VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI-590018

A DBMS Mini Project Report

on

" Rental Solutions Website "

Submitted in partial fulfillment of the requirements for the V semester and award of the degree of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi

Submitted by:

Likhith C	*1RN20CS074*
Maharsh G R	*1RN20CS079*

Under the Guidance of:

Mrs. Mamatha S Jajur
Assistant Professor
Dept. of CSE

ESTD : 2001
An Institute with a Difference

Department of Computer Science and Engineering
(NBA Accredited for academic years 2018-19, 2019-20, 2020-22)
RNS Institute of Technology
Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560 098
2022-2023

RNS Institute of Technology

Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

(NBA Accredited for academic years 2022-25)

CERTIFICATE

Certified that the mini project work entitled **“ Rental Solutions Website”** has been successfully carried out by **”Likhith C”** bearing USN **”1RN20CS074”** and **“Maharsh G R”** bearing USN **”1RN20CS079”**, bonafide students of **”RNS Institute of Technology”** in partial fulfillment of the requirements for the 5th semester of **”Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University”**, Belagavi, during academic year 2022-2023. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The project report has been approved as it satisfies the DBMS laboratory requirements of 5th semester BE, CSE.

Signature of the Guide	Signature of the HoD	Signature of the Principal
Mrs. Mamatha S Jajur	**Dr. Kiran P**	**Dr. M K Venkatesha**
Assistant Professor	Professor, HOD	Principal
Dept. of CSE	Dept. of CSE	

External Viva:

Name of the Examiners	Signature with Date
1.	
2.	

Acknowledgement

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped us in carrying out this project work. We would like to take this opportunity to thank them all.
We are grateful to Management and **Dr. M K Venkatesha**, Principal, RNSIT, Bangalore, for his support towards completing this mini project.
We would like to thank **Dr. Kiran P**, Professor & Head, Department of Computer Science & Engineering, RNSIT, Bangalore, for his valuable suggestions and expert advice.
We deeply express our sincere gratitude to our guide **Mrs. Mamatha S Jajur** , Assistant Professor, Department of CSE, RNSIT, Bangalore, for her able guidance, regular source of encouragement and assistance throughout this project.
We would like to thank all the teaching and non-teaching staff of Department of Computer Science & Engineering, RNSIT, Bengaluru-98 for their constant support and encouragement.

Abstract

A rental solutions website is an online platform where an user can search for PG or flat in a particular area.This website eases the difficulty of the user by providing the details about the PG or flat and it also provides the information like number of rooms,facilities available in PG,owner details.

The purpose of this project is to design and develop a rental management system for PGs (paying guest accommodations) and flats. The system will provide a convenient platform for landlords to list their properties for rent and for potential renters to search and find properties that meet their specific needs and preferences. The system will also facilitate the rental process by allowing for secure online payments and easy communication between landlords and renters.

The system will be designed with a user-friendly interface that will allow landlords to easily list their properties and potential renters to easily search and find properties based on various criteria such as location, price, size, and amenities. Additionally, the system will include features such as real-time availability updates, contract management, and maintenance request tracking.

The project will be implemented using a combination of technologies such as HTML, CSS, JavaScript, Node.Js and MySQL. The system will be tested and evaluated for usability, functionality, and performance.

Overall, this project aims to make the process of searching for and renting PGs and flats as seamless and efficient as possible, by providing a centralized platform for all the rental management needs..

Contents

List of Figures

Chapter 1

Introduction

1.1 Database Technologies

The essential feature of database technology is that it provides an internal representation (model) of the external world of interest. Examples are, the representation of a particular date/time/flight/aircraft in an airline reservation or of the item code/item description/quantity on hand/reorder level/reorder quantity in a stock control system.

The technology involved is concerned primarily with maintaining the internal representation consistent with external reality; this involves the results of extensive RD over the past 30 years in areas such as user requirements analysis, data modelling, process modelling, data integrity, concurrency, transactions, file organisation, indexing, rollback and recovery, persistent programming, object-orientation, logic programming, deductive database systems, active database systems... and in all these (and other) areas there remains much more to be done. The essential point is that database technology is a CORE TECHNOLOGY which has links to:

- Information management / processing
- Data analysis / statistics
- Data visualization / presentation
- Multimedia and hypermedia
- Office and document systems
- Business processes, workflow, CSCW (computer-supported cooperative work)

Relational DBMS is the modern base technology for many business applications. It offers flexibility and easy-to-use tools at the expense of ultimate performance. More recently relational systems have started extending their facilities in directions like information retrieval, objectorientation and deductive/active systems which lead to the so-called 'Extended Relational Systems'.

Information Retrieval Systems began with handling library catalogues and then extended to full free-text by utilizing inverted index technology with a lexicon or thesaurus. Modern systems utilize some KBS (knowledge-based systems) techniques to improve the retrieval. Object-Oriented DBMS started for engineering applications in which objects are complex, have versions and need to be treated as a complete entity. OODBMSs share many of the OOPL features such as identity, inheritance, late binding, overloading and overriding. OODBMSs have found favours in engineering and office systems but haven't been successful yet in traditional application areas.

Deductive / Active DBMS has evolved over the last 20 years and combines logic programming technology with database technology. This allows the database itself to react to the external events and also to maintain its integrity dynamically with respect to the real world.

1.2 Characteristics of Database Approach

Traditional form included organising the data in file format. DBMS was a new concept then, and all kinds of research was done to make it overcome the deficiencies in traditional style of data management. A modern DBMS has the following characteristics

- Real-world entity A modern DBMS is more realistic and uses real-world entities to design its architecture. It uses behaviour and attribute too. For example, a school database may use students as an entity and their age as an attribute.
- Relation-based tables DBMS allows entities and relations to form tables. A user can understand the architecture of a database by just looking at the table names.
- Isolation of data and application A database system is entirely different than its data. A database is an active entity, whereas data is said to be passive, on which the database works and organizes. DBMS also stores metadata, which is data about data, to ease its own process.
- Less redundancy DBMS follows the rules of normalization, which splits a relation when any of its attributes has redundancy in its values. Normalization is a mathematically rich and scientific process that will reduces the data redundancy.
- Consistency Consistency is a state where every relation in a database remains consistent. There exists methods and techniques, that can detect an attempt of leaving database in an inconsistent

state. DBMS can provide greater consistency as compared to earlier forms of data storing applications like file-processing systems.

- Query Language DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and the filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used.
- ACID Properties DBMS follows the concepts of Atomicity, Consistency, Isolation, and Durability (normally shortened as ACID). These concepts are applied on transactions, which manipulate data in a database. ACID properties help the database to stay healthy in multi-transactional environments and also in case of failure.
- Multiuser and Concurrent Access DBMS supports multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when users attempt to handle the same data item, but users are always unaware of them.
- Multiple views DBMS offers multiple views for different users. A user in the Sales department will have a different view of the database from the person working in the Production department. This feature enables the users to have a concentrate view of the database according to their requirements.
- Security Features like multiple views offer security to certain extent when users are unable to access the data of other users and departments. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with different features. For example, a user in the Sales department cannot see the data that belongs to the Purchase department. It can also be helpful in deciding how much data of the Sales department should be displayed to the user. Since a DBMS is not saved on the disk as traditional file systems, it is very hard for miscreants to break the code.

1.3 Applications of DBMS

Applications of Database Management Systems :

- Telecom: There is a database to keeps track of the information regarding the calls made, network usage, customer details etc. Without the database system it is hard to maintain such huge amounts of data which gets updated every millisecond.

- Industry: Whether it is a manufacturing unit, a warehouse or a distribution centre, each one needs a database to keep the records of the ins and outs. For example, a distribution centre should keep a track of the product units that were supplied to the centre as well as the products that got delivered from the distribution centre on each day; this is where DBMS comes into picture.
- Banking System: For storing information regarding a customer, keeping a track of his/her day to day credit and debit transactions, generating bank statements etc is done with through Database management systems.
- Education sector: Database systems are frequently used in schools and colleges to store and retrieve the data regarding the student , staff details, course details, exam details, payroll data, attendance details, fees details etc. There is lots of inter-related data that needs to be stored and retrieved in an efficient manner.
- Online shopping: You must be aware of the online shopping websites such as Amazon, Flip kart etc. These sites store the product information, your addresses and preferences, credit details and provide you the relevant list of products based on your query. All this involves a Database management system.

1.4 Problem Description/Statement

The problem that the rented rooms searching website aims to solve is providing a convenient platform for users to search for and find rental rooms that meet their specific needs and preferences. The website should allow users to easily search for rooms based on various criteria such as location, price, size, and amenities.

Additionally, the website should provide a user-friendly interface for renters to list their rooms for rent and for potential renters to view the listings and contact the renters. The website should also allow for secure payments and easy communication between renters and potential renters. Overall, the goal of the website is to make the process of searching for and renting a room as seamless and efficient as possible.

Chapter 2

Resource Requirements

2.1 Hardware Analysis

The Hardware requirements are very minimal and the program can be run on most of the machines.

