**Chapter 1**

# INTRODUCTION

 This Hostel Management System is developed in favor of the hostel management team which helps them to save the records of the students about their rooms and other things. It helps them from the manual work from which it is very difficult to find the record of the students and the information about those ones who had left the hostel years before.

This solution is developed on the plight of the hostel management team, through this they cannot require so efficient person to handle and manage the affairs of the students in the hostel, all you need to do is to login as administrator and you can see the information of all the students who have obtained and registered their hostel form, click verify to ascertain their eligibility and allocate them to the available hostel.

Identification of the problems of the existing hostel management leads to the development of computerized solution that will be compatible to the existing hostel management with the solution which is more users friendly and more GUI oriented. We can improve the efficiency of the hostel management, thus overcome the drawbacks of the existing management.

**1.1 Database management system (DBMS):**

A database management system (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data.

A DBMS makes it possible for end users to create, read, update and delete data in a database. The DBMS essentially serves as interface between the database and end users or application programs, ensuring that data is consistently organized and remains easily accessible.

The DBMS manages three important things: the data, the database engine that allows data to be accessed, locked and modified – and the database schema, which defines the database’s logical structure. These three foundational elements help provide concurrency, security, data integrity and uniform administration procedures. Typical database administration tasks are supported by the DBMS include change management, performance monitoring/tuning and backup and recovery. Many database management systems are also responsible for automated rollbacks, restarts and recovery as well as the logging and auditing of activity.

The DBMS perhaps most useful for providing a centralized view of data that can be accessed by multiple users , for multiple locations , in a controlled manner. A DBMS can limit what data the end user sees, as well as how that end user can view the data, providing may views of single database schema. End users can software programs are free from having to understand where the data is physically located or on what type of storage media it resides because the DBMS handles all requests.

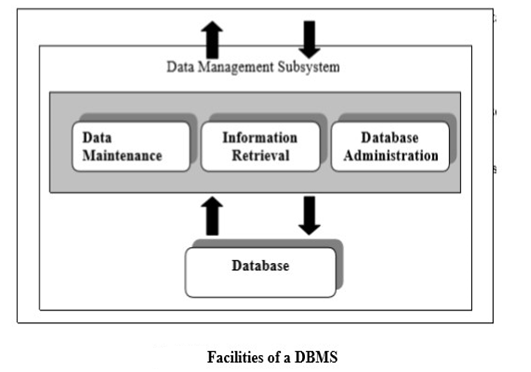


Fig 1.1 Facilities of a DBMS

**1.2 Structured Query Language (SQL):**

SQL was one of the first commercial Languages for Edgar F. Codd’s relational model, as described in his influential 1970 paper, ‘A Relational Model of Data for Large Shared Data Banks”. Despite not entirely adhering to the relation model as described by Codd, it becomes the most widely used database language.

SQL become a standard of the American National Standard Institute (ANSI) in 1986,

And of the International Organization for Standardization(ISO) in 1987.Since then, the standard has been revised to include a large set of features. Despite the existing of such standards, most SQL code is not completely portable among different database systems without adjustments.

**1.3 Front end development using Notepad:**

Front-end and Back-end are terms used to characterize program interfaces and services relative to the initial user of these interfaces and services. A “front-end” application is the one application users interact with directly. A “back-end” application serves in directly in support of the front-end services, usually by being closer to the required resource or having the capability to communicate with required resource. The back-end application may interact directly with front-end or, perhaps more typically, is a program called from an intermediate program that mediates front-end and back-end activities.

**Microsoft Visual Studio** is an [integrated development environment](https://en.wikipedia.org/wiki/Integrated_development_environment) (IDE) from [Microsoft](https://en.wikipedia.org/wiki/Microsoft). It is used to develop [computer programs](https://en.wikipedia.org/wiki/Computer_program), as well as [websites](https://en.wikipedia.org/wiki/Web_site), [web apps](https://en.wikipedia.org/wiki/Web_app), [web services](https://en.wikipedia.org/wiki/Web_service) and [mobile apps](https://en.wikipedia.org/wiki/Mobile_app). Visual Studio uses Microsoft software development platforms such as [Windows API](https://en.wikipedia.org/wiki/Windows_API), [Windows Forms](https://en.wikipedia.org/wiki/Windows_Forms), [Windows Presentation Foundation](https://en.wikipedia.org/wiki/Windows_Presentation_Foundation), [Windows Store](https://en.wikipedia.org/wiki/Windows_Store) and [Microsoft Silverlight](https://en.wikipedia.org/wiki/Microsoft_Silverlight). It can produce both [native code](https://en.wikipedia.org/wiki/Machine_code) and [managed code](https://en.wikipedia.org/wiki/Managed_code).

