

2. INTRODUCTION

2.1 PROBLEM DEFINITION

We see all the existing hostel management systems mainly focus on the student registration, room allocation and fee payment. Identification of the drawbacks of the existing system leads to the designing of computerized system that will be compatible to the existing system with the system which is more user friendly. We can improve the efficiency of the system, thus overcome the drawbacks of the existing system. We have designed this system of the hostel management especially for the college hostel, through this they cannot require so efficient person to handle. Hosteria is a hostel management system. This particular project deals with the problems on managing a hostel and solving problems related to a particular room. Here we are focusing on resolving complaints which are related to room like replacing lights, cleaning the room, supply of electrical devices. It is easy for the student as well as hostel manager to keep a track of the room damages.

2.2 OBJECTIVES OF PROJECT

This software product the hostel management to improve their services for all the students of the hostel. This also reduce the manual work of the persons in admin panel and the bundle of registers that were search when to find the information of a previous student, because through this system you can store the data of those students who had left the hostel. Students can also lodge and track complaints regarding the hostel rooms. The students of the hostel will be recognized from the ID number allocated at the room rental time. This system will also keep a track of the handyman or the workers who are resolving the problem. In the last, this system will improve the communication between the hostel manager and the student regarding the complaints.

- To provide a quick response with very accurate information as and when required
- To make the present manual system more interactive, speedy, user friendly and reduce the cost of maintenance.

3. SOFTWARE AND HARDWARE REQUIREMENTS

3.1 Software Requirements:

This software requirements refers to the server-side and user-side required software specifications.

Software tools used:

- HTML
- CSS
- Bootstrap
- Node.js
- WAMP/XAMPP

3.2 Hardware Requirements:

This hardware Constraints refers to the server-side and user-side required hardware specifications so that the module can be run with System.

- Operating System : Windows 10 Home
- Processor : intel i3 8th Gen
- RAM : Minimum 1GB
- Hard Disk : Minimum 32GB

4. TECHNOLOGIES

4.1 HTML

HTML stands for Hyper Text Markup Language. It is used to design web pages using markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages. Markup language is used to define the text document within tag which defines the structure of web pages. This language is used to annotate (make notes for the computer) text so that a machine can understand it and manipulate text accordingly. Most of markup languages are human readable. Language uses tags to define what manipulation must be done on the text.

Features of HTML

- It is easy to learn and easy to use.
- It is platform independent.
- Images, video and audio can be added to a web page.
- Hypertext can be added to text.
- It is a markup language

Advantages

- HTML is used to build the websites.
- It is supported by all browsers.
- It can be integrated with other languages like CSS, JavaScript etc.

4.2 CSS

Cascading Style Sheets, fondly referred to as CSS, is a simply designed language intended to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page. CSS is easy to learn and understood but it provides powerful control over the presentation of an HTML document.

WHY CSS?

- CSS saves time: You can write CSS once and reuse same sheet in multiple HTML pages.
- Easy Maintenance: To make a global change simply change the style, and all elements in all the webpages will be updated automatically.
- Search Engines: CSS is considered as clean coding technique, which means search engines won't have to struggle to "read" its content.
- Superior styles to HTML: CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes
- Offline Browsing: CSS can store web applications locally with the help of offline cache. Using of this we can view offline websites.

4.3 Node.js

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

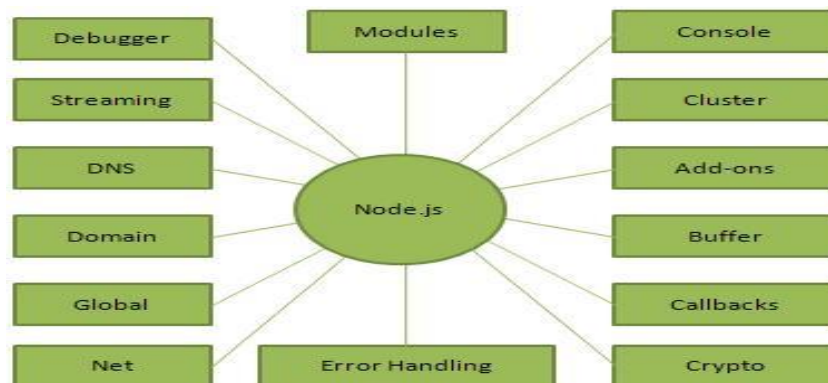


Figure.4.1.Parts of Node.js

Why Node.js

- I/O bound Application.
- Data Streaming Applications
- Data Intensive Real-time Applications (DIRT)
- JSON APIs based Applications
- Single Page Applications

