**ACKNOLWEDGEMENT**

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely privileged to have got this all along the completion of our project. All that we have done is only due to such supervision and assistance and we would not forget to thank them.

Our Sincere thanks to highly esteemed institution **SRI VENKATESHWARA COLLEGE OF ENGINEERING** for providing us this great opportunity.

We express our sincere gratitude to **Dr. SURESHA**, Principal, SVCE for providing support and encouragement.

We respect and thank **Ms.Kulkarni Varsha, Asst. Professor, Dept. of CS&E**for providing us an opportunity, giving us all support and guidance which made us complete the project duly. We are extremely thankful to **Dr. Shrishail S Math, HOD Dept. of CS&E** for providing such a nice support and guidance, although he had busy schedule.

Finally, we thank all our computer science faculty members, parents and friends who helped directly or indirectly to meet our project work with success.

**ABSTRACT**

The Property Management Database fully and formally describes the requirements for the property management database.The property management system sets out the functional and non functional requirements .It includes the description of the user interface documentation and the requirements for the Property Management Database.

The main objective of property management database is to create a property database for every customer like residential,vacant,commercial etc to buy or sell the property.Our System will search and display the property details to buy.The Property Management Database is highly interactive between the user and customer. Property Management Database is web based product and interactive.

**LIST OF FIGURES**

A. Schema Diagram

B. Entity Relationship Diagram

**LIST OF TABLES**

1. Agent
2. Branch
3. Customer
4. Property
5. Payment

**TABLE OF CONTENTS**

**Chapter no. Page.no**

1. **INTRODUCTION 1**

1.1 OBJECTIVES 1

1.2 LIMITATIONS 2

**2. STUDY OF EXISTING SYSTEM 3**

2.1 A CASE STUDY ON Property Management Database 3

2.2 PROPOSED SYSTEM 3

**3. DATABASE DESIGN 4**

3.1 SOFTWARE REQUIREMENT SPECIFICATION 6

3.1.1 COLLECTION OF REQUIREMENTS 7

3.1.2 SOFTWARE AND HARDWARE REQUIREMENTS 7

3.2 CONCEPTUAL DESIGN 8

3.2.1 SCHEMA DIAGRAM 8

3.2.2 ER DIAGRAM 9

3.3 IMPLEMENTATION 10

3.3.1 FRONT END 10

3.3.2 BACK END 16

3.3.3 TRIGGER 18

3.3.4 STORED PROCEDURE 19

1. **USER INTERFACES 20**

4.1 SCREEN SHOTS 20

1. **CONCLUSIONS**

**6. REFERENCES**

**Chapter 1**

**INTRODUCTION**

The Property Management Database keeps the record of the data and the property details and helps to maintain the record which saves time and make search of available property easily.This contains all information about the customer who sell or buy property and shows the available property details.This will contain all the records with case and also save the data of the customer who will buy or sell property.This documents will lead to a unique interpretation of product

**1.1 Objectives**

In Property Management Database product we provided all the important functions which a property administrator can perform.The main objective of this product is that it will contain all information about the customer who sell or buy the property of the customers and available property details

Main Objectives are:-

* To make the task of the customer easier by providing all the data
* Availability of the property details of the customer.
* Interface is very simple and easy to understand.
* View of the product is quite simple and attractive.

**1.1.1 Scope of Project**

The software is web based application.This software contains all records with case and also save the data of the buyer and seller.

The main application of the software specified is it can be used by agents,customers etc in an efficient manner.The aim of this software is that it contains all information about agent, customer and available property details.Customer and property record can be maintained easily.

It is easy to fetch the data from the database and also it is easier to destroy the existing data from the database.

**1.2 LIMITATIONS**

* + Property details can be displayed until the property sold.
  + Maps are not provided for the convenience of the customer.
  + It does not have registration facility.
  + Property Details can be visible only for registered users.
  + It does not cover International market.

**Chapter 2**

**STUDY OF EXISTING SYSTEM**

**2.1 CASE STUDY**

The property management system is a web based application consists of the customer information,agent information and property details of the customer.This web based application allows the user to register as buyer or seller.Once the customer has register successfully then the customer has been allowed to enter the property details or customer can view the property details.Once the customer has selects the property then the customer has redirected to the payment page.After successful payment customer has to contact the agent for further registration process.