Processor : i5 processor

Processor Speed : 1.2 GHz

RAM : 1 GB

Storage Space : 40 GB

Monitor Resolution : 1024*768 or 1336*768 or 1280*1024

2.2 Software Requirements

System specifications and software required are,

Operating System used: Windows 10

Technologies used: HTML, CSS, NodeJS, JavaScript

Server: MySQL, XAMPP

IDE used: Visual Studio Code

Browser that supports HTML

2.3 End User Requirements

The technical requirements for the project are mentioned below:

1.Node.js 2.MySQL 3.HTML 4.CSS 5.JavaScript

2.3.1 Node.js

Node.js is a JavaScript runtime that runs on the V8 JavaScript engine, which was developed by Google for use in their Chrome web browser. It allows developers to run JavaScript code on the server-side, which can be used to build a wide variety of applications, such as web servers, real-time applications, and command-line tools.

One of the key advantages of Node.js is its ability to handle a large number of simultaneous connections with high throughput. This is because it uses an event-driven, non-blocking I/O model, which makes it very efficient at handling many concurrent connections.

Node.js also includes a built-in package manager called npm (Node Package Manager), which allows developers to easily share and reuse code. There are over 1 million packages available on npm, which can be used to add functionality to Node.js applications.

Node.js is commonly used for building web servers and APIs, real-time applications such as chat and gaming applications, and command-line tools. It can also be used for building desktop applications using frameworks such as Electron, and for building IoT devices using frameworks such as Johnny-Five.

Another advantage of Node.js is its wide usage on different platforms, It's built on Chrome's JavaScript runtime, which makes it easy to run on different operating systems, including Windows, Mac, and Linux.

In summary, Node.js is a powerful, efficient, and versatile JavaScript runtime that can be used to build a wide variety of applications, from web servers and APIs to real-time applications and command-line tools. It's built on the V8 JavaScript engine and comes with a built-in package manager (npm), which makes it easy to share and reuse code. It's also cross-platform and widely adopted by many companies and developers.

2.3.2 MySQL

MySQL is a popular open-source relational database management system (RDBMS) that uses Structured Query Language (SQL) to manage and manipulate data stored in databases. It is widely used in web applications, data warehousing, and other applications that require a robust and reliable database management system. MySQL is known for its high performance, reliability, and ease of use, and it can be run on a variety of operating systems, including Windows, Linux, and macOS. It also supports various programming languages, including PHP and Java. The primary factor differentiating relational databases from other digital storage lies in how data is organized at a high level. Databases like MySQL contain records in multiple, separate, and highly codified tables, as opposed to a single all-encompassing repository, or collections of semi- or unstructured documents.

This allows RDBMSs to better optimize actions like data retrieval, updating information, or more complex actions like aggregations. A logical model is defined over all of the contents of the database, describing for example the values allowed in individual columns, characteristics of tables and views, or how indices from two tables are related.

Relational models have remained popular for several reasons. They empower users with intuitive, declarative programming languages — essentially telling the database what result is wanted in language akin to, or at least comprehensible as, written english, instead of meticulously coding up each step of the procedure leading to that result. This moves a lot of the work into the RDBMS and SQL engines, better enforcing logical rules and saving valuable resources and manpower.

2.3.3 HTML and CSS

HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets) are two essential technologies used for creating and designing websites. HTML is used to structure and organize the content of a website, while CSS is used to control the layout and appearance of that content.

HTML is a markup language that uses tags to define the structure of a web page. These tags include headings, paragraphs, lists, links, images, and more. By using these tags, developers can create a hierarchical structure for their content, making it easy for both humans and machines to understand the organization of the page.

CSS is used to apply styles to the HTML elements on a web page. Styles include things like color, font, size, spacing, and positioning. With CSS, developers can create a consistent look and feel across all pages of a website, and can also easily make changes to the layout and design without having to alter the HTML code.

CSS also allows for the separation of presentation and content, which makes it easier to maintain and update a website over time. It also enables responsive design, which allows the layout and design of a website to adapt to the size of the device or screen on which it is viewed. This is becoming increasingly important as more and more people access the internet on mobile devices.

CSS also has some advanced features like CSS Grid and Flexbox which enables the developers to create complex and responsive layouts easily. With the help of these features, developers can create grid-based layouts and align elements on a web page in a flexible and responsive way.

In conclusion, HTML and CSS are essential building blocks of the modern web. HTML provides the structure and organization of a web page, while CSS provides the layout and design. Together, they make it possible to create beautiful and functional websites that are easy to navigate and maintain.

2.3.4 JavaScript

JavaScript often abbreviated as JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. As of 2022, 98 of websites use JavaScript on the client side for webpage behavior, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users devices.

JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard.It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

Chapter 3

Database Design

3.1 Major Entities, Attributes and Relationships

- Owner

  Owner_ID - Primary Key

  Name

  Username

  Password

  Contact

  Email_ID

  Address

- PG

  PG_ID - Primary Key

  Name

  landmark

  No-of-Rooms

  Rooms_Filled

  Rooms_Vac

  Address

  Facilities

Foods

Contact

Owner_ID

Owner_ID - foreign key references to Owner (Owner_ID)

- Flat

  Name

  Flat_ID Primary Key

  Type

  Address

  Landmark

  Contact

  Owner_ID

  Owner_ID - foreign key references to Owner (Owner_ID)

