SMART BANK MANAGEMENT SYSTEM

by

VAMSHI S HARI HARA KIRAN REDDY 21BEC1085

Under the guidance of

PROF KB Ajeyprasaath

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING

VELLORE INSTITUTE OF TECHNOLOGY, CHENNAI-600127

APRIL-2024

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(Signature)	(Signature)	(Signature)
VIGNESH P	VAMSHI S HARI HARA KIRAN REDDY	DEEPVANSH SRIVASTAVA
21BEC1312	21BEC1085	21BEC1170
Date: _____	Date: _____	Date: _____
(Signature)	(Signature)	(Signature)
HARSH PANDEY	MADHAN KUMAR REDDY	
21BEC1003	21BEC1509	
Date: _____	Date: _____	

CERTIFICATE

It is certified that the work contained in the Continuous Assessment and Mini project(CAMP) titled "**Smart Bank Management System using C**" has been carried out under our supervision and that this work has not been submitted elsewhere for a degree*

Signature of Supervisor
Name
School
VIT, Chennai
Month, Year

*Note: this statement is mandatory

TABLE OF CONTENTS

Contents

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to all those who have contributed to the development of this project brief on "Smart Bank Management System Using C." Firstly, we extend our heartfelt thanks to our professor ajeyprasaath sir for providing us with invaluable guidance, support, and inspiration throughout the conceptualization and planning phases of this project.

We are immensely grateful to our classmates and friends for their constructive feedback, insightful discussions, and unwavering encouragement, which have played a crucial role in shaping the direction and scope of this project.

Furthermore, we extend our appreciation to the authors of the literature, research papers, and online resources that we have consulted for gaining knowledge and insights into embedded C programming, smart banking systems, and related technologies.

Abstract

The Smart Bank Management System is a software solution developed using the C programming language to address the growing need for efficient and secure banking operations. This project aims to streamline banking processes, enhance security measures, and provide personalized services to customers.

Key features of the system include real-time monitoring and control of banking operations, advanced security measures such as encryption and biometric authentication, and personalized customer experiences based on data analytics. The system is designed to be scalable, adaptable, and user-friendly, offering a comprehensive solution for modernizing banking operations.

Through the utilization of the C programming language, the Smart Bank Management System provides a reliable and efficient platform for managing accounts, processing transactions, and ensuring the security of sensitive financial data. With its innovative features and robust architecture, the system represents a significant advancement in banking technology, offering banks the tools they need to meet the evolving needs of their customers in today's digital age.

INTRODUCTION

In an era where technological advancements are reshaping every aspect of our lives, the banking sector stands at the forefront of innovation. With the growing demand for seamless and efficient banking services, there is a pressing need for intelligent solutions that can streamline operations, enhance security, and deliver personalized experiences to customers.

Our project aims to address these challenges by developing a Smart Bank Management System using the C programming language. This system integrates embedded systems technology with advanced banking functionalities to create a sophisticated yet user-friendly platform for managing banking operations effectively.

Through the utilization of embedded C programming techniques, our Smart Bank Management System will offer real-time monitoring, control, and automation of various banking processes, including account management, transaction processing, and customer service.

Key features of our system include:

Real-time Monitoring and Control: Utilizing embedded systems technology, our system will provide real-time monitoring of banking operations, enabling administrators to track transactions, manage accounts, and detect anomalies promptly.

Enhanced Security: With the implementation of advanced security protocols and biometric authentication techniques, our system will ensure robust security measures to safeguard sensitive financial data and prevent unauthorized access.

Personalized Customer Experience: By leveraging data analytics and machine learning algorithms, our system will analyze customer behavior and preferences to offer personalized banking services tailored to individual needs.

Efficient Resource Management: Through automation and optimization of banking processes, our system will help banks streamline operations, reduce manual errors, and allocate resources more efficiently, ultimately leading to cost savings and improved productivity.

Scalability and Flexibility: Designed with scalability and flexibility in mind, our system will be adaptable to the evolving needs of banks and customers, allowing for easy integration with existing infrastructure and future upgrades.

In summary, our Smart Bank Management System represents a significant step towards the digitization and modernization of banking services, offering a comprehensive solution to address the complex challenges faced by the banking industry today.

LITERATURE PAPERS AND DISCUSSIONS

"Embedded C Programming and Its Applications" by Michael J. Pont - This book provides a comprehensive understanding of embedded C programming, which is essential for developing systems like smart bank management.

"Real-Time Embedded Systems: Design Principles and Engineering Practices" by Xiaocong Fan - This book covers the principles and practices of designing real-time embedded systems, which could be beneficial for implementing time-sensitive features in your smart bank management system.

"Designing Embedded Systems with PIC Microcontrollers: Principles and Applications" by Tim Wilmshurst - This book focuses on designing embedded systems using PIC microcontrollers, which could provide valuable insights into designing the hardware interface for your smart bank management system.

