**CRM for a Real Estate Company**

****

Session: 2021 – 2025

**Submitted by:**

**Hammad Ejaz 2021-CS-102**

**Supervised by:**

**Dr. Awais Hassan**

Department of Computer Science

**University of Engineering and Technology**

**Lahore Pakistan**

**CONTENTS**

* **Abstract**
* **Functional Requirements**
* **Wireframes**
* **Data Structures**
* **Function Prototypes**
* **Flowchart**
* **Complete Code**
* **Test Cases**

**CRM for a Real Estate Company**

**Abstract:**

Finding customers for your business is not easy. In addition, once you find them, there comes another challenge – Establishing and maintaining strong relationships with them. CRM software helps you centralize, optimize and streamline your communication. The better you know your customers, the stronger your relationships will be with them. In addition, this will help you to increase your sales.

**Functional Requirements:**

There are two users of this system:

* Admin (Manager)
* User (Employ)

1. **Admin account:**

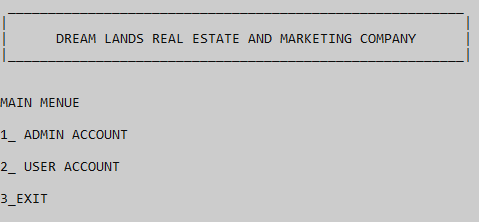
* Admin can check the total follow ups of users
* Admin can check the total visit schedule of users
* Admin has access to add leads (leads are the client information which is provided to user) in user account and delete leads
* Admin can check the total leads of all users
* Admin can check the total sales of all users
* Admin can add the projects and delete projects
* Admin can add users
* Admin has access to edit inventory
* Inventory includes:
* Maps of society
* Total number of plots
* Available plots
* Sold plots

1. **User account**

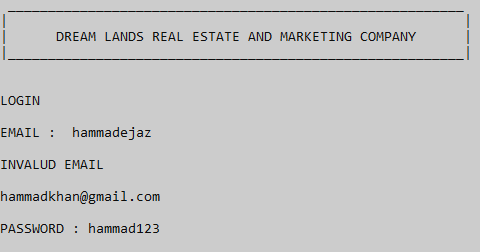
* User check the new leads and add their response.
* User check there follow ups
* User set their visit schedule with clients
* User also add their interested clients
* User also add their cold clients
* User add their sale information
* User check there total sales
* User check the inventory

