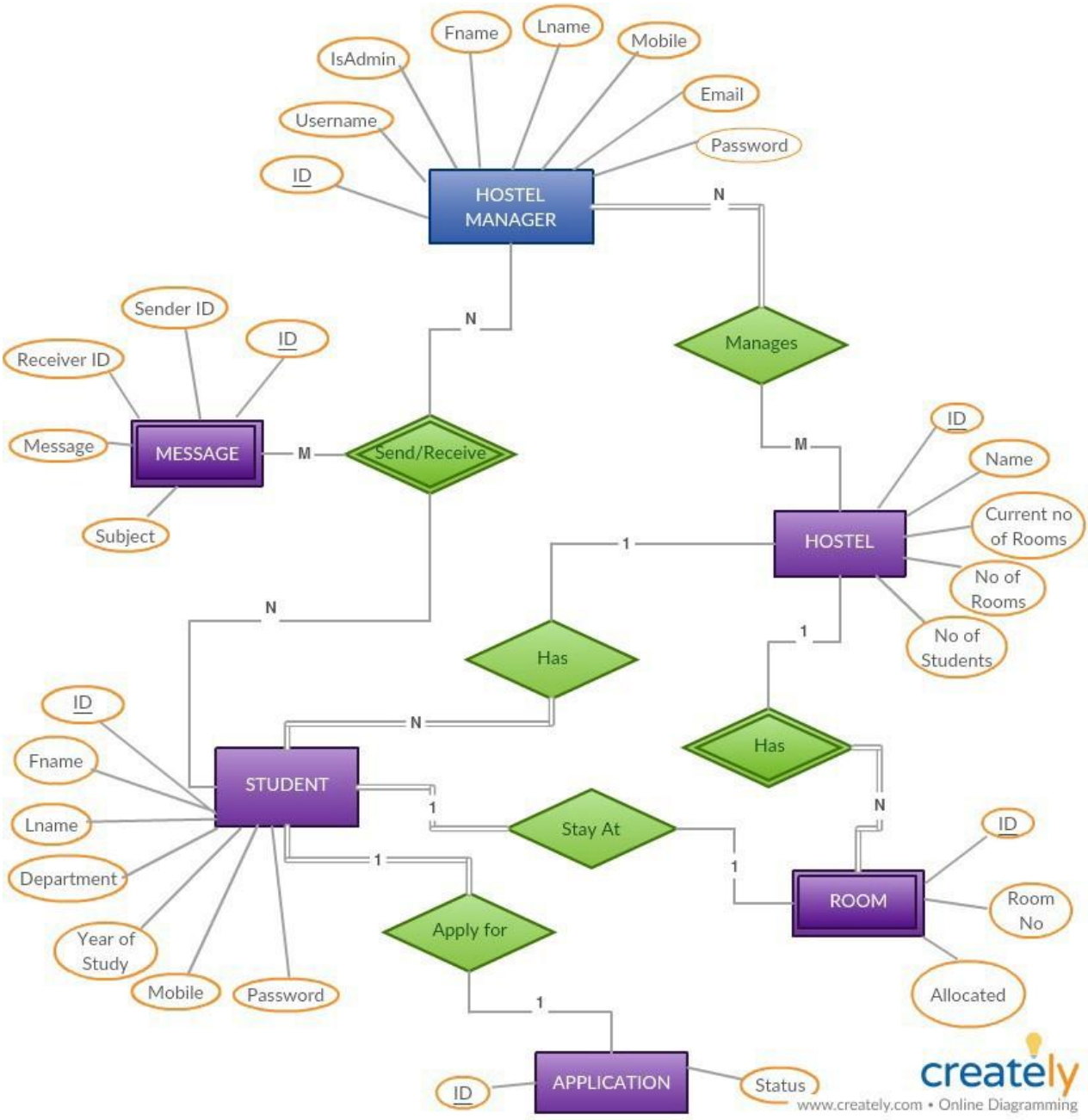
- Hostel Administrators
- Students

The student can select among the allocated hostel to a specified batch and the Hostel Manager can assign the room number in the specific hostel that the student has selected upon the availability.

CHAPTER 2

Data Design

2.1 Entity Relationship Diagram

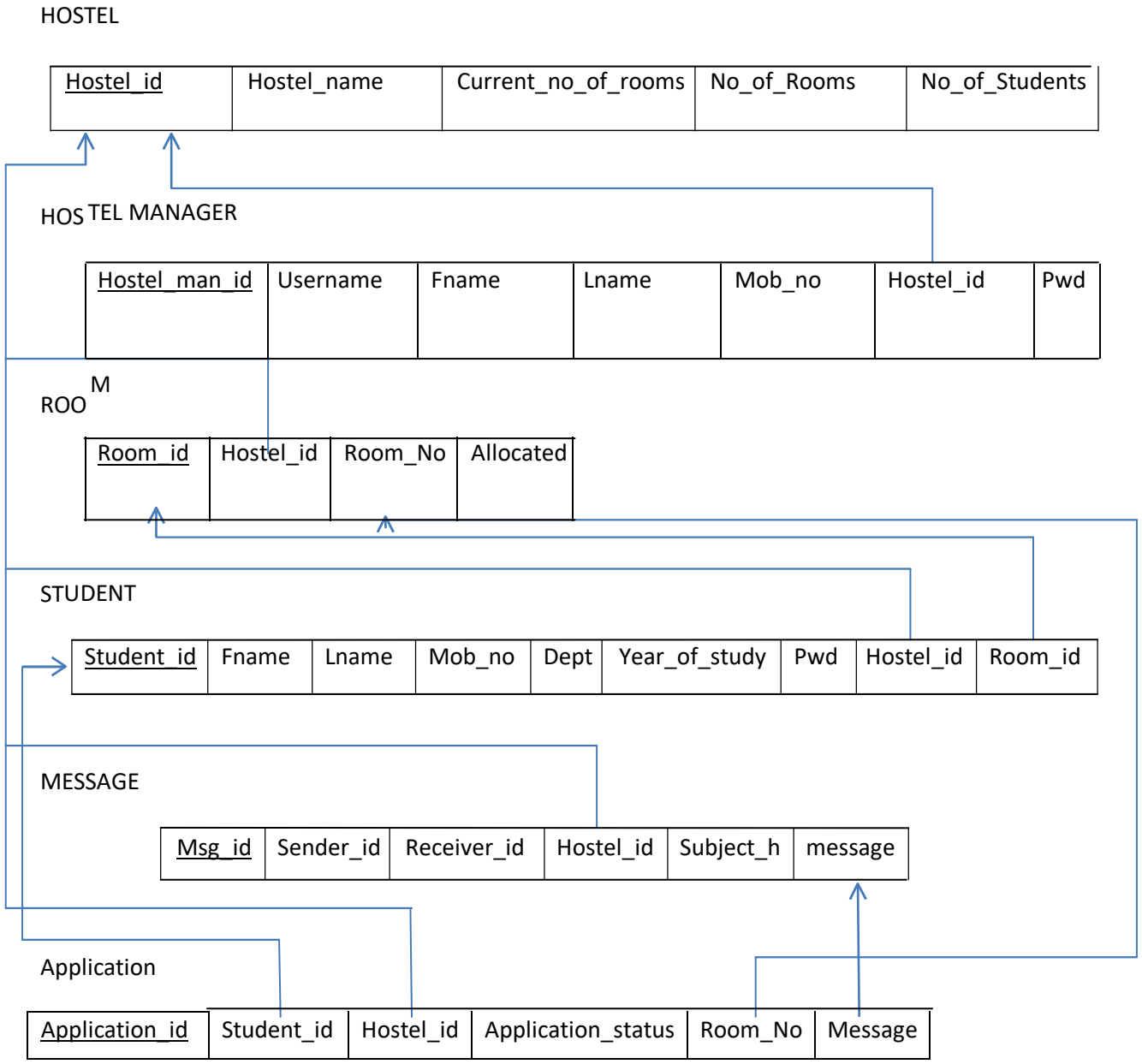


The schemas for the database application can be displayed by means of graphical notation known as Entity Relationship diagram.

The ER model describes data as entities, relationships and attributes.