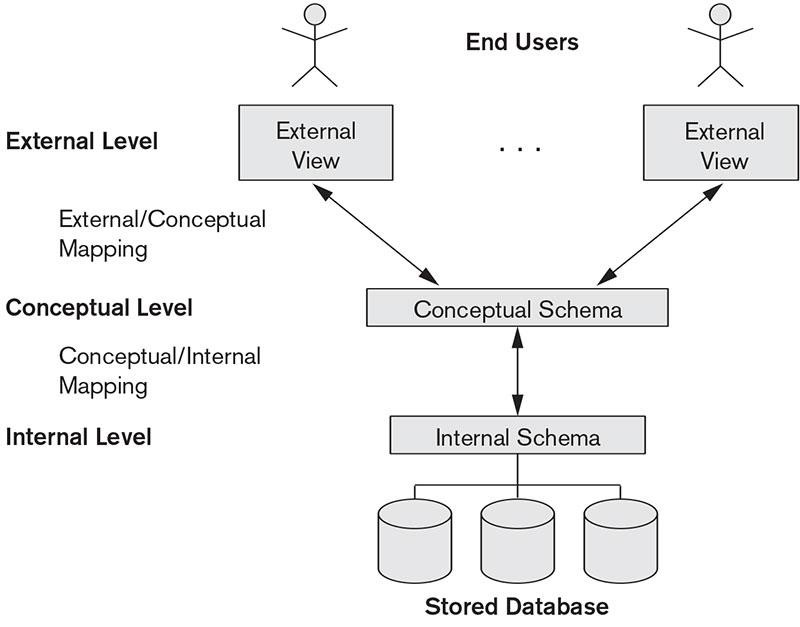
Visual Studio supports 36 different [programming languages](https://en.wikipedia.org/wiki/Programming_language) and allows the code editor and debugger to support (to varying degrees) nearly any programming language, provided a language-specific service exists.Built-in languages include C,[8] C++, C++/CLI, Visual Basic .NET, [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [F#](https://en.wikipedia.org/wiki/F_Sharp_(programming_language)),[[9]](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio#cite_note-9) [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [TypeScript](https://en.wikipedia.org/wiki/TypeScript), [XML](https://en.wikipedia.org/wiki/XML), [XSLT](https://en.wikipedia.org/wiki/XSLT), [HTML](https://en.wikipedia.org/wiki/HTML), and [CSS](https://en.wikipedia.org/wiki/Cascading_Style_Sheets). Support for other languages such as [Python](https://en.wikipedia.org/wiki/Python_(programming_language)),[[10]](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio#cite_note-10) [Ruby](https://en.wikipedia.org/wiki/Ruby_(programming_language)), [Node.js](https://en.wikipedia.org/wiki/Node.js), and [M](https://en.wikipedia.org/wiki/MUMPS) among others is available via [plug-ins](https://en.wikipedia.org/wiki/Plug-in_(computing)). [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) (and [J#](https://en.wikipedia.org/wiki/J_Sharp)) were supported in the past.

The entire project has been developed keeping in view of the distributed client server computing technology. The user interfaces are browser specific to give distributed accessibility for the overall system. The database connectivity was planned using the latest “XAMPP” technology provided by Apache consisting mainly of the Apache HTTP Server, and interpreters for scripts written in PHP and Perl programming languages. The authentication and authorization was cross checked at all the relevant stages.

**1.4 Three Schema Architecture:**

Following are the three levels of database architecture:

* Physical Level
* Conceptual Level
* External Level



## Fig 1.2 The Three Schema Architecture

In the above diagram,

* It shows the architecture of DBMS.
* Mapping is the process of transforming request response between various database levels of architecture.
* Mapping is not good for small database, because it takes more times.
* In External/Conceptual mapping, DBMS transforms a request on an external schema against the conceptual schema.
* In Conceptual/Internal mapping, it is necessary to transform the request form the conceptual to internal levels.

1. **Physical Level:**
   * + Physical level describes the physical storage structure of data in database.
     + It is also known as Internal Level.
     + This level is very close to physical storage of data.
     + At lowest level, it is stored in the form of bits with the physical addresses on the secondary storage device.
     + At highest level, it can be viewed in the form of files.
     + The internal schema defines the various stored data types. It uses a physical data model.
2. **Conceptual Level:** 
   * + Conceptual level describes the structure of the whole database for a group of users.
     + It is also a known as the data model.
     + Conceptual schema is a representation of the entire content of the database.
     + This schema contain all the information to build relevant external records.
     + It hides the internal details of physical storage.
3. **External Level:** 
   * + External level is related to the data which is viewed by individual end users.
     + This level includes a no. of user views or external schemas.
     + This level is closest to the user.
     + External view describe the segment of the database that is required for a particular user group and hides the rest of the database from that user group.

**1.5 NORMALIZATION:**

Here are the most commonly used normal forms:

**1). First normal form (1NF):**

* + - As per the rule of first normal form, an attribute(column) ofa table cannot hold multiple values. It should hold only atomic values.

**2). Second normal form (2NF):**

* + - A table is said to be in 2NF if both the following conditions hold:
    - Table is in 1NF (First normal form)
    - No non-prime attribute is dependent on the proper subset of any candidate key of table.
    - An attribute that is not part of any candidate key is known as non-prime attribute.

**3). Third normal form (3NF):**

* + - A table design is said to be in 3NF if both the following conditions hold:
    - Table must be in 2NF
    - Transitive functional dependency of non-prime attribute on any super key should be removed
    - An attribute that is not part of any candidate key is known as non-prime attribute.
    - In other words 3NF can be explained like this : A table is in 3NF if it is in 2NF and for each functional dependency X->Y at least one of the following condition hold:
    - X is a super key of table
    - Y is a prime attribute of table
    - An attribute that is a part of one of the candidate keys is known as prime attribute.

**Fourth Normal Form(4NF):** is a normal form used in normalization.Introduced by Ronald Fagin in 1997,4NF is the next level of normalization after Boycee-Codd normal form (BCNF).Whereas the second,third,and Boyce-Codd normal forms are concerned with functional dependencies ,4NF is concerned with a more general type of dependency known as multivalued dependency.A tables is in 4NF if and only if ,for every one of its non-trivial multivalued dependies X,Y ,X is a super-key that is ,X is either a candidate key or a super set.

**Fifth Normal Form(5NF) :** A database is said to be decomposes in 5th Normal Form ,if and only if, it is in 4th normal form and if the table further to eliminate redundancy and anomaly and when we joins the decomposed tables by means of candidate keys we should not be losing the original data or any new record set should not arise. In simple words joining two or more decomposed tables should not lose records or new records.

**Boyce and Codd normal form (BCNF):** It is an advance version of 3NF that’s why it is also referred as 3.5NF. BCNF is stricter than 3NF. A table compiles with BCNF if it is in 3NF and for every functional dependency X->Y, X should be the super key of the table.