**Wireframes:**

**Main Menu:**



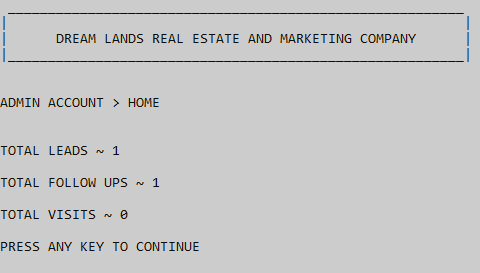
**Admin account**



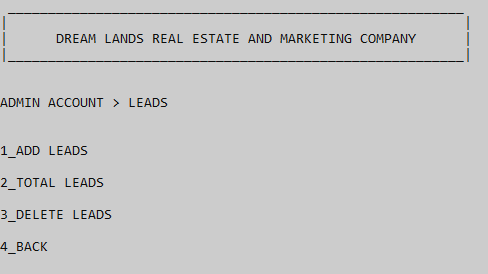
**Admin main menu**



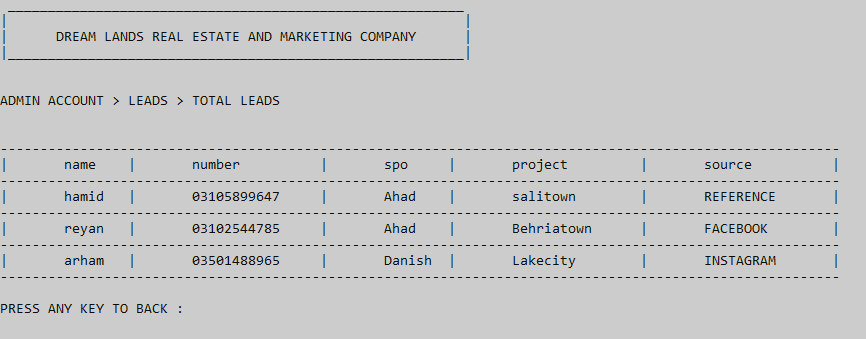
**Home**



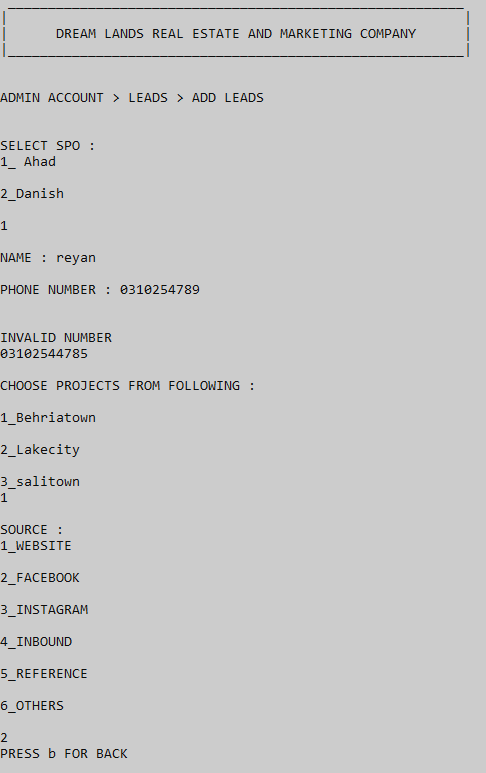
**Leads**

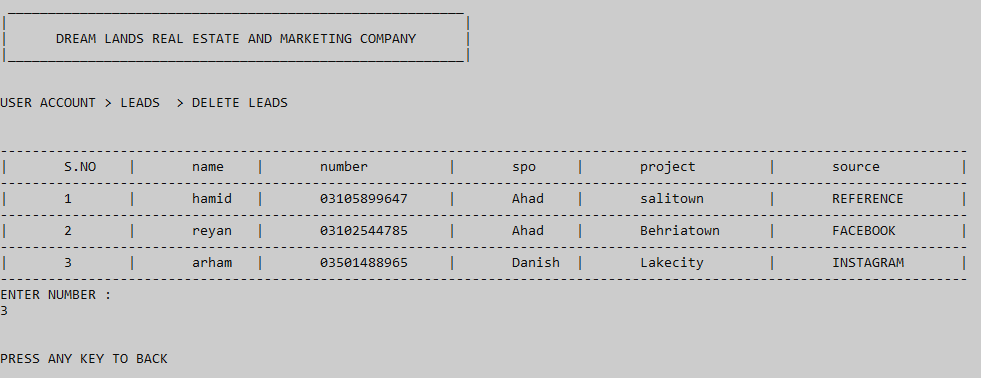


**Total leads**

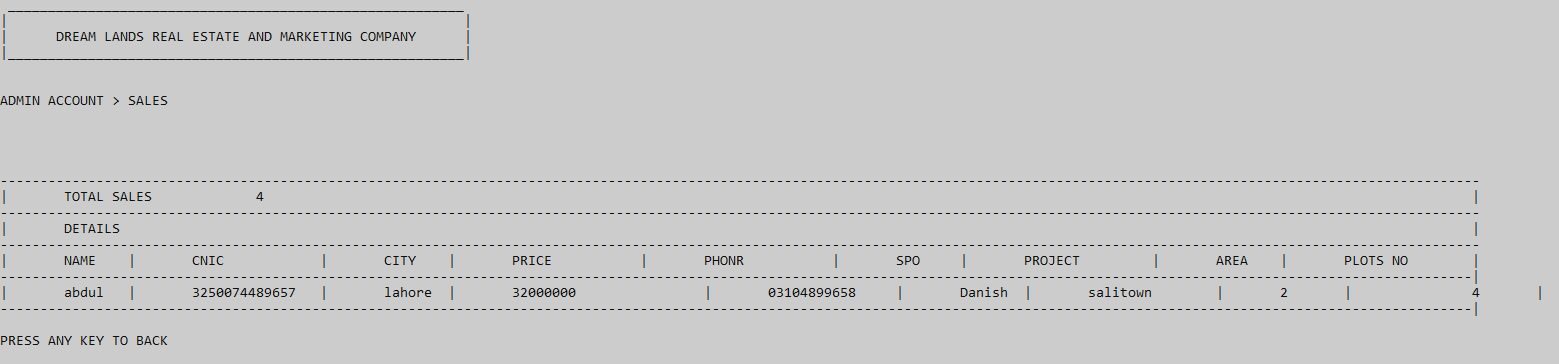


**Add leads**

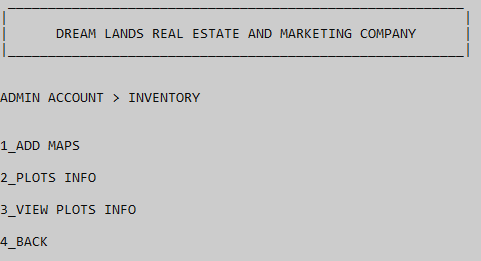


**Delete leads**

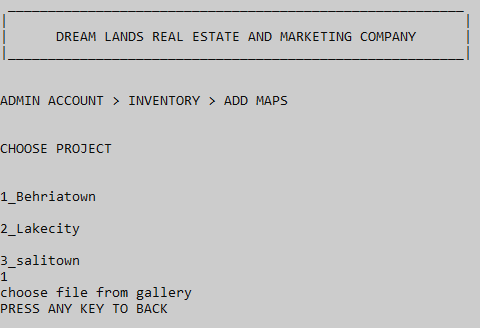
**Sales**



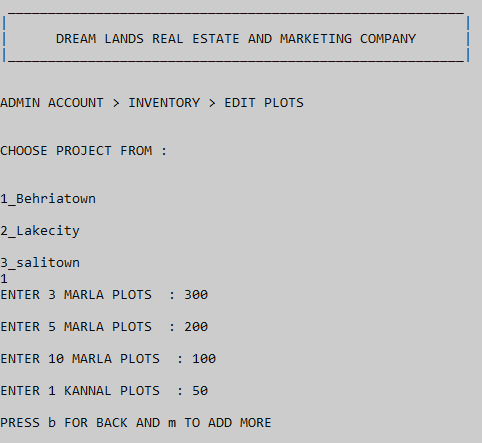
**Inventory**



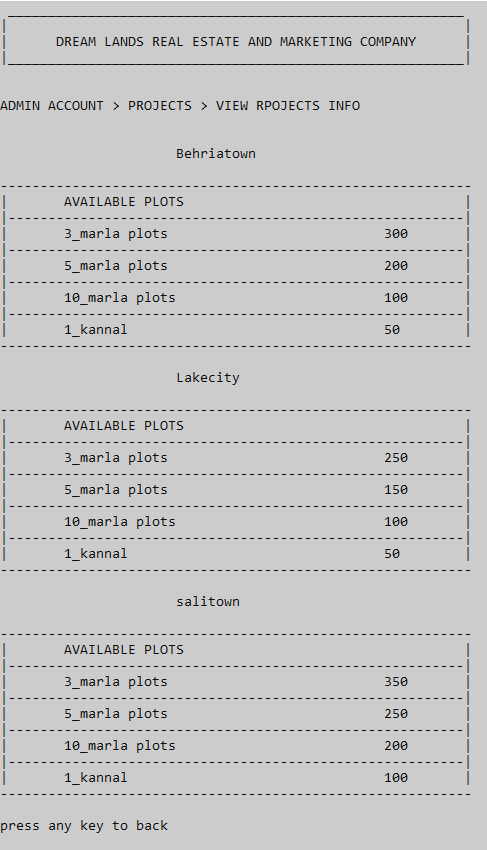
**Add maps**



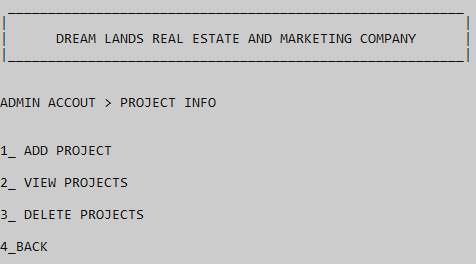
**Edit plots info**



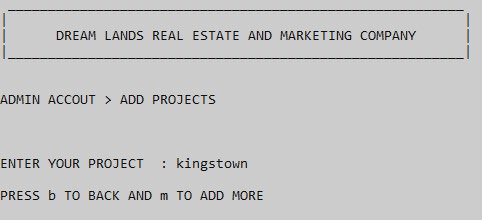
**View plots info**

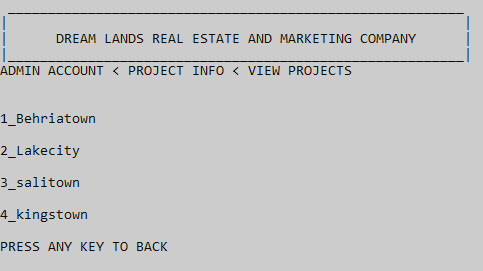


**Projects**

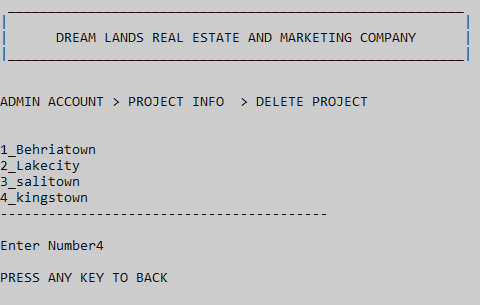


**Add project**



**View project**

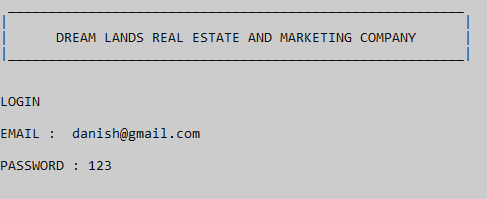
**Delete project**



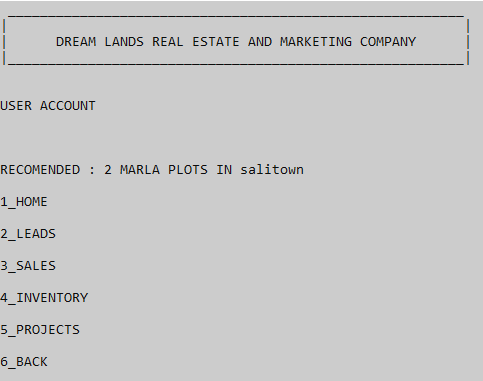
**Add user**



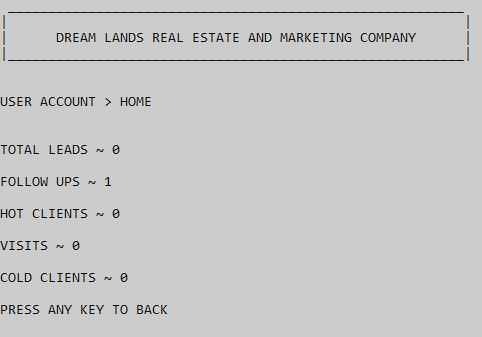
**User account**



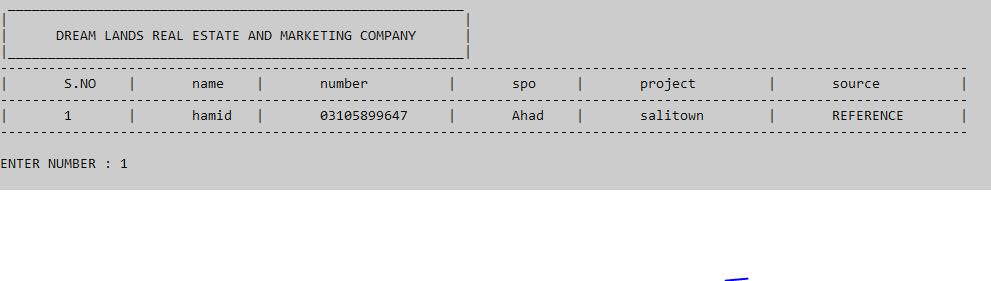
**User main menu**



**Home**

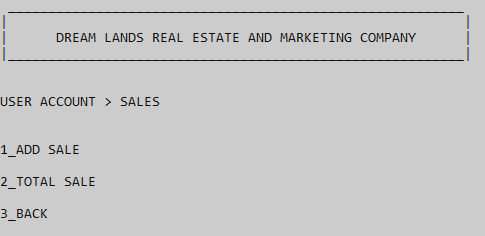


**Leads**

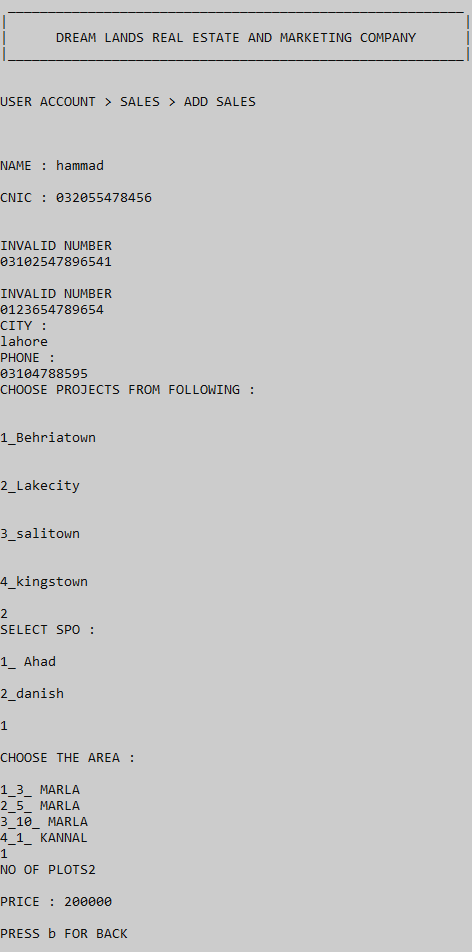




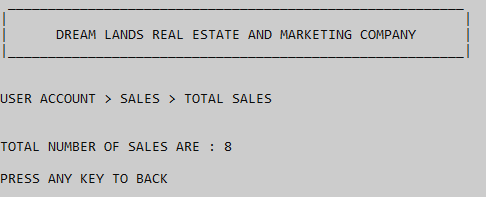
**Sale**



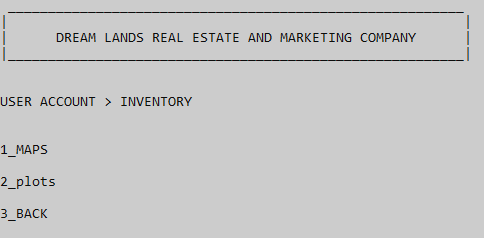
**Add sale**



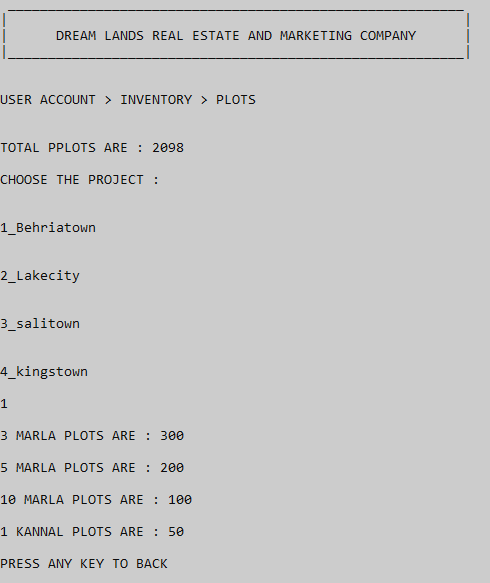
**Total sale**



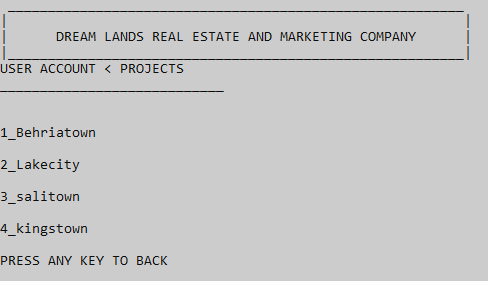
**Inventory**



**Plots**



**Projects**



**Data Structures:**

**Files:**

* Adminadduserstore.txt
* Admininvstore.txt
* Adminleadsstore.txt
* Adminprojectstore.txt
* Useraddsalestore.txt
* Useraddsalestore1.txt
* Userschedulestore.txt
* Userschedulestore1.txt

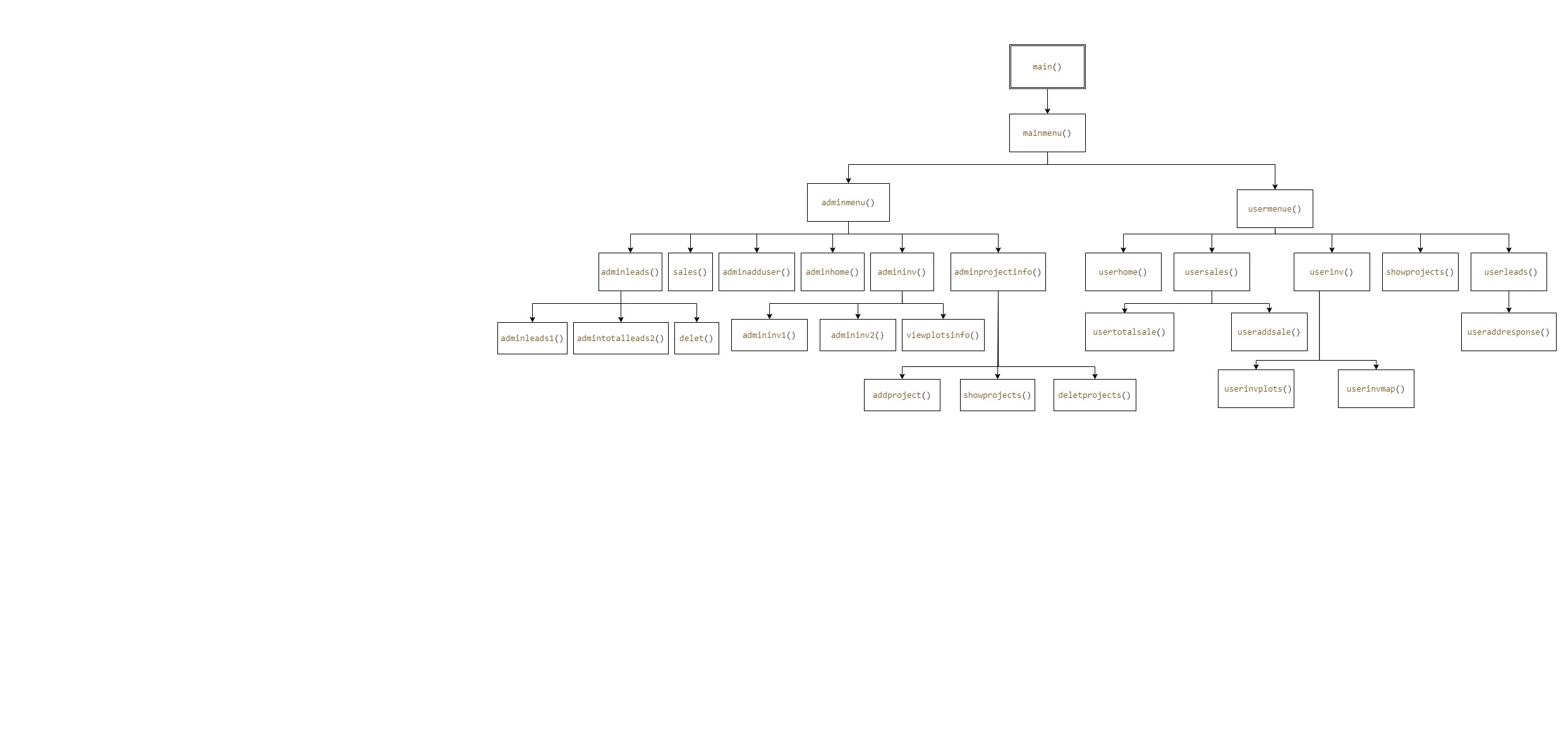
**Arrays:**

* string addleads\_spo[size];
* string addleads\_project[size];
* string addleads\_source[size];
* string addleads\_spo1[size];
* string addleads\_project1[size];
* string addleads\_source1[size];
* string source[6];
* string area[size];
* string projects[size];
* string sname[size];
* string addleads\_name[size];
* string addleads\_name1[size];
* string scity[size];
* string sproject[size];
* string sspo[size];
* string scheduale[size];
* string response[size];
* string addleads\_phone[size];
* string scheduale1[size];
* string response1[size];
* string addleads\_phone1[size];
* string scnic[size];
* string sphone[size];
* long sprice[size];
* int snum[size];
* int sarea[size];
* int tmarla[size];
* int thmarla[size];
* int fmarla[size];
* int kannal[size];
* int total\_plots[size];
* int most\_sold\_area[size];
* int most\_sold\_marla[size];