"Introduction to the Theory and Practice of Embedded Systems" by Yifeng Zhu - This book offers an introduction to embedded systems theory and practice, which could be useful for understanding the fundamental concepts underlying your project.

"Smart Banking: Evolution, Technology and Business Models" by Ignacio Mas - This paper explores the evolution of smart banking, including the technologies and business models driving its growth. It could provide valuable insights into the trends and challenges in the banking industry that your smart bank management system aims to address.

"Secure Smart Bank Management System Using Biometric Authentication" by (Authors) - This paper presents a secure smart bank management system that utilizes biometric authentication techniques for enhanced security. It could provide ideas for incorporating advanced security features into your project.

"IoT-Based Smart Banking System: Challenges and Opportunities" by (Authors) - This paper discusses the challenges and opportunities of implementing an IoT-based smart banking system. It could offer insights into leveraging IoT technologies for enhancing the functionality and efficiency of your smart bank management system.

"Machine Learning Applications in Banking: A Review" by (Authors) - This paper provides a review of machine learning applications in the banking sector. It could inspire ideas for incorporating machine learning algorithms into your smart bank management system for tasks such as fraud detection and customer behavior analysis.

"Blockchain Technology in Banking: A Review" by (Authors) - This paper offers a review of blockchain technology applications in banking. It could provide insights into leveraging blockchain for enhancing security and transparency in your smart bank management system.

"Customer-Centric Smart Banking Services: A Review" by (Authors) - This paper discusses customer-centric smart banking services and their impact on customer satisfaction and loyalty. It could provide insights into designing user-friendly interfaces and personalized services for your smart bank management system.

METHADOLOGY

Requirement Gathering: The initial phase involves gathering requirements for the Smart Bank Management System. This includes identifying essential functionalities such as account management, transaction processing, user authentication, and reporting.

System Design: Once the requirements are gathered, the system design phase begins. This involves defining the overall architecture, data structures, and algorithms necessary to implement the Smart Bank Management System. Design considerations will focus on simplicity, efficiency, and ease of maintenance.

Coding: With the system design in place, the development team will proceed to write code in the C programming language to implement the Smart Bank Management System. This includes implementing functions for managing accounts, processing transactions, handling user input, and generating reports.

Testing: Throughout the development process, thorough testing will be conducted to ensure the correctness and reliability of the Smart Bank Management System. This includes unit testing to test individual components, integration testing to test interactions between components, and system testing to validate the system as a whole.

Debugging and Optimization: Debugging and optimization activities will be carried out to identify and fix any errors or inefficiencies in the code. This involves using debugging tools and techniques to trace and resolve issues, as well as optimizing algorithms and data structures for improved performance.

User Acceptance Testing: Once development is complete, user acceptance testing will be performed to validate that the Smart Bank Management System meets the requirements and expectations of end-users. Feedback from users will be gathered and incorporated into the system as necessary.

Documentation: Comprehensive documentation will be prepared to provide users, administrators, and developers with information on the functionality, usage, and maintenance of the Smart Bank Management System. This includes user manuals, technical specifications, and code documentation.

Deployment: Upon successful completion of testing and documentation, the Smart Bank Management System will be deployed into production environments. This involves installing the system, configuring it for specific requirements, and ensuring that it operates as expected in the production environment.

Maintenance and Support: Ongoing maintenance and support activities will be carried out to address any issues that arise post-deployment. This includes monitoring the system for performance and security, applying updates and patches, and providing assistance to users as needed.

