DAV UNIVERSITY

JALANDHAR

PROJECT
OF
OBJECT ORIENTED PROGRAMMING
(CSE201)

SUBMITTED TO:-
MISS. BALRAJPREET KAUR

SUBMITTED BY:-
PAARTH GILHOTRA
12201114
3rd SEMESTER(B08B)

TABLE OF CONTENTS

AKNOWLEDGEMENT

Apart from the efforts of me, the success of any project depends largely on the encouragement and guidelines of many others, I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I express my heartfelt gratitude to my parents for constant encouragement while carrying out this project.

I would like to express a deep sense of thanks & gratitude to my guide teacher **MISS.BALRAJPREET KAUR** for guiding me immensely through the course of the project

Last but not the least I would to thanks all those who had helped directly and indirectly towards the completion of this project.

BANK MANAGEMENT SYSTEM

INTRODUCTION

Bank management is a critical aspect of the financial industry, where the efficient handling of various banking operations is of utmost importance. To streamline these operations and provide better services to customers, the development of a robust and reliable Bank Management System (BMS) is essential. In this project, we will focus on creating a Bank Management System using the C++ programming language, aiming to improve the efficiency, accuracy, and security of banking operations.

The Bank Management System project is designed to address the complexities involved in managing a bank's day-to-day operations, including customer accounts, transactions, and administrative tasks like update and check the details of the existing account. It provides an efficient platform for both customers and bank employees to access and manage their accounts, ensuring data accuracy and security while offering a user-friendly interface.

The Bank Management System project is designed to address the complexities involved in managing a bank's day-to-day operations, including customer accounts, transactions, and administrative tasks like update and check the details of the existing account. It provides an efficient platform for both customers and bank employees to access and manage their accounts, ensuring data accuracy and security while offering a user-friendly interface.

DISADVANTAGES OF THE MANUAL

- C++ programs may be limited to specific platforms or operating systems, potentially reducing accessibility for a broader user base.
- High Development and Maintenance Costs:

  Developing and maintaining a BMS in C++ can be costly and time-consuming, requiring skilled programmers and frequent updates.
- Reduced Portability:

  C++ programs may be less portable across different platforms and environments, limiting adaptability and integration with other applications.
- Increased Code Complexity:

  C++ code can be complex and challenging to maintain, leading to more frequent errors and bugs.
- Hardware Dependencies:

  C++ programs may require specific hardware configurations to run efficiently, leading to additional hardware costs.
- Steeper Learning Curve:

  C++ is known for its complexity and steep learning curve, necessitating extensive training for employees.

ADVANTAGES

- Efficiency:
  C++ is a high-performance language, allowing for the development of a fast and responsive system. This ensures quick response times for customer transactions and administrative tasks, improving overall efficiency.
- Robustness:
  C++ is known for its strong type-checking and memory management, reducing the likelihood of errors and system crashes. This robustness is crucial in maintaining the stability of banking operations.
- Data Security:
  C++ enables the implementation of strong security measures to protect sensitive customer data. This language provides low-level control over memory and data structures, making it possible to build a secure system.
- Real-time Processing:
  C++ is well-suited for real-time applications. This capability is essential for processing financial transactions instantly and providing customers with up-to-date account information.
- Reliability:
  C++ is known for its reliability, ensuring that the BMS functions consistently and accurately, which is crucial in the banking sector to prevent errors and data discrepancies.
- Low-level Control:
  C++ provides low-level memory access and control, allowing developers to optimize system performance and resource management, which is critical in high-transaction environments.