**Functions:**

1. void header();
2. void admin();
3. void adminhome();
4. void worngpassword();
5. void adminleads1();
6. void admintotalleads2();
7. void admininv2();
8. void sales();
9. void admininv1();
10. void adminadduser();
11. void userhome();
12. void userhome1();
13. void useraddresponse(int);
14. void useraddresponse1(int);
15. void useraddsale();
16. void usertotalsale();
17. void userinvmap();
18. void userinvplots();
19. void printprojects();
20. void source1();
21. void printarea();
22. void sorting\_sales\_data();
23. void addproject();
24. void showprojects();
25. void delet();
26. void deletprojects();
27. int mainmenu();
28. int adminmenu();
29. int adminleads();
30. int admininv();
31. int usermenue();
32. int userleads();
33. int userleads1();
34. int userleads1();
35. int usersales();
36. int userinv();
37. int totalsal();
38. int totalplo();
39. int totalfollowup();
40. int totalvisits();
41. int totalhotclients();
42. int totalcold();
43. int totalfollowup1();
44. int totalvisits1();
45. int totalhotclients1();
46. int totalcold1();
47. int small(int j);
48. int adminprojectinfo();
49. int print\_user();
50. string parsedata(string a, int f);
51. bool isValidEMAIL(string text);
52. int admin\_project\_load();
53. int user\_addsale\_load();
54. void user\_schedule\_load();
55. void user\_schedule\_load1();
56. void admin\_adduser\_load();
57. void admin\_inv\_load();
58. int admin\_leads\_load();
59. int admin\_leads\_load1();
60. void admin\_project\_store();
61. void user\_addsale\_store();
62. void user\_schedule\_store();
63. void user\_schedule\_store1();
64. void admin\_adduser\_store();
65. void admin\_leads\_store();
66. void admin\_leads\_store1();
67. void admin\_inv\_store();
68. void viewplotsinfo();
69. void sold\_plots(int i, int n);
70. void recomendation();
71. int mostsoldmarla();
72. int mostlargearea();
73. int userleads11();
74. int admintotalfollowup();
75. int admintotalvisits();

**Flowchart:**



**Code:**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

// functions declared

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

void header();

void admin();

void adminhome();

void worngpassword();

void adminleads1();

void admintotalleads2();

void admininv2();

void sales();

void admininv1();

void adminadduser();

void userhome();

void userhome1();

void useraddresponse(int);

void useraddresponse1(int);

void useraddsale();

void usertotalsale();

void userinvmap();

void userinvplots();

void printprojects();

void source1();

void printarea();

void sorting\_sales\_data();

void addproject();

void showprojects();

void delet();

void deletprojects();

int mainmenu();

int adminmenu();

int adminleads();

int admininv();

int usermenue();

int userleads();

int userleads1();

int userleads1();

int usersales();

int userinv();

int totalsal();

int totalplo();

int totalfollowup();

int totalvisits();

int totalhotclients();

int totalcold();

int totalfollowup1();

int totalvisits1();

int totalhotclients1();

int totalcold1();

int small(int j);

int adminprojectinfo();

int print\_user();

string parsedata(string a, int f);

bool isValidEMAIL(string text);

int admin\_project\_load();

int user\_addsale\_load();

void user\_schedule\_load();

void user\_schedule\_load1();

void admin\_adduser\_load();

void admin\_inv\_load();

int admin\_leads\_load();

int admin\_leads\_load1();

void admin\_project\_store();

void user\_addsale\_store();

void user\_schedule\_store();

void user\_schedule\_store1();

void admin\_adduser\_store();

void admin\_leads\_store();

void admin\_leads\_store1();

void admin\_inv\_store();

void viewplotsinfo();

void sold\_plots(int i, int n);

void recomendation();

int mostsoldmarla();

int mostlargearea();

int userleads11();

int admintotalfollowup();

int admintotalvisits();

// variables

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

int n = 0, totalsales = 0, countu = 0, m = 0;

string realproject = "iqbalgarden", uphone1 = "", uname1 = "", ucat1 = "", umail1 = "", upassword1 = "", email = "", password = "";

int s = 0, c = 0, t = 1, d = 0, u = 0, k = 0, y = 0, z = 0, x, soldplots = 0, s1 = 0;

// arrays declared

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

const long size = 1000;

string addleads\_spo[size];

string addleads\_project[size];

string addleads\_source[size];

string addleads\_spo1[size];

string addleads\_project1[size];

string addleads\_source1[size];

string source[6] = {"WEBSITE", "FACEBOOK", "INSTAGRAM", "INBOUND", "REFERENCE", "OTHERS"};

string area[size] = {"3\_ MARLA", "5\_ MARLA", "10\_ MARLA", "1\_ KANNAL"};

string projects[size];

string sname[size];

string addleads\_name[size];

string addleads\_name1[size];

string scity[size];

string sproject[size];

string sspo[size];

string scheduale[size];

string response[size];

string addleads\_phone[size];

string scheduale1[size];

string response1[size];

string addleads\_phone1[size];

string scnic[size];

string sphone[size];

long sprice[size];

int snum[size];

int sarea[size];

int tmarla[size];

int thmarla[size];

int fmarla[size];

int kannal[size];

int total\_plots[size];

int most\_sold\_area[size];

int most\_sold\_marla[size];

// main body of project

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

int main()

{

admin\_adduser\_load();

s = admin\_leads\_load();

s1 = admin\_leads\_load1();

admin\_inv\_load();

u = user\_addsale\_load();

user\_schedule\_load();

user\_schedule\_load1();

t = admin\_project\_load();

int op;

for (int i = 0; i < 4; i++)

{

header();

op = mainmenu();

if (op == 1)

{

header();

admin();

// admin account

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

if (email == "hammadkhan@gmail.com" && password == "hammad123")

{

int op;

for (int i = 0; i < 8; i++)

{

header();

op = adminmenu();

if (op == 1)

{

header();

adminhome();

}

if (op == 2)

{

int op;

for (int i = 0; i < 4; i++)

{

header();

op = adminleads();

if (op == 1)

{

header();

adminleads1();

}

if (op == 2)

{

header();

admintotalleads2();

}

if (op == 3)

{

header();

delet();

}

if (op == 4)

{

break;

}

}

}

if (op == 3)

{

header();

sales();

}

if (op == 4)

{

int op;

for (int i = 0; i < 4; i++)

{

header();

op = admininv();

if (op == 1)

{

header();

admininv1();

}

if (op == 2)

{

header();

admininv2();

}

if (op == 3)

{

header();

viewplotsinfo();

}

if (op == 4)

{

break;

}

}

}

if (op == 5)

{

int op;

for (int i = 0; i < 3; i++)

{

header();

op = adminprojectinfo();

if (op == 1)

{

header();

addproject();

}

if (op == 2)

{

header();

cout << "ADMIN ACCOUNT < PROJECT INFO < VIEW PROJECTS" << endl;

showprojects();

}

if (op == 3)

{

header();

deletprojects();

}

if (op == 4)

{

break;

}

}

}

if (op == 6)

{

header();

adminadduser();

}

if (op == 7)

{

break;

}

}

}

else

{

header();

worngpassword();

}

}

// user account

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

else if (op == 2)

{

header();

admin();

if (email == "ahadejaz@gmail.com" && password == "123") // U S E R 1

{

int op1;

for (int i = 0; i < 7; i++)

{

header();

op1 = usermenue();

if (op1 == 1)

{

header();

userhome();

}

if (op1 == 2)

{

int op;

header();

op = userleads11();

header();

if (op != -1)

{

useraddresponse(op);

}

}

if (op1 == 3)

{

int op;

for (int i = 1; i < 4; i++)

{

header();

op = usersales();

if (op == 1)

{

header();

useraddsale();

}

if (op == 2)

{

header();

usertotalsale();

}

if (op == 3)

{

break;

}

}

}

if (op1 == 4)

{

int op;

for (int i = 1; i < 4; i++)

{

header();

op = userinv();

if (op == 1)

{

header();

userinvmap();

}

if (op == 2)

{

header();

userinvplots();

}

if (op == 3)

{

break;

}

}

}

if (op1 == 5)

{

header();

cout << "USER ACCOUNT < PROJECTS" << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

showprojects();

}

if (op1 == 6)

{

break;

system("cls");

}

}

}

else if (email == umail1 && password == upassword1) // U S E R 2

{

int op1;

for (int i = 0; i < 7; i++)

{

header();

op1 = usermenue();

if (op1 == 1)

{

header();

userhome1();

}

if (op1 == 2)

{

int op;

header();

op = userleads1();

header();

if (op != -1)

{

useraddresponse1(op);

}

}

if (op1 == 3)

{

int op;

for (int i = 1; i < 4; i++)

{

header();

op = usersales();

if (op == 1)

{

header();

useraddsale();

}

if (op == 2)

{

header();

usertotalsale();

}

if (op == 3)

{

break;

}

}

}

if (op1 == 4)

{

int op;

for (int i = 1; i < 4; i++)

{

header();

op = userinv();

if (op == 1)

{

header();

userinvmap();

}

if (op == 2)

{

header();

userinvplots();

}

if (op == 3)

{

break;

}

}

}

if (op1 == 5)

{

header();

cout << "USER ACCOUNT < PROJECTS" << endl;

cout << "\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

showprojects();

}

if (op1 == 6)

{

break;

system("cls");

}

}

}

else

{

header();

worngpassword();

}

}

else if (op == 3)

{

break;

}

}

}

// end of body

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

// fuction declared

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



void header()

{

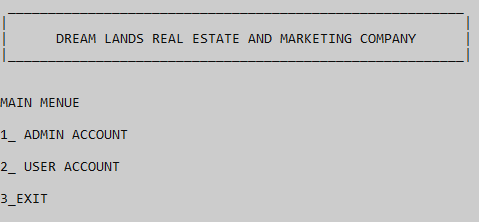
cout << " \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

cout << "| |" << endl;

cout << "| DREAM LANDS REAL ESTATE AND MARKETING COMPANY |" << endl;

cout << "|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|" << endl;

}



int mainmenu()

{

int op;

cout << endl;

cout << endl;

cout << "MAIN MENUE" << endl;

cout << endl;

cout << "1\_ ADMIN ACCOUNT" << endl;

cout << endl;

cout << "2\_ USER ACCOUNT " << endl;

cout << endl;

cout << "3\_EXIT" << endl;z

cout << endl;

cin >> op;

while (op > 3 || op < 0)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

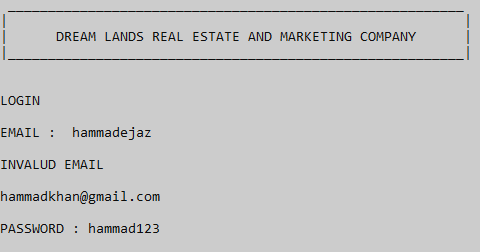
cin >> op;

}

system("cls");

return op;

}



void admin()

{

bool valid;

cout << endl;

cout << endl;

cout << "LOGIN" << endl;

char b;

cout << endl;

cout << "EMAIL : ";

cin >> email;

valid = isValidEMAIL(email);

while (valid == false)

{

cout << endl;

cout << "INVALUD EMAIL " << endl;

cout << endl;

cin >> email;

valid = isValidEMAIL(email);

}

cout << endl;

cout << "PASSWORD : ";

cin >> password;

system("cls");

}

void worngpassword()

{

char b;

cout << endl;

cout << endl;

cout << "WRONG PASSWORD" << endl;

cout << endl;

cout << "PRESS ANY KEY TO BACK";

cin >> b;

system("cls");

}



int adminmenu()

{

int op;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT" << endl;

cout << endl;

cout << endl;

cout << "1\_HOME" << endl;

cout << endl;

cout << "2\_LEADS" << endl;

cout << endl;

cout << "3\_SALES" << endl;

cout << endl;

cout << "4\_INVENTORY" << endl;

cout << endl;

cout << "5\_PROJECTS INFO" << endl;

cout << endl;

cout << "6\_ADD USERS" << endl;

cout << endl;

cout << "7\_BACK" << endl;

cout << endl;

cin >> op;

while (op > 7 || op < 0)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

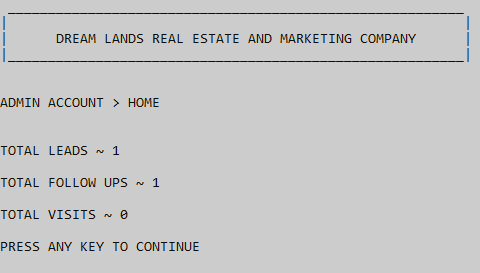
cin >> op;