**2.2 PROPOSED SYSTEM**

* + The proposed system which allows the customer/user to quickly and easily search a property for buy and sell.
  + The registered user can upload his property for sale.
  + The system will give accurate information regarding the property which helps to view all property information directly from anywhere.
  + It is user friendly and flexible.
  + Customer should be able to entry all the necessary description of their property for sale.
  + Customer should be able to access property details services like buying and selling property.
  + Customer should be able to update his property,delete his account and view agent details.

**CHAPTER 3**

**DATABASE DESIGN**

Database table is made up of more than two records.Database is stored in the disk and contains tables,query and report.The general objective is to make information access easy,quick,inexpensive and flexible for the user.A collection of records make up a table.To design and store data to the database table are prepared.Two essential types of database are:

* Primary key: The field that is unique for all records occurrences.
* Foreign key: The field used to set relation between tables.Normalization is a

technique to avoid redundancy in the tables

The tables used in this projected are listed below consisting of its structure.

Table Name: customer

Description: This table stores the details of all the customers and the agent id for all customers.

|  |  |  |
| --- | --- | --- |
| Fields | Data Type | Description |
| c\_id | int () | Stores the customer id |
| c\_name | varchar () | Stores the customer name |
| c\_address | varchar () | Stores the customer address |
| c\_phone | bigint () | Stores the customer phone number |
| c\_type | varchar () | Stores the type of the customer |
| status | varchar () | stores the customer status |
| a\_id | int () | stores the agent id for each customer |

Table name: agent

Description: This table stores the details of all the agents and the branch id for all agents.

|  |  |  |
| --- | --- | --- |
| Fields | Data Type | Description |
| a\_id | int () | Stores the agent id |
| a\_name | varchar () | Stores the agent name |
| a\_salary | int () | Stores the agent salary |
| a\_phone | bigint () | Stores the agent phone |
| b\_id | int () | Stores the branch id for each agent |
| a\_address | varchar () | Stores the agent address |

Table name: branch

Description: This table stores the details of all the branches.

|  |  |  |
| --- | --- | --- |
| Fields | Data Type | Description |
| b\_id | int () | Stores the branch id |
| b\_name | varchar () | Stores the branch name |
| b\_phone | bigint () | Stores the branch phone |

Table name: property

Description: This table stores all property details and customer id for all property.

|  |  |  |
| --- | --- | --- |
| Fields | Data Type | Description |
| p\_id | int () | Stores the property id |
| p\_length | int () | Stores the length of the property |
| p\_width | int () | Stores the width of the property |
| p\_cost | int () | Stores the cost of the property |
| p\_type | varchar () | Stores the type of the property |
| p\_location | varchar () | Stores the location of the property |
| c\_id | int () | Stores the customerid for each property |

Table name: payment

Description: This table stores the payment details of all transactions completed by the customer including the property id and agent id.

|  |  |  |
| --- | --- | --- |
| Fields | Data Type | Description |
| py\_id | int () | Stores the payment id |
| py\_cost | int () | Stores the payment cost |
| p\_id | int () | Stores the property id for each payment |
| c\_id | int () | Stores the customer id for each payment |
| a\_id | int () | Stores the agent id for each payment |
| p\_mode | varchar () | Stores the payment mode |
| c\_no | bigint () | Stores the card no |

**3.1 SOFTWARE REQUIREMENT SPECIFICATION**

A Software requirements specification is a complete description of the behaviour of the system to be developed.It includes set of use cases that describes all the interactions that the user will have with the software. Use cases are also known as functional requirements.Non functional requirements impose constraints on the design or implementation.

**3.1.1 COLLECTION OF REQUIREMENTS**

**3.1.2 SOFTWARE AND HARDWARE REQUIREMENTS**

The software used in building this project are as specified:

* **Operating System:** Windows 10
* **Database:** mySQL
* **Local Host:** xampp

The physical components are:

* + **Processor:** intel corei5 and above
  + **Processor speed:** 2.4GHz
  + **RAM:** 8GB
  + **Storage Space:** 1GB or above
  + **Monitor Resolution:** colour monitor with minimum resolution of 1920\*1080 and above

**3.2 CONCEPTUAL DESIGN**

**3.2.1 SCHEMA DIAGRAM**

The scheme diagram is the skeleton structure that represents the logical view of entire database.