**Chapter 2**

**REQUIREMENT SPECIFICATION**

**2.1 Software Requirement:**

* Technology: HTML,CSS,PHP,JAVASCRIPT, XAMPP. Database: My SQL.
* Runtime Environment: Web Server: Tomcat Apache 9.0.
* Operating system: Windows 10.0
* Browser : Google Chrome

**2.2 Hardware Requirement:**

* Intel P4 1.5 GHz Processor or Above
* RAM 512MB and Above
* HDD 20 GB Hard Disk Space or Above

**2.3 About Connectivity:**

**Creating a Database Using MySQL:**

* Let create a MySQL database using MySQL Command Line Client.
* Provide the password for root user login.
* Create a database named MyNewDatabase and then exit.

Ex: CREATE DATABASE MyNewDatabase;

**Making a Connection Using PHP:**

* [Create MySQL Database at the Localhost](https://www.cloudways.com/blog/connect-mysql-with-php/#createmysql)
  + [Create Database](https://www.cloudways.com/blog/connect-mysql-with-php/#createdatabase)
  + [Create a Folder in htdocs](https://www.cloudways.com/blog/connect-mysql-with-php/#createfolder)
  + [Create Database Connection File In PHP](https://www.cloudways.com/blog/connect-mysql-with-php/#connection)
  + [Create new php file to check your database connection](https://www.cloudways.com/blog/connect-mysql-with-php/#newphp)
  + [Run it](https://www.cloudways.com/blog/connect-mysql-with-php/#run)
* [Create MySQL Database at Cloudways Server](https://www.cloudways.com/blog/connect-mysql-with-php/#mysql)
  + [Create Database Connection](https://www.cloudways.com/blog/connect-mysql-with-php/#databaseconnection)
  + [MySQLi Procedural Query](https://www.cloudways.com/blog/connect-mysql-with-php/#mysqli)
  + [Connect MySQL Database with PHP Using PDO](https://www.cloudways.com/blog/connect-mysql-with-php/#connectmysql)
  + [Check Connection](https://www.cloudways.com/blog/connect-mysql-with-php/#check)
* [Remote MySQL](https://www.cloudways.com/blog/connect-mysql-with-php/#remote)
* [Top MySQL Management tools](https://www.cloudways.com/blog/connect-mysql-with-php/#topmysql)
  + [MySQL Workbench](https://www.cloudways.com/blog/connect-mysql-with-php/#workbench)
  + [Navicat For MySQL](https://www.cloudways.com/blog/connect-mysql-with-php/#navicat)
  + [MySQL Yog](https://www.cloudways.com/blog/connect-mysql-with-php/#yog)
  + [Cloudways MySQL Database Manager](https://www.cloudways.com/blog/connect-mysql-with-php/#cloudways)

**Create a new project in the PHP:**

* Go to File Menu and select **New** | **PHP Project**. -Or- In **PHP** Explorer view, right-click and select **New** | **PHP Project**.
* The **New Project** wizard will open. Enter a name for your **new project** into the **Project** Name field.
* Click Finish to complete the **creation** of your **project**.

**Preparing the Interface:**

Begin by preparing a simple interface for the two pages. Both index.jsp and response.jsp implement an HTML table to display data in a structured fashion. index.jsp also requires an HTML form that includes a drop-down list.

# CHAPTER 3:

# SYSTEM DESIGN AND ANALYSIS

**Step 1: Mapping of Regular Entity Types**

For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E. Include only the simple component attributes of E as the primary key for R. If the chosen key of E is a composite, then the set of simple attributes that form it will together form the primary key of R.

**Step 2: Mapping of Weak Entity Types**

For each weak entity type W in the ER schema with owner entity type E, create a relation R & include all simple attributes(or simple components of composite attributes) of was attributes of R. In addition , include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s). The primary key of R is the combination of the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any.

**Step 3: Mapping of Binary 1:1 Relationship Type**

For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R.

**Step 4**: **Mapping of Binary 1: N Relationship Types**

For each regular binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type. Include as foreign key in S the primary key of the relation T that represent the other entity type participating in R. Include any simple attributes of the 1:N relationship type as attributes of S.

**Step 5**: **Mapping of Binary M: N Relationship Types**

For each regular binary M:N relationship type R, create a new relation S to represent R. Include as foreign key attribute in S the primary key of the relations that represent the participating entity type; their combination will form the primary key of S. Also Include any simple attributes of the M:N relationship p type as attributes of S.

**Step 6**: **Mapping of Multi valued attributes**

For each multi valued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R- of the relation that represents the entity type or relationship type that has A as an attribute. The primary key of R is the combination of A and K. If the multivalued attribute is a composite, we include its simple component.

**Step 7**: **Mapping of N-ary Relationship Types**

For each n-ary relationship type R, where n>2, create a new relation S to represent R. Include as foreign key attribute in S the primary key of the relations that represent the participating entity types. So Include any simple attribute of the n-ary relationship type as attribute of S.

# 3.2 ENTITY RELATIONSHIP DIAGRAM

# 

# 

# Fig.3.1 ER diagram of Hostel Management System

# 3.3 SCHEMA DIAGRAM

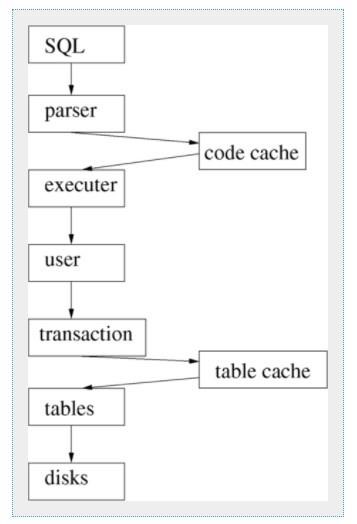
# 

# Fig.3.2 Schema diagram of Hostel Management System

**Chapter 4:**

**IMPLEMENTATION**

A database management system handles the requests generated from the SQL interface, producing or modifying data in response to these requests. This involves a multilevel processing system.



## Fig .4.1 DBMS Execution and Parsing

level structure processes the SQL submitted by the user or application.

**Parser:** The SQL must be parsed and tokenized. Syntax errors are reported back to the user. Parsing can be time consuming, so good quality DBMS implementations cache queries after they have been parsed so that if the same query is submitted again the cache copy can be used instead. To make the best use of this most systems use placeholders in queries, like :

**Executer:** This takes the SQL tokens and basically translate it into relational algebra. Each relational algebra fragment is optimized, and the passed down the levels to be acted on.