- The system has 6 regular entities.
- The attributes are shown above.
- All attributes are atomic attributes. With those underlined are primary key of the entities.
- The relationship “Manages” has a cardinality ratio of N:M.
  Any number of Hostel Manager manages any number of hostel.
- The relationship “Has” has a cardinality ratio of 1:N . Each hostel has n number of rooms and students.
- The relationship “Stay At” has a cardinality ratio of1:1. Each student stays at 1 room .
- The relationship “Apply for” cardinality ratio of 1:1. 1 student can apply for only 1 hostel.
- The relationship “Send/Receive” has a cardinality ratio of N:M. Any number of students and managers can send or receive any number of messages to each other.

2.2 CONCEPTUAL SCHEMA



2.3 ENTITIES AND ATTRIBUTES:

An entity may be an object with a physical existence (for e.g. A particular person, car or employee) or it may be an object with a conceptual existence (for e.g. a company, a job, or a university course)
Each entity has attributes i.e. the particular properties that describe it. The attribute values that describe each entity become a major part of the data store in the database. When an attribute of one entity type refers to another entity type, a relationship exists.

This section of the document explains the entities used in the project, their attributes and how they will work together. Basically, this is intended to make the design more easy and understandable for everyone.

Entities

1. Hostel
2. Hostel Manager
3. Room
4. Student

5 Message

6. Application

2.3.1 Hostel

An Institution has many hostels and each hostel is represented using this 'Hostel' entity.

Hostel model takes part in the following relationships.

1. Manager manages Hostel.
2. Hostel has Students.
3. Hostel has Rooms.

Attributes

Name	Data Type	Type
Hostel ID	int	Primary Key attribute
Name	varchar	Non_key attribute
Current no of Rooms	int	Non_key attribute
No of Rooms	int	Non_key attribute
No of Students	int	Non_key attribute

2.3.2 Hostel Manager

Every hostel has an administrator and n manager which can be added or removed by administrator. It is represented using the 'hostel manager' entity.

Manager entity takes part in following relationships.

1.Manager manages Hostel.

2.Manager sends/receives Messages

Attributes

Name	Data Type	Type
Hostel manager ID	int	Primary Key attribute
Username	varchar	Non_key attribute
IsAdmin	int	Non_key attribute
Fname	varchar	Non_key attribute
Lname	varchar	Non_key attribute
Email	varchar	Non_key attribute
Password	varchar	Non_key attribute
Hostel_ID	int	Foreign_key attribute

2.3.3 Room

Every Hostel has rooms and they are represented using 'room' entity & participates in the following relationships.

1.Hostel has Rooms.
2.Student stays at room.

Attributes

Name	Data Type	Type
Room ID	int	Primary Key attribute
Hostel ID	int	Foreign key attribute
Room no	int	Non_key attribute
Allocated	tinyint	Non_key attribute

2.3.4 Student

Every hostel has students and they are represented by the 'student' entity.

Student entity participates in the following relationships.

1. Hostel has Students.
2. Students apply for Application.
3. Students stay at room

Attributes

Name	Data Type	Type
Student ID	varchar	Primary Key attribute
Fname	varchar	Non key attribute
Lname	varchar	Non key attribute
Mobile no	varchar	Non key attribute
Department	varchar	Non key attribute
Year of Study	varchar	Non key attribute
Password	varchar	Non key attribute
Hostel ID	int	Foreign key attribute
Room ID	int	Foreign key attribute

2.3.5 Message

Message is represented using this 'Message' entity takes part in the following relationships.

1. Hostel Manager send/receive Messages
2. Student send/receive Message.

Attributes

Name	Data Type	Type
Message ID	int	Primary Key attribute
Sender ID	varchar	Non key attribute
Receiver ID	varchar	Non key attribute
Hostel ID	int	Foreign key attribute
Subject	varchar	Non key attribute
Message	varchar	Non key attribute
Msg Date	varchar	Non key attribute
Msg Time	varchar	Non key attribute

2.3.6 Application

Student has to apply for an application to request occupancy in a particular hostel.

1. Students apply for Application.

Attributes

Name	Data Type	Type
Student ID	varchar	Primary Key attribute
Student ID	int	Foreign key attribute
Hostel ID	int	Foreign key attribute
Mobile no	varchar	Non key attribute
Application Status	Tiny int	Non key attribute
Room no	int	Non key attribute
Message	varchar	Non key attribute

2.4 NORMALIZATION:

- NORMAL FORM (1NF)
  - There are no repeated or duplicate fields.
  - Each cell contains only one single value and each record is unique.
  - Hence, the database is in the first normal form.

- SECOND NORMAL FORM (2NF)
  - All non-key fields depend on all components of the primary key which is guaranteed when the primary key is a single field.
  - Hence, the database is in 2NF as it satisfies all the required conditions. complaint

- THIRD NORMAL FORM (3NF)
  - All non-key fields depend only on the primary key
  - All entities are under 3NF, hence we can conclude that the entities satisfy all the normal forms.

CHAPTER 3

IMPLEMENTATION

SOFTWARE AND HARDWARE REQUIREMENTS

The hideous methods to search for the location and details of a suitable accommodation which is practically tedious in the current system is overcome by this project which will save the valuable time.

3.1 SOFTWARE REQUIREMENTS:

Table 3.1 : Software Requirements

NAME OF THE COMPONENT	SPECIFICATION
Operating System	Windows 98, Windows XP, Windows 7, Windows 10
Language	HTML, CSS, PHP,Javascipt
Database	MYSQL DATABASE
Browser	Any of Mozilla, Google Chrome, Opera etc.
Web Server	Apache
Software Development kit	XAMPP 3.2.2
Scripting Language enable	HTML, PHP, CSS
Database Connection	PHP Connection

HARDWARE REQUIREMENTS:

Table 3.2: Hardware Requirements

NAME OF THE COMPONENT	SPECIFICATION
Processor	Core i7 processor
RAM	8GB
Hard Disk	512GB
Monitor	14” color monitor
Keyboard	122 keys

3.2 FRONT END

3.2.1 HTML & CSS:

- Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications.
- Cascading Style Sheets (CSS)is a style sheet language used for describing the presentation of a document written in a markup language like HTML
- **HTML** is used to create the actual content of the page, such as written text, and **CSS** is responsible for the design or style of the website, including the layout, visual effects and background color.

3.2.2 **HTML AND CSS CODE FOR STUDENT LOGIN** (index.php)

```
<!DOCTYPE html>

<html lang="en">

<head>

  <title>HMS</title>
```

```
    <meta charset="UTF-8" />

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

    <link href="web/css/style.css" rel="stylesheet" type="text/css" />

    <link href="web/css/fontawesome-all.css" rel="stylesheet" />

    <link
href="//fonts.googleapis.com/css?family=Raleway:100,100i,200,200i,300,300i,400,400i,500,500i,600,60
0i,700,700i,800,800i,900,900i"

        rel="stylesheet">

</head>

<body>

    <h1>Hostel Room Allocation System</h1>

    <div class=" w3l-login-form">

        <h2>Student Login</h2>

        <form action="includes/login.inc.php" method="POST">
        <div class=" w3l-form-group">

            <label>Student Roll No:</label>

            <div class="group">

                <i class="fas fa-user"></i>

                <input type="text" class="form-control" name="student_roll_no" placeholder="Roll No"
required="required" />
```

```
            </div>
          </div>
          <div class=" w3l-form-group">
            <label>Password:</label>
            <div class="group">
              <i class="fas fa-unlock"></i>
              <input type="password" class="form-control" name="pwd" placeholder="Password"
required="required" />
            </div>
          </div>
          <!--<div class="forgot">
            <a href="#">Forgot Password?</a>
            <p><input type="checkbox">Remember Me</p>
          </div>-->
          <button type="submit" name="login-submit">Login</button>
        </form>
         <p class=" w3l-register-p">Login as<a href="login-hostel_manager.php" class="register">
Hostel- Manager/Admin</a></p>
       <p class=" w3l-register-p">Don't have an account?<a href="signup.php" class="register">
Sign up</a></p>
    </div></body></html>
```

3.3 BACK END

3.3.1 PHP

- Hypertext Preprocessor is a server-side scripting language designed for Web development,& also used as a general-purpose programming language.
- PHP is implemented in the application as below :
    - Establishing the connection to the Database
    - Requesting data from the Database through sql queries
    - Retrieving the requested data from the Database
    - Displaying the retrieved data

3.3.2 PHP CODE

Before reading or writing a database, a connection must be made to it. Windows hosts have a similar folder hierarchy to a Windows PC, so the location of the files on a Windows server will likely have a path of the form C:\xampp\htdocs\Hostel- Management-System\includes\login.inc.php. You should be able to extract the value of the path to the root folder of your host using the PHP.