PROGRAM CODE

```
#include<iostream>
#include<windows.h>
#include<conio.h>
using namespace std;
class Bank{
        private:
                string username,password;
                string username1,password1;
                int total;
                struct person{
                        string name,email,ID;
                        int contact,cash;
                }person[100];
        public:
                Bank(){
                        total=0;
                }
                void choice();
                void start();
                void signUp();
                void login();
                void Data();
                void show();
                void update();
                void search();
                void transactions();
                void del();
};
Void Bank::choice(){
        char ch;
        while(true){
        cout<<"\n\nPRESS..!!"<<endl;
        cout<<"1. Create new account"<<endl;
        cout<<"2. View customers list"<<endl;
```

```
cout<<"3. Update information of existing account"<<endl;
cout<<"4. Check the details of an existing account"<<endl;
cout<<"5. For transactions"<<endl;
cout<<"6. Remove existing account"<<endl;
cout<<"7. Exit"<<endl;
ch=getch();
system("ClS");
switch(ch){
        case '1':
                Bank::Data();
                break;
        case '2':
                Bank::show();
                break;
        case '3':
                Bank::update();
                break;
        case '4':
               Bank::search();
                break;
        case '5':
                Bank::transactions();
                break;
        case '6':
                Bank::del();
                break;
        case '7':
                exit(0);
                break;
        default:
          cout<<"\aInvalid input"<<endl;
          break;
          }
   }
}
void Bank::start(){
char arr[]={'B','A','N','K',' ','M','A','N','A','G','E','M','E'
,'N','T',' ','S','Y','S','T','E','M'};
cout<<"\n\n\n\n\n\t\t\t";
for(int i=0;i<24;i++){
cout<<arr[i];
Sleep(30);
}
Sleep(1000);
system("CLS");
}
void Bank::signUp(){
cout<<"\n\n\t\tBank Management System"<<endl;
cout<<"\n\t\t  SIGN UP"<<endl;
cout<<"\t\tEnter Username: ";
cin>>username;
cout<<"\t\tEnter Password: ";
cin>>password;
cout<<"\t\tYour new id is creating please wait";
for(int i=0;i<5;i++){
cout<<".";
Sleep(100);
}
}
```

```
 void Bank::login(){
system("CLS");
cout<<"\n\n\t\t LOGIN"<<endl;
cout<<"\t\tEnter Username: ";
cin>>username1;
cout<<"\t\tEnter Password: ";
cin>>password1;
if(username==username1&&password==password1){
system("CLS");
Bank::choice();
}
else if(username!=username1&&password==password1){
cout<<"\t\t\aInvalid username please try again";
Sleep(3000);
Bank::login();
}
else if(username==username1&&password!=password1){
cout<<"\t\t\aInvalid password please try again";
Sleep(3000);
Bank::login();
}
else{
cout<<"\t\t\aInvalid username and password please try again";
Sleep(3000);
Bank::login();
}
}
```

```
void Bank::Data(){
    cout<<"\nEnter data of person"<<endl<<endl;
    cout<<"Enter Name: ";
    cin>>person[total].name;
    cout<<"Enter ID: ";
    cin>>person[total].ID;
    cout<<"Enter Contact: ";
    cin>>person[total].contact;
    cout<<"Enter Email: ";
    cin>>person[total].email;
    cout<<"Enter Cash: ";
    cin>>person[total].cash;
    total++;
    }
    void Bank::show(){
    for(int i=0;i<total;i++){
    cout<<"\nData of person "<<i+1<<endl;
    cout<<"Name: "<<person[i].name<<endl;
    cout<<"ID: "<<person[i].ID<<endl;
    cout<<"Contact: "<<person[i].contact<<endl;
    cout<<"Email: "<<person[i].email<<endl;
    cout<<"Cash: "<<person[i].cash<<endl;
    }
    }

void Bank::update(){
    string id;
    cout<<"Enter ID of person for update"<<endl;
    cin>>id;
    for(int i=0;i<total;i++){
    if(id==person[i].ID){
    cout<<"\nPrevious data of person "<<i+1<<endl;
    cout<<"Name: "<<person[i].name<<endl;
    cout<<"ID: "<<person[i].ID<<endl;
    cout<<"Contact: "<<person[i].contact<<endl;
    cout<<"Email: "<<person[i].email<<endl;
    cout<<"Cash: "<<person[i].cash<<endl;
    cout<<"\nEnter new data"<<endl<<endl;
    cout<<"Enter Name: ";
    cin>>person[i].name;
    cout<<"Enter ID: ";
    cin>>person[i].ID;
    cout<<"Enter Contact: ";
```

```
cin>>person[i].contact;

cout<<"Enter Email: ";

cin>>person[i].email;

cout<<"Enter Cash: ";

cin>>person[i].cash;

cout<<"Your required data is updated"<<endl;

break;

}

if(i==total-1){

cout<<"No such record found"<<endl;

}

}

}void Bank::search(){

string id;

cout<<"Enter ID of person for check"<<endl;

cin>>id;

for(int i=0;i<total;i++){

if(id==person[i].ID){

cout<<"\nData of person "<<i+1<<endl;

cout<<"Name: "<<person[i].name<<endl;

cout<<"ID: "<<person[i].ID<<endl;

cout<<"Contact: "<<person[i].contact<<endl;
```