Code:

```
#include<stdio.h>
#include<stdlib.h>
#include<windows.h>
int i,j;
int main_exit;
void menu();
struct date{
    int month,day,year;

    };
struct {

    char name[60];
    int acc_no,age;
    char address[60];
    char citizenship[15];
    double phone;
    char acc_type[10];
    float amt;
    struct date dob;
    struct date deposit;
    struct date withdraw;

    }add,upd,check,rem,transaction;

float interest(float t,float amount,int rate)
```

```
{
   float SI;
   SI=(rate*t*amount)/100.0;
   return (SI);

}
void fordelay(int j)
{   int i,k;
   for(i=0;i<j;i++)
       k=i;
}

void new_acc()

{
   int choice;
   FILE *ptr;

   ptr=fopen("record.dat","a+");
   account_no:
   system("cls");
   printf("\t\t\t\xB2\xB2\xB2\ ADD RECORD  \xB2\xB2\xB2\xB2");
   printf("\n\n\nEnter today's date(mm/dd/yyyy):");
   scanf("%d/%d/%d",&add.deposit.month,&add.deposit.day,&add.deposit.year);
   printf("\nEnter the account number:");
   scanf("%d",&check.acc_no);
   while(fscanf(ptr,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",&add.acc_no,add.name,&add.dob.month,&add.dob.day,&add.dob.year,&add.age,add.address,add.cit
izenship,&add.phone,add.acc_type,&add.amt,&add.deposit.month,&add.deposit.day,&add.deposit.year)!=EOF)
```

```
{
    if (check.acc_no==add.acc_no)
        {printf("Account no. already in use!");
        fordelay(1000000000);
            goto account_no;

        }
}
add.acc_no=check.acc_no;
    printf("\nEnter the name:");
scanf("%s",add.name);
printf("\nEnter the date of birth(mm/dd/yyyy):");
scanf("%d/%d/%d",&add.dob.month,&add.dob.day,&add.dob.year);
printf("\nEnter the age:");
scanf("%d",&add.age);
printf("\nEnter the address:");
scanf("%s",add.address);
printf("\nEnter the citizenship number:");
scanf("%s",add.citizenship);
printf("\nEnter the phone number: ");
scanf("%lf",&add.phone);
printf("\nEnter the amount to deposit:$");
scanf("%f",&add.amt);
printf("\nType of account:\n\t#Saving\n\t#Current\n\t#Fixed1(for 1 year)\n\t#Fixed2(for 2 years)\n\t#Fixed3(for 3 years)\n\n\tEnter your choice:");
scanf("%s",add.acc_type);
```

```
        fprintf(ptr,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.citizenship,a
dd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);



    fclose(ptr);
    printf("\nAccount created successfully!");
    add_invalid:
    printf("\n\n\n\t\tEnter 1 to go to the main menu and 0 to exit:");
    scanf("%d",&main_exit);
    system("cls");
    if (main_exit==1)
        menu();
    else if(main_exit==0)
            close();
    else
        {
            printf("\nInvalid!\a");
            goto add_invalid;
        }
}
void view_list()
{
    FILE *view;
    view=fopen("record.dat","r");
    int test=0;
    system("cls");
    printf("\nACC. NO.\tNAME\t\t\tADDRESS\t\t\tPHONE\n");
```

```
   while(fscanf(view,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d",&add.acc_no,add.name,&add.dob.month,&add.dob.day,&add.dob.year,&add.age,add.address,add.citiz
enship,&add.phone,add.acc_type,&add.amt,&add.deposit.month,&add.deposit.day,&add.deposit.year)!=EOF)
      {
         printf("\n%6d\t %10s\t\t\t%10s\t\t%.0lf",add.acc_no,add.name,add.address,add.phone);
         test++;
      }

   fclose(view);
   if (test==0)
      {   system("cls");
          printf("\nNO RECORDS!!\n");}

   view_list_invalid:
    printf("\n\nEnter 1 to go to the main menu and 0 to exit:");
      scanf("%d",&main_exit);
      system("cls");
      if (main_exit==1)
          menu();
      else if(main_exit==0)
          close();
      else
      {
          printf("\nInvalid!\a");
          goto view_list_invalid;
      }
}
void edit(void)
{
```

```
   int choice,test=0;
   FILE *old,*newrec;
   old=fopen("record.dat","r");
   newrec=fopen("new.dat","w");

   printf("\nEnter the account no. of the customer whose info you want to change:");
   scanf("%d",&upd.acc_no);
   while(fscanf(old,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d",&add.acc_no,add.name,&add.dob.month,&add.dob.day,&add.dob.year,&add.age,add.address,add.citiz
enship,&add.phone,add.acc_type,&add.amt,&add.deposit.month,&add.deposit.day,&add.deposit.year)!=EOF)
   {
      if (add.acc_no==upd.acc_no)
      {   test=1;
         printf("\nWhich information do you want to change?\n1.Address\n2.Phone\n\nEnter your choice(1 for
address and 2 for phone):");
         scanf("%d",&choice);
         system("cls");
         if(choice==1)
            {printf("Enter the new address:");
            scanf("%s",upd.address);
            fprintf(newrec,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,upd.address,add.citizenship,a
dd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);
            system("cls");
            printf("Changes saved!");
            }
         else if(choice==2)
            {
               printf("Enter the new phone number:");
            scanf("%lf",&upd.phone);
```

```
            fprintf(newrec,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.citizenship,u
pd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);
            system("cls");
            printf("Changes saved!");
            }

      }
      else
         fprintf(newrec,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.citizenship,a
dd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);
   }
   fclose(old);
   fclose(newrec);
   remove("record.dat");
   rename("new.dat","record.dat");