4.4 BootStrap

Bootstrap is the most popular front end framework in the recent time. It is sleek, intuitive, and powerful mobile first front-end framework for faster and easier web development. It uses HTML, CSS and Javascript. This tutorial will teach you the basics of Bootstrap Framework using which you can create web projects with ease. The tutorial is divided into sections such as Bootstrap Basic Structure, Bootstrap CSS, Bootstrap Layout Components and Bootstrap Plugins. Each of these sections contain related topics with simple and useful examples

Advantages of BootStrap

- **Mobile first approach** – Bootstrap 3, framework consists of Mobile first styles throughout the entire library instead them of in separate files.
- **Browser Support** – It is supported by all popular browsers.
- **Easy to get started** – With just the knowledge of HTML and CSS anyone can get started with Bootstrap. Also the Bootstrap official site has a good documentation.
- **Responsive design** – Bootstrap's responsive CSS adjusts to Desktops, Tablets and Mobiles.

4.5 WAMP

WAMP is quite a well known term among Website Hosting Services industry. WAMP is acronym for the combination of Windows, Apache, MySQL and PHP/Python/Perl. In this combination the first three are constant ones and for the fourth one it varies among PHP, Python and Perl. There may be a few occasions

in which Python and Perl can be used together. The reason behind the popularity of WAMP is because it provides four important elements Operating System, Database Web Server and scripting application which are required for a web hosting server. When all these four elements are used as combined then such a usage is called as 'Server Stack'. In this (WAMP) server stack you use 'Microsoft' Windows as an operating system, 'Apache' as a Web Server, MySQL to work as a Database and you can choose one from PHP, Python and Perl to be used as scripting language.

5. IMPLEMENTATION

The project consists of few modules. They are as follows

- Student Module
- Handyman Module
- Admin Module
- Login Module

5.1 Student Module

The students residing at the hostel should Login or Sign-up to the web application. The student who already has an account, has to enter the credentials in to the website and the student who is new to the website needs to register by entering the details of his/her Name, Username and Password. If the user already exists, then an alert is shown that the user is already registered in the website. The student is given access to the website only if he/she enters the login details correctly.

Once the student has logged in then he can lodge a complaint and track the complaint. After the complaint is resolved then the student can give the rate to the handyman.

Student Module Code:

```
function changeStudentPassword(req,res,con){
    var student_id=req.query.student_id;
    var old_pass=req.query.old_Pass;
    var new_pass=req.query.new_Pass;
    var sql="update student_info set password=? where student_id=? and
password=?";
    con.query(sql,[new_pass,student_id,old_pass],function(err,result){
        if(err!=null){
            console.log(err);
```

```
        res.end("false");
    }
    else if(result.affectedRows==1)
        res.end("true");
    else
        res.end("old");
    })
};

function changeStudentPhone(req,res,con){
    var student_id=req.query.student_id;
    var phone=req.query.new_phone;
    var sql="update student_info set phone_no=? where student_id=?";
    con.query(sql,[phone,student_id],function(err,result){
        if(err!=null){
            console.log(err);
            res.end("false");
        }
        else if(result.affectedRows==1)
            res.end("true");
        //console.log(result);
        //console.log(req.query);
    })
};

module.exports={
    changeStudentPassword:function(req,res,con){
        changeStudentPassword(req,res,con);
    },
    changeStudentPhone:function(req,res,con){
        changeStudentPhone(req,res,con);
    }
}
```


5.2 Handyman Module

Handyman is a worker who resolves the problem like replacing a light, fan etc. The Handyman is given a username and password. After entering the correct login credentials the Home page is redirected to handyman dashboard. Dashboard consists of the complaints that are to be resolved.

Once the complaint is resolved then the OTP which is given to the student is taken. The complaint which is lodge also have the time slot in which he can resolve the problem.