}

system("cls");

return op;

}



void adminhome() // O P T I O N 1

{

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > HOME" << endl;

cout << endl;

cout << endl;

char b;

int tfollow, tvisit;

tfollow = admintotalfollowup();

tvisit = admintotalvisits();

cout << "TOTAL LEADS ~ " << s + s1<< endl;

cout << endl;

cout << "TOTAL FOLLOW UPS ~ " << tfollow << endl;

cout << endl;

cout << "TOTAL VISITS ~ " << tvisit << endl;

cout << endl;

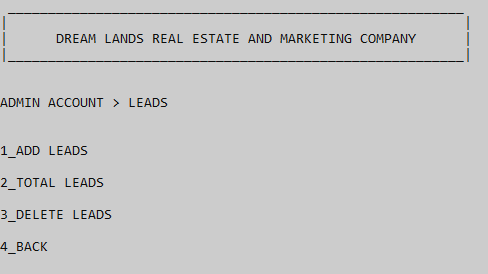
cout << "PRESS ANY KEY TO CONTINUE " << endl;

cout << endl;

cin >> b;

system("cls");

}



int adminleads() // O P T I O N 2

{

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > LEADS " << endl;

cout << endl;

cout << endl;

int op;

cout << "1\_ADD LEADS" << endl;

cout << endl;

cout << "2\_TOTAL LEADS" << endl;

cout << endl;

cout << "3\_DELETE LEADS" << endl;

cout << endl;

cout << "4\_BACK" << endl;

cin >> op;

while (op > 4 || op < 0)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

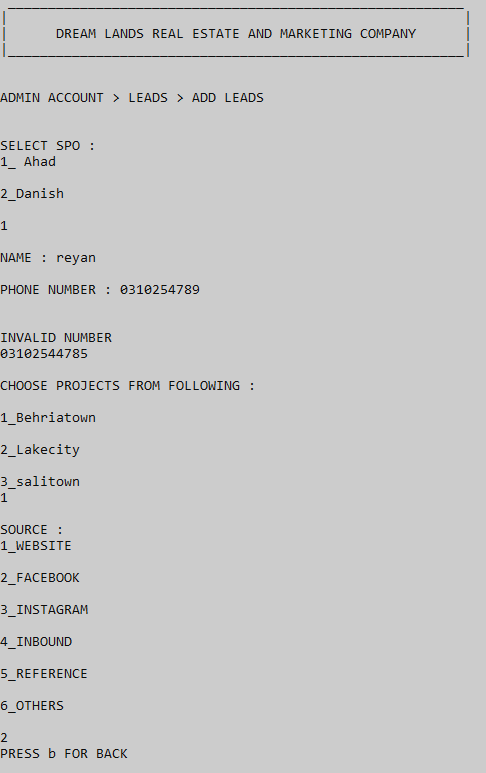
cin >> op;

}

system("cls");

return op;

}



void adminleads1()// S U B O P T I O N 1

{

int n = 0, op;

char b;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > LEADS > ADD LEADS" << endl;

cout << endl;

cout << endl;

cout << "SELECT SPO : ";

op = print\_user();

if (op == 1)

{

for (int i = s; i < size; i++) // F O R U S E R 1

{

addleads\_spo[i] = "Ahad";

cout << endl;

cout << "NAME : ";

cin >> addleads\_name[i];

cout << endl;

cout << "PHONE NUMBER : ";

cin >> addleads\_phone[i];

cout << endl;

while (addleads\_phone[i].length() != 11)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

cin >> addleads\_phone[i];

cout << endl;

}

cout << "CHOOSE PROJECTS FROM FOLLOWING : " << endl;

printprojects();

cin >> n;

while (n > t || n < 0)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

cin >> n;

}

addleads\_project[i] = projects[n - 1];

cout << endl;

cout << "SOURCE : " << endl;

source1();

cin >> n;

while (n > 6 || n < 0)

{

cout << "INVALID NUMBER " << endl;

cin >> n;

}

addleads\_source[i] = source[n - 1];

response[i] = "-";

scheduale[i] = "-";

admin\_leads\_store();

cout << "PRESS b FOR BACK ";

cin >> b;

while (b != 'b')

{

cout << "INVALID CHARACTER" << endl;

cin >> b;

}

system("cls");

if (b == 'b')

{

s++;

break;

}

if (d == 0)

{

s = s - 1;

}

}

}

if (op == 2)

{

for (int i = s1; i < size; i++) // F O R U S E R 2

{

addleads\_spo1[i] = uname1;

cout << endl;

cout << "NAME : ";

cin >> addleads\_name1[i];

cout << endl;

cout << "PHONE NUMBER : ";

cin >> addleads\_phone1[i];

cout << endl;

while (addleads\_phone1[i].length() != 11)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

cin >> addleads\_phone1[i];

}

cout << "CHOOSE PROJECTS FROM FOLLOWING : " << endl;

printprojects();

cin >> n;

while (n > t || n < 0)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

cin >> n;

}

addleads\_project1[i] = projects[n - 1];

cout << endl;

cout << "SOURCE : " << endl;

source1();

cin >> n;

while (n > 6 || n < 0)

{

cout << "INVALID NUMBER " << endl;

cin >> n;

}

addleads\_source1[i] = source[n - 1];

response1[i] = "-";

scheduale1[i] = "-";

admin\_leads\_store1();

cout << "PRESS b FOR BACK ";

cin >> b;

while (b != 'b')

{

cout << "INVALID CHARACTER" << endl;

cin >> b;

}

system("cls");

if (b == 'b')

{

s1++;

break;

}

if (d == 0)

{

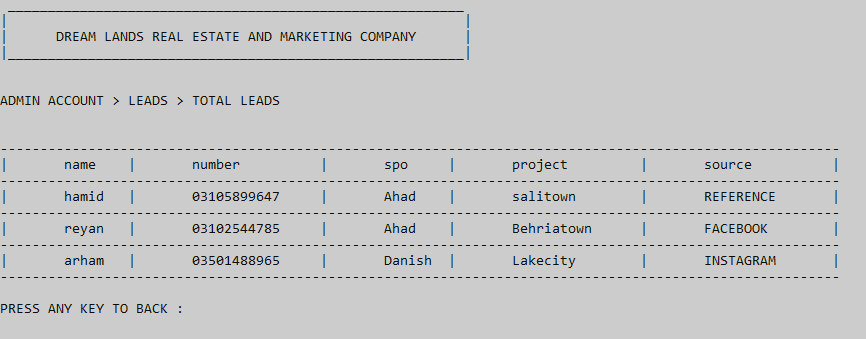
s1 = s1 - 1;

}

}

}

}



void admintotalleads2() // S U B O P T I O N 2

{

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > LEADS > TOTAL LEADS" << endl;

cout << endl;

cout << endl;

string name, spo, pro, sour, num;

char b;

cout << "---------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "name" << "\t" << "|" << "\t" << "number" << "\t\t" << "|" << "\t" << "spo" << "\t" << "|" << "\t" << "project" << "\t\t" << "|" << "\t" << "source" << "\t\t" << "|" << endl;

cout << "---------------------------------------------------------------------------------------------------------" << endl;

if (s + s1 < 0)

{

cout << "NO RECORD HAVE TO SHOW " << endl;

}

else

{

for (int i = 0; i < s; i++) // F O R U S E R 1

{

name = addleads\_name[i];

spo = addleads\_spo[i];

num = addleads\_phone[i];

pro = addleads\_project[i];

sour = addleads\_source[i];

cout << "|" << "\t" << name << "\t" << "|" << "\t" << num << "\t" << "|" << "\t" << spo << "\t" << "|" << "\t" << pro << "\t" << "|" << "\t" << sour << "\t" << "|" << endl;

cout << "---------------------------------------------------------------------------------------------------------" << endl;

}

for (int j = 0; j < s1; j++)// F O R U S E R 2

{

name = addleads\_name1[j];

spo = addleads\_spo1[j];

num = addleads\_phone1[j];

pro = addleads\_project1[j];

sour = addleads\_source1[j];

cout << "|" << "\t" << name << "\t" << "|" << "\t" << num << "\t" << "|" << "\t" << spo << "\t" << "|" << "\t" << pro << "\t" << "|" << "\t" << sour << "\t" << "|" << endl;

cout << "---------------------------------------------------------------------------------------------------------" << endl;

}

}

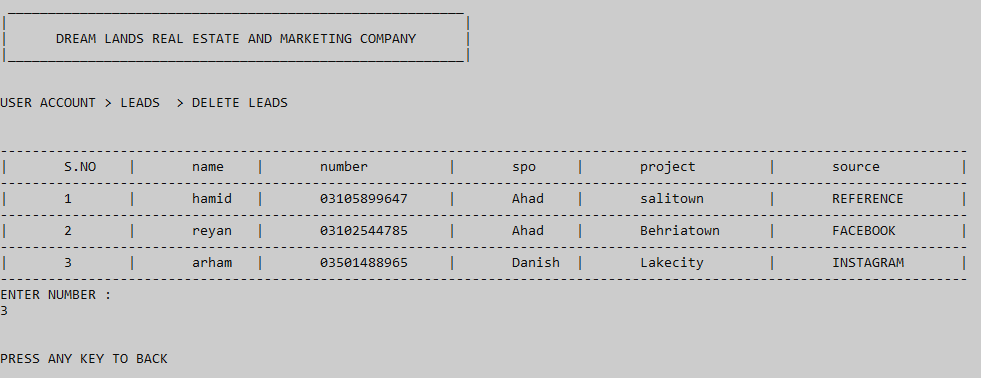
cout << endl;

cout << "PRESS ANY KEY TO BACK : " << endl;

cin >> b;

system("cls");

}



void delet() // S U B O P T I O N 3

{

int o;

char b;

cout << endl;

cout << endl;

cout << "USER ACCOUNT > LEADS > DELETE LEADS" << endl;

cout << endl;

cout << endl;

string name, spo, pro, sour, num;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "S.NO" << "\t" << "|" << "\t" << "name" << "\t" << "|" << "\t" << "number" << "\t\t" << "|" << "\t" << "spo" << "\t" << "|" << "\t" << "project" << "\t\t" << "|" << "\t" << "source" << "\t\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

if (s < 0)

{

cout << endl;

cout << "NO RECORD HAVE TO SHOW " << endl;

}

else

{

for (int i = 0; i < s; i++) // D E L E T E T H E L E A D S O F U S E R 1

{

name = addleads\_name[i];

spo = addleads\_spo[i];

num = addleads\_phone[i];

pro = addleads\_project[i];

sour = addleads\_source[i];

cout << "|" << "\t" << i + 1 << "\t" << "|" << "\t" << name << "\t" << "|" << "\t" << num << "\t"<< "|" << "\t" << spo << "\t" << "|" << "\t" << pro << "\t" << "|" << "\t" << sour << "\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

}

for (int i = 0; i < s1; i++) // D E L E T E T H E L E A D S O F U S E R 2

{

name = addleads\_name1[i];

spo = addleads\_spo1[i];

num = addleads\_phone1[i];

pro = addleads\_project1[i];

sour = addleads\_source1[i];

cout << "|" << "\t" << s + i + 1 << "\t" << "|" << "\t" << name << "\t" << "|" << "\t" << num << "\t" << "|" << "\t" << spo << "\t" << "|" << "\t" << pro << "\t" << "|" << "\t" << sour << "\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

}

}

cout << "ENTER NUMBER : " << endl; // HERE I GET THE SERIAL NUMBER OF LEADS WHICH USER WANT TO DELETE

cin >> o;

cout << endl;

while (o > s + s1)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> o;

}

if (o < s)

{

for (int i = o; i < s; i++)

{

addleads\_name[i - 1] = addleads\_name[i];

addleads\_phone[i - 1] = addleads\_phone[i];

addleads\_spo[i - 1] = addleads\_spo[i];

addleads\_project[i - 1] = addleads\_project[i];

addleads\_source[i - 1] = addleads\_source[i];

}

s--;

admin\_leads\_store();

}

if (o > s)

{

o = o - s;

for (int i = o; i < s1; i++)

{

addleads\_name1[i - 1] = addleads\_name1[i];

addleads\_phone1[i - 1] = addleads\_phone1[i];

addleads\_spo1[i - 1] = addleads\_spo1[i];

addleads\_project1[i - 1] = addleads\_project1[i];

addleads\_source1[i - 1] = addleads\_source1[i];

}

s1--;

admin\_leads\_store1();