if(test!=1)
      {   system("cls");
          printf("\nRecord not found!!\a\a\a");
          edit_invalid:
            printf("\nEnter 0 to try again,1 to return to main menu and 2 to exit:");
            scanf("%d",&main_exit);
            system("cls");
              if (main_exit==1)

                 menu();
            else if (main_exit==2)
                 close();
```

```
            else if(main_exit==0)
                edit();
            else
                {printf("\nInvalid!\a");
                goto edit_invalid;}
        }
    else
        {printf("\n\n\nEnter 1 to go to the main menu and 0 to exit:");
        scanf("%d",&main_exit);
        system("cls");
        if (main_exit==1)
            menu();
        else
            close();
        }
}

void transact(void)
{   int choice,test=0;
    FILE *old,*newrec;
    old=fopen("record.dat","r");
    newrec=fopen("new.dat","w");

        printf("Enter the account no. of the customer:");
    scanf("%d",&transaction.acc_no);
    while (fscanf(old,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d",&add.acc_no,add.name,&add.dob.month,&add.dob.day,&add.dob.year,&add.age,add.address,add.citiz
enship,&add.phone,add.acc_type,&add.amt,&add.deposit.month,&add.deposit.day,&add.deposit.year)!=EOF)
   {
```

```
        if(add.acc_no==transaction.acc_no)
        {   test=1;

if(strcmpi(add.acc_type,"fixed1")==0||strcmpi(add.acc_type,"fixed2")==0||strcmpi(add.acc_type,"fixed3")==0)
            {
               printf("\a\a\a\n\nYOU CANNOT DEPOSIT OR WITHDRAW CASH IN FIXED ACCOUNTS!!!!!");
               fordelay(1000000000);
               system("cls");
               menu();

            }
            printf("\n\nDo you want to\n1.Deposit\n2.Withdraw?\n\nEnter your choice(1 for deposit and 2 for
withdraw):");
            scanf("%d",&choice);
            if (choice==1)
            {
               printf("Enter the amount you want to deposit:$ ");
               scanf("%f",&transaction.amt);
               add.amt+=transaction.amt;
               fprintf(newrec,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.citizenship,a
dd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);
               printf("\n\nDeposited successfully!");
            }
            else
            {
               printf("Enter the amount you want to withdraw:$ ");
               scanf("%f",&transaction.amt);
               add.amt-=transaction.amt;
```

```
                fprintf(newrec,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.citizenship,a
dd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);

                printf("\n\nWithdrawn successfully!");

             }



          }

          else

          {

             fprintf(newrec,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.citizenship,a
dd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);

          }

  }

  fclose(old);

  fclose(newrec);

  remove("record.dat");

  rename("new.dat","record.dat");

  if(test!=1)

  {

     printf("\n\nRecord not found!!");

     transact_invalid:

    printf("\n\n\nEnter 0 to try again,1 to return to main menu and 2 to exit:");

    scanf("%d",&main_exit);

    system("cls");

    if (main_exit==0)

      transact();

   else if (main_exit==1)

      menu();

   else if (main_exit==2)
```

```
        close();
    else
    {
        printf("\nInvalid!");
        goto transact_invalid;
    }

  }
  else
  {
      printf("\nEnter 1 to go to the main menu and 0 to exit:");
        scanf("%d",&main_exit);
        system("cls");
        if (main_exit==1)
            menu();
        else
            close();
  }

}
void erase(void)
{
    FILE *old,*newrec;
    int test=0;
    old=fopen("record.dat","r");
    newrec=fopen("new.dat","w");
    printf("Enter the account no. of the customer you want to delete:");
    scanf("%d",&rem.acc_no);
```

```
   while (fscanf(old,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d",&add.acc_no,add.name,&add.dob.month,&add.dob.day,&add.dob.year,&add.age,add.address,add.citiz
enship,&add.phone,add.acc_type,&add.amt,&add.deposit.month,&add.deposit.day,&add.deposit.year)!=EOF)
  {
      if(add.acc_no!=rem.acc_no)
          fprintf(newrec,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.citizenship,a
dd.phone,add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);


      else
          {test++;
          printf("\nRecord deleted successfully!\n");
          }
  }
  fclose(old);
  fclose(newrec);
  remove("record.dat");
  rename("new.dat","record.dat");
  if(test==0)
      {
          printf("\nRecord not found!!\a\a\a");
          erase_invalid:
            printf("\nEnter 0 to try again,1 to return to main menu and 2 to exit:");
            scanf("%d",&main_exit);


              if (main_exit==1)
                  menu();
              else if (main_exit==2)
                  close();
              else if(main_exit==0)
```

```
            erase();
          else
            {printf("\nInvalid!\a");
            goto erase_invalid;}
      }
   else
      {printf("\nEnter 1 to go to the main menu and 0 to exit:");
      scanf("%d",&main_exit);
      system("cls");
      if (main_exit==1)
         menu();
      else
         close();
      }

}

void see(void)
{
   FILE *ptr;
   int test=0,rate;
   int choice;
   float time;
   float intrst;
   ptr=fopen("record.dat","r");