HandyMan Module Code:

```
function getWork(req,res,con){
    var value=req.query;
    var id=value.handyman_id;
    var sql="select
complaint_id,c.student_id,subject,gender,date,type,catagory,cost,description,time
_slot,descriptionFull,email,phone_no,room_no,name from complaint_info as
c,student_info as s where handyman_id="+id+" and status<2 and
c.student_id=s.student_id order by complaint_id;"
    con.query(sql,function(err,result){
        if(err!=null){
            res.end(String(err));
            return;
        }
        res.end(JSON.stringify(result));
        return;
    });
}

function complaintSolved(req,res,con){
    var value=req.query;
    var complaint_id=value.complaint_id;
```

```
var otp=value.otp;
var handyman_id=value.handyman_id;
var sql="update complaint_info set status=2 where
complaint_id="+complaint_id+" and otp="+otp;
con.query(sql,function(err,result){
    if(err!=null){
        res.end("false");
        return;
    }
    if(result.affectedRows==0){
        res.end("Wrong OTP");
        return;
    }
    if(result.changedRows==0){
        res.end("Reload");
        return;
    }
    sql="update handyman_info set solved=solved+1,today=today+1
where handyman_id="+handyman_id;
    con.query(sql,function(err,result){
        if(err!=null){
            res.end("false");
            return;
        }
        else{
            res.end("true");
        }
    });
});
}
```

```
function complaintHistory(req,res,con){
    var handyman_id=req.query.handyman_id;
    var type=req.query.type;
    var status="(1,2,3)";
    if(type=="Solved"){
        status="(2,3)";
    }
    if(type=="Unsolved"){
        status="(1)";
    }
}
```

5.3 Admin Module

In this module, the admin or the hostel manager can add handyman and delete the handyman. The admin can keep track of the complaints lodge against each handyman and check the status of the work.

Admin Module Code:

```
var fs=require('fs');

function getAllEquipment(req,res,con){
    var sql="select distinct(equipment) from master_data";
    con.query(sql,function(err,result){
        if(err){
            console.log(err);
            res.end("false");
        }
        res.end(JSON.stringify(result));
    })
}

function getAllHistory(req,res,con){
```

```

var values=req.query;
var sInfoRadio=values.sInfoRadio;
var sInfoValue="";
var handyman_id="";
try{
    sInfoValue=values.sInfoValue.trim();
}
catch(err){
    sInfoValue=""
}
try{
    handyman_id=values.handyman_id.trim();
}
catch(err){
    handyman_id=""
}
//console.log(sInfoValue+" "+sInfoRadio);
//?subject=any&catagory=any&time_slot=any&type=any&status=any&rati
ng=any
var subject=values.subject.trim();
var catagory=values.catagory.trim();
var time_slot=values.time_slot.trim();
var type=values.type.trim();
var status=values.status.trim();
var rating=values.rating.trim();
var sql="select * from complaint_info c,handyman_info h,student_info s
where c.handyman_id=h.handyman_id and c.student_id=s.student_id ";
if(handyman_id!=""){
    sql+="and c.handyman_id="+handyman_id+" ";
}
if(sInfoValue!=""){