**agent**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| a\_id | a\_name | a\_salary | a\_phone | a\_address | b\_id |

**branch**

|  |  |  |
| --- | --- | --- |
| b\_id | b\_name | b\_phone |

**customer**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| c\_id | c\_name | c\_phone | c\_address | c\_type | status | a\_id |

**property**

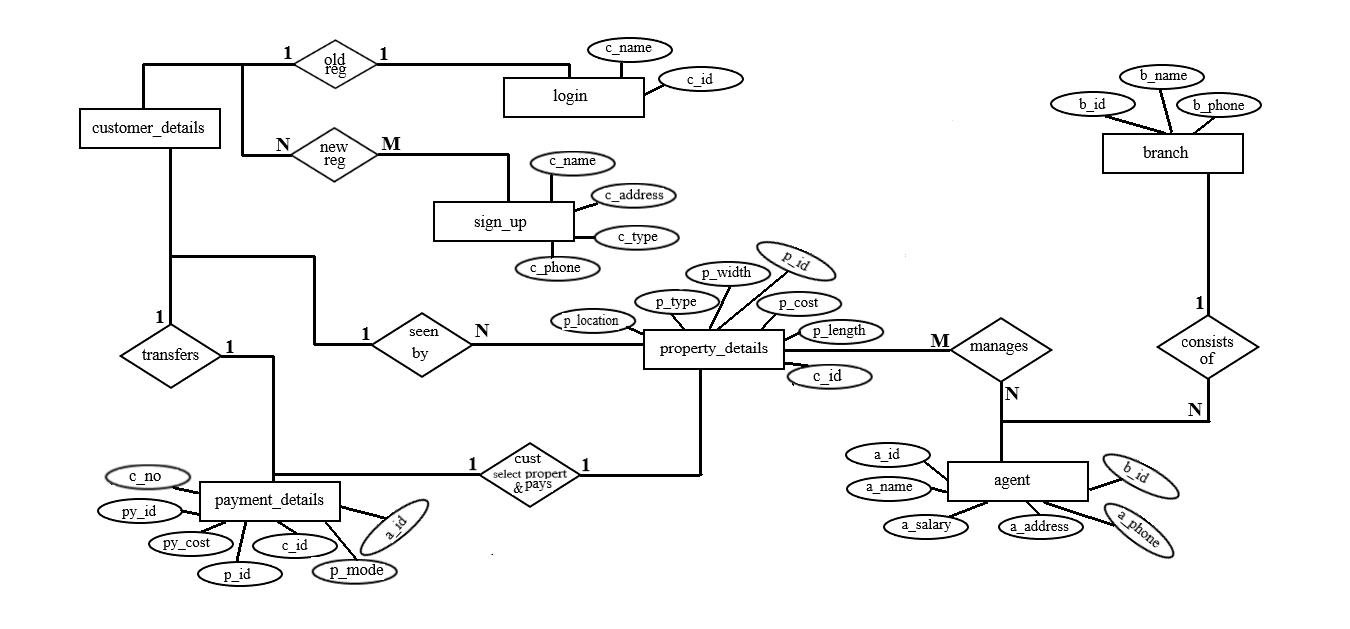
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| p\_id | p\_length | p\_width | p\_cost | p\_type | p\_location | c\_id |

**payment**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| py\_id | py\_cost | p\_id | c\_id | a\_id | p\_mode | c\_no |

**3.2.2 ER DIAGRAM**

An entity-relationship (ER) diagram is a specialized graphic that illustrate the interrelationships between entities in the database.ER diagram often use symbols to represent three different types of information.Boxes are commonly used to represent entities.Diamonds are normally used to represent relationships and Ovals are used to represent attributes.

****

**3.3 IMPLEMENTATION**

Implementation is a stage of the project where the front end and the back end design is turned into the working system.

**3.3.1 FRONT END**

The few front end implementation codes are described below.

* **Login page:**
* **PHP code:**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$db="project";

// Create connection

$conn = new mysqli($servername, $username, $password,$db);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if(isset($\_POST["sub"])) {

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

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

$sel = "SELECT c\_id, c\_name FROM customer";

$result = mysqli\_query($conn,$sel);

while($row=mysqli\_fetch\_array($result)){

$tid=$row['c\_id'];

$tname=$row['c\_name'];

if(($tid==$id)&&($tname==$name)){

include 'output.php';

}

elseif(($tid!=$id)&&($tname!=$name)){

echo "Incorrect ID or Password"."<br>";

}

}

}

elseif(isset($\_POST["su"])) {

include 'signup.html';

}

?>

* **HTML code:**

<!DOCTYPE html>

<html>

<head>

<style>

body

{

background: url(login.jpg);

background-size: cover;

background-repeat: no-repeat;