}

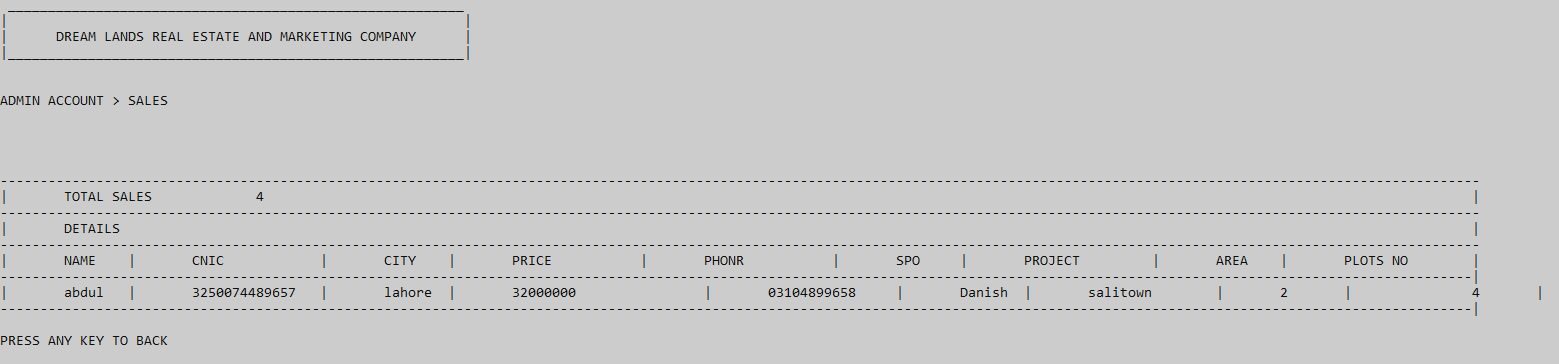
cout << endl;

cout << "PRESS ANY KEY TO BACK";

cin >> b;

system("cls");

}



void sales() // O P T I O N 3

{

int total;

char b;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > SALES" << endl;

cout << endl;

cout << endl;

total = totalsal();

cout << endl;

cout << endl;

cout << "-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "TOTAL SALES " << total << "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t" << "|" << endl;

cout << "-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "DETAILS " << "\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t\t" << "|" << endl;

sorting\_sales\_data();

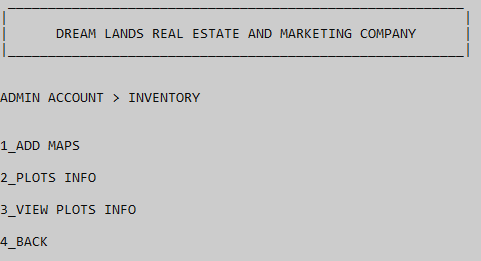
cout << endl;

cout << "PRESS ANY KEY TO BACK" << endl;

cin >> b;

system("cls");

}



int admininv() // O P T I O N 4

{

int op;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > INVENTORY" << endl;

cout << endl;

cout << endl;

cout << "1\_ADD MAPS" << endl;

cout << endl;

cout << "2\_PLOTS INFO" << endl;

cout << endl;

cout << "3\_VIEW PLOTS INFO" << endl;

cout << endl;

cout << "4\_BACK" << endl;

cout << endl;

cin >> op;

while (op > 4 || op < 0)

{

cout << endl;

cout << "INVALID NUMBER " << endl;

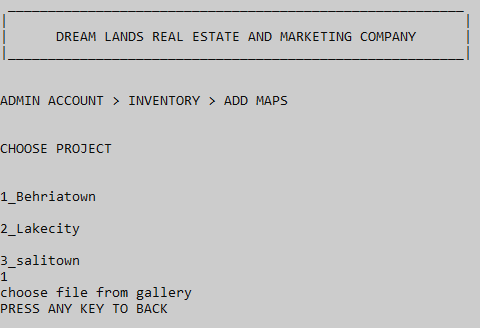
cin >> op;

}

system("cls");

return op;

}



void admininv1() // S U B O P T I O N 1

{

char b;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > INVENTORY > ADD MAPS" << endl;

cout << endl;

cout << endl;

cout << "CHOOSE PROJECT " << endl;

cout << endl;

printprojects();

cin >> n;

while (n > t || n < 0)

{

cout << "INVALID NUMBER" << endl;

cin >> n;

}

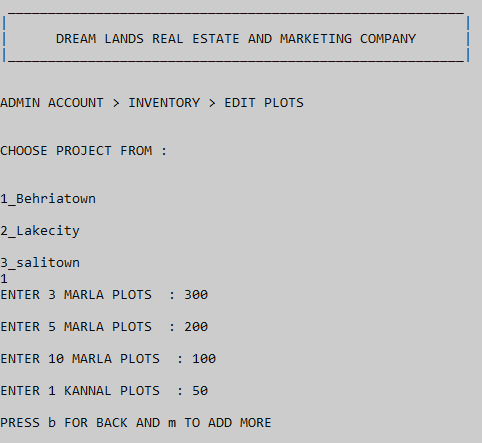
cout << "choose file from gallery" << endl; // J U S T T O S H O W T H E F E A T U R E

cout << "PRESS ANY KEY TO BACK" << endl;

cin >> b;

system("cls");

}



void admininv2() // S U B O P T I O N 2

{

char b;

int n = 0;

y = y + z;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > INVENTORY > EDIT PLOTS" << endl;

cout << endl;

cout << endl;

for (int i = y; i < size; i++)

{

cout << "CHOOSE PROJECT FROM : " << endl;

cout << endl;

printprojects();

cin >> n;

while (n > t || n < 0)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cout << endl;

cin >> n;

}

cout << "ENTER 3 MARLA PLOTS : ";

cin >> thmarla[n - 1];

cout << endl;

cout << "ENTER 5 MARLA PLOTS : ";

cin >> fmarla[n - 1];

cout << endl;

cout << "ENTER 10 MARLA PLOTS : ";

cin >> tmarla[n - 1];

cout << endl;

cout << "ENTER 1 KANNAL PLOTS : ";

cin >> kannal[n - 1];

cout << endl;

admin\_inv\_store();

cout << "PRESS b FOR BACK AND m TO ADD MORE ";

cin >> b;

while (b != 'b' && b != 'm')

{

cout << endl;

cout << "INVALID CHARACTER" << endl;

cout << endl;

cin >> b;

}

if (b == 'm')

{

z = z + 1;

continue;

}

if (b == 'b')

{

z = z + 1;

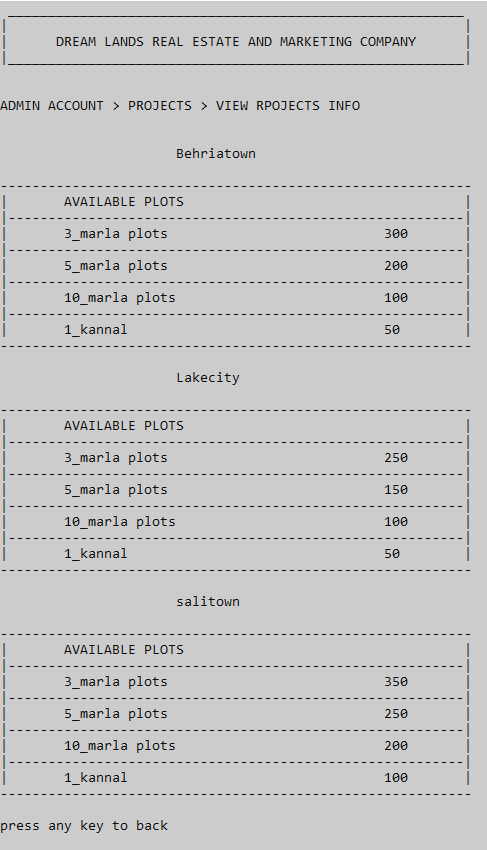
system("cls");

break;

}

}

}



void viewplotsinfo() // S U B O P T I O N 3

{

char b;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > PROJECTS > VIEW RPOJECTS INFO" << endl;

cout << endl;

cout << endl;

for (int i = 0; i < t; i++)

{

cout << " " << projects[i] << endl;

cout << endl;

cout << "-----------------------------------------------------------" << endl;

cout << "|" << "\t" << "AVAILABLE PLOTS" << " |" << endl;

cout << "|---------------------------------------------------------|" << endl;

cout << "|" << "\t" << "3\_marla plots " << "\t\t\t" << thmarla[i] << "\t" << " |" << endl;

cout << "|---------------------------------------------------------|" << endl;

cout << "|" << "\t" << "5\_marla plots " << "\t\t\t" << fmarla[i] << "\t" << " |" << endl;

cout << "|---------------------------------------------------------|" << endl;

cout << "|" << "\t" << "10\_marla plots " << "\t\t\t" << tmarla[i] << "\t" << " |" << endl;

cout << "|---------------------------------------------------------|" << endl;

cout << "|" << "\t" << "1\_kannal " << "\t\t\t\t" << kannal[i] << "\t" << " |" << endl;

cout << "-----------------------------------------------------------" << endl;

cout << endl;

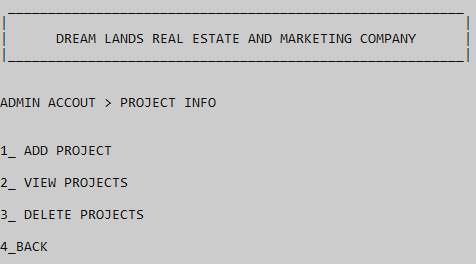
}

cout << "press any key to back " << endl;

cin >> b;

system("cls");

}



int adminprojectinfo() // O P T I O N 5

{

int op;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUT > PROJECT INFO " << endl;

cout << endl;

cout << endl;

cout << "1\_ ADD PROJECT" << endl;

cout << endl;

cout << "2\_ VIEW PROJECTS" << endl;

cout << endl;

cout << "3\_ DELETE PROJECTS" << endl;

cout << endl;

cout << "4\_BACK" << endl;

cin >> op;

cout << endl;

while (op > 4)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

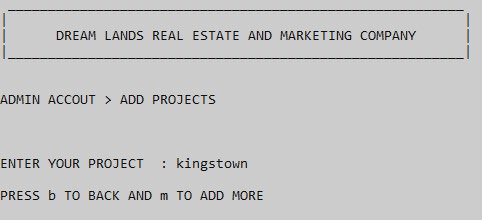
cin >> op;

}

system("cls");

return op;

}



void addproject() // S U B O P T I O N 1

{

char b;

// t = t + d;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUT > ADD PROJECTS " << endl;

cout << endl;

cout << endl;

d = 0;

for (int i = t; i < size; i++)

{

d = d + 1;

cout << endl;

cout << "ENTER YOUR PROJECT : ";

cin >> projects[i];

thmarla[i] = 0;

fmarla[i] = 0;

tmarla[i] = 0;

kannal[i] = 0;

cout << endl;

cout << "PRESS b TO BACK AND m TO ADD MORE " << endl;

cin >> b;

while (b != 'b' && b != 'm')

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> b;

};

if (b == 'm')

{

continue;

system("cls");

}

else if (b == 'b')

{

break;

system("cls");

}

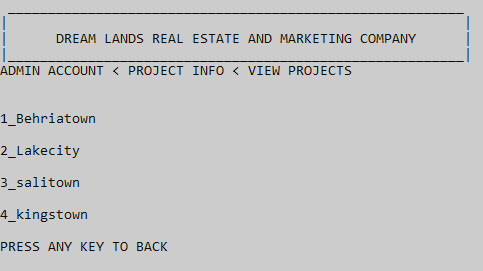
}

t = t + d;

admin\_project\_store();

system("cls");

}



void showprojects() // S U B O P T I O N 2 (A L S O U S E D I N U S E R A C C O U N T)

{

char b;

cout << endl;

cout << endl;

string name;

if (t == 0)

{

cout << "NO PROJECT IS ADDED FIRST ADD THE PROJECTS" << endl;

cout << "PRESS 0 TO CONTINUE" << endl;

}

else

{

for (int i = 0; i < t; i++)

{

name = projects[i];

cout << i + 1 << "\_" << name << endl;

cout << endl;

}

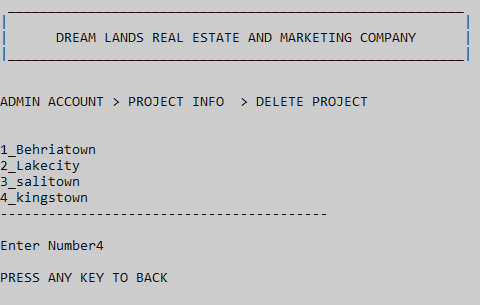
}

cout << "PRESS ANY KEY TO BACK" << endl;

cin >> b;

system("cls");