**User:** The concept of the user is required at this stage. This gives the query context, and also allows security to be implemented on a per-user basis.

**Transaction:** The queries are executed in the transaction model. The same query from the same user can be executing multiple times in different transactions. Each transaction is quite separate.

**Tables:** The idea of the table structure is controlled at low level.

**Table cache**: Disks are slow, yet a disk is the best way of storing long-term data. Memory is much faster, so it makes sense to keep as much table information as possible in memory. The disk remains synchronized to memory as part of the transaction control system.

**Disks:** Underlying almost all the database system is the disk storage system. This provides storage for the DBMS system tables, user information, schema definition, and the user data itself. It also provides the means for transaction logging.

**4.1 Table Creation:-**

**Hostel**

SQL>create table hostel(hostel\_id varchar(20) primary key , hostel\_type varchar(20) , block varchar(20) , no\_of\_rooms varchar(20));

**Student**

SQL>create table student(name varchar(100) , gender varchar(10) , usn varchar(20) primary key , h\_id varchar(20) , room\_no varchar(20) ,batch varchar(20) , constraint cons1 foreign key (h\_id) references hostel(h\_id));

**Department**

SQL>create table department(usn varchar(20) , dname varchar(40) , constraint cons2 foreign key (usn) references student(usn));

**Student\_details**

SQL>create table student\_details (usn varchar(20) , father\_name varchar(20) , address varchar(20) , ph\_no varchar(20) , email varchar(20) , constraint cons3 foreign key (usn) references student(usn));

**Fees**

SQL>create table fees (usn varchar(20) , fees\_submitted varchar(20) , total\_fees varchar(20) , recipt\_no numeric(10) , due\_date varchar(20) , constraint cons4 foreign key(usn) references student (usn));

**Fees\_log**

SQL>create table fees\_log (usn varchar(20) , fee\_left varchar(10) , date varchar(20) , constraint cons5 foreign key(usn) references student(usn));

**4.2 Source Code:-**

**Code for SIGN\_UP:**

<?php

$name = $\_POST['uname'];

$pass = $\_POST['psw'];

$q = "select \* from signin where username='$name'";

$result=mysqli\_query($con,$q);

$num=mysqli\_num\_rows($result);

if($num==1)

{

?>

<script>

window.alert('User name already exists.');

window.location.href='../signup.php';

</script>

<?php

}

else

{

$qy="insert into signin(username,password) values ('$name','$pass')";

mysqli\_query($con,$qy);

?>

<script>

window.alert('Registration Successful.');

window.location.href='../login.php';

</script>

<?php

**Code for LOGIN:**

<?php

if(isset($\_POST['submit']))

{

$name = $\_POST['uname'];

$pass = $\_POST['psw'];

$q = "select \* from signin where username='$name' && password='$pass'";

$result=mysqli\_query($con,$q);

$num=mysqli\_num\_rows($result);

if($num==1)

{

?>

<script>

window.location.href='../student/hostel.php';

</script>

<?php

}

else

{

?>

<script>

window.alert('Invalid Username and Password');

window.location.href='../login.php';

</script>

<?php

**Code for INSERT:**

<?php

session\_start();

$con = mysqli\_connect('localhost','root','root');

if($con)

{

echo "connection successful";

}

else{

echo "no connection";

}

$con->set\_charset("utf8");

mysqli\_select\_db($con,'project');

$name = $\_POST['Name'];

$email = $\_POST['Email'];

$message=$\_POST['Message'];

$query = "insert into complaint\_forum(name,email,message) values ('$name','$email','$message')";

$result=mysqli\_query($con,$query);

?>

**Code for DELETE:**

<?php

if(isset($\_POST["num"]))

{

$num=$\_POST["num"];

$a=implode(",",$num);

echo $a;

$q="delete from complaint where name in ('$a')";

mysqli\_query($con,$q);

?>

**Code For DISPLAY:**

<?php

$usn = $\_POST['usn'];

$q = "select \* from student s,student\_details d,department b,fees f ,fees\_log fl,hostel h

where s.usn=d.usn and s.usn=b.usn and s.usn=f.usn and s.usn=fl.usn and s.hostel\_id=h.hostel\_id and s.usn='$usn'";

$result=mysqli\_query($con,$q);

$num=mysqli\_num\_rows($result);

if($num==1){

$row = mysqli\_fetch\_array($result);

$sname = $row['name'];

$susn = $row['usn'];

$img = $row['img\_src'];

$sgender = $row['gender'];

$address = $row['address'];

$phoneno = $row['phone\_no'];

$email = $row['email'];

$fname = $row['f\_name'];

$batch = $row['batch'];

$branch = $row['department\_name'];

$roomno = $row['room\_no'];

$feessubmitted = $row['fees\_submitted'];

$feesleft = $row['fee\_left'];

$dateofsubmission = $row['date'];

$reciptno = $row['recipt\_no'];

$duedate = $row['due\_date'];

$block = $row['block'];

$hosteltype = $row['hostel\_type'];

}

else{

?>

<script>

window.alert('USN not exist!');

window.location.href='../../student/studentsearch.php';

</script>

<?php

}?>

**Code for UPDATE:**

<?php

$usn = $\_POST['usn'];

$amount = $\_POST['amount'];

$reciptno=$\_POST['reciptno'];

$duedate=$\_POST['duedate'];

$q = "update fees set fees\_submitted='$amount',recipt\_no='$reciptno',due\_date='$duedate' where usn='$usn'";

$result=mysqli\_query($con,$q)or die(mysqli\_error($con));;

?>

**CHAPTER 5**

# TESTING

**5.1 Introduction:**

The purpose of testing is to discover errors. Testing is the process of trying to every convincible fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies, and/or finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner.