FOR SIGN IN (login.inc.php)

```
<?php
 session_start();
 $conn = mysqli_connect('localhost','root');
 if($conn){ echo" connection successful";
 }
 else{
 echo " no connection"
 }
mysqli_select_db($conn, 'hostel_management_system');
```

```
$roll = $_POST['student_roll_no'];
$password = $_POST['pwd'];
if (empty($roll) || empty($password))
{
   header("Location:../index.php?error=emptyfiels;
  exit();
 }
 else{
$sql = " SELECT * FROM Student WHERE Student_id = '$roll' and pwd = '$password'";
$result = mysqli_query($conn, $sql);
$num = mysqli_num_rows($result);
$row = mysqli_fetch_assoc($result);
if($num == 1){
      $_SESSION['roll'] = $row['Student_id'];
     $_SESSION['fname'] = $row['Fname'];
     $_SESSION['lname'] = $row['Lname'];
     $_SESSION['mob_no'] =$row['Mob_no'];
      $_SESSION['department'] =$row['Dept'];
      $_SESSION['year_of_study'] = $row['Year_of_study'];
      $_SESSION['hostel_id'] = $row['Hostel_id'];
      $_SESSION['room_id'] =
       $row['Room_id'];
       if(isset($_SESSION['department'])){
            echo "<script type='text/javascript'>alert('Set')</script>";
        }else {
      echo "<script type='text/javascript'>alert('Not SET')</script>";}
       header("Location:../home.php"); echo "succesful login";
        else{
        header("Location: ../index.php?error=strangeerr");
      exit();}} ?>
```

FOR LOG OUT (logout.inc.php)

```
<?php
  session_start();
  session_unset();
  session_destroy();
  $_SESSION = array();
  header("Location: ../index.php")
 ?>
```

FOR REMOVING HOSTEL MANAGER (hm_remove.php)

```
<?php
if (isset($_POST['hm_remove_submit'])) {
  session_start();
  $conn = mysqli_connect('localhost','root');
  if($conn){
  echo" connection successful";
  }
  else{
  echo " no connection";
 }
 mysqli_select_db($conn, 'hostel_management_system');
  $username = $_POST['hm_uname'];
  $hostel_name = $_POST['hostel_name'];
  $Adminpassword = $_POST['pass'];
  if (empty($username) || empty($hostel_name) || empty($Adminpassword)) {
    header("Location: ../admin/create_hm.php?error=emptyfields");
    exit();
  }
  else {
      $sql = "SELECT *FROM Hostel_Manager WHERE Username = '$username'";
      $result = mysqli_query($conn, $sql);
      if($row = mysqli_fetch_assoc($result)){
```

```
    $sql2 = "SELECT *FROM Hostel WHERE Hostel_name = '$hostel_name'";
    $result2 = mysqli_query($conn, $sql2);
    if($row2 = mysqli_fetch_assoc($result2)){
      $HNO = $row2['Hostel_id'];
      if ($HNO == $row['Hostel_id']) {
      if ($Adminpassword!=$_SESSION['PSWD']) {
          header("Location: ../admin/create_hm.php?error=wrongpwd");
          exit();
        }
        else {
          $sql3 = "DELETE FROM Hostel_Manager WHERE Username = '$username'";
          $result3 = mysqli_query($conn, $sql3);
          if($result3){
            header("Location: ../admin/create_hm.php?DeletionSuccessful");
            exit();
          }
          else {
            header("Location: ../admin/create_hm.php?error=DeletionFailed");
            exit();
          }
        }
      }
      else {
        header("Location: ../admin/create_hm.php?error=wronghostel");
        exit();
      }
    }
    else {
      header("Location: ../admin/create_hm.php?error=nohostel");
      exit();
}
    if($password != $row['Pwd']){
```

```
      header("Location: ../index.php?error=wrongpwd");
        exit();
      }
      else {
        $_SESSION['roll'] = $row['Student_id'];
        $_SESSION['fname'] = $row['Fname'];
        $_SESSION['lname'] = $row['Lname'];
        $_SESSION['mob_no'] = $row['Mob_no'];
        $_SESSION['department'] = $row['Dept'];
        $_SESSION['year_of_study'] = $row['Year_of_study'];
        $_SESSION['hostel_id'] = $row['Hostel_id'];
        $_SESSION['room_id'] = $row['Room_id'];
        if(isset($_SESSION['department'])){
          echo "<script type='text/javascript'>alert('Set')</script>";
        }
        else {
          echo "<script type='text/javascript'>alert('Not SET')</script>";
        }
        header("Location: ../home.php?login=success");
     }
    }
    else{
      header("Location: ../admin/create_hm.php?error=nouser");
      exit();
    }
  }
}
else {
  header("Location: ../index.php");
  exit();
}
```

3.5 Triggers

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database.

1) CREATE TRIGGER TRI1 AFTER UPDATE ON ROOM
   BEGIN
   UPDATE HOSTEL SET No_of_students = No_of_students + 1
   WHERE hostel.Hostel_id = Hostel_id AND new.Allocated = 1;
   END
   DEFINER=`root`@`localhost`

2) CREATE TRIGGER TRI2 AFTER DELETE ON STUDENT
   BEGIN
   UPDATE HOSTEL SET No_of_students = No_of_students-1
   WHERE hostel.Hostel_id=old.hostel_id
   END
   DEFINER=`root`@`localhost`

3) CREATE TRIGGER TRI3 AFTER UPDATE ON ROOM
   BEGIN
   UPDATE HOSTEL SET No_of_students = No_of_students - 1
   WHERE hostel.Hostel_id = Hostel_id AND new.Allocated = 0;
   END
   DEFINER=`root`@`localhost'

3.6 Stored Procedure

A stored procedure is a set of Structured Query Language (SQL) statements with an assigned Name, which are stored in a relational database management system as a group, so it can be reused and shared by multiple programs.

This stored procedure is used to view all students from student table..

```
select * from students
DEFINER=`root`@`localhost`
```

3.7 Queries

CRUD operations are performed in the backend**.**

1. $query999 = "SELECT * FROM Hostel WHERE Hostel_id = '$HOID'";
   This query is used to display hostel manager information.

2. $query4 = "UPDATE Student SET Hostel_id = NULL, Room_id = NULL WHERE Student_id = '$roll'";
   This query is used to update student table when his room is vacated by manager.

3. $sql3 = "DELETE FROM Hostel_Manager WHERE Username = '$username'";
   This query is used to delete by admin to remove hostel manager.

4. $sql = "SELECT *FROM Hostel_Manager WHERE Username = '$username' and pwd = '$password'";
   This query is used when hostel manager wants to log in.

5. $qy = "INSERT INTO Student (Student_id, Fname, Lname, Mob_no, Dept, Year_of_study, Pwd) VALUES ('".$roll."', '".$fname."', '".$lname."', '".$mobile."', '".$dept."', '".$year."', '".$password."')";
   This query to used when student is registered in hostel room allocation system.