```

```
        if(sInfoRadio=="name"){
            sql+="and s.name='"+sInfoValue+"' ";
        }
        if(sInfoRadio=="roll"){
            sql+="and roll_no='"+sInfoValue+"' ";
        }
        if(sInfoRadio=="room"){
            var r=sInfoValue.split(" ")[1];
            var g=sInfoValue.split(" ")[0];
            //console.log(r+" "+g);
            sql+="and room_no="+r+" and gender='"+g+"' ";
        }
    }
    if(subject!="Any"){
        sql+="and subject='"+subject+"' ";
    }
    if(catagory!="Any"){
        sql+="and c.catagory='"+catagory+"' ";
    }
    if(time_slot!="Any"){
        sql+="and time_slot='"+time_slot+"' ";
    }
    if(type!="Any"){
        sql+="and type='"+type+"' ";
    }
    if(status!="Any"){
        if(status=="Lodged")
            status="(0)";
        if(status=="Unsolved")
            status="(0,1)";
        if(status=="Progress")
```

```
        status="(1)";
        if(status=="Solved")
            status="(2,3)";
        if(status=="Closed")
            status="(3)";
        sql+="and status in "+status+" ";
    }
    if(rating!="Any"){
        if(rating=="Poor")
            rating="(1,2)";
        if(rating=="Average")
            rating="(3)";
        if(rating=="Good")
            rating="(4,5)";
        sql+="and rating in "+rating+" ";
    }

    con.query({sql:sql,nestTables: true},function(err,result){
        if(err){
            console.log(err);
            res.end("false");
        }
        res.end(JSON.stringify(result));
    })
};

function assignWorkHelper(Data,slot,result,status,res,con){
    var id=null;
    var min=10;
    var sequence=-1;
    for(i in Data){
```

```
        if(min>Data[i][slot]){
            min=Data[i][slot];
            id=Data[i]['handyman_id'];
            sequence=i;
        }
    }
    if(min>=5)
        return -1;
    else{
        if(status==0)
            var sql="update complaint_info set handyman_id="+id+",
status=1 ,assignDate=NOW() where complaint_id="+result.complaint_id;
        else
            var sql="update complaint_info set handyman_id="+id+",
status=1 where complaint_id="+result.complaint_id;
        con.query(sql,function(err,result){
            if(err){
                console.log(err);
                res.end("false");
                return;
            }
            else{
                if(result.affectedRow==1){
                    return sequence;
                }
                return sequence;
            }
        });
    }
    return sequence;
}
```

```
function assignWork(electricianData,carpenterData,maidData,con,res){
    var sql="select complaint_id,time_slot,catagory,status from complaint_info
where status<2";
    con.query(sql,function(err,result){
        if(err){
            console.log(err);
            res.end("false");
            return;
        }
        else{
            //console.log(result);
            var slot="slot1";
            var e=0;
            var c=0;
            var m=0;
            for(i in result){
                if(result[i].time_slot=="10:00-12:00")
                    slot="slot1";
                if(result[i].time_slot=="13:00-15:00")
                    slot="slot2";
                if(result[i].time_slot=="16:00-18:00")
                    slot="slot3";
                if(result[i].catagory=="Electrician" ){
                    e=assignWorkHelper(electricianData,slot,result[i],result[i].status,res,con);
                    //console.log("e="+e)
                    if(e>=0){
                        electricianData[e][slot]++;
                        electricianData[e]['issued']++;
                    }
                }
            }
        }
    })
}
```



```

    }
    else if(result[i].category=="Carpenter" ){
c=assignWorkHelper(carpenterData,slot,result[i],result[i].status,res,con);
        if(c>=0){
            carpenterData[c][slot]++;
            carpenterData[c]['issued']++;
        }
    }
    else if(result[i].category=="Maid"){
m=assignWorkHelper(maidData,slot,result[i],result[i].status,res,con);
        if(m>=0){
            maidData[m][slot]++;
            maidData[m]['issued']++;
        }
    }
}
for(i in electricianData){
    sql="update handyman_info set
slot1="+electricianData[i]['slot1']+",slot2="+electricianData[i]['slot2']+",slot3="
+electricianData[i]['slot3']+",issued="+electricianData[i]['issued']+" where
handyman_id="+electricianData[i]['handyman_id'];
    con.query(sql,function(err,result){
        if(err!=null){
            console.log(err);
            res.end("false");
            return;
        }
    });
}
for(i in carpenterData){
    //console.log(electricianData[i]['handyman_id']);

```

```

        //console.log(electricianData[i]['issued']);
        sql="update handyman_info set
slot1="+carpenterData[i]['slot1']+",slot2="+carpenterData[i]['slot2']+",slot3="+c
arpenterData[i]['slot3']+",issued="+carpenterData[i]['issued']+" where
handyman_id="+carpenterData[i]['handyman_id'];

        //sql="update handyman_info set
slot1="+electricianData[i]['slot1']+",slot2="+electricianData[i]['slot2']+",slot3="
+electricianData[i]['slot3']+",issued=issued"+electricianData[i]['issued']+"wher
e handyman_id="+electricianData[i]['handyman_id'];

        con.query(sql,function(err,result){
            if(err!=null){
                console.log(err);
                res.end("false");
                return;
            }
        });
    }
    for(i in maidData){
        sql='update handyman_info set
slot1='+maidData[i]['slot']+"slot2="+maidData[i]['slot2']+",slot3="+maidData[i]
['slot3']+",issued="+maidData[i]['issued']+" where
handyman_id="+maidData[i]['handyman_id'];

        con.query(sql, function(err,result){
            if(err!=null){
                console.log(err);
                res.end("false");
                return;
            }
        });
    }
}