**5.2 LEVELS OF TESTING:**

**5.2.1 UNIT TESTING:**

Unity testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures and Operating procedures .For unit testing first we adopted the code testing strategy, which examined the logic of program

**5.2.2 USER ACCEPTANNCE TESTING:**

User acceptance testing of the system is the key factor for the success of the system. A system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system at the time of development and making change whenever required. This is done with regard to the input screen design and output screen design.

**5.2.3 GUI Testing:**

GUI testing is use to ensure the visual clarity of the system, flexibility of the system, user friendliness of the system. The various component which are to be tested are: i. Relative layout ii. Various Link and Buttons.

**5.2.4 Validation testing:**

At the culmination of black box testing, software is completely assembled is a package. Interfacing errors have uncovered and the correct and final states of tests i.e. validation is defined with a simple definition that validation succeeds when the software function in a manner that can be reasonably accepted by the customer.

**5.2.5 Output Testing:**

After performing validation testing, the next step is output testing of the proposed system. Since the system cannot be useful if it does not procedure the required output. Asking the user about the user about its required format in which the system is required tests the output displayed or generated by the system under consideration.

**Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Case Id** | **Test Case** | **Test**  **Condition** | **Excepted**  **Output** | **Actual Output** | **Pass /**  **Fail** |
| 1 | Validation  Test Case | Required  Field  Validation | Mandatory  field should not be blank | You have to enter value in mandatory  Field | Pass |
| Regular  Expression | A predefine format should be follow | Check proper format | Pass |
| Compare  Validation | Check with  predefine control | Compare with  control | Pass |
| 2 | Login on | Username | Username format Must be Input | Username format must be input. | Pass |
| Confirmation  Password | Password &  confirm password match | Password &  Confirm password match. | Pass |
| Text field | All information must be input | All information must be input | Pass |
|  | | | |

**CHAPTER 6**

# CONCLUSION AND FUTURE ENHANCEMENT

**CONCLUSION**

The development of the hostel management system is a user friendly system to indents to automate the hostel from manual systems. It’s a web application based system that will manage all the operations of the hostel. This system offers useability,stability and cost effectiveness also flexible and adaptable by any device or browser.

**FUTURE ENHANCEMENT**

It is easy to extend the system that we have proposed. A person could see any of the issued, unissued or all the rooms according to his/her will. In future we can implement some features for “HOSTEL MANAGEMENT SYSTEM” project. In this system its possible to categorize room rent for middle class students and poor students. Some poor students are given a particular concession for the entire year

# BIBLIOGRAPHY

Robin Nixon “Learning PHP,MySQL & JavaScript”.

Raghu Ramakrishnan “Database Management System”.

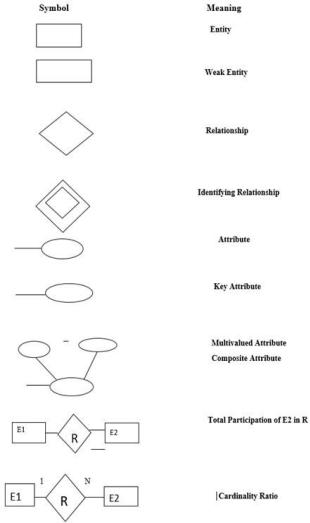
[http://www.Quora.com](http://www.quora.com/).

[http://www.w3schools.com](http://www.w3schools.com/).

[http://www.Github.com](http://www.github.com/).

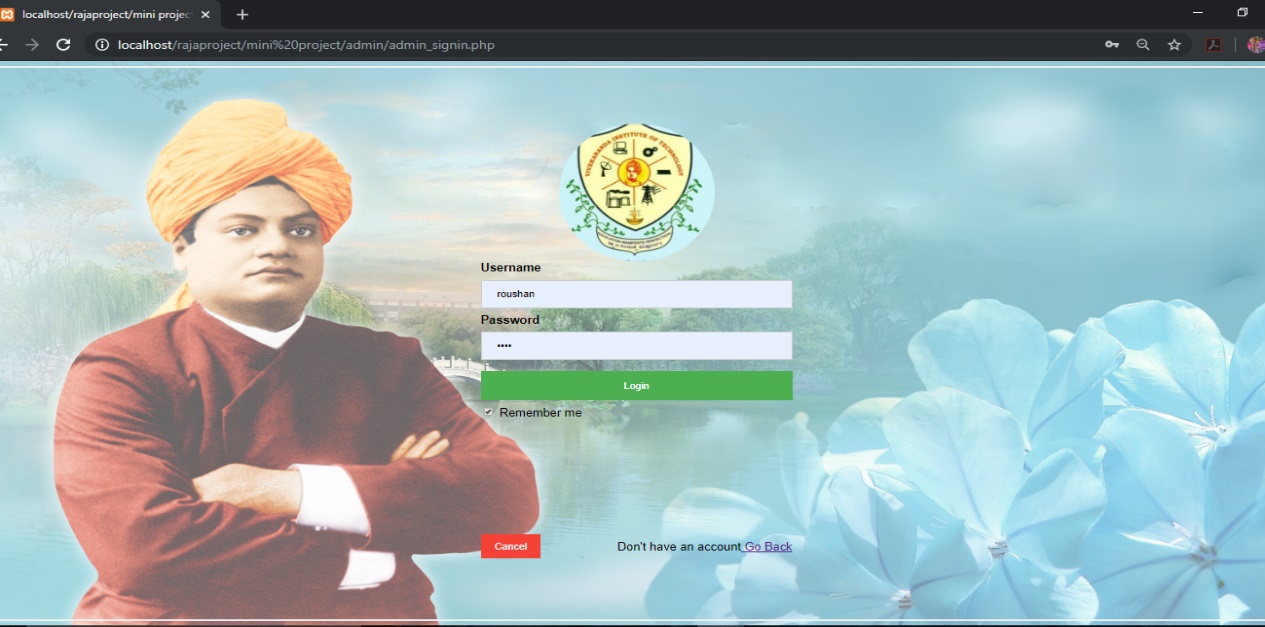
**APPENDIX**

**Notations for ER diagram**



**Fig : Notation for ER diagram**

# 1. LOGIN PAGE

****

**Fig:1 Login Page**

Login page provide Admin username and admin password for login purpose. The username and the password should be valid.