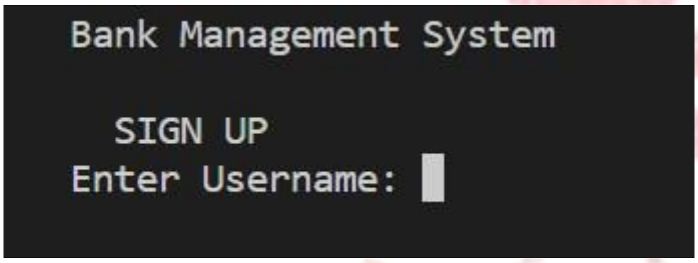
```
    cout<<"Email: "<<person[i].email<<endl;
    cout<<"Cash: "<<person[i].cash<<endl;
    break;
    }
    if(i==total-1){
    cout<<"No such record found"<<endl;
    }}}

void Bank::transactions(){
string id;
char ch;
int cash;
cout<<"Enter id of person which you want to
transaction"<<endl;
cin>>id;
for(int i=0;i<total;i++){
if(id==person[i].ID){
cout<<"Name: "<<person[i].name<<endl;
cout<<"ID: "<<person[i].ID<<endl;
cout<<"Contact: "<<person[i].contact<<endl;
cout<<"Email: "<<person[i].email<<endl;
cout<<"\nExisting Cash "<<person[i].cash<<endl;
```

```
cout<<"\nPress 1 to deposit cash"<<endl;

cout<<"Press 2 to withdraw cash"<<endl;

ch=getch();

switch(ch){

case '1':

cout<<"Enter cash which you want to
deposit"<<endl;

cin>>cash;

person[i].cash+=cash;

cout<<"New amount is "<<person[i].cash<<endl;

break;

case '2':

back:

cout<<"Enter cash which you want to
withdraw"<<endl;

cin>>cash;

if(cash>person[i].cash){

cout<<"Your existing cash is just
"<<person[i].cash<<endl;

Sleep(3000);

goto back;

}
person[i].cash-=cash;

cout<<"New amount is "<<person[i].cash<<endl;

break;

default:

cout<<"\aInvalid Input"<<endl;

break;

}

break;

}

if(i==total-1){

cout<<"No such record found"<<endl;

}}}


void Bank::del(){

if(total!=0){

char ch;

cout<<"Press 1 to remove specific record"<<endl;

cout<<"Press 2 to remove full record"<<endl;

ch=getch();

if(ch=='1'){

string id;

cout<<"Enter id of person which you want to
```

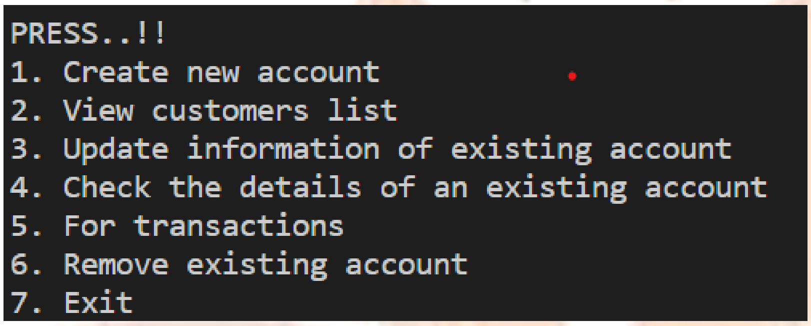
```
 remove"<<endl;


cin>>id;

for(int i=0;i<total;i++){

if(id==person[i].ID){

for(int j=i;j<total;j++){

person[j].name=person[j+1].name;

person[j].ID=person[j+1].ID;

person[j].email=person[j+1].email;

person[j].contact=person[j+1].contact;

person[j].cash=person[j+1].cash;

total--;

cout<<"Your required record is deleted"<<endl;

break;

}

}

if(i==total-1){

cout<<"No such record found"<<endl;}

}

}

else if(ch=='2'){

cout<<"All record is deleted"<<endl;
```

```
}
else{
cout<<"\aInvalid Input"<<endl;
}
}
else{
cout<<"No data is entered"<<endl;
}
}

main(){
        Bank b;
        b.start();
        b.signUp();
        b.login();
}
```