CHAPTER 4

TESTING

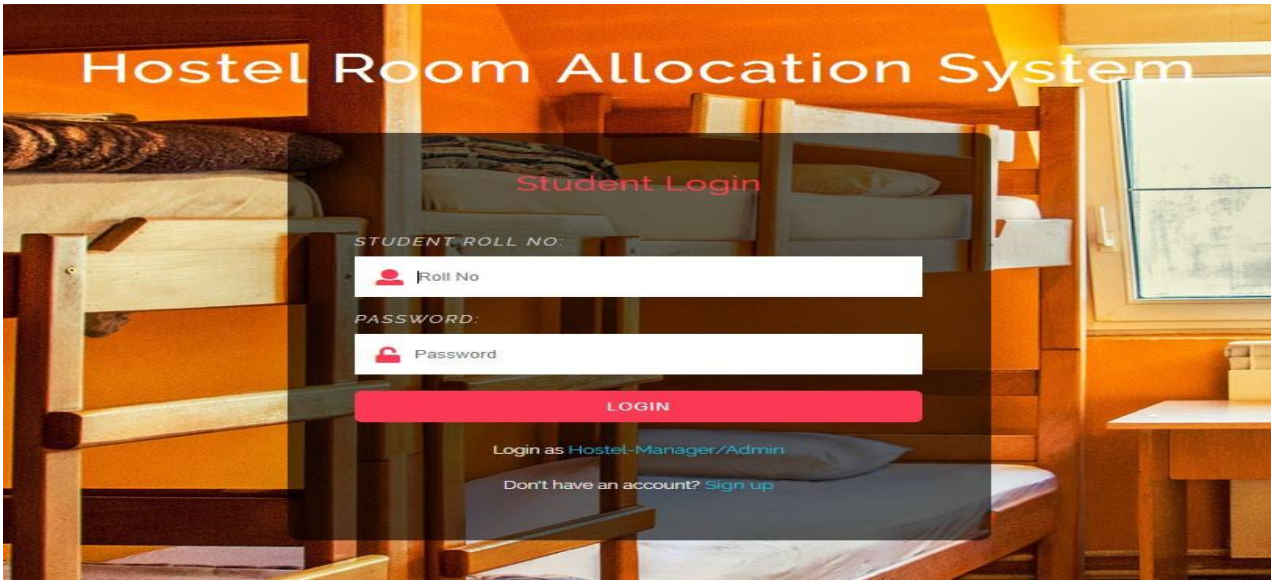
Table 4.1: Testing

S.NO	INPUT	OUTPUT	REMARKS
1	Open in chrome, myphpadmin	Localhost refused to connect Try checking connection ERR_CONNECTION	Xampp not started ie; Apache server and mysql not started.
2	Open in chrome, the localhost	Localhost refused to connect Try checking connection ERR_CONNECTION	MySQL not started
3	Insertion INSERT INTO ROOM(3, 4, 21, 0);	Error: duplicate entry '3' for key 'primary'	UNIQUE constraint violated. Integrity constraint.
5	Insertion INSERT INTO student VALUES ('1BI16IS013', Ayush, Rawat, 9456313690, ISE, 2018, 6, 11)	Cannot add or update a child row: a foreign key constraint fails. CONSTRAINT fkhid FOREIGN KEY(hostel_id) REFERENCES HOSTEL(hostel_id);	hostel_id and room_id are foreign constraint hence can't be inserted until the hostel_id is added in the hostel table.

CHAPTER 5

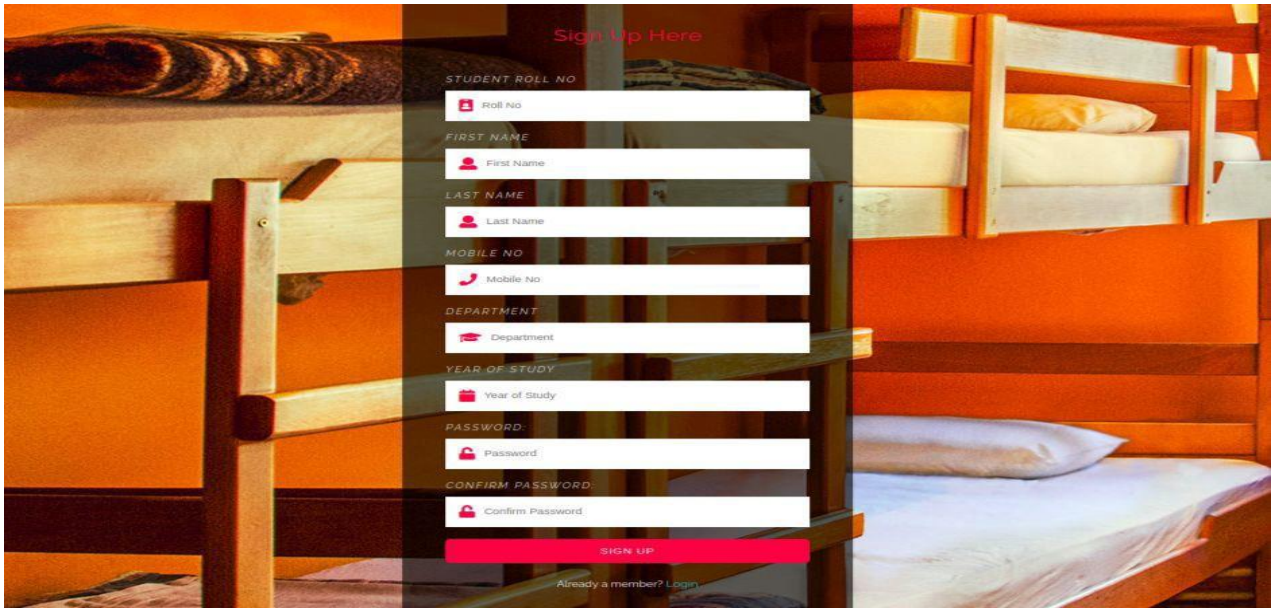
SNAPSHOTS

5.1 Login Page



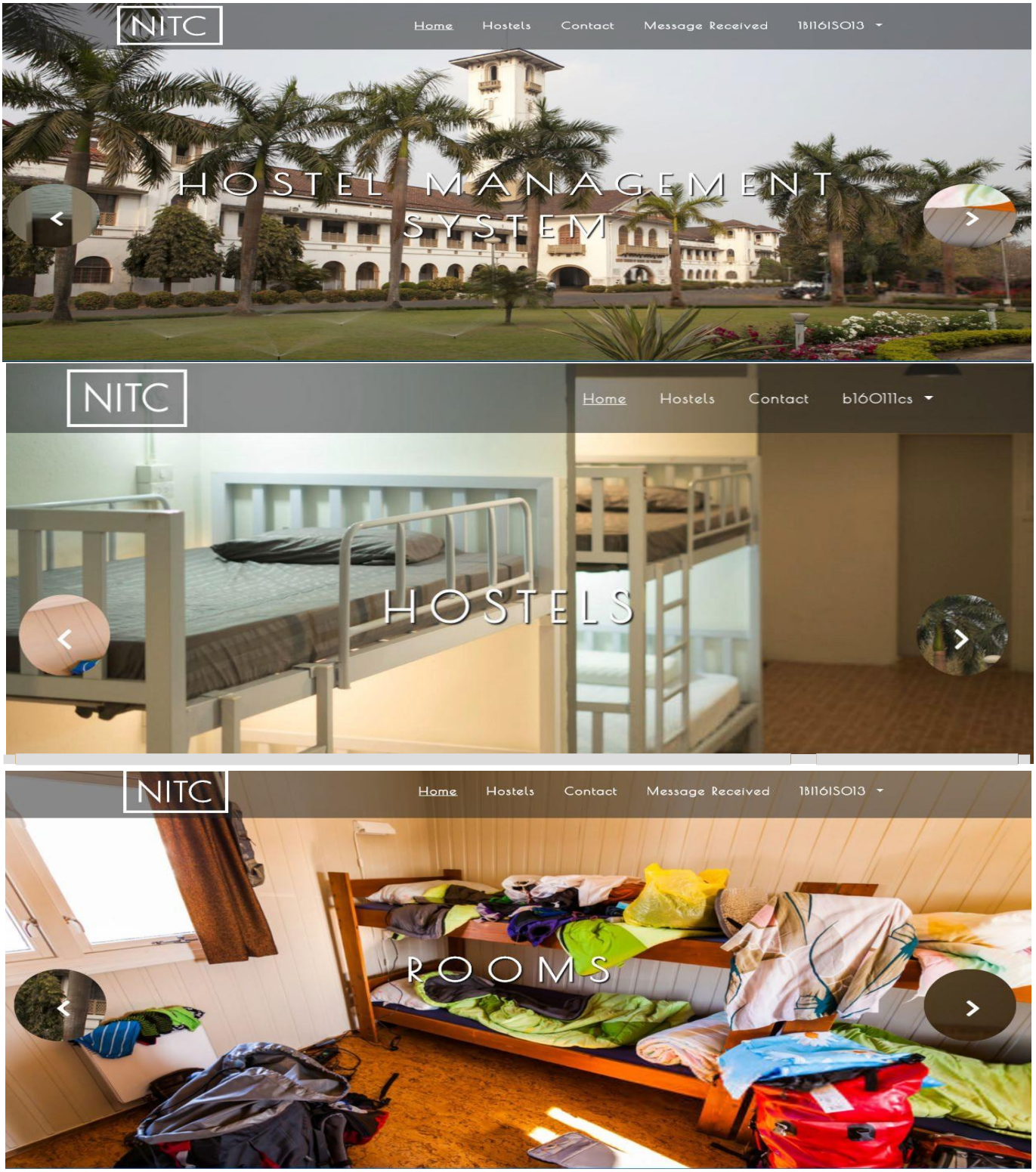
Users who have previously registered for the Hostel Room Allocation System Web Application . Login to advance to the next screen and begin using the application.