```

```
});
}
```

5.4 Login Module

```
var express = require('express');
var session = require('express-session');
var path=require('path');
var app=express();
var mysql = require('mysql');
var fs = require('fs');
var bodyParser=require('body-parser');
var adminJS = require('./nodeAdmin.js');
var handyManJS=require('./nodeHandyMan.js');
var studentJS=require('./nodeStudent.js');
app.use(bodyParser());
var MySQL_Connection_Details;
var con;
fs.readFile('MySQL_Connection_Details','utf8',function(err, data) {
    MySQL_Connection_Details=JSON.parse('{' +data+'}');
    con = mysql.createConnection(
        MySQL_Connection_Details
    );
    con.connect(function(err) {
        if (err){
            console.log("Error!")
            throw err;
        }
        console.log("Connected!");
    });
});
app.use(express.static(__dirname + '/dist'));
app.use(express.static(__dirname + ''));
app.use(express.static(__dirname + '/fa'));
app.get('/',function(req,resp){

    resp.sendFile(path.resolve('index.html'));
    return;
    //console.log(req);
});
```

```
        //console.log(resp);
    });

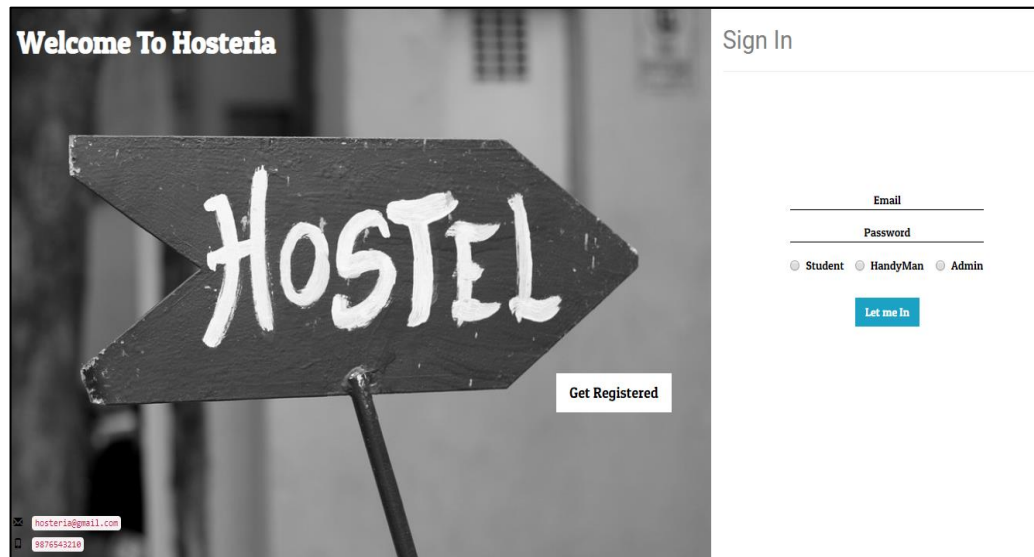
    app.get('/w3cc.html',function(req,resp){
        resp.sendFile('studen.html',{root: path.join(__dirname, ")});
    })

    app.get('/admin.html',function(req,resp){
        resp.sendFile('admin.html',{root: path.join(__dirname, ")});
    })
    // app.get('/main.html',function(req,resp){
    //     resp.sendFile('main.html',{root: path.join(__dirname, ")});
    // })

    app.get('/getData',function(req,resp){
        var sess=req.query;
        //console.log(sess);
        if(sess.email && sess.type=='S')
        {

            var sql = "SELECT * FROM student_info WHERE
email='"+sess.email+"'";
            con.query(sql, function (err, result) {
                if (err){
                    console.log(err);
                    resp.end("false");
                    return;
                }
                if(result.length==0){
                    resp.end("false");
                }
            })
        }
    })
}
```

6. OBSERVATIONS



The screenshot shows the Hosteria homepage. On the left, there is a large black arrow-shaped sign with the word "HOSTEL" written in white. Below the sign, there is a "Get Registered" button. In the bottom left corner, there is contact information: an email icon followed by "hosteria@gmail.com" and a phone icon followed by "9876543210". On the right side, there is a "Sign In" section. It contains two input fields for "Email" and "Password". Below these fields are three radio buttons labeled "Student", "HandyMan", and "Admin". At the bottom of the sign-in section is a blue "Let me In" button.

Figure.6.1 Hosteria Homepage



The screenshot shows the Hosteria signup page. On the left, there is a large black arrow-shaped sign with the word "HOSTEL" written in white. Below the sign, there is an "Already Registered" button. In the bottom left corner, there is contact information: an email icon followed by "hosteria@gmail.com" and a phone icon followed by "9876543210". On the right side, there is a "Sign Up" section. It contains six input fields for "Full Name", "Roll no", "Room no.", "Email", "Phone no.", and "Password". Below these fields are two radio buttons labeled "Male" and "Female". At the bottom of the sign-up section is a blue "Register Me" button.