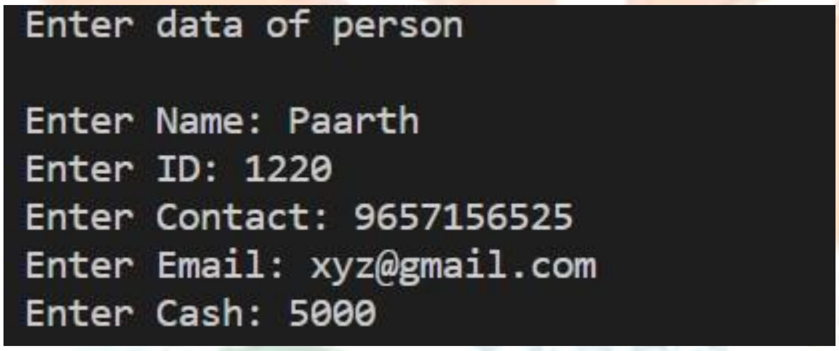
OUTPUT



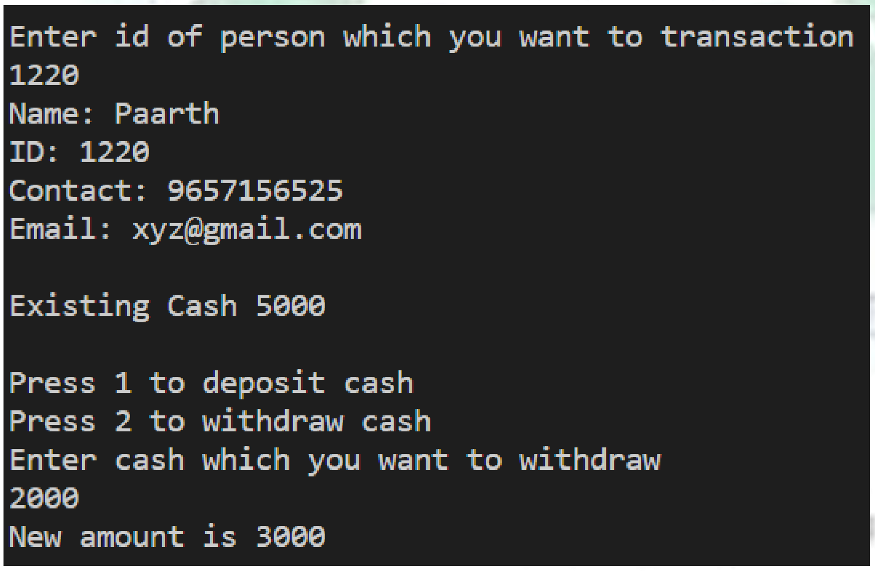
```
   Bank Management System

     SIGN UP
   Enter Username: █
```



```
PRESS..!!
1. Create new account
2. View customers list
3. Update information of existing account
4. Check the details of an existing account
5. For transactions
6. Remove existing account
7. Exit
```



```
 Enter data of person

 Enter Name: Paarth
 Enter ID: 1220
 Enter Contact: 9657156525
 Enter Email: xyz@gmail.com
 Enter Cash: 5000
```



```
Enter id of person which you want to transaction
1220
Name: Paarth
ID: 1220
Contact: 9657156525
Email: xyz@gmail.com

Existing Cash 5000

Press 1 to deposit cash
Press 2 to withdraw cash
Enter cash which you want to withdraw
2000
New amount is 3000
```