3.2 ER Diagram

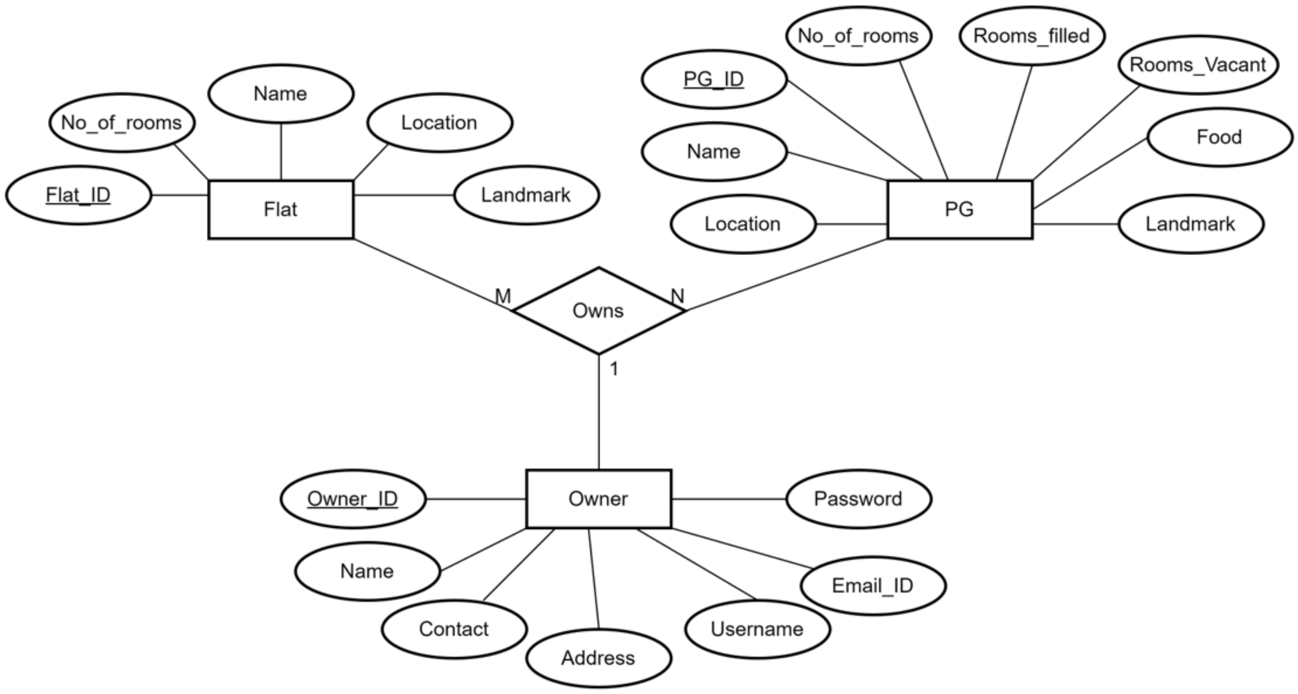


Figure 3.1: ER Diagram

3.3 Schema Diagram

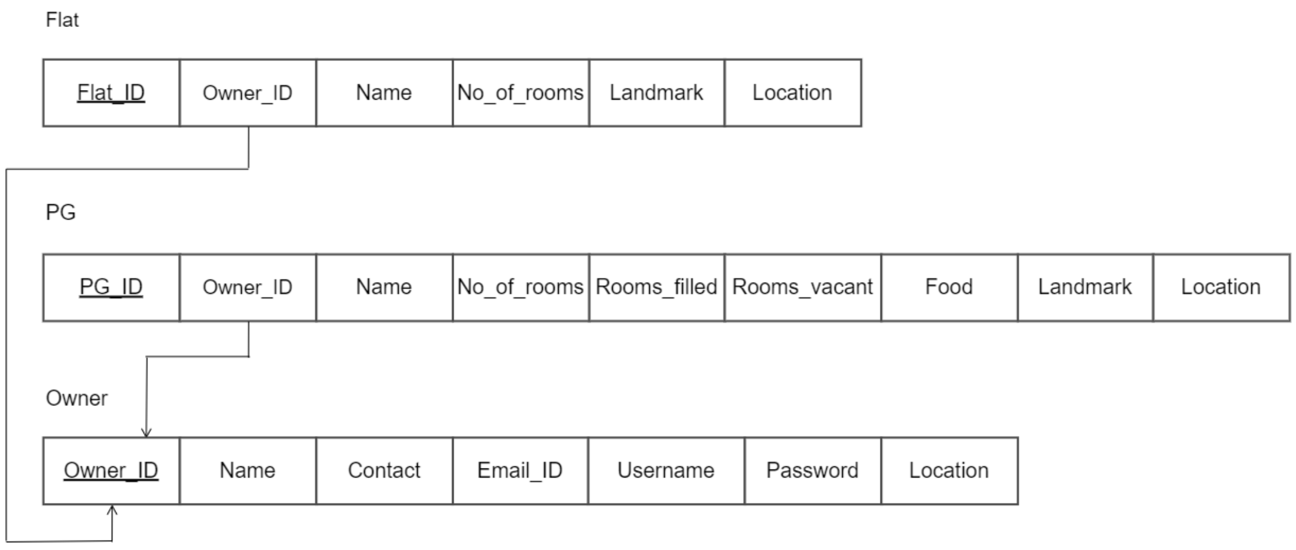


Figure 3.2: Schema Diagram

Chapter 4

Implementation

4.1 Database Connectivity

```
const mysql = require('mysql');
const con = mysql.createConnection({
    host:"localhost",
    user:"root",
    password:"",
    database:"db",
    multipleStatements: true
});
con.connect((err)=>{
    if(err){
        console.log("error in connection");
    }
    else
    {
        console.log("connection");
    }
})
module.exports = con;
```

4.2 Code Snippets of Major Functionalities

4.2.1 Owner-Signup and Login Page

```
app.post('/ownersignup',(req,res)=>{
    console.log(req.body);
    const name = req.body.name;
    const username = req.body.username;
    const email_id = req.body.emailid;
    const contact = req.body.ph;
    const pwd = req.body.pass;
    const address = req.body.address;
   var sql = "INSERT INTO ownersignup(name,username,email_ID,contact,pwd,address)
   VALUES('"+name+"','"+username+"','"+email_id+"','"+contact+"','"+pwd+"',
   '"+address+"')";
        con.query(sql,function(err,result){
            if(err) {console.log("error");}
            alert("registration successfull");
            res.redirect('/ownerlogin');
        })
    })
    var uname;
    var pass;
    app.post('/ownerlogin',(req,res)=>{
         uname = req.body.username;
         pass = req.body.password;
        if(uname && pass){
            con.query('select * from ownersignup where username = ? and
            pwd = ?',[uname,pass],(err,result,fields)=>{
                if(result.length>0){
                    res.redirect('/ownerprofile');
               }
                else{
```

```
                alert("Incorrect Username and Password");
            }
        });
    }
    else{
        alert("Please enter Username and Password");
    }
})
```

4.2.2 Tenant-Signup and Login Page

```
app.post('/tenantsignup',(req,res)=>{
    console.log(req.body);
    const name = req.body.name;
    const username = req.body.username;
    const email_ID = req.body.emailid;
    const contact = req.body.ph;
    const pwd = req.body.pass;
   var sql = "INSERT INTO signup(name,username,email_ID,contact,pwd)
   VALUES('"+name+"','"+username+"','"+email_ID+"','"+contact+"','"+pwd+"')";
        con.query(sql,function(err,result){
            if(err) {console.log("ayyo");}
            alert("registration successfull");
            res.redirect('/tenantlogin');
        })
    })

app.post('/tenantlogin',(req,res)=>{
    var username = req.body.username;
    var pwd = req.body.password;
    if(username && pwd){
        con.query('select * from signup where username = ? and pwd = ?',
        [username,pwd],(err,result,fields)=>{
            if(result.length>0){
                res.redirect('/listofrooms');

            }
            else{
                alert("Incorrect Username and Password");
            }
        });
    }
    else{
        alert("Please enter Username and Password");
    }
})
```

4.2.3 Listing PGs and Flats

```
app.get("/listofrooms",function(req,res){
    var sql = "select Name,landmark,Address from pg";
    con.query(sql,function(err,result){
        if(err){console.log(err)};
        res.render("./listofrooms",{Room:result});
    })
})

app.get("/flatlist",function(req,res){
    var sql = "select Name,landmark,address from flat";
    con.query(sql,function(err,result){
        if(err){console.log(err)};
        res.render("./flatlist",{Room:result});
    })
})
```

4.2.4 Searching PGs and Flats using landmark

```
app.get('/search',function(req,res){
    var landmark = req.query.landmark;
    var sql = "select * from pg where landmark like '%"+landmark+"'";
    con.query(sql,function(err,result){
        if(err) {console.log(err)}
        res.render('./listofrooms',{Room:result});
    })
})

app.get('/searchflat',function(req,res){
    var landmark = req.query.landmark;
    var sql = "select * from flat where landmark like '%"+landmark+"'";
    con.query(sql,function(err,result){
        if(err) {console.log(err)}
        res.render('./flatlist',{Room:result});
    })
})
```

4.2.5 Owner Profile

```
    app.get("/ownerprofile",function(req,res){
        var sql = "select * from owner where username = '"+uname+"' and
        pwd = '"+pass+"';
        select p.PG_ID,p.Name,p.landmark,p.Address from pg p,owner o
        where o.username = '"+uname+"' and
        o.Owner_ID=p.Owner_ID;
        select f.Flat_ID,f.Name,f.landmark,f.address from flat f,owner o
        where o.username='"+uname+"' and o.Owner_ID = f.Owner_ID";
        con.query(sql,function(err,result){
            if(err){console.log(err)};
            res.render("./ownerprofile",{o:result[0],Room:result[1],flat:result[2]});
        })
    })
```

4.2.6 PG and Flat Details

```
     app.get('/pg:id', function (req, res) {
```

```
    var pid = req.params.id;
    console.log(pid);
    con.query(‘SELECT * from pg where Name=’${req.params.id}’‘,
    function (error, results, fields) {
      if (error) throw error;
      if (results.length > 0) {
         res.render(’roomdet’,{Room:results});
      } else {
        console.log("Not found");
      }
    });
  })

  app.get(’/flat:id’, function (req, res) {
    var pid = req.params.id;
    console.log(pid);
    con.query(‘SELECT * from flat where Name=’${req.params.id}’‘,
    function (error, results, fields) {
      if (error) throw error;
      if (results.length > 0) {
         res.render(’flatdet’,{Room:results});
      } else {
        console.log("Not found");
      }
    });
  })
```

4.2.7 Deleting PGs and Flats

```
app.get(’/deletepg’,(req,res)=>{
        var sql = "delete from pg where PG_ID=?";

        var PG_ID = req.query.PG_ID;
        con.query(sql,[PG_ID],(err,result)=>{
            if(err) {console.log(error)}
            res.redirect('/ownerprofile');
        })
      })

      app.get('/deleteflat',(req,res)=>{
        var sql = "delete from flat where Flat_ID=?";

        var Flat_ID = req.query.Flat_ID;
        con.query(sql,[Flat_ID],(err,result)=>{
            if(err) {console.log(error)}
            res.redirect('/ownerprofile');
        })
      })
```

4.2.8 Adding PGs and Flats

```
 var Owner_ID
      app.get('/addpg',(req,res)=>{

        Owner_ID = req.query.Owner_ID;
        console.log(Owner_ID)
        res.render('addpg')
      })

      app.get('/addflat',(req,res)=>{

        Owner_ID = req.query.Owner_ID;
        console.log(Owner_ID)
        console.log(req.query)
        res.render('addflat')
    })
 app.post('/addpg',(req,res)=>{
```

```
 console.log(req.body);
 const Name = req.body.Name;
 const landmark = req.body.landmark;
 const No_Rooms = req.body.Noofrooms;
 const Rooms_Filled = req.body.roomsfilled;
 const Rooms_vac = req.body.roomsvac;
 const Address = req.body.address;
 const facilities = req.body.facilities;
 const foods = req.body.food;
 const contact = req.body.contact;
 var sql = "INSERT INTO pg(Name,landmark,No_Rooms,Rooms_Filled,Rooms_vac,
 Address,facilities,foods,Owner_ID,contact)VALUES(’"+Name+"’,’"+landmark+"’,
 ’"+No_Rooms+"’,’"+Rooms_Filled+"’,’"+Rooms_vac+"’,’"+Address+"’,’"+facilities+
 ’"+foods+"’,’"+Owner_ID+"’,’"+contact+"’)";
     con.query(sql,function(err,result){
         if(err) throw err;
         alert("pg added");
         res.redirect(’/ownerprofile’);
     })
})



 app.post(’/addflat’,(req,res)=>{
     console.log(req.body);
     const Name = req.body.Name;
     const landmark = req.body.landmark;
     const contact = req.body.contact;
     const address = req.body.address;
     const type = req.body.type;
     console.log(req.query);
     var sql = "Insert into flat(Name,landmark,contact,address,type,Owner_ID)
     values(’"+Name+"’,’"+landmark+"’,’"+contact+"’,’"+address+"’,’"+type+"’,
     ’"+Owner_ID+"’)";
```

```
            con.query(sql,function(err,result){
                if(err) throw err;
                alert("flat added");
                res.redirect('/ownerprofile');
            })
        })
```