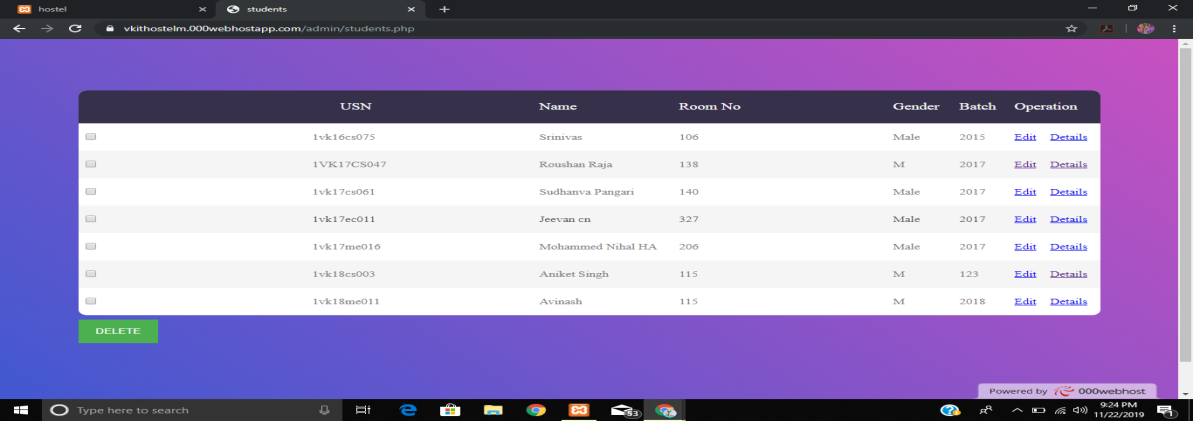
**2. ADMIN INTERFACE**

****

**Fig:2 Admin Home Page**

Home page form provide the different operation that can be perform by the user. User can click on buttons to perform specific action.

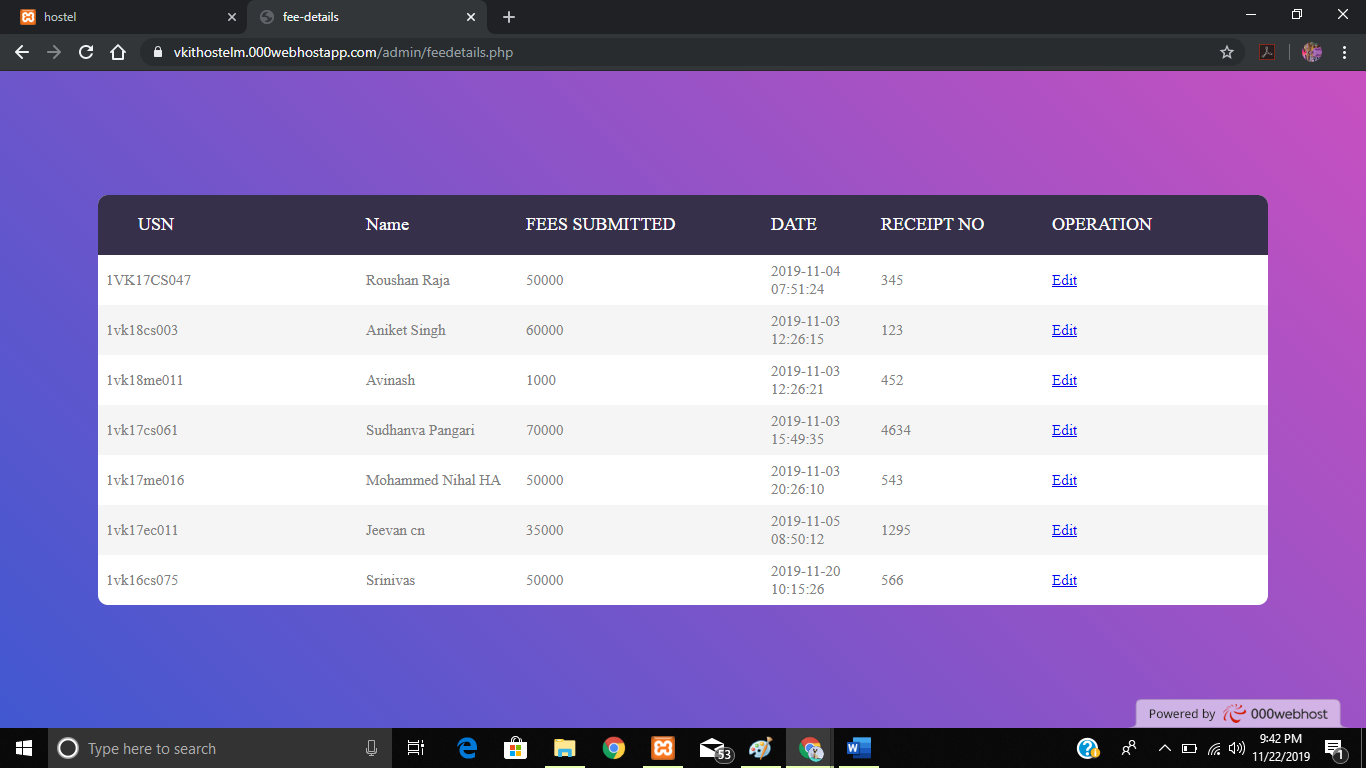
**3. STUDENTS VIEW**

****

**Fig:3 Student View**

Students table provide the information about the students.The admin can insert,delete,update,search the details of students.