5.2 Sign up Page



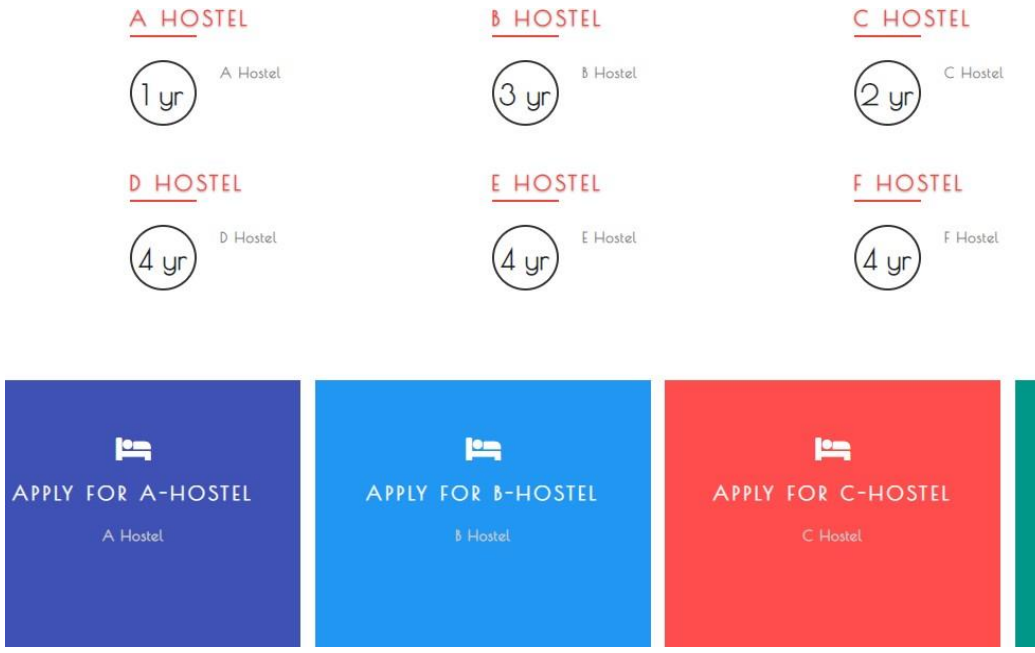
Students who are not a part of the portal can register themselves here. Hostel Manager cannot sign up, he/she can only login. Hostel manager can only be created by Admin!

5.3 Student Home Page



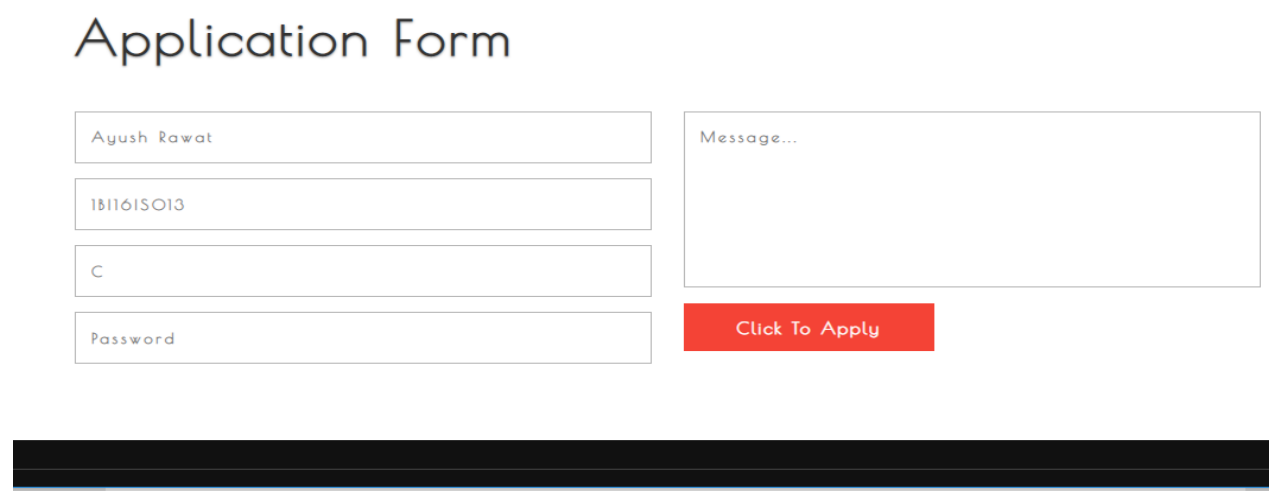
The homepage basically cointains a navigation bar which takes the user to different pages to perform different activities.

5.4 Hostels Page



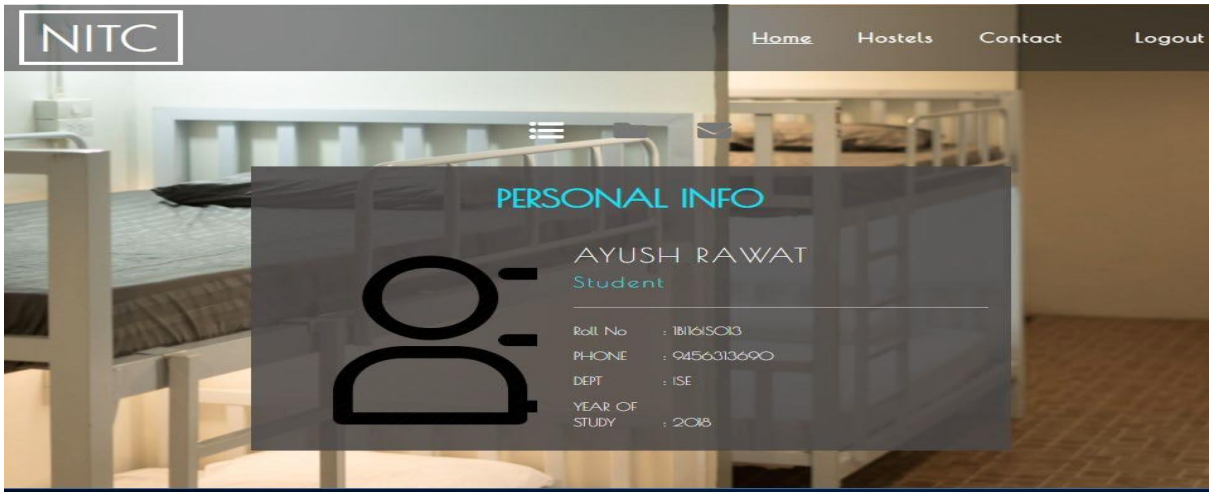
Hostel Page is accessible only to students and they can request for room in any hostel by just clicking on the corresponding hostel link and confirming by entering their password.