4.2.9 Adding PGs and Flats

```
 var P_ID
    app.get('/updatepg',(req,res)=>{
        P_ID = req.query.PG_ID;
        console.log(P_ID)
        res.render('updatepg')
      })
 app.post('/updatepg',(req,res)=>{
            const Name = req.body.Name;
            const landmark = req.body.landmark;
            const Noofrooms = req.body.Noofrooms;
            const roomsfilled = req.body.roomsfilled;
            const roomsvac = req.body.roomsvac;
            const address = req.body.address;
            const facilities = req.body.facilities;
            const food = req.body.food;
            con.query('CALL update_pg(?, ?, ?, ?, ?, ?, ?, ?, ?)'
            ,[Name,landmark,Noofrooms,roomsfilled,roomsvac,
            address,facilities,food,P_ID], (error, results) => {
                if (error) throw error;
                res.redirect('/ownerprofile')
           });
            }) //update

        var FID
         app.get('/updateflat',(req,res)=>{
```

```
        FID = req.query.Flat_ID;
        console.log(FID)
        res.render('updateflat')
})
app.post('/updateflat',(req,res)=>{
    const Name = req.body.Name;
    const landmark = req.body.landmark;
    const contact = req.body.contact;
    const address = req.body.address;
     const type = req.body.type;
     console.log(FID)
     console.log(req.query)
    con.query('CALL updateflat(?, ?, ?, ?, ?,?)'
    ,[Name,landmark,contact,address,type,FID], (error, results) => {
        if (error) throw error;
        res.redirect('/ownerprofile')
   });
    }) //update
```

Chapter 5

Results and Snapshots

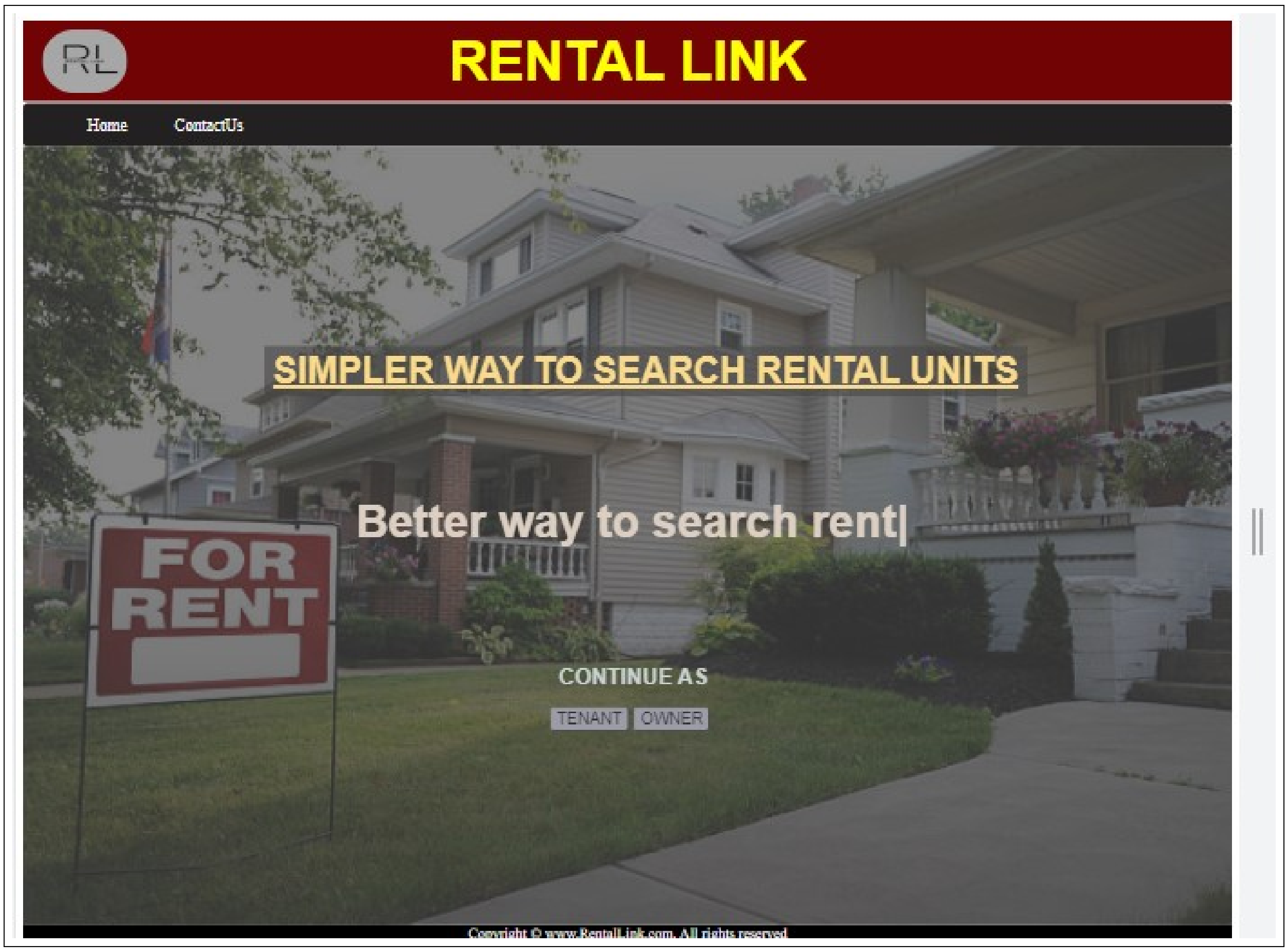

RL
RENTAL LINK
Home
ContactUs
SIMPLER WAY TO SEARCH RENTAL UNITS
Better way to search rent|
CONTINUE AS
TENANT
OWNER
Copyright © www.RentalLink.com. All rights reserved