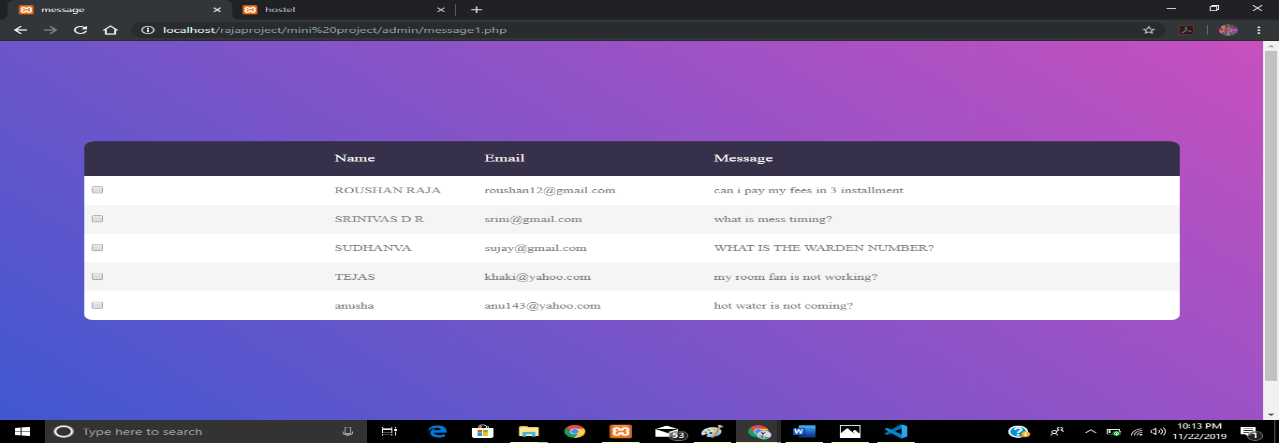
**4. FEES DETAILS VIEW**

****

**Fig:4 Fees Details View**

Fees details page provide the information about the fees submitted by students. Here, the admin can insert,delete and update the fees details.

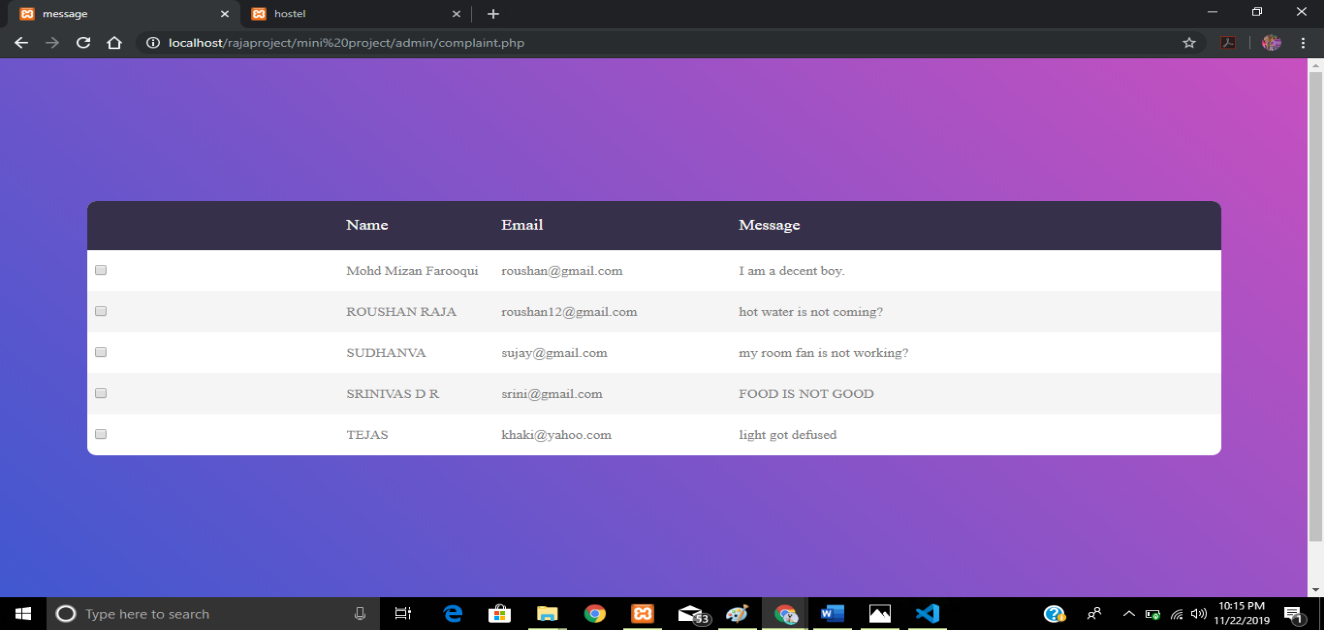
**5. MESSAGE VIEW**

****

**Fig:5 Message View**

Message table provide the information about the message sended .Here, admin can delete the information about the message.

**6. COMPLAINT VIEW**

****

**Fig:6 Complaint View**

This page provide the information about the complaint raised.