}



void deletprojects() // S U B O P T I O N 3

{

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > PROJECT INFO > DELETE PROJECT" << endl;

cout << endl;

cout << endl;

string name;

int o;

char b;

if (t == 0) // I N T H E C A S E I F N O P R O J E C T I S A D D E D

{

cout << endl;

cout << "NO PROJECT IS ADDED FIRST ADD THE PROJECTS" << endl;

}

else

{

for (int i = 0; i < t; i++)

{

name = projects[i];

cout << i + 1 << "\_" << name << endl;

}

}

cout << "-----------------------------------------" << endl;

cout << endl;

cout << "Enter Number";

cin >> o;

cout << endl;

while (o > t || o < 0)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> o;

}

for (int i = o - 1; i < t; i++)

{

projects[i] = projects[i + 1];

}

t--;

cout << "PRESS ANY KEY TO BACK" << endl;

cin >> b;

system("cls");

}



void adminadduser() // O P T I O N 3

{

// A L R E A D Y H A V E A B Y D E F A U L T U S E R B U T

// Y O U C A N A D D A N O T H E R O N E

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > ADD USERS" << endl;

cout << endl;

cout << endl;

bool valid;

char b;

cout << "ENTER NAME : ";

cin >> uname1;

cout << endl;

cout << "ENTER EMAIL : ";

cin >> umail1;

cout << endl;

valid = isValidEMAIL(umail1);

while (valid == false)

{

cout << endl;

cout << "INVALID EMAIL " << endl;

cin >> umail1;

valid = isValidEMAIL(umail1);

}

cout << "ENTER CATAGORY" << endl;

cin >> ucat1;

cout << endl;

cout << "ENTER PHONE NUMBER : ";

cin >> uphone1;

cout << endl;

while (uphone1.length() != 11)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> uphone1;

}

cout << "ENTER PASSWORD : ";

cin >> upassword1;

cout << endl;

admin\_adduser\_store();

cout << "PRESS ANY KEY TO BACK";

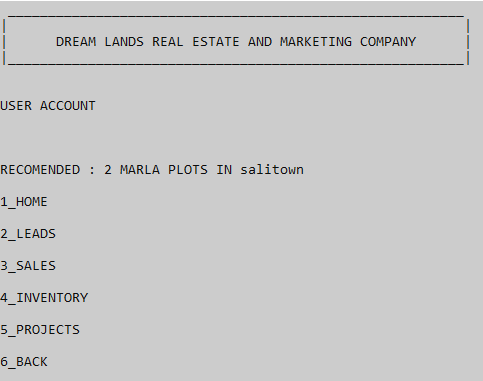
cout << endl;

cin >> b;

system("cls");

}

// U S E R S F U N C T I O N



int usermenue() // S A M E F O R B O T H U S E R 1 A N D U S E R 2

{

int op;

cout << endl;

cout << endl;

cout << "USER ACCOUNT" << endl;

cout << endl;

cout << endl;

recomendation();

cout << "1\_HOME" << endl;

cout << endl;

cout << "2\_LEADS" << endl;

cout << endl;

cout << "3\_SALES" << endl;

cout << endl;

cout << "4\_INVENTORY" << endl;

cout << endl;

cout << "5\_PROJECTS" << endl;

cout << endl;

cout << "6\_BACK" << endl;

cout << endl;

cin >> op;

cout << endl;

while (op > 6 || op < 0)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

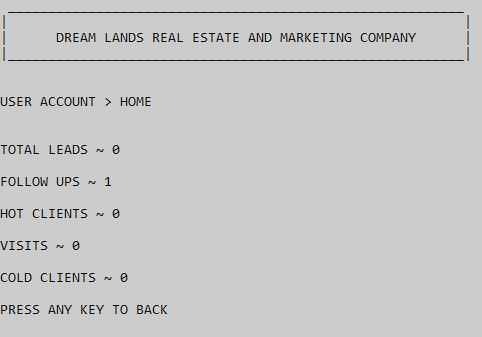
cin >> op;

}

system("cls");

return op;

}



void userhome() // O P T I O N 1 (F O R U S E R 1)

{

char b;

int totalvisit, totalhot, totalcoldclients, totalfollow;

totalvisit = totalvisits();

totalhot = totalhotclients();

totalfollow = totalfollowup();

totalcoldclients = totalcold();

cout << endl;

cout << endl;

cout << "USER ACCOUNT > HOME" << endl;

cout << endl;

cout << endl;

cout << "TOTAL LEADS ~ " << s << endl;

cout << endl;

cout << "FOLLOW UPS ~ " << totalfollow << endl;

cout << endl;

cout << "HOT CLIENTS ~ " << totalhot << endl;

cout << endl;

cout << "VISITS ~ " << totalvisit << endl;

cout << endl;

cout << "COLD CLIENTS ~ " << totalcoldclients << endl;

cout << endl;

cout << "PRESS ANY KEY TO BACK" << endl;

cout << endl;

cin >> b;

system("cls");

}

void userhome1()// O P T I O N 1 (F O R U S E R 2)

{

char b;

int totalvisit, totalhot, totalcoldclients, totalfollow;

totalvisit = totalvisits1();

totalhot = totalhotclients1();

totalfollow = totalfollowup1();

totalcoldclients = totalcold1();

cout << endl;

cout << endl;

cout << "USER ACCOUNT > HOME" << endl;

cout << endl;

cout << endl;

cout << "TOTAL LEADS ~ " << s1 << endl;

cout << endl;

cout << "FOLLOW UPS ~ " << totalfollow << endl;

cout << endl;

cout << "HOT CLIENTS ~ " << totalhot << endl;

cout << endl;

cout << "VISITS ~ " << totalvisit << endl;

cout << endl;

cout << "COLD CLIENTS ~ " << totalcoldclients << endl;

cout << endl;

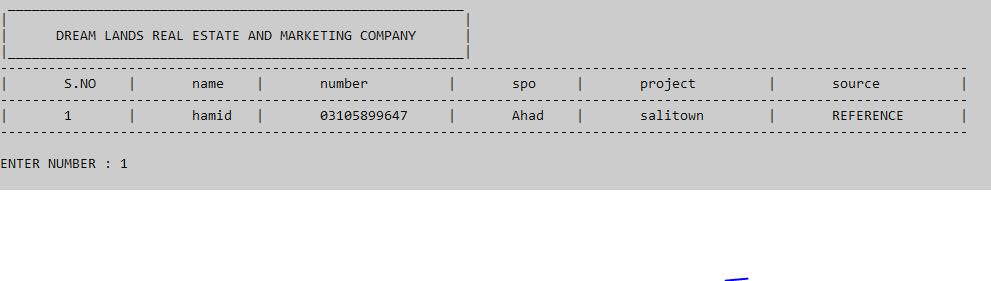
cout << "PRESS ANY KEY TO BACK" << endl;

cout << endl;

cin >> b;

system("cls");

}



int userleads11()// O P T I O N 2 (F O R U S E R 1)

{

int op = -1;

int o;

string name, spo, pro, sour, num;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "S.NO" << "\t" << "|" << "\t" << "name"<< "\t" << "|" << "\t" << "number" << "\t\t" << "|" << "\t" << "spo" << "\t" << "|" << "\t" << "project" << "\t\t" << "|" << "\t" << "source" << "\t\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

if (s < 0)

{

cout << "NO RECORD HAVE TO SHOW ENTER 0" << endl;

}

else

{

for (int i = 0; i < s1 + 1; i++)

{

name = addleads\_name[i];

spo = addleads\_spo[i];

num = addleads\_phone[i];

pro = addleads\_project[i];

sour = addleads\_source[i];

cout << "|" << "\t" << i + 1 << "\t" << "|" << "\t" << name << "\t" << "|" << "\t" << num << "\t" << "|" << "\t" << spo << "\t" << "|" << "\t" << pro << "\t" << "|" << "\t" << sour << "\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

}

}

cout << endl;

cout << "ENTER NUMBER : ";

cin >> o;

while (o > s || o < 0)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> o;

}

system("cls");

return op + o;

}

int userleads()// O P T I O N 2 (F O R U S E R 2)

{

int op = -1;

int o;

string name, spo, pro, sour, num;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "S.NO" << "\t" << "|" << "\t" << "name" << "\t" << "|" << "\t" << "number" << "\t\t" << "|" << "\t" << "spo" << "\t" << "|" << "\t" << "project" << "\t\t" << "|" << "\t" << "source" << "\t\t"<< "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

if (s < 0)

{

cout << "NO RECORD HAVE TO SHOW ENTER 0" << endl;

}

else

{

for (int i = 0; i < s +1; i++)

{

name = addleads\_name[i];

spo = addleads\_spo[i];

num = addleads\_phone[i];

pro = addleads\_project[i];

sour = addleads\_source[i];

cout << "|" << "\t" << i + 1 << "\t" << "|" << "\t" << name << "\t" << "|" << "\t" << num << "\t" << "|" << "\t" << spo << "\t" << "|" << "\t" << pro << "\t" << "|" << "\t" << sour << "\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

}

}

cout << endl;

cout << "ENTER NUMBER : ";

cin >> o;

while (o > s || o < 0)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> o;

}

system("cls");

return op + o;

}



void useraddresponse(int op)//S U B O P T I O N 1 (F O R U S E R 1)

{

char b;

int n;

n = op;

cout << endl;

cout << endl;

cout << "USER ACCOUNT > LEADS" << endl;

cout << endl;

cout << endl;

cout << "NAME : " << addleads\_name[n] << endl;

cout << endl;

cout << "PHONE NO : " << addleads\_phone[n] << endl;

cout << endl;

cout << "PROJECT : " << addleads\_project[n] << endl;

cout << endl;

cout << "SOURCE : " << addleads\_source[n] << endl;

cout << endl;

if (response[n] != "")

{

cout << endl;

cout << "YOUR PREVIOUS RESPONSE : " << endl;

cout << endl;

cout << "\t\t" << response[n] << endl;

}

cout << endl;

cout << "ENTER YOUR NEW RESPONSE : " << endl;

cout << endl;

cin.get();

getline(cin, response[n]);

cout << endl;

cout << "ENTER SCHEDULE : " << endl;

cout << endl;

cout << "\*\_ followup" << endl;

cout << endl;

cout << "\*\_ visit" << endl;

cout << endl;

cout << "\*\_ hotclient" << endl;

cout << endl;

cout << "\*\_ coldclient" << endl;

cin >> scheduale[n];

while (scheduale[n] != "followup" && scheduale[n] != "visit" && scheduale[n] != "hotclient" && scheduale[n] != "coldclient")

{

cout << " enter invalid schedule " << endl;

cin >> scheduale[n];

}

user\_schedule\_store();

cout << endl;

cout << "PRESS ANY KEY TO BACK" << endl;

cin >> b;

system("cls");

}

void useraddresponse1(int op)//S U B O P T I O N 1 (F O R U S E R 2)

{

char b;

int n;

n = op;

cout << endl;

cout << endl;

cout << "USER ACCOUNT > LEADS" << endl;

cout << endl;

cout << endl;

cout << "NAME : " << addleads\_name1[n] << endl;

cout << endl;

cout << "PHONE NO : " << addleads\_phone1[n] << endl;

cout << endl;

cout << "PROJECT : " << addleads\_project1[n] << endl;

cout << endl;

cout << "SOURCE : " << addleads\_source1[n] << endl;

cout << endl;

if (response1[n] != "")

{

cout << endl;

cout << "YOUR PREVIOUS RESPONSE : " << endl;

cout << endl;

cout << "\t\t" << response1[n] << endl;

}

cout << endl;

cout << "ENTER YOUR NEW RESPONSE : " << endl;

cout << endl;

cin.get();

getline(cin, response1[n]);

cout << endl;

cout << "ENTER SCHEDULE : " << endl;

cout << endl;

cout << "\*\_ followup" << endl;

cout << endl;

cout << "\*\_ visit" << endl;

cout << endl;

cout << "\*\_ hotclient" << endl;

cout << endl;

cout << "\*\_ coldclient" << endl;

cin >> scheduale1[n];

while (scheduale1[n] != "followup" && scheduale1[n] != "visit" && scheduale1[n] != "hotclient" && scheduale1[n] != "coldclient")

{

cout << " enter invalid schedule " << endl;

cin >> scheduale1[n];

}

user\_schedule\_store1();