   printf("Do you want to check by\n1.Account no\n2.Name\nEnter your choice:");
   scanf("%d",&choice);
   if (choice==1)
```

```
{   printf("Enter the account number:");
    scanf("%d",&check.acc_no);

    while (fscanf(ptr,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d",&add.acc_no,add.name,&add.dob.month,&add.dob.day,&add.dob.year,&add.age,add.address,add.citiz
enship,&add.phone,add.acc_type,&add.amt,&add.deposit.month,&add.deposit.day,&add.deposit.year)!=EOF)
    {
       if(add.acc_no==check.acc_no)
       {   system("cls");
          test=1;

          printf("\nAccount NO.:%d\nName:%s \nDOB:%d/%d/%d \nAge:%d \nAddress:%s \nCitizenship No:%s
\nPhone number:%.0lf \nType Of Account:%s \nAmount deposited:$ %.2f \nDate Of
Deposit:%d/%d/%d\n\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.ci
tizenship,add.phone,
          add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);
          if(strcmpi(add.acc_type,"fixed1")==0)
             {
                time=1.0;
                rate=9;
                intrst=interest(time,add.amt,rate);
                printf("\n\nYou will get $%.2f as interest on
%d/%d/%d",intrst,add.deposit.month,add.deposit.day,add.deposit.year+1);
             }
          else if(strcmpi(add.acc_type,"fixed2")==0)
             {
                time=2.0;
                rate=11;
                intrst=interest(time,add.amt,rate);
                printf("\n\nYou will get $.%.2f as interest on
%d/%d/%d",intrst,add.deposit.month,add.deposit.day,add.deposit.year+2);
```

```
            }
          else if(strcmpi(add.acc_type,"fixed3")==0)
            {
               time=3.0;
               rate=13;
               intrst=interest(time,add.amt,rate);
               printf("\n\nYou will get $.%.2f as interest on
%d/%d/%d",intrst,add.deposit.month,add.deposit.day,add.deposit.year+3);

            }
          else if(strcmpi(add.acc_type,"saving")==0)
            {
               time=(1.0/12.0);
               rate=8;
               intrst=interest(time,add.amt,rate);
               printf("\n\nYou will get $.%.2f as interest on %d of every month",intrst,add.deposit.day);

             }
          else if(strcmpi(add.acc_type,"current")==0)
            {

               printf("\n\nYou will get no interest\a\a");

             }

       }
    }
```

```
   }
   else if (choice==2)
   {   printf("Enter the name:");
       scanf("%s",&check.name);
       while (fscanf(ptr,"%d %s %d/%d/%d %d %s %s %lf %s %f
%d/%d/%d",&add.acc_no,add.name,&add.dob.month,&add.dob.day,&add.dob.year,&add.age,add.address,add.citiz
enship,&add.phone,add.acc_type,&add.amt,&add.deposit.month,&add.deposit.day,&add.deposit.year)!=EOF)
       {
          if(strcmpi(add.name,check.name)==0)
          {   system("cls");
              test=1;
              printf("\nAccount No.:%d\nName:%s \nDOB:%d/%d/%d \nAge:%d \nAddress:%s \nCitizenship No:%s
\nPhone number:%.0lf \nType Of Account:%s \nAmount deposited:$%.2f \nDate Of
Deposit:%d/%d/%d\n\n",add.acc_no,add.name,add.dob.month,add.dob.day,add.dob.year,add.age,add.address,add.ci
tizenship,add.phone,
              add.acc_type,add.amt,add.deposit.month,add.deposit.day,add.deposit.year);
              if(strcmpi(add.acc_type,"fixed1")==0)
                 {
                    time=1.0;
                    rate=9;
                    intrst=interest(time,add.amt,rate);
                    printf("\n\nYou will get $.%.2f as interest on
%d/%d/%d",intrst,add.deposit.month,add.deposit.day,add.deposit.year+1);
                 }
              else if(strcmpi(add.acc_type,"fixed2")==0)
                 {
                    time=2.0;
                    rate=11;
                    intrst=interest(time,add.amt,rate);
                    printf("\n\nYou will get $.%.2f as interest on
%d/%d/%d",intrst,add.deposit.month,add.deposit.day,add.deposit.year+2);
```

```
            }
          else if(strcmpi(add.acc_type,"fixed3")==0)
            {
               time=3.0;
               rate=13;
               intrst=interest(time,add.amt,rate);
               printf("\n\nYou will get $.%.2f as interest on
%d/%d/%d",intrst,add.deposit.month,add.deposit.day,add.deposit.year+3);

            }
          else if(strcmpi(add.acc_type,"saving")==0)
            {
               time=(1.0/12.0);
               rate=8;
               intrst=interest(time,add.amt,rate);
               printf("\n\nYou will get $.%.2f as interest on %d of every month",intrst,add.deposit.day);

             }
          else if(strcmpi(add.acc_type,"current")==0)
            {

               printf("\n\nYou will get no interest\a\a");

             }

       }
    }
```

```
}



fclose(ptr);
 if(test!=1)
    {   system("cls");
        printf("\nRecord not found!!\a\a\a");
        see_invalid:
          printf("\nEnter 0 to try again,1 to return to main menu and 2 to exit:");
          scanf("%d",&main_exit);
          system("cls");
            if (main_exit==1)
                menu();
            else if (main_exit==2)
                close();
            else if(main_exit==0)
                see();
            else
                {
                    system("cls");
                    printf("\nInvalid!\a");
                    goto see_invalid;}
    }
else
    {printf("\nEnter 1 to go to the main menu and 0 to exit:");
    scanf("%d",&main_exit);}
    if (main_exit==1)
    {
```