}

div

{

border-radius: 25px;

margin: 200px auto ;

padding:30px;

color: white;

text-align: center;

background-color: rgba(180,180,180,0.45);

width: 375px;

}

.xlarge

{

font-size: 40px;

color: black;

}

.large

{

font-size: 20px

}

</style>

<title>PropertyManagementSystem</title>

</head>

<body>

<div>

<h1 class=xlarge ><i>Welcome to Property Management System</i></h1>

<form action="login.php" method="post">

<b class=large>Identidication Number:</b><br>

<input type="text" name="id" ><br>

<b class=large>Name:</b><br>

<input type="text" name="name" ><br>

<input type="submit" name="sub" value="submit">

<input type="submit" name="su" value="signup">

</form>

</div>

</body>

</html>

* **Property Input Details page:**
* **PHP code:**

<?php

$servername = "localhost";

$username = "root";

$password = "";

$db="project";

// Create connection

$conn = new mysqli($servername, $username, $password,$db);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if(isset($\_POST["sub"])) {

if(isset($\_POST["height"])) {

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

}

if(isset($\_POST["width"])) {

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

}

if(isset($\_POST["loc"])) {

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

}

if(isset($\_POST["price"])) {

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

}

if(isset($\_POST["type"])) {

$type=($\_POST["type"]);

if($type=="vacant")

$type="vacant";

if($type=="residential")

$type="residential";

else if($type=="commercial")

$type="commercial";

}

}

$ins1="INSERT INTO property(p\_height,p\_width,p\_cost,p\_type,p\_location) VALUES ('$height','$width','$price','$type','$loc')";

if(!mysqli\_query($conn,$ins1)){

echo 'not inserted';

}

else{

echo 'inserted';

}

?>

* **HTML code:**

<!DOCTYPE html>

<html>

<head>

<style>

body

{

background: url(propertyinput.jpg);

background-size: cover;

background-repeat: no-repeat;

}

div

{

border-radius: 25px;

margin: 200px auto ;

padding:30px;

color: white;

text-align: center;

background-color: rgba(180,180,180,0.75);

width: 375px;

}

.xlarge

{

font-size: 20px;

color: black;

}

.large

{

color: black;

}

</style>

<title>PropertyDetails</title>

</head>

<body>

<div>

<h1 class=large><i>Enter Property Details</i></h1>

<form action="propertyinput.php" method="post">

<b class=xlarge>Enter Property Dimension:</b><br>

<input placeholder="Height" type="text" name="height" ><br>

<input placeholder="Width" type="text" name="width" ><br>

<b class=xlarge>Enter the Location of your property:</b><br>

<input type="text" name="loc" ><br>

<b class=xlarge>Enter Type of property:</b><br>

<input type="radio" name="type" value="vacant">Vacant Property<br>

<input type="radio" name="type" value="residential" checked>Residential Property<br>

<input type="radio" name="type" value="commercial">Commercial Property <br>

<b class=xlarge>Enter the Price you would like to sell <br> your property for:</b><br>

<input type="text" name="price" ><br><br>

<input type="submit" name="sub" value="submit">

</form>

</div>

</body>

</html>

**3.3.2 BACK END**

The few back end sql code are described below

* To create customer table

SQL>create table customer (

c\_id int(11) primary key,

c\_name varchar(20),

c\_address varchar(20) ,

c\_phone bigint(10),

c\_type varchar(20),

status int(11),

a\_id int(11) );

* + To insert values into customer table

SQL>insert into customer values

(2, 'shreyas', 'sadashivnagar', 9884500158, 'buyer', NULL, 4);

SQL>insert into customer values

(3, 'rahul', 'yelahanka', 9590854666, 'buyer', NULL, 3);

SQL>insert into customer values

(4, 'akshay', 'yelahanka', 8556452123, 'seller', NULL, 3);

SQL>insert into customer values

(5, 'rakesh', 'sahakar nagar', 9884512626, 'buyer', NULL, 7);

* To create property table

SQL>create table property (

p\_id int(11) primary key,

p\_length int(10),

p\_width int(10),

p\_cost int(10),

p\_type varchar(20),

p\_location varchar(20),

c\_id int(11) );

* + To insert values into property table

SQL>insert into property values

(1, 20, 30, 2500000, 'vacant', 'jpnagar', 2);

SQL>insert into property values

(2, 40, 60, 3500000, 'residential', 'yelahanka', 3);

SQL>insert into property values

(3, 35, 45, 4000000, 'commercial', 'jayanagar', 4);