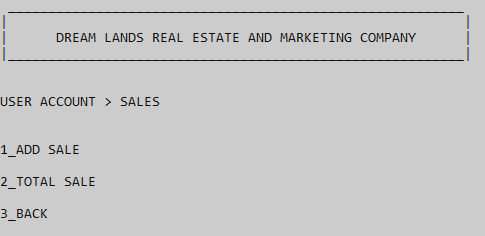
cout << endl;

cout << "PRESS ANY KEY TO BACK" << endl;

cin >> b;

system("cls");

}



int usersales() // O P T I O N 3 S A M E F O R B O T H U S E R 1 A N D 2

{

int op;

cout << endl;

cout << endl;

cout << "USER ACCOUNT > SALES" << endl;

cout << endl;

cout << endl;

cout << "1\_ADD SALE" << endl;

cout << endl;

cout << "2\_TOTAL SALE" << endl;

cout << endl;

cout << "3\_BACK" << endl;

cout << endl;

cin >> op;

while (op > 3 || op < 0)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> op;

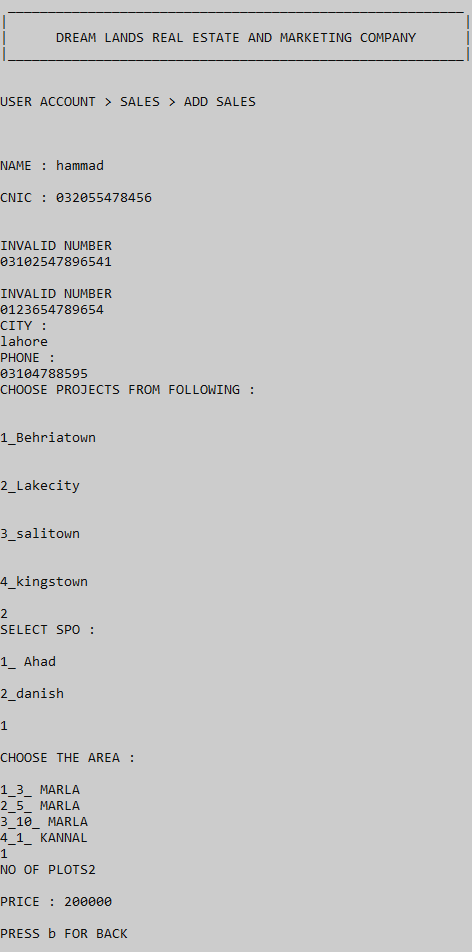
}

countu = countu + m;

system("cls");

return op;

}



void useraddsale()// S U B O P T I O N 1 S A M E F O R B O T H U S E R 1 A N D 2

{

int n = 0, op;

char b;

cout << endl;

cout << endl;

cout << "USER ACCOUNT > SALES > ADD SALES" << endl;

cout << endl;

cout << endl;

for (int i = u; i < size; i++)

{

cout << endl;

cout << "NAME : ";

cin >> sname[i];

cout << endl;

cout << "CNIC : ";

cin >> scnic[i];

cout << endl;

while (scnic[i].length() != 13)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> scnic[i];

}

cout << "CITY : ";

cout << endl;

cin >> scity[i];

cout << "PHONE : ";

cout << endl;

cin >> sphone[i];

while (sphone[i].length() != 11)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> sphone[i];

}

cout << "CHOOSE PROJECTS FROM FOLLOWING : " << endl;

cout << endl;

printprojects();

cin >> n;

most\_sold\_area[n - 1] = most\_sold\_area[n - 1] + 1;

while (n > t)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> n;

}

sproject[i] = addleads\_project[n - 1];

cout << "SELECT SPO : " << endl;

op = print\_user();

if (op == 1)

{

sspo[i] = "ahad";

}

if (op == 2)

{

sspo[i] = uname1;

}

cout << endl;

cout << "CHOOSE THE AREA : " << endl;

cout << endl;

printarea();

cin >> sarea[i];

if (sarea[i] == 1) // U S E D R E C O M E N D A T I O N

{

most\_sold\_marla[0] = most\_sold\_marla[0] + 1;

}

if (sarea[i] == 2)

{

most\_sold\_marla[1] = most\_sold\_marla[1] + 1;

}

if (sarea[i] == 3)

{

most\_sold\_marla[2] = most\_sold\_marla[2] + 1;

}

if (sarea[i] == 4)

{

most\_sold\_marla[3] = most\_sold\_marla[3] + 1;

}

while (sarea[i] != 1 && sarea[i] != 2 && sarea[i] != 3 && sarea[i] != 4)

{

cin >> sarea[i];

}

cout << "NO OF PLOTS";

cin >> snum[i];

cout << endl;

sold\_plots(i, n);

cout << "PRICE : ";

cin >> sprice[i];

cout << endl;

cout << "PRESS b FOR BACK";

cin >> b;

while (b != 'b')

{

cout << endl;

cout << "INVALID CHARACTER" << endl;

cin >> b;

}

if (b == 'b')

{

k = k + 1;

u = u + k;

user\_addsale\_store();

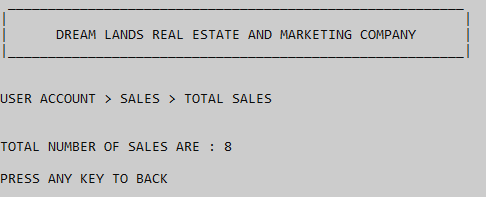
break;

}

system("cls");

}

}



void usertotalsale()// S U B O P T I O N 2 S A M E F O R B O T H U S E R 1 A N D 2

{

char b;

int totalsales;

totalsales = totalsal();

cout << endl;

cout << endl;

cout << "USER ACCOUNT > SALES > TOTAL SALES" << endl;

cout << endl;

cout << endl;

cout << "TOTAL NUMBER OF SALES ARE : " << totalsales << endl;

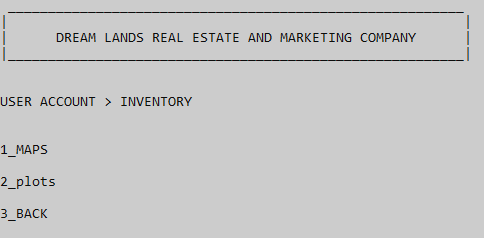
cout << endl;

cout << "PRESS ANY KEY TO BACK ";

cin >> b;

system("cls");

}



int userinv()// O P T I O N 4 S A M E F O R B O T H U S E R 1 A N D 2

{

int op;

cout << endl;

cout << endl;

cout << "USER ACCOUNT > INVENTORY" << endl;

cout << endl;

cout << endl;

cout << "1\_MAPS" << endl;

cout << endl;

cout << "2\_plots" << endl;

cout << endl;

cout << "3\_BACK" << endl;

cout << endl;

cin >> op;

while (op > 3)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> op;

}

system("cls");

return op;

}

void userinvmap()// S U B O P T I O N 1 S A M E F O R B O T H U S E R 1 A N D 2

{

char b;

cout << endl;

cout << endl;

cout << "ADMIN ACCOUNT > INVENTORY > MAPS" << endl;

cout << endl;

cout << endl;

cout << "PROJECT IS : " << realproject << endl; // just to show the maps feature i use veriable you can declared an array if you want

cout << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

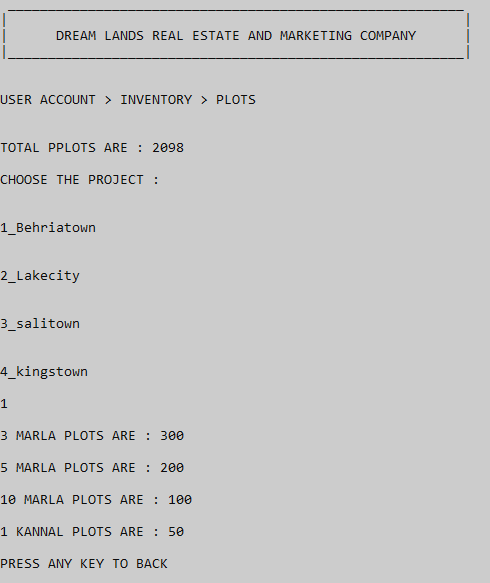
cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "PRESS ANY KEY TO BACK" << endl;

cin >> b;

system("cls");

}



void userinvplots()// S U B O P T I O N 2 S A M E F O R B O T H U S E R 1 A N D 2

{

char b;

int op;

cout << endl;

cout << endl;

cout << "USER ACCOUNT > INVENTORY > PLOTS" << endl;

cout << endl;

cout << endl;

if (t == 0)

{

cout << endl;

cout << "NO PROJECT IS ADDED FIRST ADD THE PROJECTS" << endl;

cout << endl;

cout << "PRESS ANY KEY TO CONTINUE" << endl;

cin >> b;

system("cls");

}

else

{

op = totalplo();

cout << "TOTAL PPLOTS ARE : " << op << endl;

cout << endl;

cout << "CHOOSE THE PROJECT : " << endl;

cout << endl;

printprojects();

cin >> op;

while (op > t)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> op;

}

cout << endl;

cout << "3 MARLA PLOTS ARE : " << thmarla[op - 1] << endl;

cout << endl;

cout << "5 MARLA PLOTS ARE : " << fmarla[op - 1] << endl;

cout << endl;

cout << "10 MARLA PLOTS ARE : " << tmarla[op - 1] << endl;

cout << endl;

cout << "1 KANNAL PLOTS ARE : " << kannal[op - 1] << endl;

cout << endl;

cout << "PRESS ANY KEY TO BACK";

cin >> b;

system("cls");

}

}

// E N D O F T H E U S E R F U N C T I O N S ------------------------------

// F U N C T I O N U S E D F O R C A L C U L A T I O N T Y P E S T U F F ----------------------------

int totalplo()

{

int totalplots;

int thmarla1 = 0, fmarla1 = 0, tmarla1 = 0, kannal1 = 0;

for (int i = 0; i < 4; i++)

{

thmarla1 = thmarla1 + thmarla[i];

fmarla1 = fmarla1 + fmarla[i];

tmarla1 = tmarla1 + tmarla[i];

kannal1 = kannal1 + kannal[i];

}

totalplots = (thmarla1 + fmarla1 + tmarla1 + kannal1) - soldplots;

return totalplots;

}

int totalsal()

{

int totalsales = 0;

for (int i = 0; i < u + 1; i++)

{

totalsales = totalsales + snum[i];

}

return totalsales;

}

int totalfollowup()

{

int followup = 0;

for (int i = 0; i < s; i++)

{

if (scheduale[i] == "followup")

{

followup = followup + 1;

}

}

return followup;

}

int totalvisits()

{

int visits = 0;

for (int i = 0; i < s; i++)

{

if (scheduale[i] == "visit")

{

visits = visits + 1;

}

}

return visits;

}

int totalhotclients()

{

int hotclients = 0;

for (int i = 0; i < s; i++)

{

if (scheduale[i] == "hotclient")

{

hotclients = hotclients + 1;

}

}

return hotclients;

}

int totalcold()

{

int cold = 0;

for (int i = 0; i < s; i++)

{

if (scheduale[i] == "coldclient")

{

cold = cold + 1;

}

}

return cold;

}

int totalfollowup1()

{

int followup = 0;

for (int i = 0; i < s; i++)

{

if (scheduale1[i] == "followup")

{

followup = followup + 1;

}

}

return followup;

}

int totalvisits1()

{

int visits = 0;

for (int i = 0; i < s; i++)

{

if (scheduale1[i] == "visit")

{

visits = visits + 1;

}

}

return visits;

}

int totalhotclients1()

{

int hotclients = 0;

for (int i = 0; i < s; i++)

{

if (scheduale1[i] == "hotclient")

{

hotclients = hotclients + 1;

}

}

return hotclients;

}

int totalcold1()

{

int cold = 0;

for (int i = 0; i < s; i++)

{

if (scheduale1[i] == "coldclient")

{

cold = cold + 1;

}

}

return cold;

}

void source1()

{

string name;

for (int i = 0; i < 6; i++)

{

name = source[i];

cout << i + 1 << "\_" << name << endl;

cout << endl;

}

}

int admintotalfollowup()

{

int a , b , c;

a = totalfollowup();

b = totalfollowup1();

c = a + b;

return c;

}

int admintotalvisits()

{

int a , b , c;

a = totalvisits();

b = totalvisits1();

c = a + b;

return c;

}

void printarea()

{

string name;

for (int i = 0; i < 4; i++)

{

name = area[i];

cout << i + 1 << "\_" << name << endl;

}

}

int small(int j)

{

x = j;

int a = sarea[j];

for (int i = j; i < u; i++)

{

if (sarea[i] < a)

{

a = sarea[i];

x = i;

}

}

return x;

}

void sorting\_sales\_data()

{

int l, s6, s7, s8;

string s1, s2, s3, s4, s5, s9;

for (int j = 0; j < u; j++)