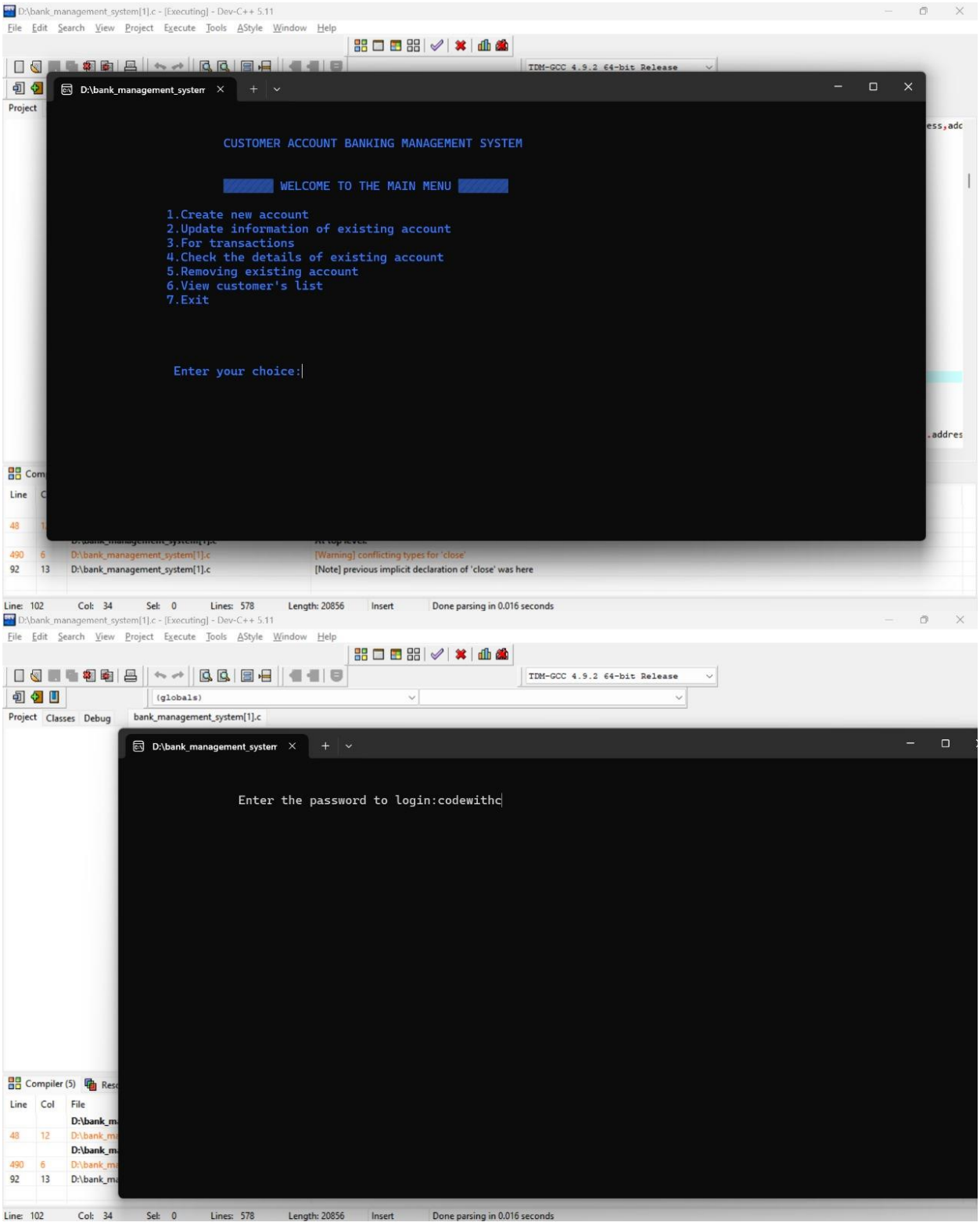
```
            system("cls");
            menu();
        }

        else
          {

             system("cls");
            close();
            }

}



void close(void)
{
    printf("\n\n\n\nThis C Mini Project is developed by Code With C team!");
    }



void menu(void)
{   int choice;
    system("cls");
    system("color 9");
    printf("\n\n\t\t\tCUSTOMER ACCOUNT BANKING MANAGEMENT SYSTEM");
    printf("\n\n\n\t\t\t\xB2\xB2\xB2\xB2\xB2\xB2\xB2 WELCOME TO THE MAIN MENU
\xB2\xB2\xB2\xB2\xB2\xB2\xB2");
```