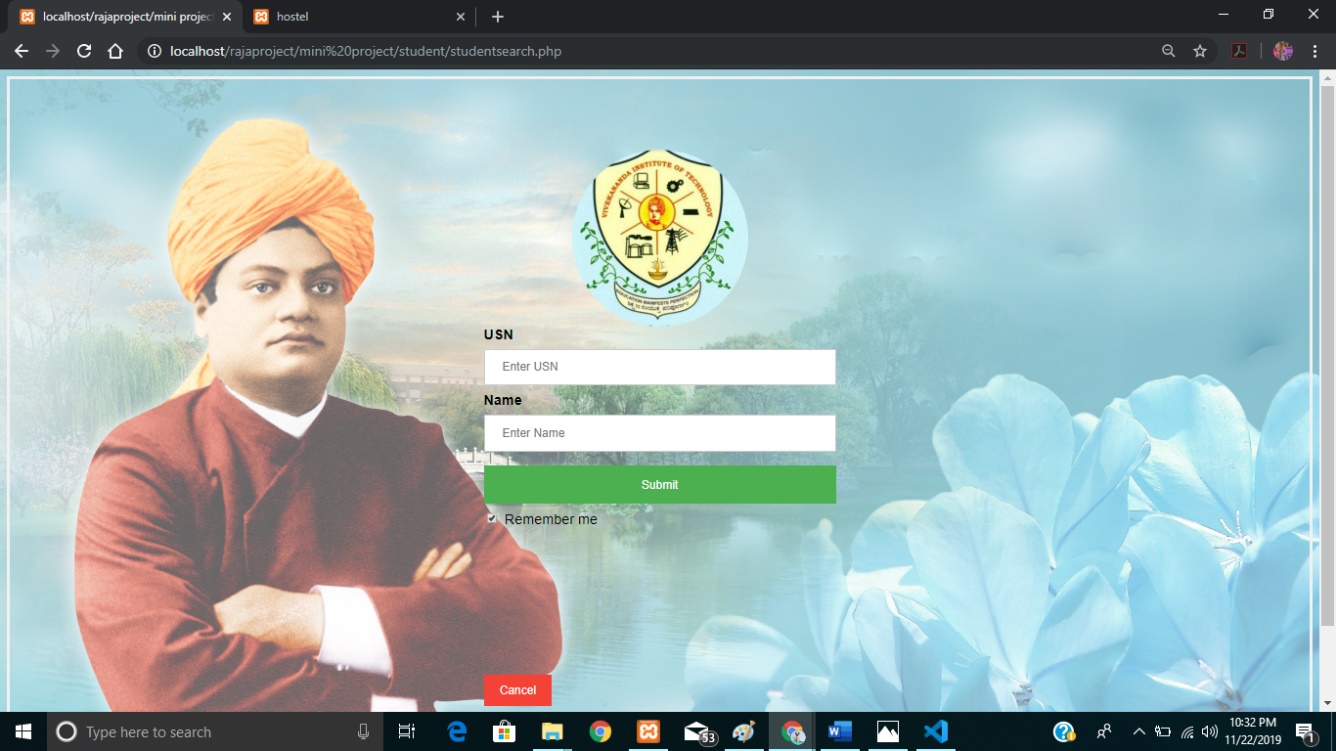
**7. STUDENT DETAILS VIEW**

****

**Fig:7 Student details**

This page provide the information about the detail of students. Here,admin can insert,delete and uodate the information about the students.

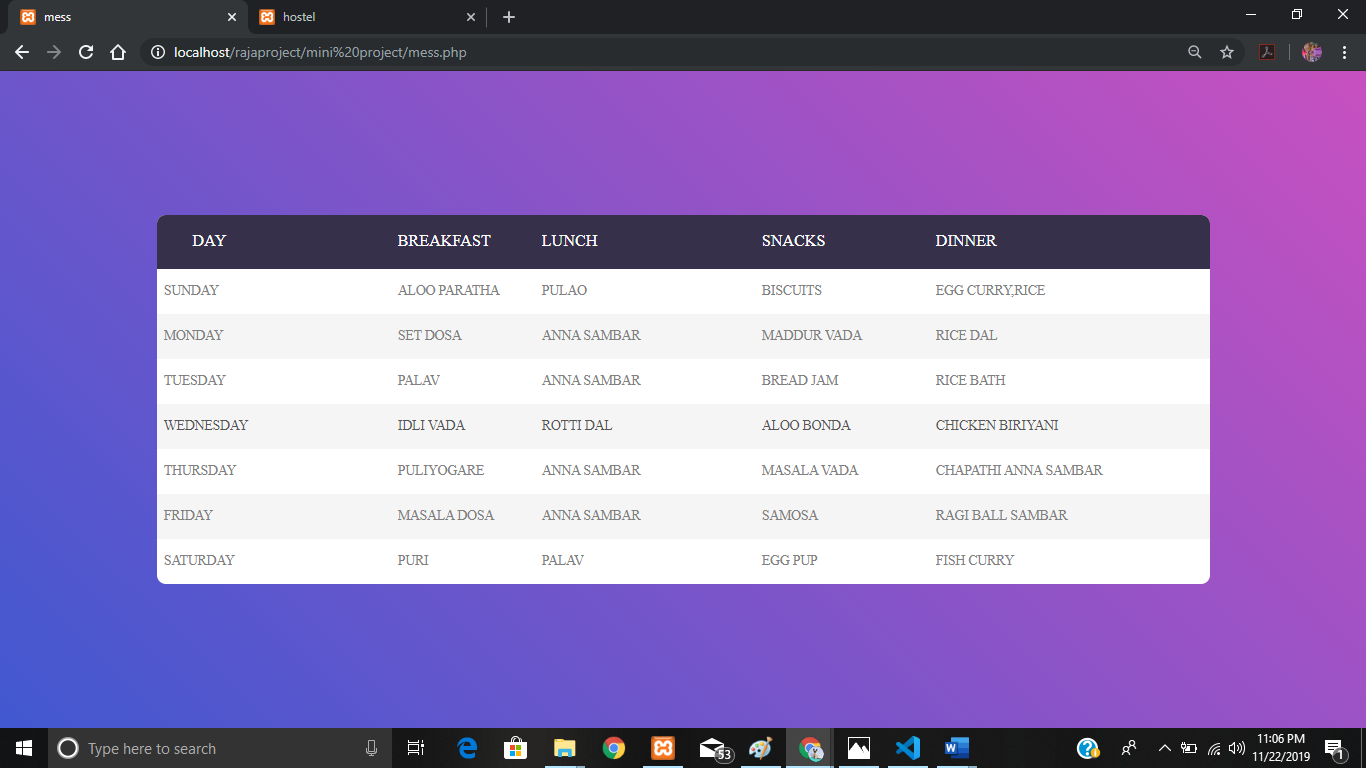
**8. STUDENT SEARCH**



**Fig:8 Student search**

This page provide the information about the students. Here, students can view the information about themselves.

**9. MESS ROUTINE VIEW**

****

**Fig:9 Mess Routine**

This page provide the information about the mess routine. Here,students can view the information about mess items.