Figure 5.1: Home

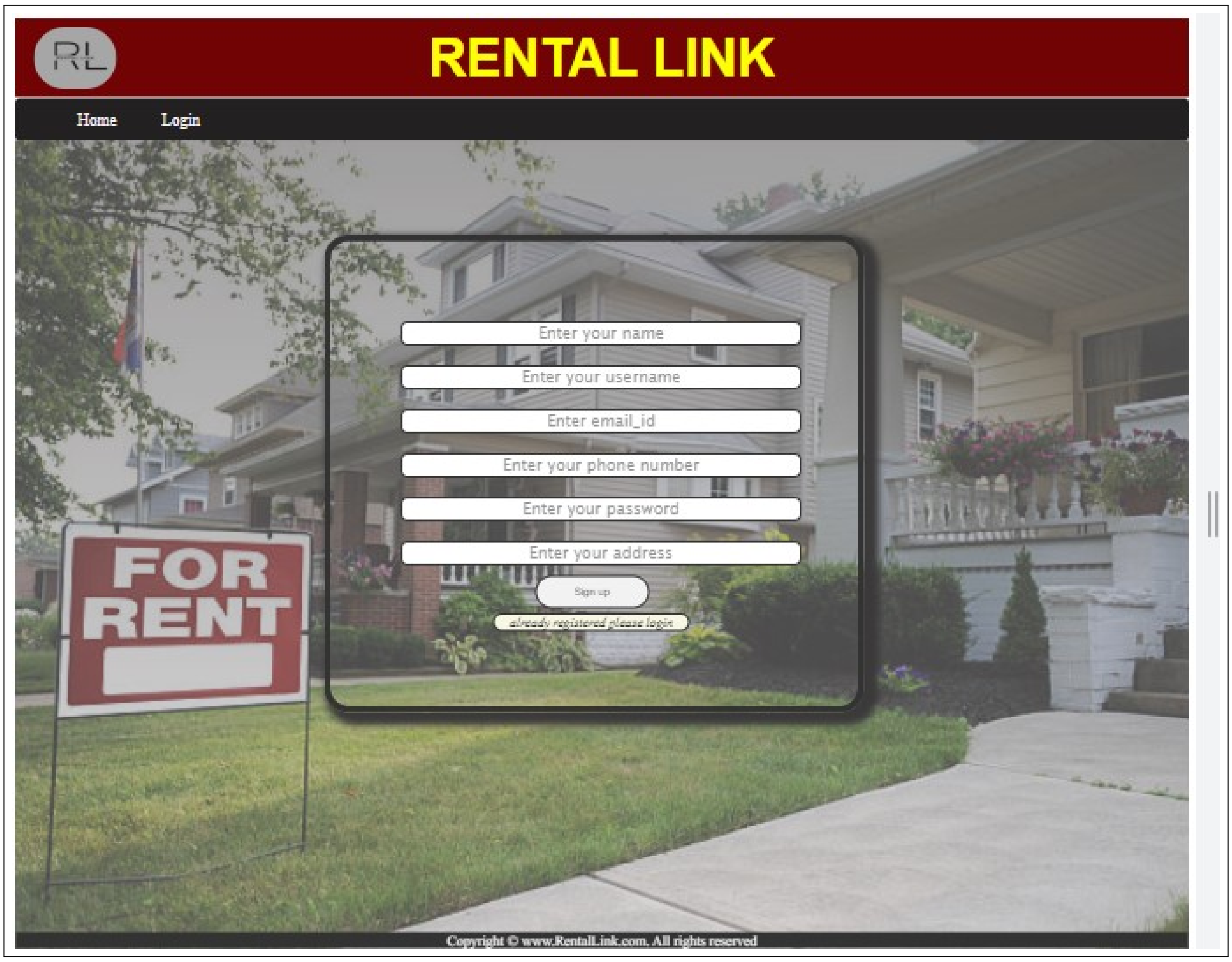


Figure 5.2: Owner-Signup

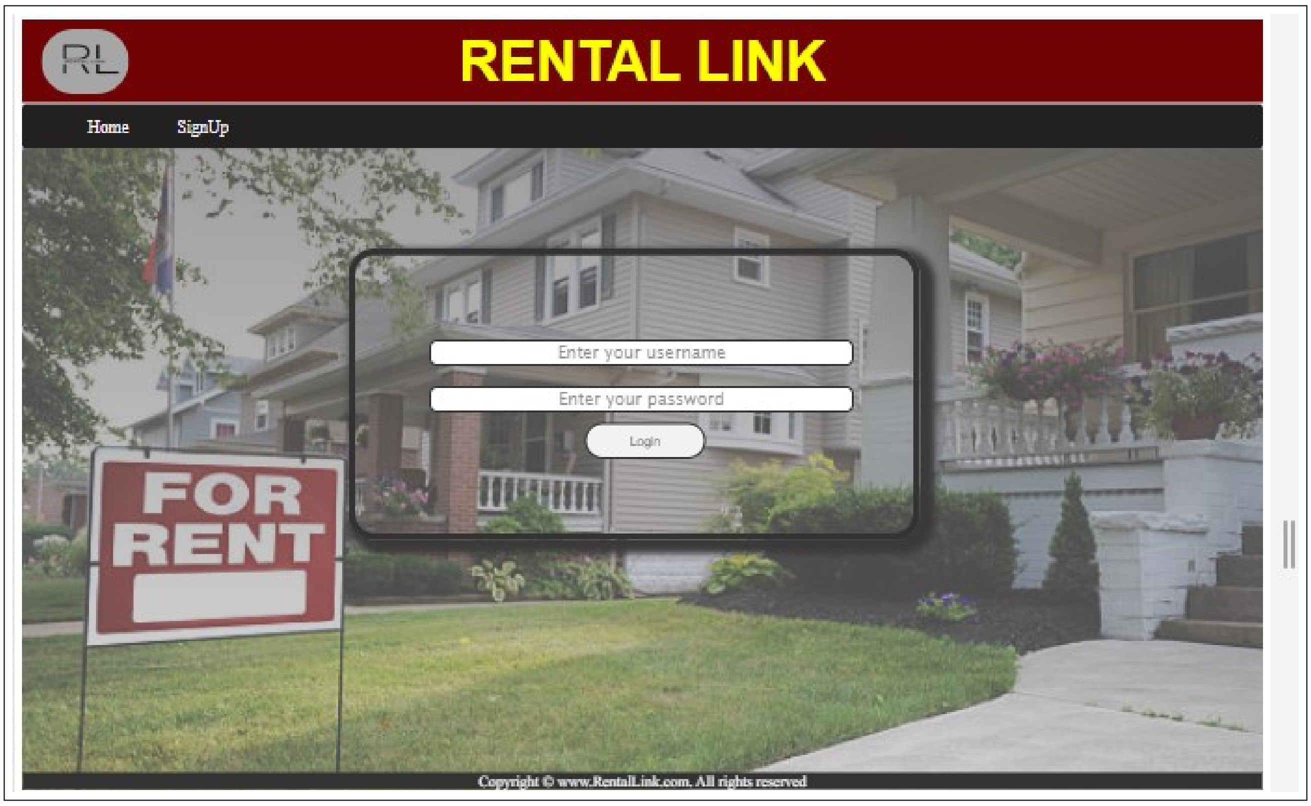


Figure 5.3: Owner-Login

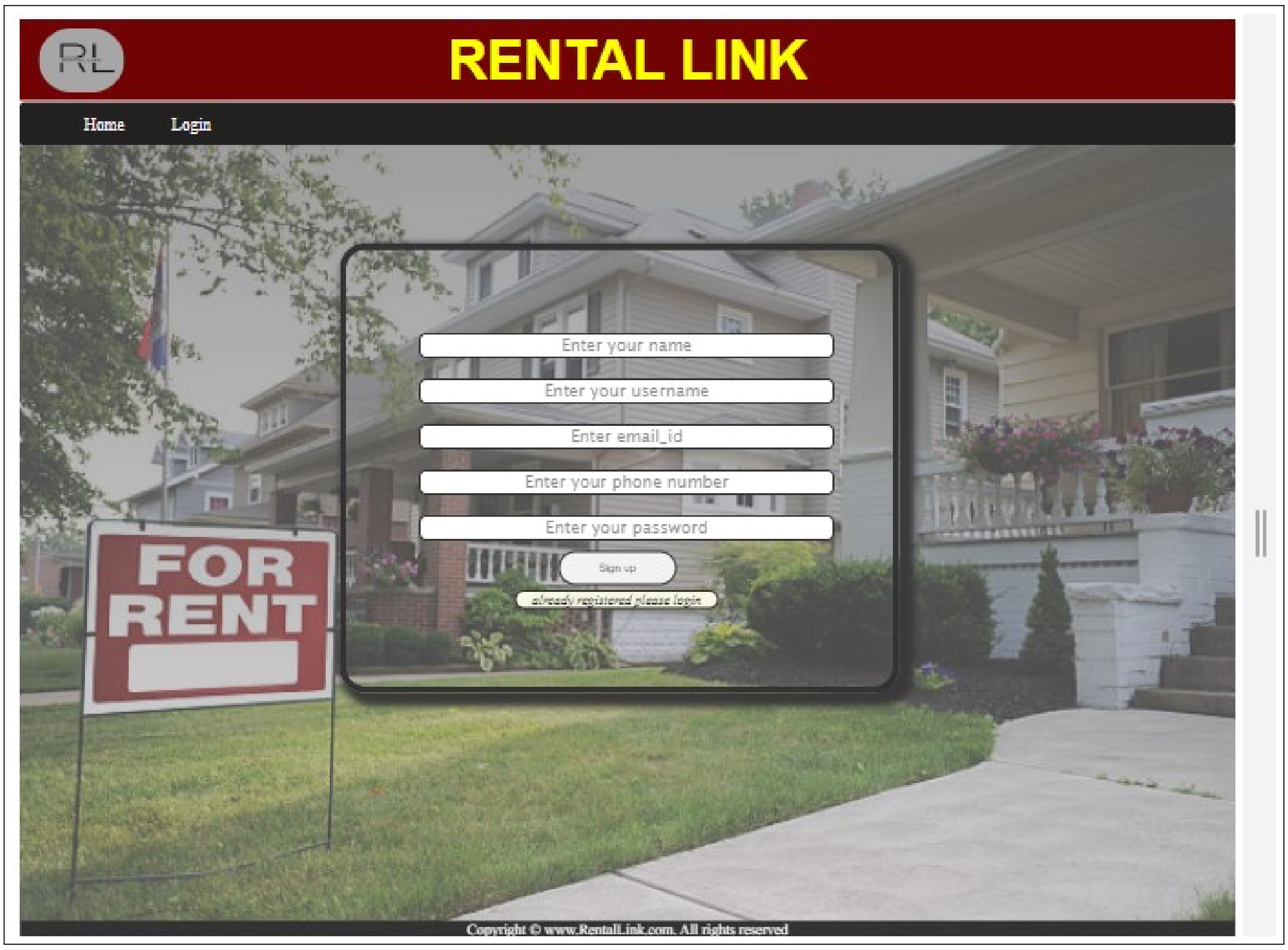


Figure 5.4: Tenant-signup

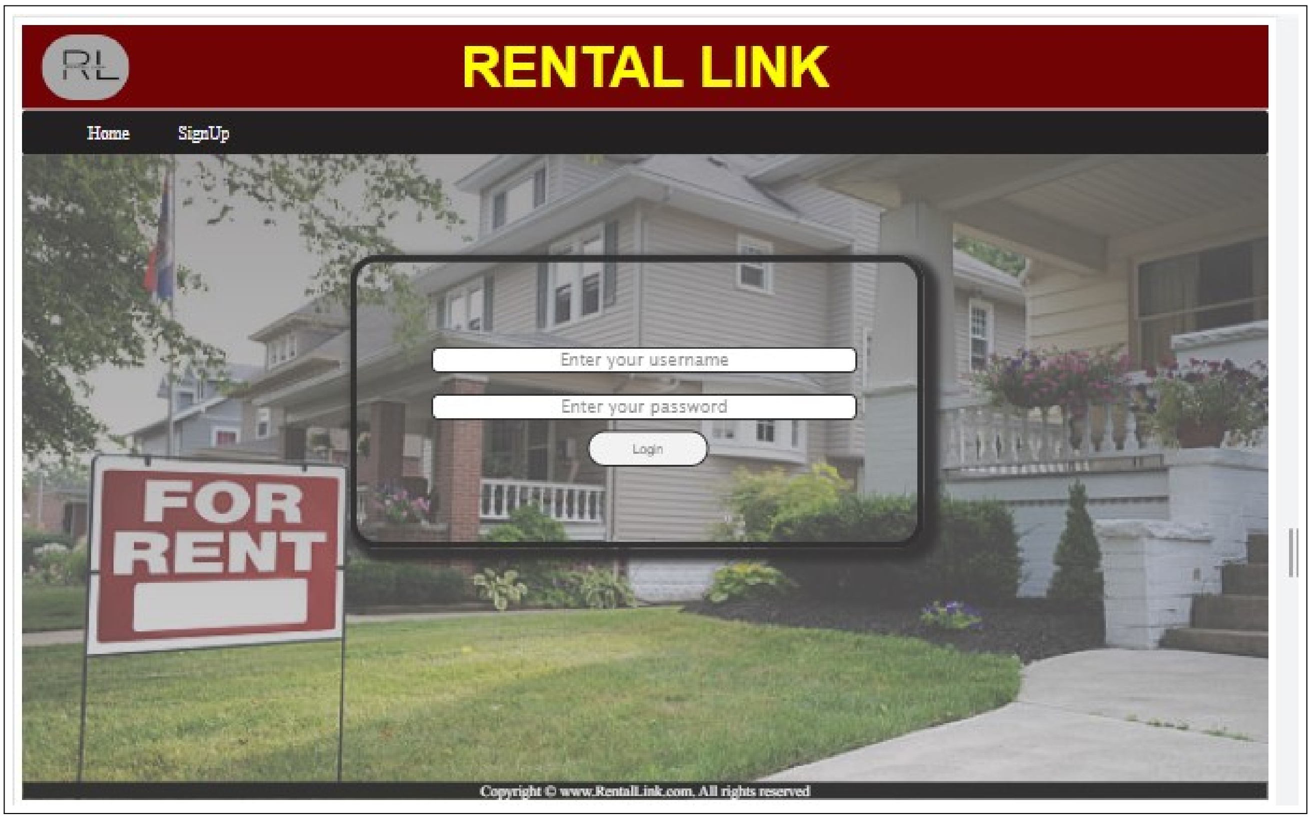
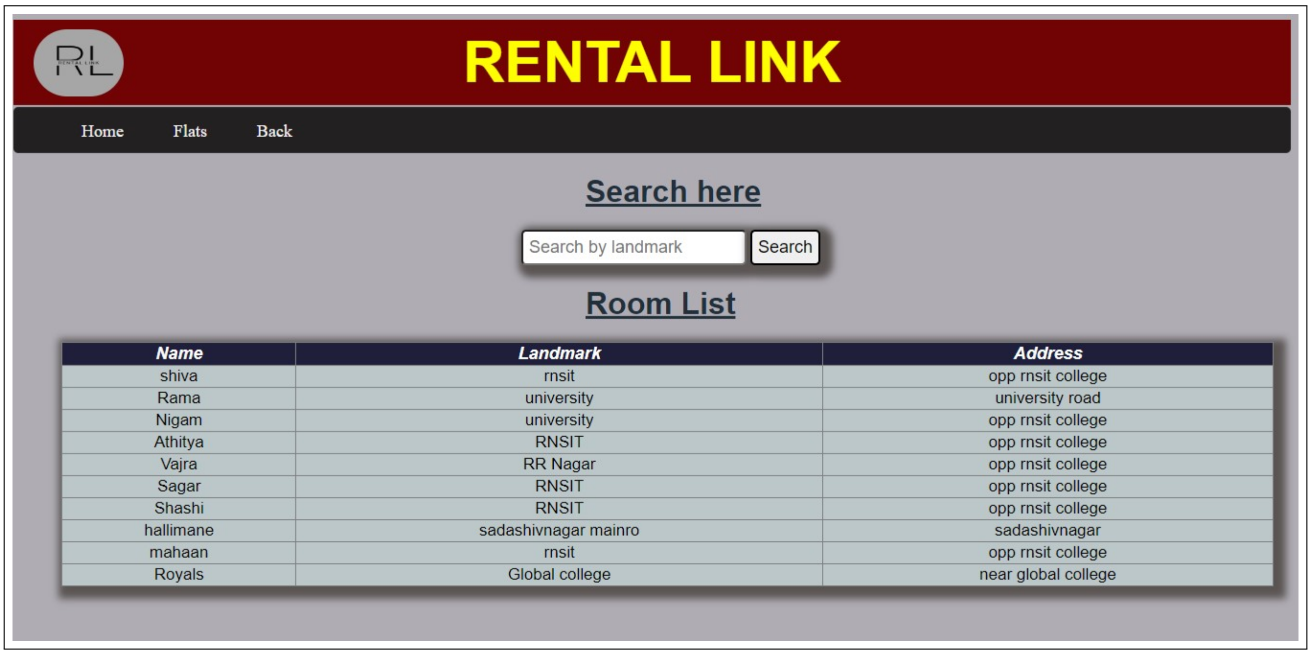
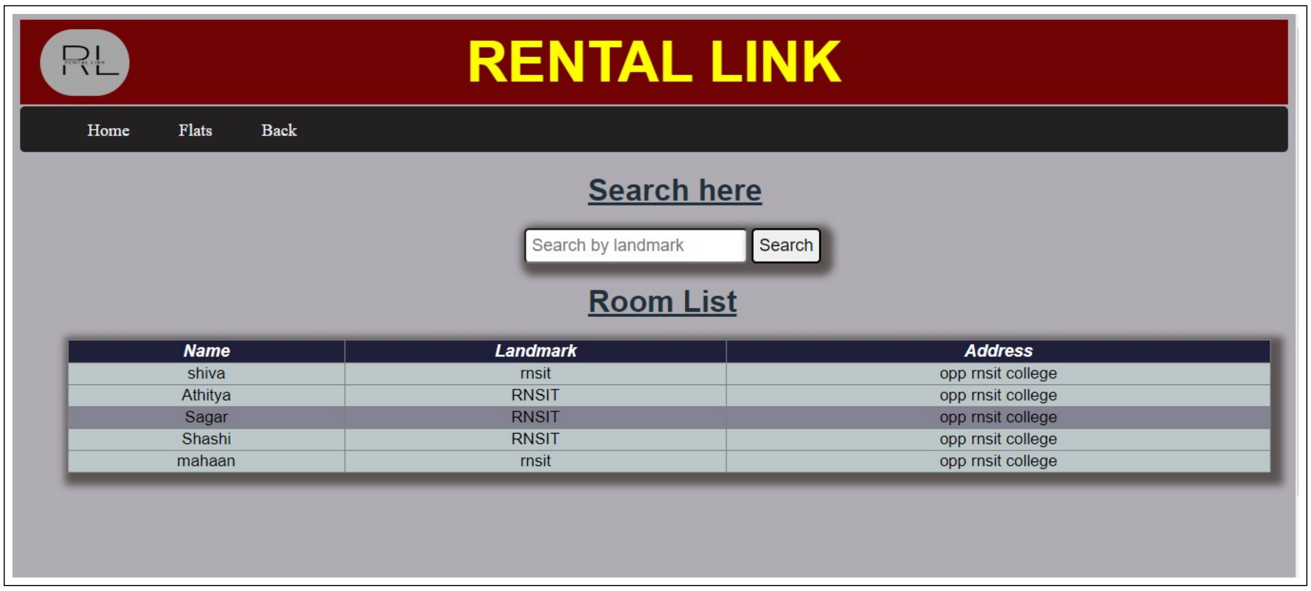


Figure 5.5: Tenant-Login



Name	Landmark	Address
shiva	rnsit	opp rnsit college
Rama	university	university road
Nigam	university	opp rnsit college
Athitya	RNSIT	opp rnsit college
Vajra	RR Nagar	opp rnsit college
Sagar	RNSIT	opp rnsit college
Shashi	RNSIT	opp rnsit college
hallimane	sadashivnagar mainro	sadashivnagar
mahaan	rnsit	opp rnsit college
Royals	Global college	near global college