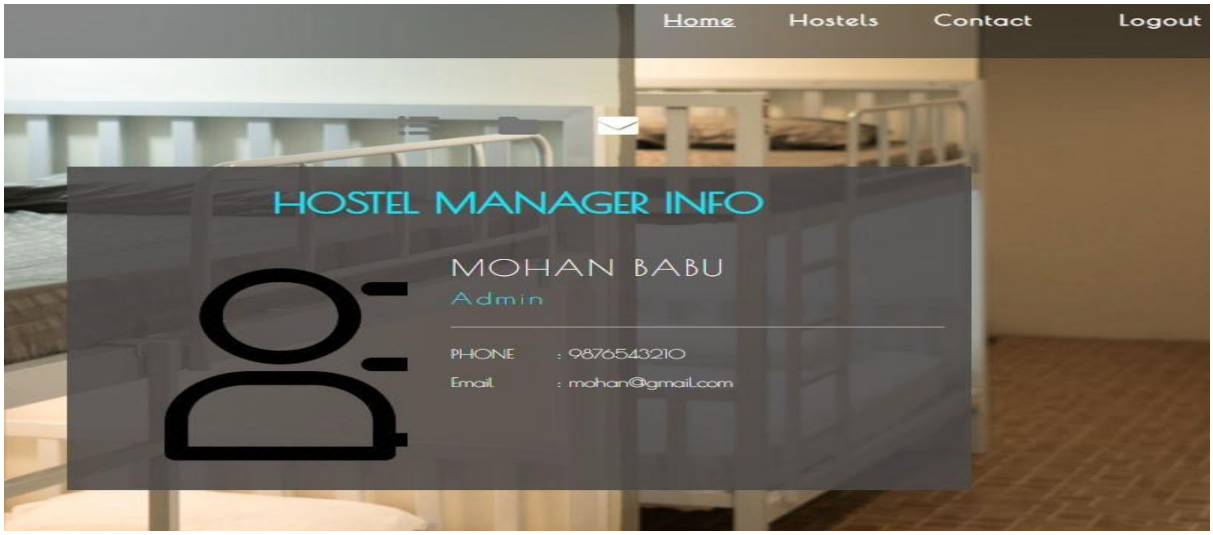
5.5 Application Form



Student should enter password to confirm . If the student is already allocated room, he won't be allowed to submit any application forms.

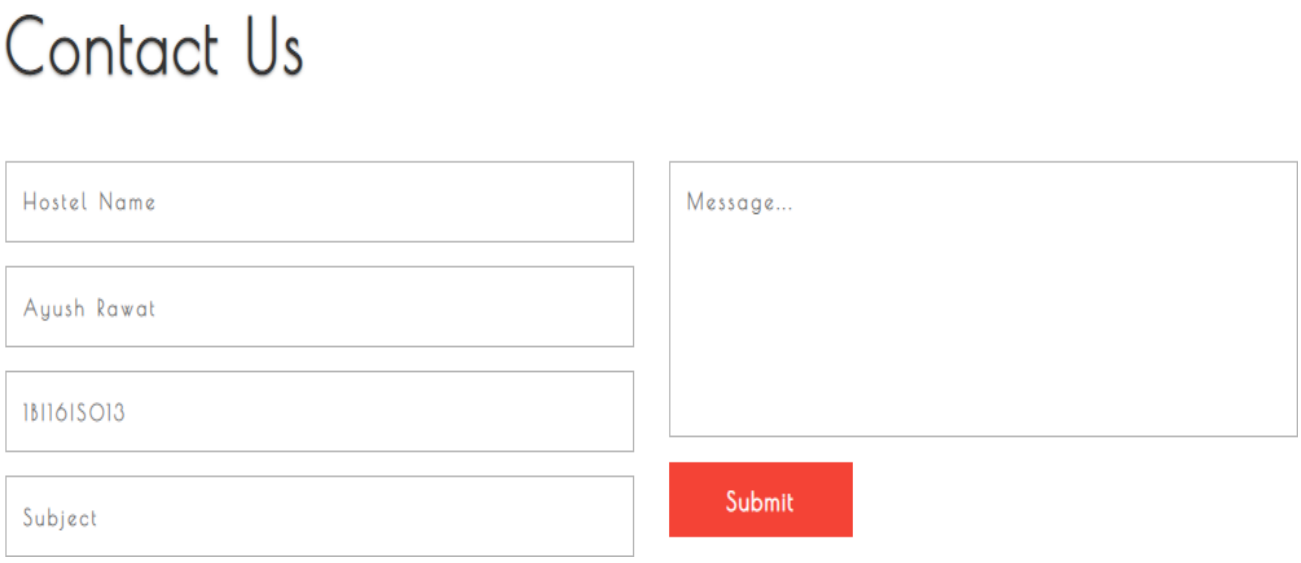
5.6 Student Profile Page





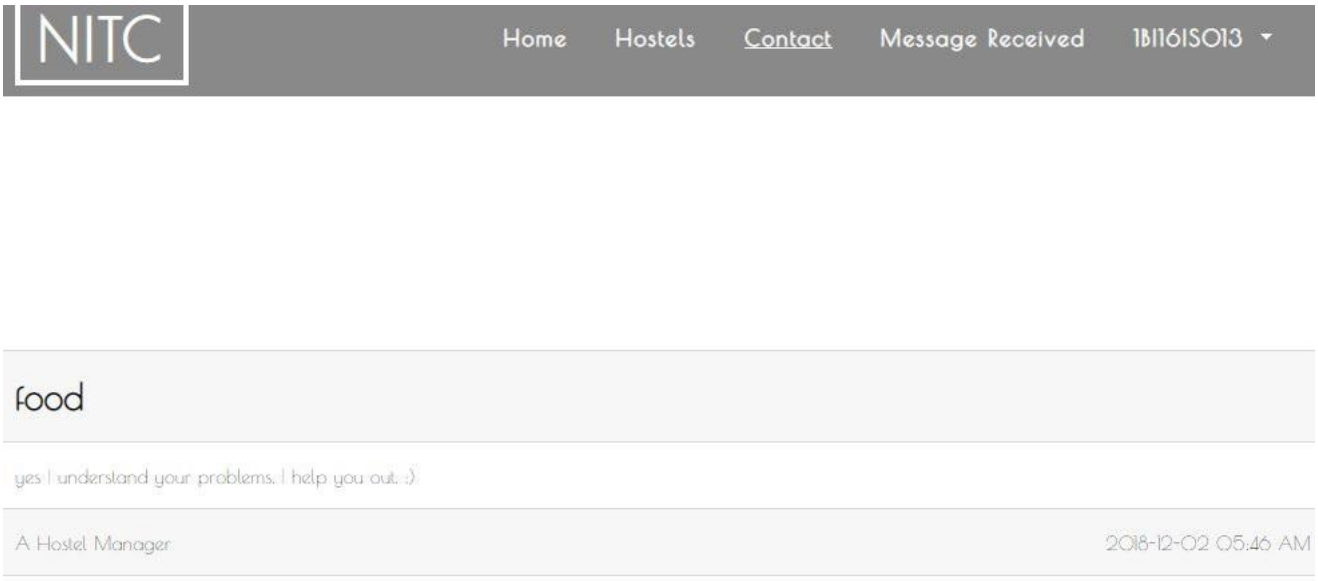
For a student his user profile will also contain the hostel details and hostel manager info if he is allocated a room.

5.7 Contact Form



Students can send messages to hostel manager using this form by clicking ‘Contact’ and ‘Submit’.