SQL>insert into property values

(6, 30, 54, 500000, 'residential', 'vidyanagar', 2);

**3.3.3 TRIGGER**

**3.3.4 STORED PROCEDUER**

**CHAPTER 4**

**USER INTERFACES**

**4.1 SCREEN SHOTS**

* **SIGN UP PAGE:**
* **LOGIN PAGE:**

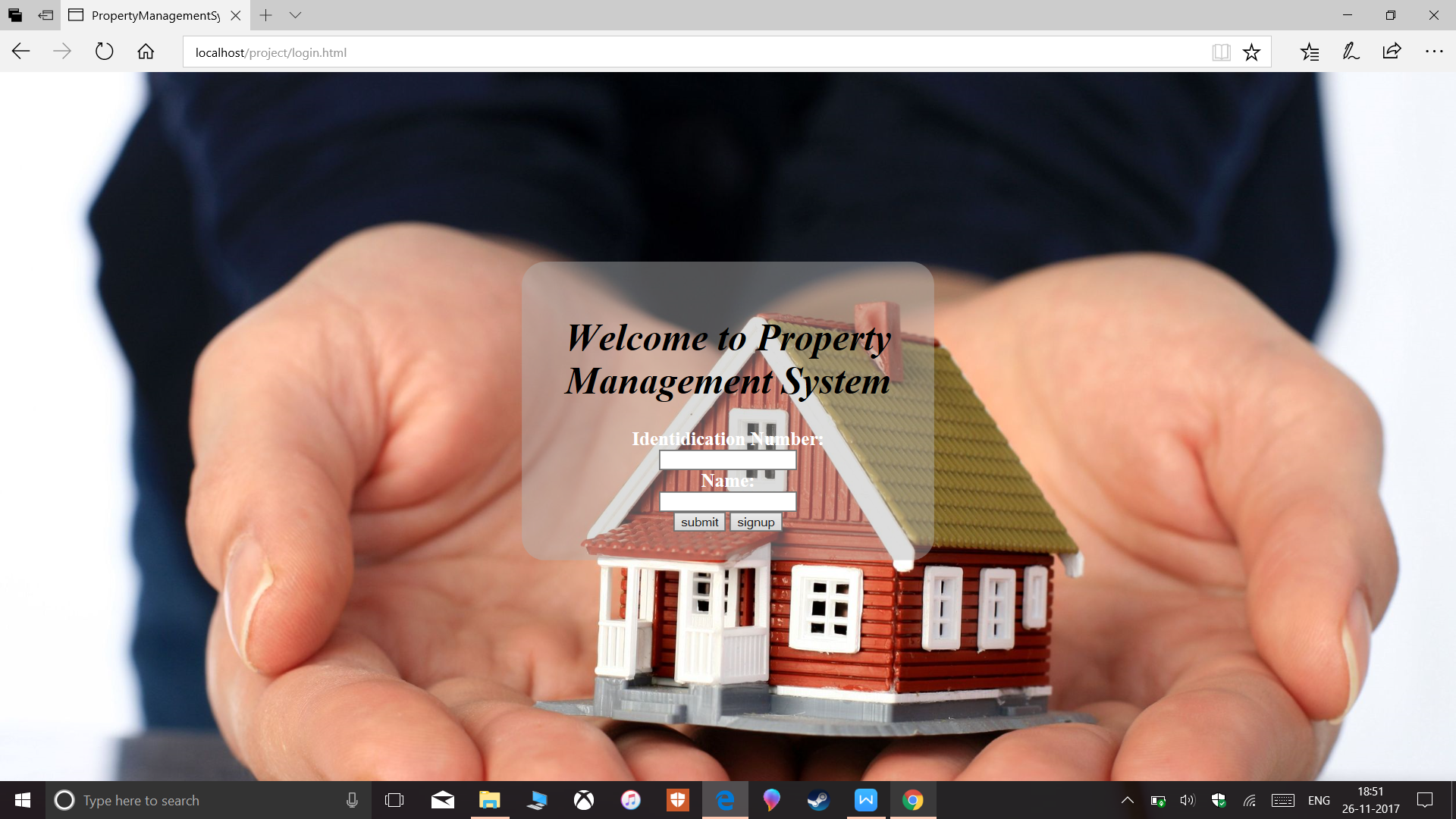
****

Fig 4a: login page

Above figure 4a is the login page of the web based application i.e., Property Management Data Base.