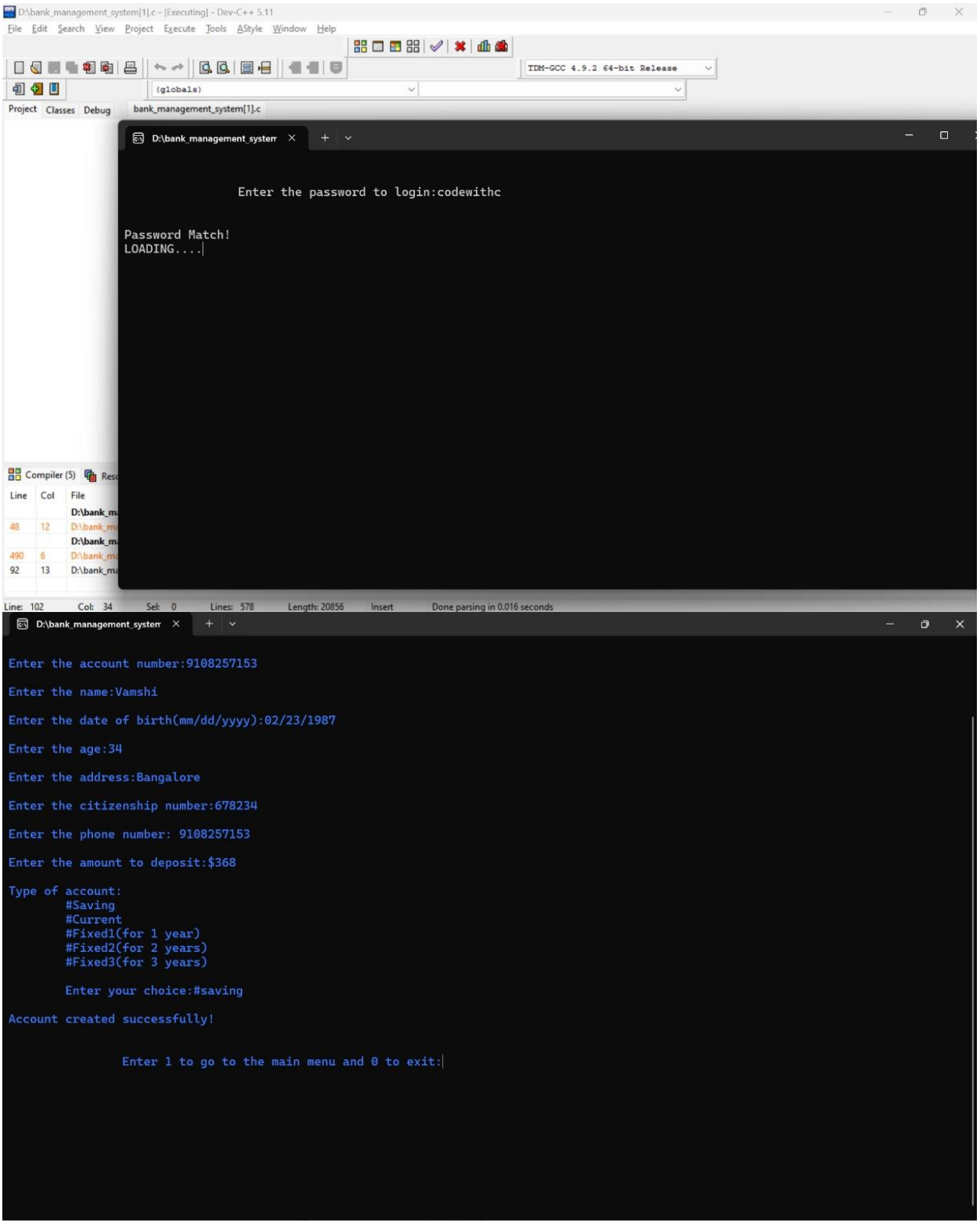
```
   printf("\n\n\t\t1.Create new account\n\t\t2.Update information of existing account\n\t\t3.For transactions\n\t\t4.Check the details of existing account\n\t\t5.Removing existing account\n\t\t6.View customer's list\n\t\t7.Exit\n\n\n\n\n\t\t Enter your choice:");
   scanf("%d",&choice);

   system("cls");
   switch(choice)
   {
      case 1:new_acc();
      break;
      case 2:edit();
      break;
      case 3:transact();
      break;
      case 4:see();
      break;
      case 5:erase();
      break;
      case 6:view_list();
      break;
      case 7:close();
      break;

   }

}
int main()
```

```
{
   char pass[10],password[10]="codewithc";
   int i=0;
   printf("\n\n\t\tEnter the password to login:");
   scanf("%s",pass);
   /*do
   {
   //if (pass[i]!=13&&pass[i]!=8)
      {
         printf("*");
         pass[i]=getch();
         i++;
      }
   }while (pass[i]!=13);
   pass[10]='\0';*/
   if (strcmp(pass,password)==0)
      {printf("\n\nPassword Match!\nLOADING");
      for(i=0;i<=6;i++)
      {
         fordelay(100000000);
         printf(".");
      }
            system("cls");
         menu();
      }
   else
      {   printf("\n\nWrong password!!\a\a\a");
         login_try:
```

```
        printf("\nEnter 1 to try again and 0 to exit:");
        scanf("%d",&main_exit);
        if (main_exit==1)
                {


                   system("cls");
                   main();
                }


        else if (main_exit==0)
                {
                system("cls");
                close();}
        else
                {printf("\nInvalid!");
                fordelay(1000000000);
                system("cls");
                goto login_try;}


    }
    return 0;
}
```