Figure.6.2 Signup page

Dashboard

- Lodge
- History
- Info**
- Log Out

Welcome **cherry** [Change Info](#)

Student
 F 105
 1242
 cherry@gmail.com
 9898765432

0 Registered **0** Completed **0** Pending

Figure.6.3 Student Dashboard

Dashboard

- Lodge**
- History
- Info
- Log Out

Lodge a complaint

hostel ▼
 Table ▼
 Carpenter
 13:00-15:00 ▼
 Broken ▼
 Tell us more about the problem

Lodge

Figure.6.4 Student Complaint Form

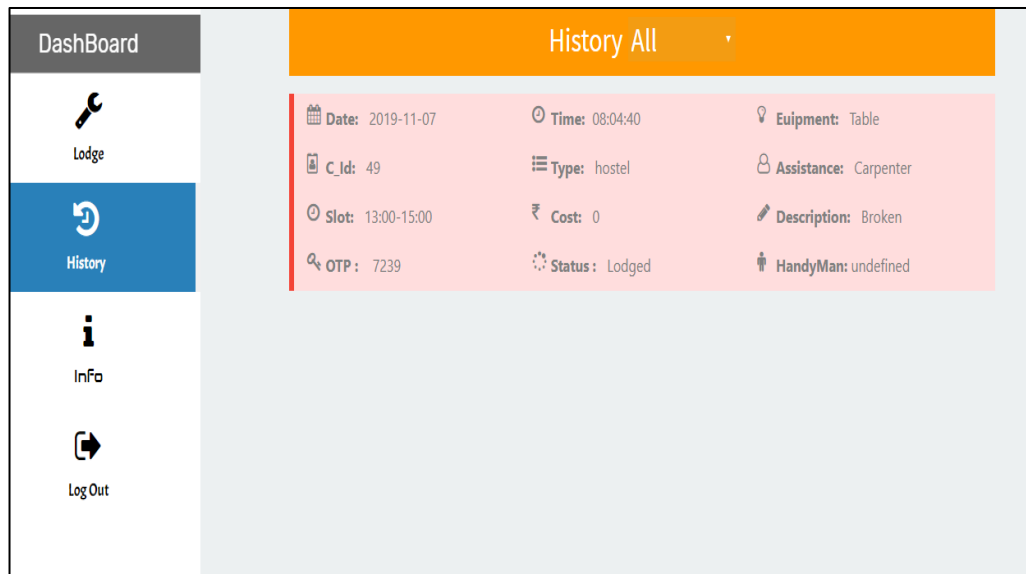


Figure.6.5 Student Complaint History

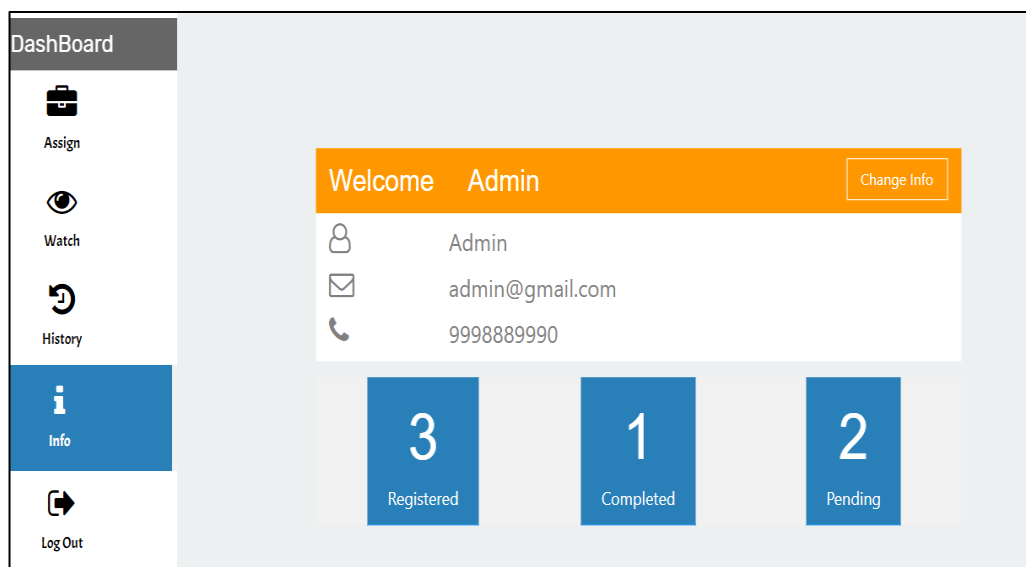


Figure.6.6 Admin Dashboard

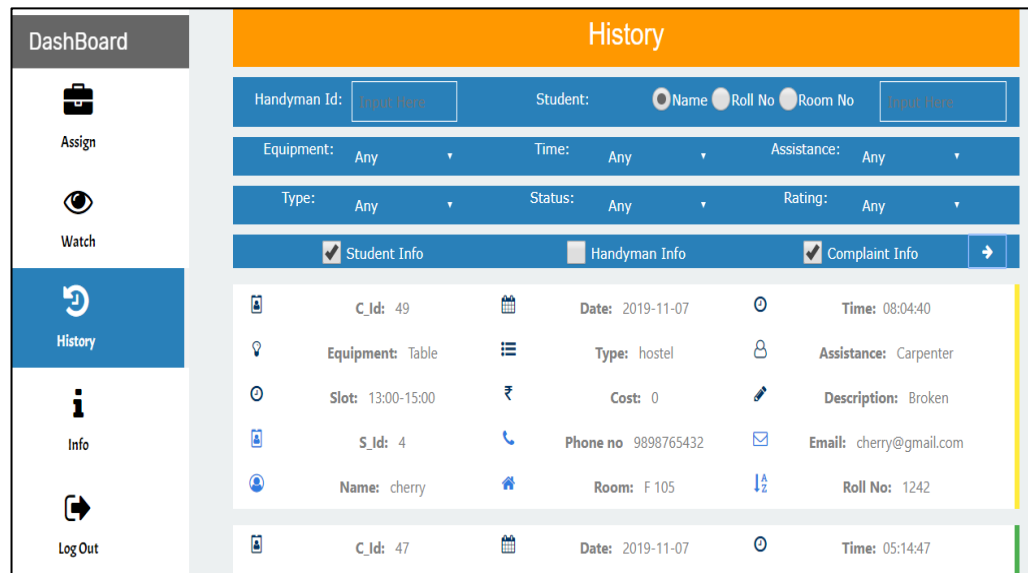


Figure.6.7 Admin Complaints History

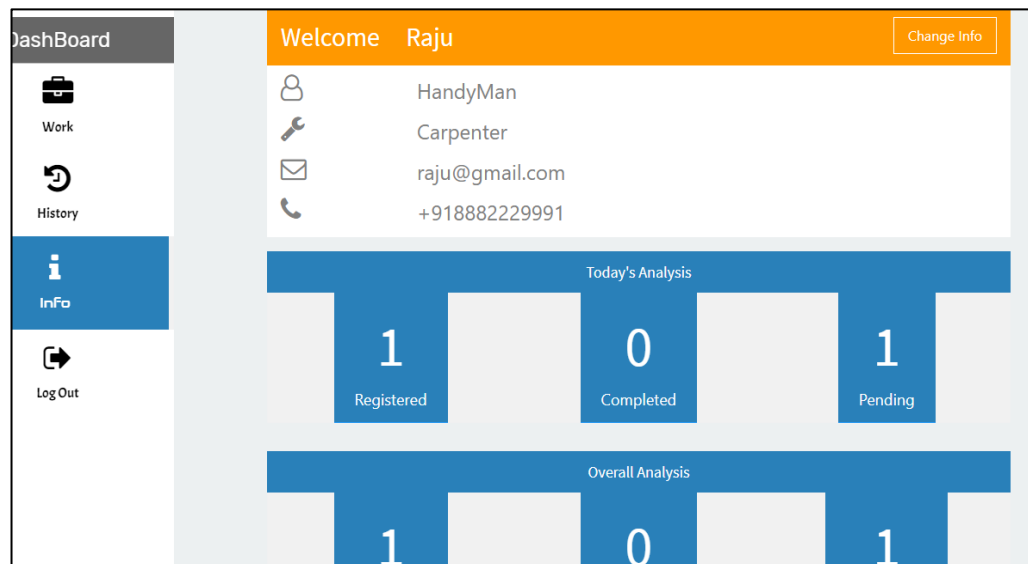


Figure.6.8 Handyman Dashboard

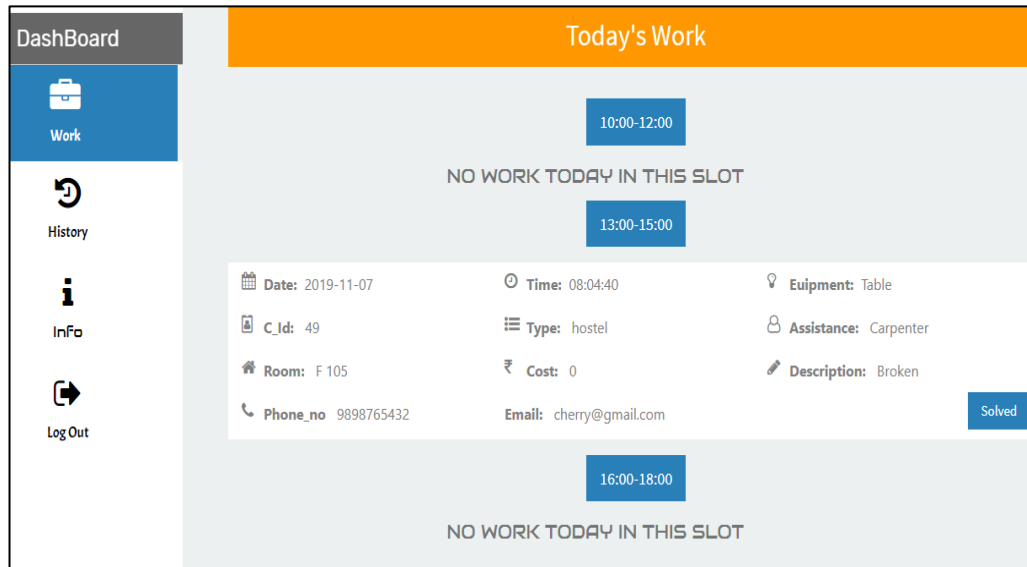


Figure.6.9 Work page

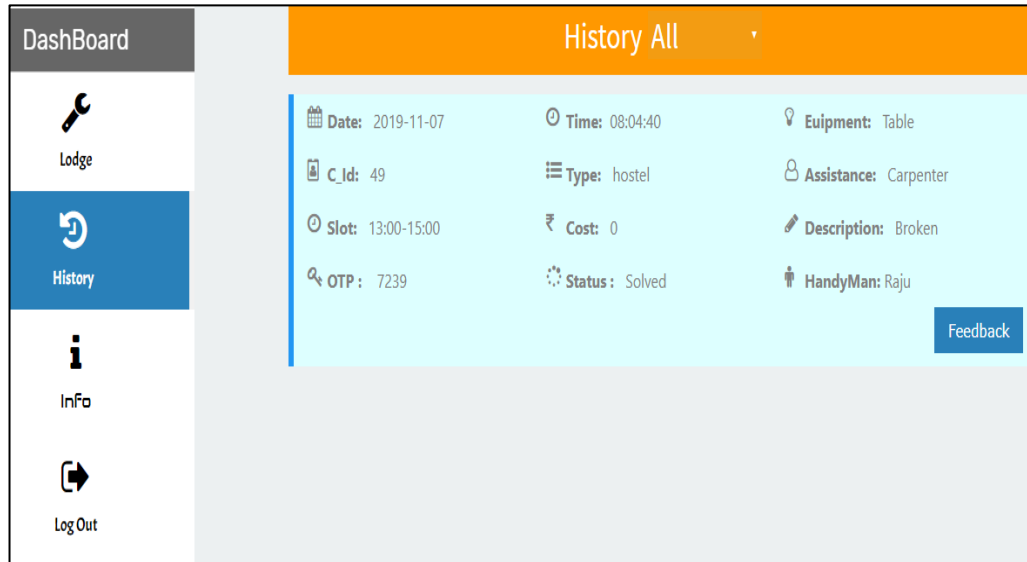


Figure.6.10 History of Complaints