{

l = small(j);

s1 = sname[j];

s2 = scnic[j];

s3 = scity[j];

s4 = sproject[j];

s5 = sspo[j];

s6 = sprice[j];

s7 = snum[j];

s8 = sarea[j];

s9 = sphone[j];

sname[j] = sname[l];

scnic[j] = scnic[l];

scity[j] = scity[l];

sproject[j] = sproject[l];

sspo[j] = sspo[l];

sprice[j] = sprice[l];

snum[j] = snum[l];

sarea[j] = sarea[l];

sphone[j] = sphone[l];

sname[l] = s1;

scnic[l] = s2;

scity[l] = s3;

sproject[l] = s4;

sspo[l] = s5;

sprice[l] = s6;

snum[l] = s7;

sarea[l] = s8;

sphone[l] = s9;

}

cout << "-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "NAME" << "\t" << "|" << "\t" << "CNIC" << "\t\t" << "|" << "\t" << "CITY" << "\t" << "|" << "\t" << "PRICE" << "\t\t" << "|" << "\t" << "PHONR" << "\t\t" << "|" << "\t" << "SPO" << "\t" << "|" << "\t" << "PROJECT" << "\t\t" << "|" << "\t" << "AREA" << "\t" << "|" << "\t" << "PLOTS NO" << "\t" << "|" << endl;

cout << "----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|" << endl;

for (int i = 0; i < u; i++)

{

cout << "|" << "\t" << sname[i] << "\t" << "|" << "\t" << scnic[i] << "\t" << "|" << "\t" << scity[i] << "\t" << "|" << "\t" << sprice[i] << "\t\t" << "|" << "\t" << sphone[i] << "\t" << "|" << "\t" << sspo[i] << "\t" << "|" << "\t" << sproject[i] << "\t" << "|" << "\t" << sarea[i] << "\t" << "|" << "\t\t" << snum[i] << "\t" << "|" << endl;

cout << "----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|" << endl;

}

}

void printprojects()

{

string name;

if (t == 0)

{

cout << endl;

cout << "NO PROJECT IS ADDED FIRST ADD THE PROJECTS" << endl;

cout << endl;

cout << "PRESS 0 TO CONTINUE" << endl;

}

else

{

for (int i = 0; i < t; i++)

{

name = projects[i];

cout << endl;

cout << i + 1 << "\_" << name << endl;

cout << endl;

}

}

}

bool isValidEMAIL(string text)

{

int atIndex = -1, dotIndex = -1;

for (int i = 0; i < text.length(); i++)

{

if (text[i] == '@')

{

atIndex = i;

break;

}

}

if (atIndex != -1)

{

for (int i = atIndex; i < text.length(); i++)

{

if (text[i] == '.')

{

dotIndex = i;

break;

}

}

}

if (atIndex != -1 && dotIndex != -1)

{

return true;

}

return false;

}

void sold\_plots(int i, int n)

{

if (sarea[i] == 1)

{

thmarla[n - 1] = thmarla[n - 1] - snum[i];

}

else if (sarea[i] == 2)

{

fmarla[n - 1] = fmarla[n - 1] - snum[i];

}

else if (sarea[i] == 3)

{

tmarla[n - 1] = tmarla[n - 1] - snum[i];

}

else if (sarea[i] == 4)

{

kannal[n - 1] = kannal[n - 1] - snum[i];

}

}

int print\_user()

{

int op;

cout << endl;

cout << "1\_ Ahad" << endl;

cout << endl;

if (uname1 != "")

{

cout << "2\_" << uname1 << endl;

cout << endl;

}

cin >> op;

return op;

}

// F U N C T I O N S T O S T O R E A N D L O A D D A T A

void admin\_inv\_store()

{

fstream file;

file.open("admininvstore.txt", ios::out);

for (int i = 0; i < t; i++)

{

file << thmarla[i] << ",";

file << fmarla[i] << ",";

file << tmarla[i] << ",";

file << kannal[i] << ",";

file << endl;

}

file.close();

}

void admin\_leads\_store()

{

fstream file;

file.open("adminleadsstore.txt", ios::out);

for (int i = 0; i < s; i++)

{

file << addleads\_name[i] << ",";

file << addleads\_phone[i] << ",";

file << addleads\_spo[i] << ",";

file << addleads\_project[i] << ",";

file << addleads\_source[i] << ",";

file << endl;

}

file.close();

}

void admin\_adduser\_store()

{

fstream file;

file.open("adminadduserstore.txt", ios::out);

file << uname1 << ",";

file << umail1 << ",";

file << ucat1 << ",";

file << uphone1 << ",";

file << upassword1 << ",";

file << endl;

file.close();

}

void user\_schedule\_store()

{

fstream file;

file.open("userschedulestore.txt", ios::out);

for (int i = 0; i < s; i++)

{

file << scheduale[i] << ",";

file << response[i] << ",";

file << endl;

}

file.close();

}

void user\_addsale\_store()

{

fstream file;

file.open("useraddsalestore.txt", ios::out);

for (int i = 0; i < u; i++)

{

file << sname[i] << ",";

file << scnic[i] << ",";

file << scity[i] << ",";

file << sphone[i] << ",";

file << sproject[i] << ",";

file << sspo[i] << ",";

file << sarea[i] << ",";

file << sprice[i] << ",";

file << snum[i] << ",";

file << endl;

}

// file << "\b";

file.close();

}

void admin\_project\_store()

{

fstream file;

file.open("adminprojectstore.txt", ios::out);

for (int i = 0; i < t; i++)

{

file << projects[i];

file << endl;

}

// file << "\b";

file.close();

}

void admin\_inv\_load()

{

fstream file;

string line;

file.open("admininvstore.txt", ios::in);

int i = 0;

while (!file.eof())

{

getline(file, line);

if (line == "")

{

break;

}

thmarla[i] = stoi(parsedata(line, 1));

fmarla[i] = stoi(parsedata(line, 2));

tmarla[i] = stoi(parsedata(line, 3));

kannal[i] = stoi(parsedata(line, 4));

i++;

}

// file << "\b";

file.close();

}

int admin\_leads\_load()

{

fstream file;

string line;

int count = 0;

file.open("adminleadsstore.txt", ios::in);

int i = 0;

while (!file.eof())

{

getline(file, line);

if (line == "")

{

break;

}

addleads\_name[i] = parsedata(line, 1);

addleads\_phone[i] = parsedata(line, 2);

addleads\_spo[i] = parsedata(line, 3);

addleads\_project[i] = parsedata(line, 4);

addleads\_source[i] = parsedata(line, 5);

i++;

count = i;

}

// file << "\b";

file.close();

return count;

}

void admin\_adduser\_load()

{

fstream file;

string line;

file.open("adminadduserstore.txt", ios::in);

int i = 0;

while (!file.eof())

{

getline(file, line);

if (line == "")

{

break;

}

uname1 = parsedata(line, 1);

umail1 = parsedata(line, 2);

ucat1 = parsedata(line, 3);

uphone1 = parsedata(line, 4);

upassword1 = parsedata(line, 5);

i++;

}

// file << "\b";

file.close();

}

void user\_schedule\_load()

{

fstream file;

string line;

file.open("userschedulestore.txt", ios::in);

int i = 0;

while (!file.eof())

{

getline(file, line);

if (line == "")

{

break;

}

scheduale[i] = parsedata(line, 1);

response[i] = parsedata(line, 2);

i++;

}

// file << "\b";

file.close();

}

int user\_addsale\_load()

{

int count = 0;

fstream file;

string line;

file.open("useraddsalestore.txt", ios::in);

int i = 0;

while (!file.eof())

{

getline(file, line);

if (line == "")

{

break;

}

sname[i] = parsedata(line, 1);

scnic[i] = parsedata(line, 2);

scity[i] = parsedata(line, 3);

sphone[i] = parsedata(line, 4);

sproject[i] = parsedata(line, 5);

sspo[i] = parsedata(line, 6);

sarea[i] = stoi(parsedata(line, 7));

sprice[i] = stoi(parsedata(line, 8));

snum[i] = stoi(parsedata(line, 9));

i++;

count = i;

}

// file << "\b";

file.close();

return count;

}

int admin\_project\_load()

{

int count = 0;

fstream file;

string line;

file.open("adminprojectstore.txt", ios::in);

while (!file.eof())

{

line = "";

file >> line;

if (line == "")

{

break;

}

projects[count] = line;

count++;

}

// file << "\b";

file.close();

return count;

}

void admin\_leads\_store1()

{

fstream file;

file.open("adminleadsstore1.txt", ios::out);

for (int i = 0; i < s1; i++)

{

file << addleads\_name1[i] << ",";

file << addleads\_phone1[i] << ",";

file << addleads\_spo1[i] << ",";

file << addleads\_project1[i] << ",";

file << addleads\_source1[i] << ",";

file << endl;

}

// file << "\b";

file.close();

}

void user\_schedule\_store1()

{

fstream file;

file.open("userschedulestore1.txt", ios::out);

for (int i = 0; i < s; i++)

{

file << scheduale1[i] << ",";

file << response1[i] << ",";

file << endl;

}

// file << "\b";

file.close();

}

void user\_schedule\_load1()

{

fstream file;

string line;

file.open("userschedulestore1.txt", ios::in);

int i = 0;

while (!file.eof())

{

getline(file, line);

if (line == "")

{

break;

}

scheduale1[i] = parsedata(line, 1);

response1[i] = parsedata(line, 2);

i++;

}

// file << "\b";

file.close();

}

int admin\_leads\_load1()

{

fstream file;

string line;

int count = 0;

file.open("adminleadsstore1.txt", ios::in);

int i = 0;

while (!file.eof())

{

getline(file, line);

if (line == "")

{

break;

}

addleads\_name1[i] = parsedata(line, 1);

addleads\_phone1[i] = parsedata(line, 2);

addleads\_spo1[i] = parsedata(line, 3);

addleads\_project1[i] = parsedata(line, 4);

addleads\_source1[i] = parsedata(line, 5);

i++;

count = i;

}

// file << "\b";

file.close();

return count;

}

string parsedata(string a, int f)

{

int comma = 1;

string item = "";

for (int i = 0; a[i] != '\0'; i++)

{

if (a[i] == ',')

{

comma++;

}

else if (comma == f)

{

item = item + a[i];

}

}

return item;

}

int userleads1()

{

int op = -1;

int o;

string name, spo, pro, sour, num;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

cout << "|" << "\t" << "S.NO" << "\t" << "|" << "\t" << "name" << "\t" << "|" << "\t" << "number" << "\t\t" << "|" << "\t" << "spo" << "\t" << "|" << "\t" << "project" << "\t\t" << "|" << "\t" << "source" << "\t\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

if (s < 0)

{

cout << "NO RECORD HAVE TO SHOW ENTER 0" << endl;

}

else

{

for (int i = 0; i < s1; i++)

{

name = addleads\_name1[i];

spo = addleads\_spo1[i];

num = addleads\_phone1[i];

pro = addleads\_project1[i];

sour = addleads\_source1[i];

cout << "|" << "\t" << i + 1 << "\t" << "|" << "\t" << name << "\t" << "|" << "\t" << num << "\t" << "|" << "\t" << spo << "\t" << "|" << "\t" << pro << "\t" << "|" << "\t" << sour << "\t" << "|" << endl;

cout << "-------------------------------------------------------------------------------------------------------------------------" << endl;

}

}

cout << endl;

cout << "ENTER NUMBER : ";

cin >> o;

while (o > s || o < 0)

{

cout << endl;

cout << "INVALID NUMBER" << endl;

cin >> o;

}

system("cls");

return op + o;

}

// F O R R E C O M E N D A T I O N

int mostlargearea()

{

x = 0;

int a = most\_sold\_area[0];

for (int i = 0; i < t; i++)

{

if (most\_sold\_area[i] < a)

{

a = sarea[i];

x = i;

}

}

return x;

}

int mostsoldmarla()

{

x = 0;

int a = most\_sold\_marla[0];

for (int i = 4; i < 4; i++)

{

if (most\_sold\_marla[i] < a)

{

a = sarea[i];

x = i;

}

}

return x;

}

void recomendation()

{

int a, b;

a = mostlargearea();

b = mostsoldmarla();

cout << endl;

cout << "RECOMENDED : " << sarea[b] << " MARLA PLOTS IN " << sproject[a] << endl;

cout << endl;

}

**Weakness:**

* The program cannot handle products more than Size.
* It does not allow us to upload maps.
* It does not have any functionality for date and time.

**Conclusion:**

Through this project, I learnt about file handling and concepts of loops.