Figure 5.6: List of PGs

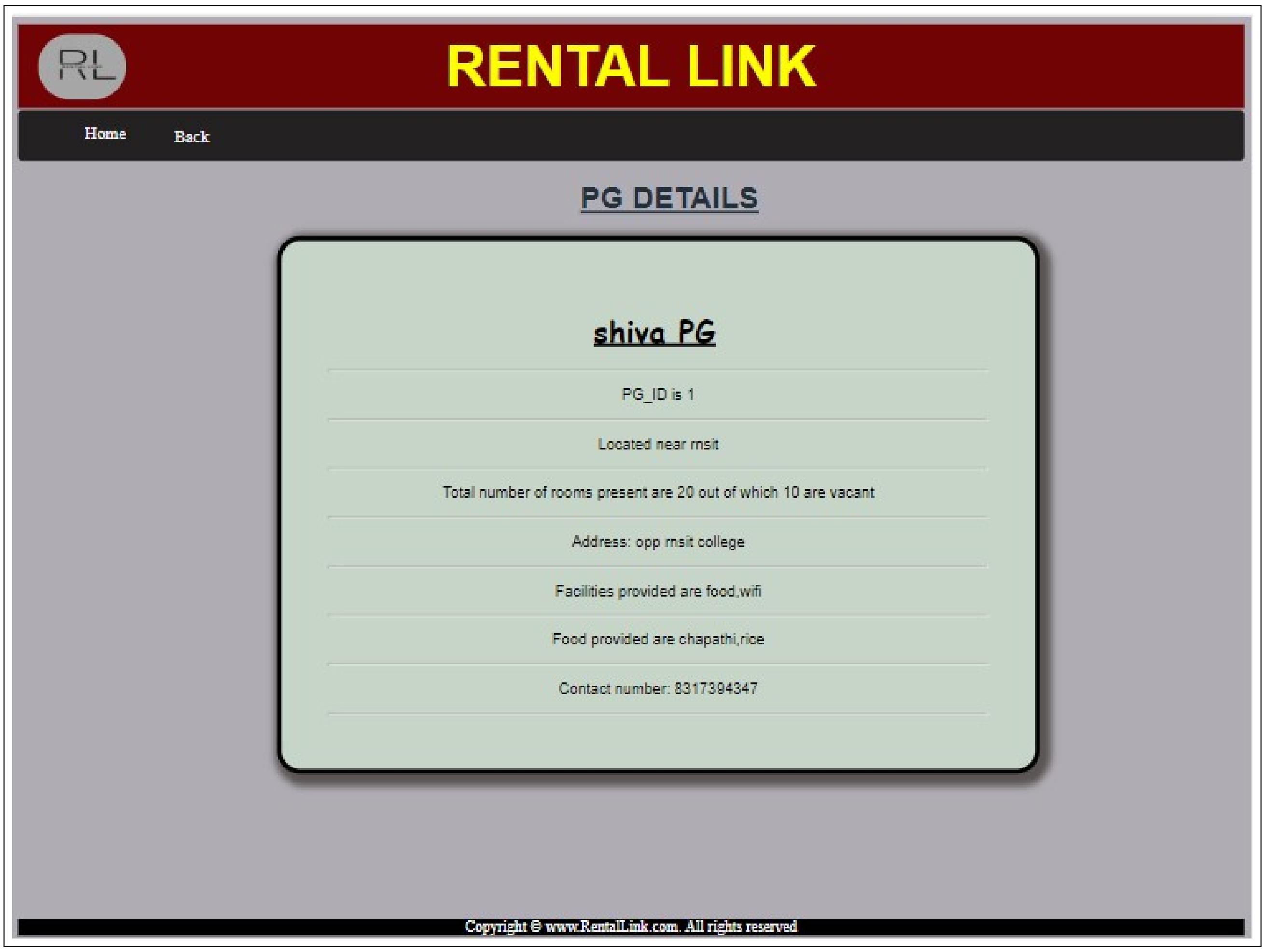


Search here

Room List

Name	Landmark	Address
shiva	rnsit	opp rnsit college
Athitya	RNSIT	opp rnsit college
Sagar	RNSIT	opp rnsit college
Shashi	RNSIT	opp rnsit college
mahaan	rnsit	opp rnsit college

Figure 5.7: List of PGs in a particular area



PG DETAILS

shiva PG

PG_ID is 1

Located near rnsit

Total number of rooms present are 20 out of which 10 are vacant

Address: opp rnsit college

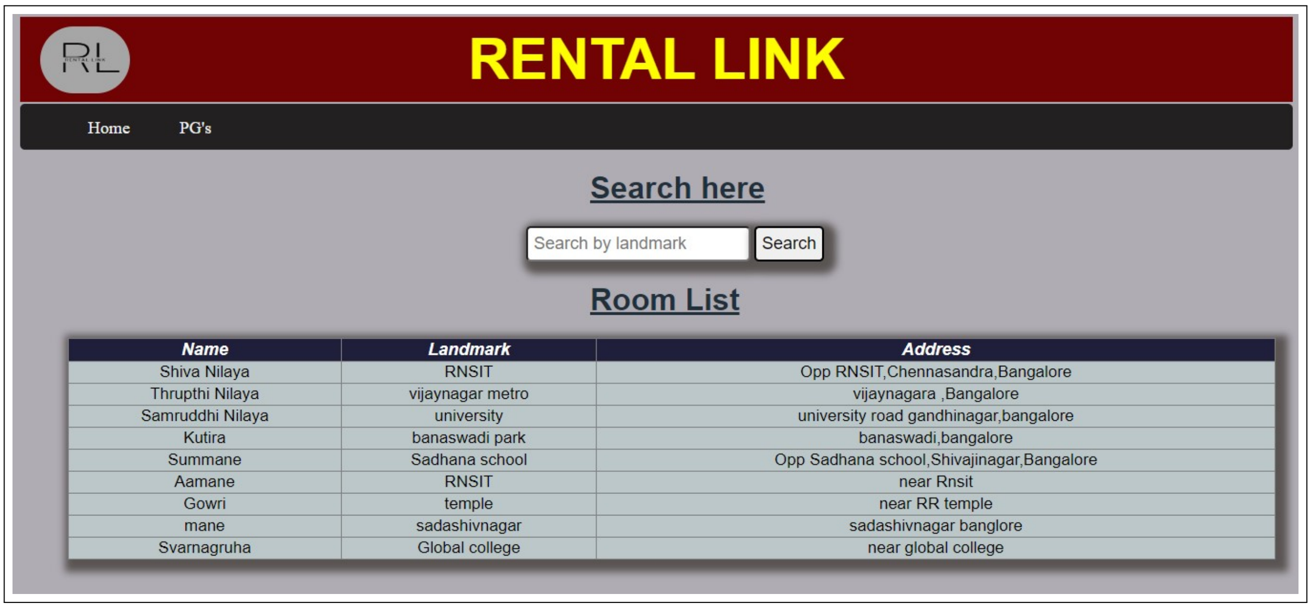
Facilities provided are food,wifi

Food provided are chapathi,rice

Contact number: 8317394347

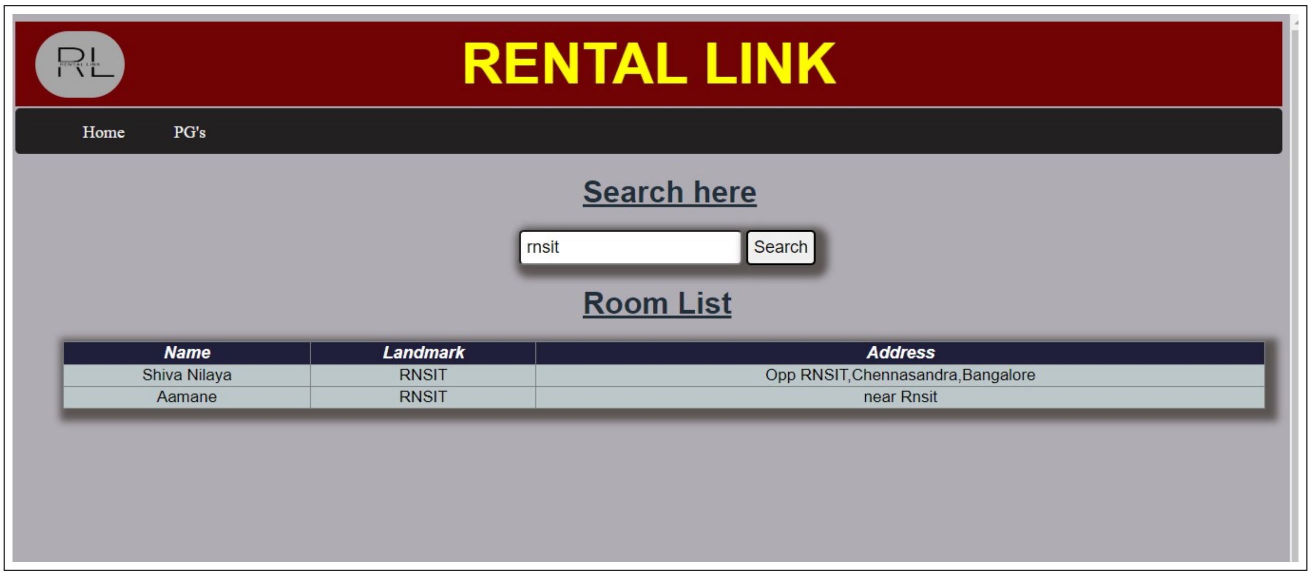
Copyright © www.RentalLink.com. All rights reserved