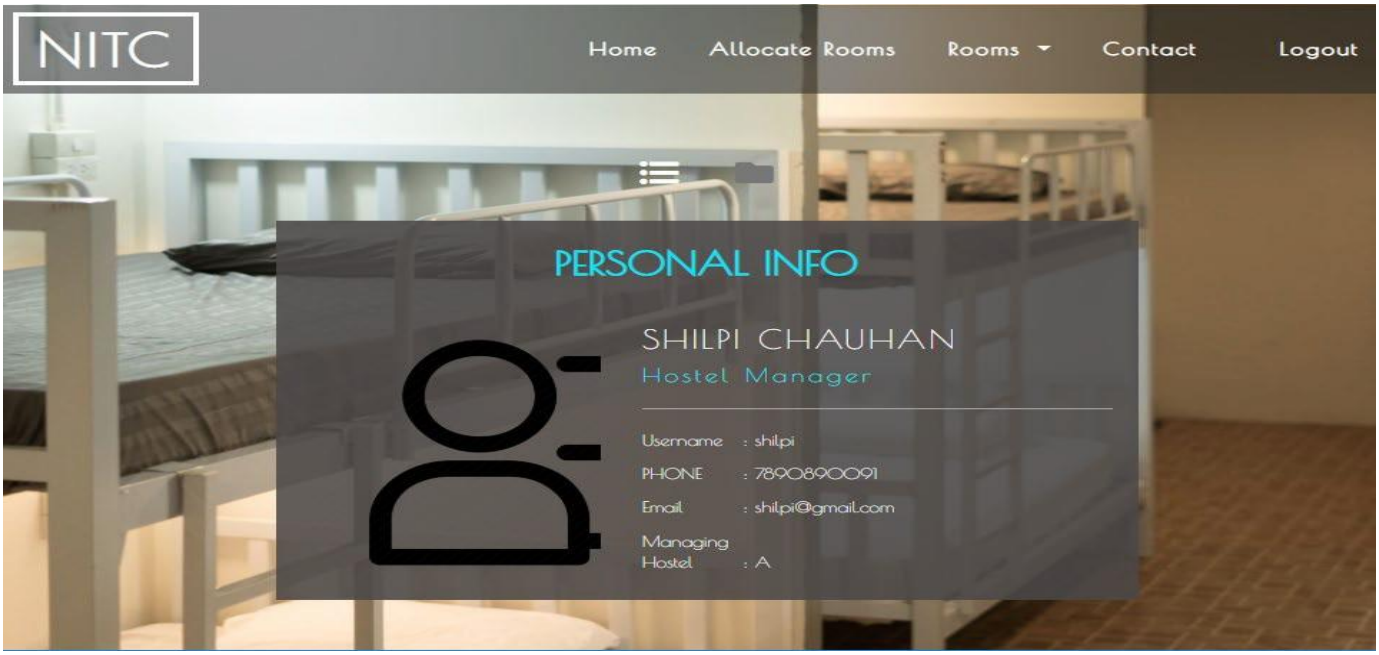
5.8 Messages Received Page



Messages replied by the manager to student is shown here.

5.9 Hostel- Manager/ Admin Login Page

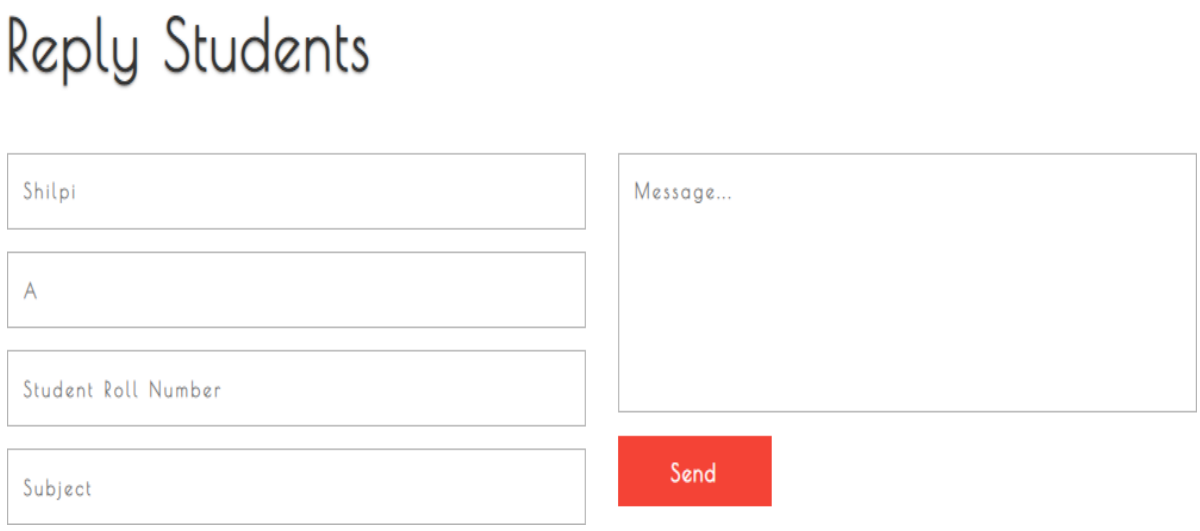
5.8 Hostel Manager Profile Page



5.8 Applications Received

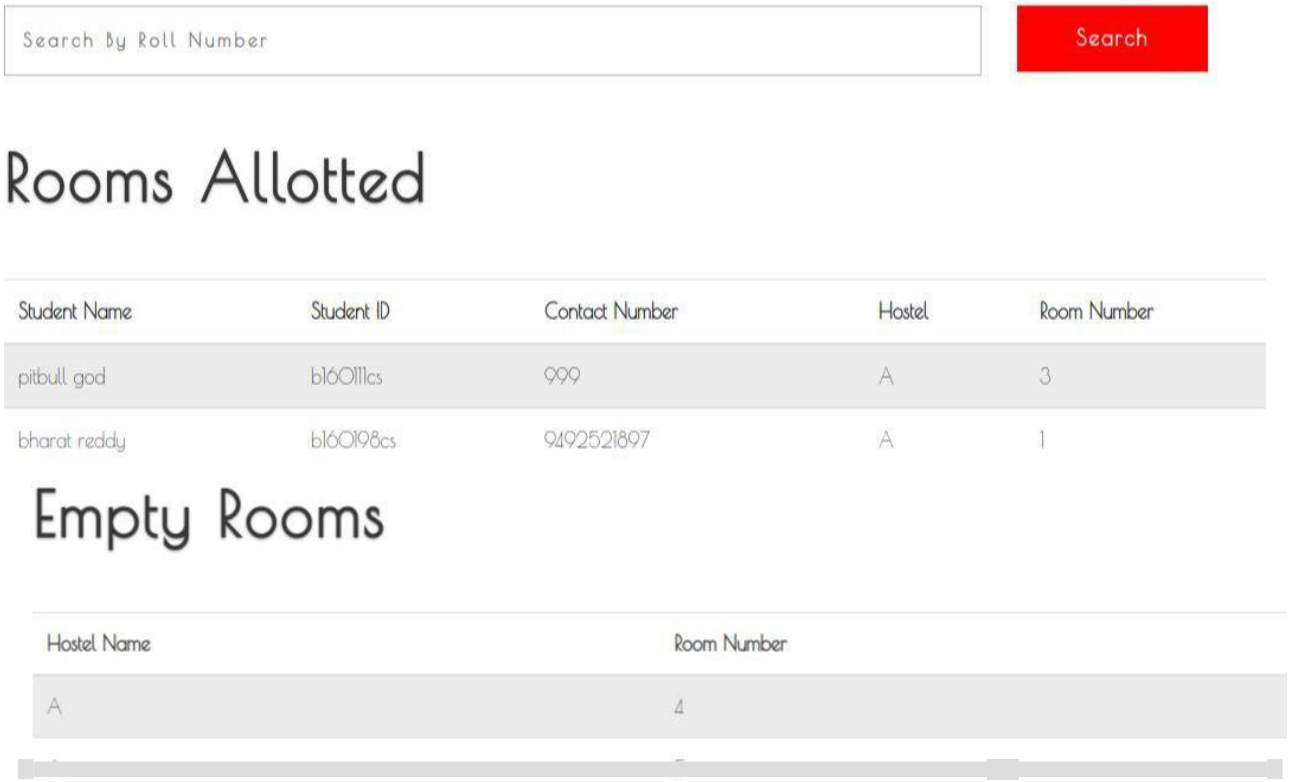
This tab can only be accessed by hostel manager. He can view all the application received and allocate Rooms accordingly .