The image shows a web application interface with a sidebar on the left and a main content area. The sidebar contains four menu items: 'Lodge' with a wrench icon, 'History' with a circular arrow icon, 'Info' with an 'i' icon, and an arrow icon at the bottom. The main content area displays a 'Service Feedback' modal form. The form has a blue header bar with the title 'Service Feedback' and a close button 'x'. Below the header, there are five star icons for rating. A text input field is labeled 'Write a review'. At the bottom of the form is a blue bar with a 'Submit' button. A small 'Feedback' button is also visible on the right side of the main content area.

Figure.6.11 Feedback Form

7. CONCLUSION

Hosteria is a hostel management system which mainly focuses on registering the complaints and tracking them. The complaints are related to the hostel room only. Students can also avail some of the services like heater, kettle, etc. The handyman will have a separate login where in which he can view the pending complaints and the time slots in which he can go and resolve the complaint. Once the complaint is resolved then he must enter the OTP which is given to the student. The hostel manager or the admin of the website has given the special privileges regarding the maintenance of the data. This website helps the hostel manager and the students to easily communicate regarding the issues in the hostel room. It also helps the manager to charge the handyman according to the assigned work. Through this application, the complaints can be easily solved.

In future, this application will also have various features like automatic room allocation for the newly registered students and calculation of the mess bills and payments.

REFERENCES

1. https://www.w3schools.com/html/html_forms.asp
2. https://www.tutorialspoint.com/nodejs/nodejs_npm.htm
3. https://www.w3schools.com/js/js_random.asp
4. <https://www.yourhtmlsource.com/stylesheets/>
5. <https://fonts.google.com/?category=Sans+Serif,Monospace>