Figure 5.8: Details of PG



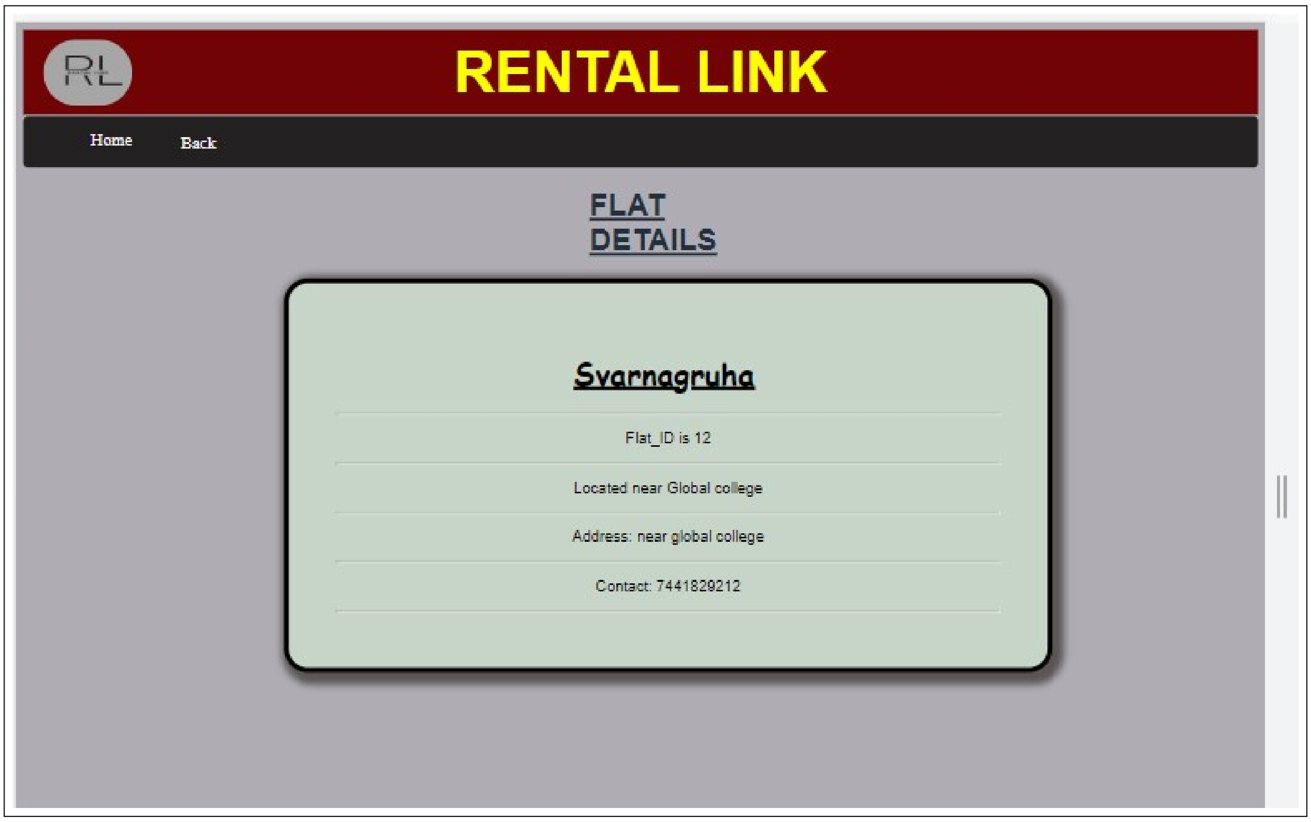
Name	Landmark	Address
Shiva Nilaya	RNSIT	Opp RNSIT,Chennasandra,Bangalore
Thrupthi Nilaya	vijaynagar metro	vijaynagara ,Bangalore
Samruddhi Nilaya	university	university road gandhinagar,bangalore
Kutira	banaswadi park	banaswadi,bangalore
Summane	Sadhana school	Opp Sadhana school,Shivajinagar,Bangalore
Aamane	RNSIT	near Rnsit
Gowri	temple	near RR temple
mane	sadashivnagar	sadashivnagar banglore
Svarnagruha	Global college	near global college

Figure 5.9: List of flats



Name	Landmark	Address
Shiva Nilaya	RNSIT	Opp RNSIT,Chennasandra,Bangalore
Aamane	RNSIT	near Rnsit

Figure 5.10: List of Flats in a particular area



Svarnagruha

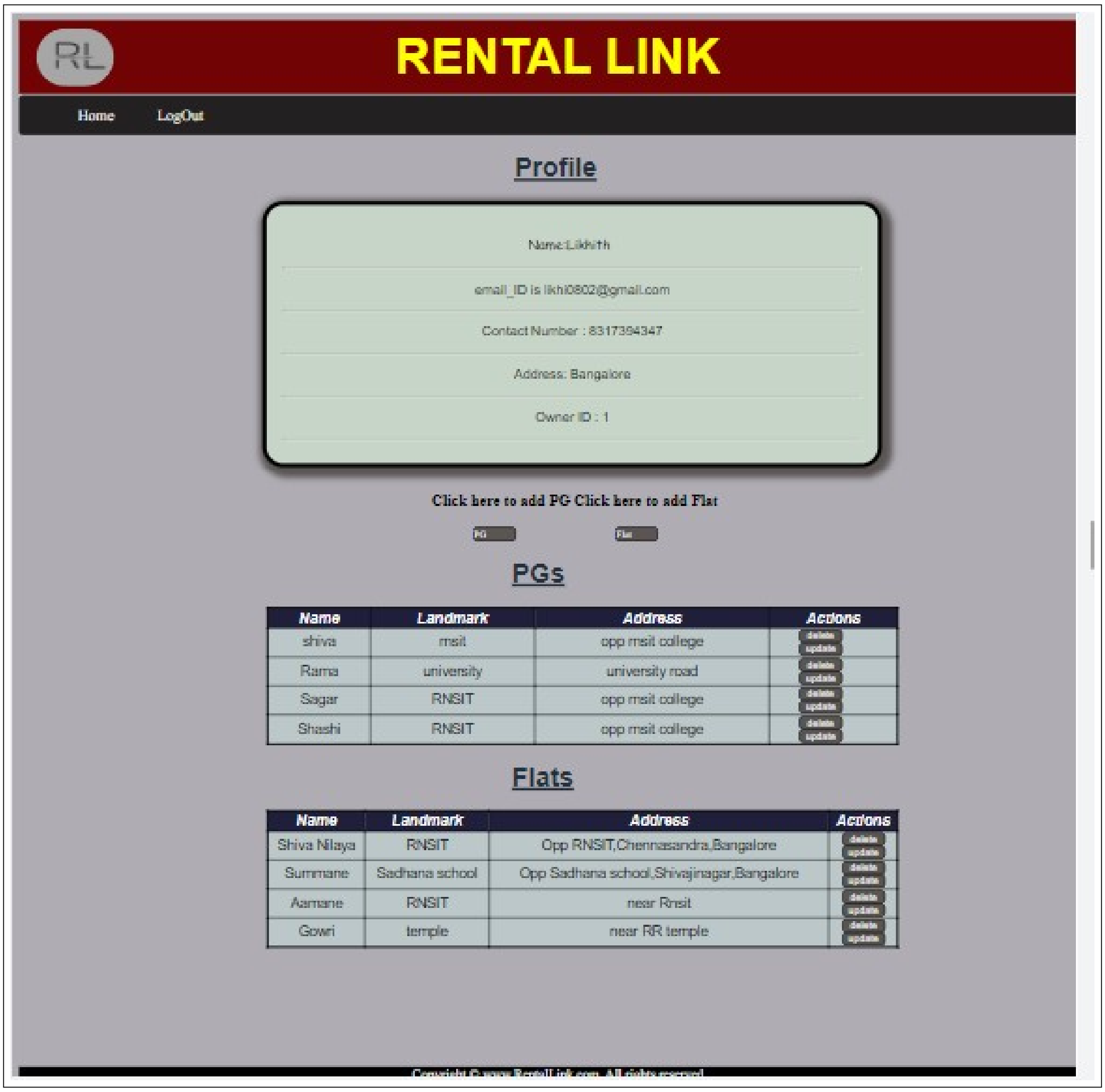
Flat_ID is 12

Located near Global college

Address: near global college

Contact: 7441829212

Figure 5.11: Details of Flat



Profile

Name:Likhith

email_ID is likhi0802@gmail.com

Contact Number : 8317394347

Address: Bangalore

Owner ID : 1

Click here to add PG Click here to add Flat

PGs

Name	Landmark	Address	Actions
shiva	rnsit	opp rnsit college	delete update
Rama	university	university road	delete update
Sagar	RNSIT	opp rnsit college	delete update
Shashi	RNSIT	opp rnsit college	delete update

Flats

Name	Landmark	Address	Actions
Shiva Nilaya	RNSIT	Opp RNSIT,Chennasandra,Bangalore	delete update
Summane	Sadhana school	Opp Sadhana school,Shivajinagar,Bangalore	delete update
Aamane	RNSIT	near Rnsit	delete update
Gowri	temple	near RR temple	delete update

Copyright © www.RentalLink.com. All rights reserved.

Figure 5.12: Owner Profile Page

Chapter 6

Conclusion & Future Enhancements

6.1 Conclusion

Rental Solution help people to explore the rental homes available in a distant or less known area.

It mainly helps students when they are supposed to move to different places for their studies by helping them to search rooms and paying guests in their desired locality.

Our website also help people to contact the owners of the rental rooms and paying guests by providing their contact details. It also helps the owners in building their network and attract more tenants and also enhance their service and business.

This website provides owners to maintain a profile where they can manage their rental properties, where they can add, delete and also can update its details.

In conclusion this website will reduce the wastage of time required to search rooms in a less known place, and also helps the owners to maintain a record of their rental properties.

6.2 Future Enhancements

Currently this website can be used by tenants only to search for the rental rooms in a particular area and helps owners to keep a record of their properties and maintain them, in future following developments can be seen

Tenant can maintain a profile where he can get the details of his rental and can also provide reviews and feedback about his/her experience so that others can judge the property by this feedback,and also helps the owners to improve the demands and services and also a satisfactory maintenance.

In case of paying guests, this website can integrate with basic essentials and service providers like electricians,plumbers,daily utility suppliers, wifi and cable operators,hotels and mess providers and others, so that all essentials required can be under one roof and can be accessed with ease.

Payment gateway like internet banking and UPIs can also be introduced where tenants can easily pay their rents and receive receipts accordingly.

A facility to proceed to book or reserve the rental properties by tenants can also be provided through the website.

References

- GeekforGeeks

  https://www.geeksforgeeks.org/
- Stack Overflow

  https://stackoverflow.com/
- “Node.js and MySQL Web development: Luke welling”
- “MySQL tutorial: Colt Stan Lee”