5.9 Reply Students



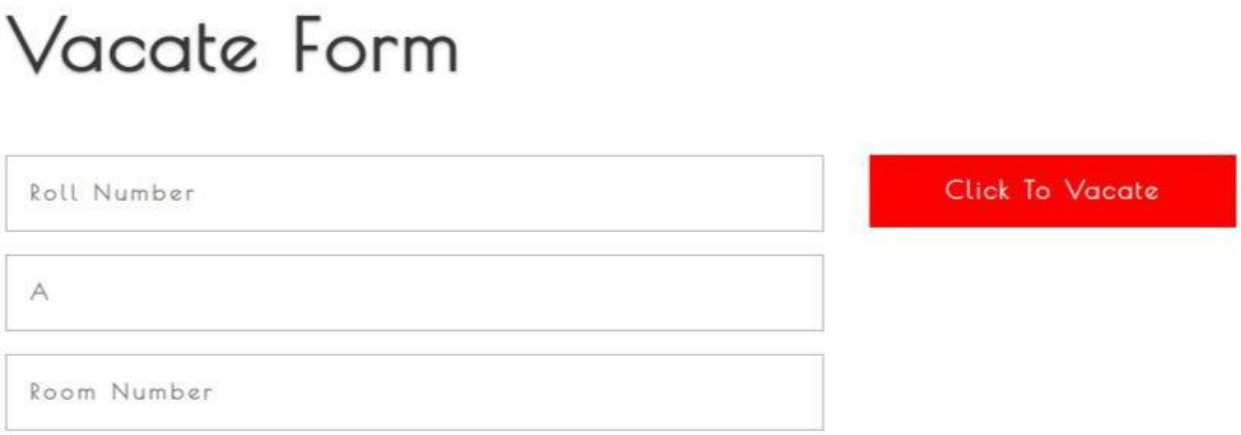
Messages sent from students are received by hostel manager and reply back.

5.10 Alloted and Empty Rooms



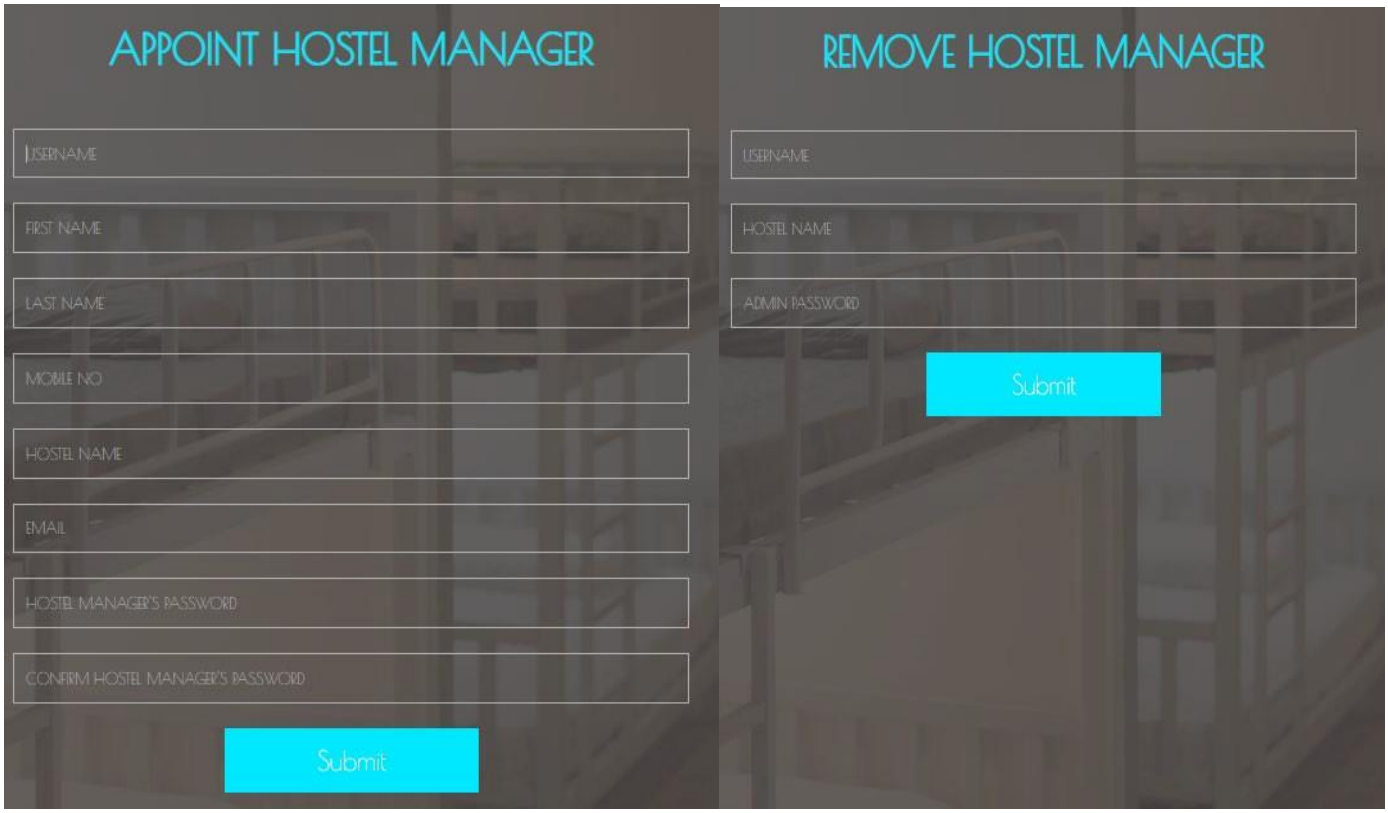
Each hostel manager can look at the list of allocated rooms and empty rooms in his/her hostel by going to this tab.

5.11Vacate Rooms



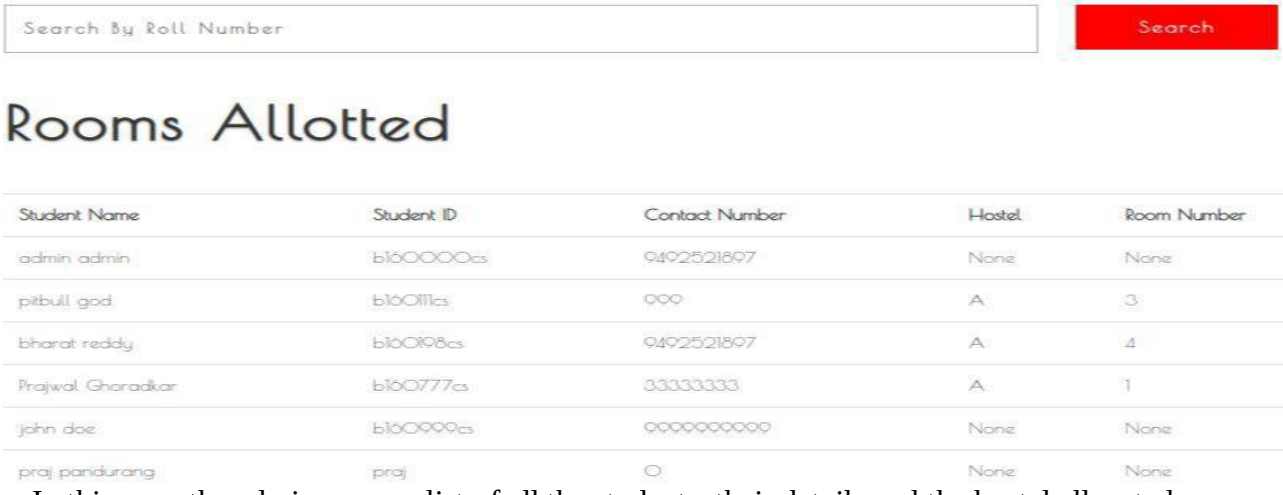
Hostel Manager can vacate an allocated room by filling the student and room details in this form and clicking on “click to vacate” button.

5.12 Appoint/Remove Hostel Manager



This page can only be accessed by hostel admin. He can allocate a new hostel admin o remove an existing hostel manager.

5.13 Students Page



In this page the admin can see list of all the students, their details and the hostel allocated to them along with the room.

6. CONCLUSION

From proper analysis of positive points and constraints on component, it can be concluded that the product is a highly efficient and will be of great help to the general public. This can be run on any system. The website is user friendly and is available to all. Manual method can be completely wiped off due to introduction of the webpage.

7. FUTURE IMPROVEMENTS

- The system is just a simple database design but still designed at maximum possible excellence.
- Still I accept drawbacks, as it is a human effort.
- Website security must be increased.
- Some complex queries to acquire multiple table data to retrieve can be included also ‘delete column’ can be provided to delete entries.
- There can be many more future enhancement and improvement in the database system designed.

8. RERERENCES

- https://www.w3schools.com/
- https://stackoverflow.com/
- Elmasri and Navathe ,Fundamental Database Management,7th edition
- https://www.apachefriends.org
- Elmasri & Navathe : Fundamental Database System ,7th edition 2017
- Ramakrishnan and Gherke : Database Management Systems, 3rd Edit