RESULTS AND OUTPUT:




D:\bank_management_system[1].c - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Project Classes Debug
bank_management_system[1].c
D:\bank_management_system
Enter the password to login:codewithc
Password Match!
LOADING....
Compiler (5)
Line Col File
Line: 102 Col: 34 Sel: 0 Lines: 578 Length: 20856 Insert Done parsing in 0.016 seconds
D:\bank_management_system
Enter the account number:9108257153
Enter the name:Vamshi
Enter the date of birth(mm/dd/yyyy):02/23/1987
Enter the age:34
Enter the address:Bangalore
Enter the citizenship number:678234
Enter the phone number: 9108257153
Enter the amount to deposit:$368
Type of account:
#Saving
#Current
#Fixed1(for 1 year)
#Fixed2(for 2 years)
#Fixed3(for 3 years)
Enter your choice:#saving
Account created successfully!
Enter 1 to go to the main menu and 0 to exit:

Similarly we shall be able to add details of several users and efficiently store and manage the data

CONCLUSION:

In conclusion, the development of the Smart Bank Management System using the C programming language represents a significant step towards modernizing and optimizing banking operations. Through the implementation of robust functionalities and user-friendly interfaces, our system aims to revolutionize the way banks manage their operations and interact with customers.

By leveraging the power of C programming, we have created a reliable and efficient system capable of handling critical banking tasks such as account management, transaction processing, and user authentication. The simplicity and versatility of the C language have allowed us to design a system that is not only efficient in terms of performance but also easy to maintain and scale as needed.

Furthermore, the integration of advanced security measures ensures that sensitive financial data remains protected from unauthorized access and potential security threats. With features such as encryption, access control, and biometric authentication, our system provides a secure environment for conducting banking transactions and safeguarding customer information.

Looking ahead, the Smart Bank Management System has the potential to revolutionize the banking industry by streamlining operations, enhancing security, and delivering personalized experiences to customers. As technology continues to evolve, our system will adapt and evolve alongside it, ensuring that banks remain at the forefront of innovation and customer satisfaction.

In summary, the Smart Bank Management System represents a significant achievement in the field of banking technology, offering a comprehensive solution to meet the evolving needs of the industry and its customers.

Future scope

The future scope of the Smart Bank Management System using C presents numerous opportunities for further enhancement and expansion. Some potential avenues for future development include:

Integration with emerging technologies: Incorporating emerging technologies such as blockchain, artificial intelligence, and Internet of Things (IoT) could further enhance the functionality and security of the system. For example, integrating blockchain technology for immutable transaction records or utilizing AI for advanced fraud detection algorithms.

Mobile and digital banking: Expanding the system to support mobile banking applications and digital banking channels would cater to the increasing demand for convenient and accessible banking services. Developing mobile apps compatible with various platforms could provide customers with on-the-go access to their accounts and transactions.

Enhanced analytics and reporting: Implementing advanced analytics capabilities could enable banks to gain deeper insights into customer behavior, preferences, and trends. Enhanced reporting features could provide management with actionable insights for strategic decision-making and personalized customer engagement strategies.

Enhanced security features: Continuously improving security measures to stay ahead of emerging cyber threats and regulatory requirements is crucial. This could involve implementing biometric authentication methods, multi-factor authentication, and encryption techniques to protect sensitive customer data and transactions.

Scalability and interoperability: Designing the system with scalability and interoperability in mind would allow for seamless integration with third-party applications and services. This could facilitate collaboration with fintech companies, payment gateways, and other banking partners to expand service offerings and reach new markets.

Compliance and regulatory updates: Staying compliant with evolving regulatory standards and requirements is essential for maintaining trust and credibility in the banking industry. Regular updates and enhancements to ensure compliance with regulations such as GDPR, PSD2, and KYC/AML requirements are necessary.

Customer-centric innovations: Focusing on customer-centric innovations such as personalized banking experiences, proactive financial advice, and seamless omnichannel experiences could differentiate the bank in the competitive market landscape. Implementing chatbots, virtual assistants, and predictive analytics could enhance customer engagement and satisfaction.

Accessibility and inclusivity: Ensuring accessibility and inclusivity for all users, including those with disabilities or limited technological literacy, is paramount. Designing user interfaces with accessibility features and providing alternative communication channels such as text-to-speech or voice recognition could enhance the user experience for all customers.

Overall, the future scope of the Smart Bank Management System using C is vast and encompasses various opportunities for innovation, differentiation, and continuous improvement to meet the evolving needs of the banking industry and its customers.

References

Kernighan, Brian W., and Dennis M. Ritchie. "The C Programming Language." Prentice Hall, 1988.

Balagurusamy, E. "Programming in ANSI C." Tata McGraw-Hill Education, 2008.

Prata, Stephen. "C Primer Plus." Addison-Wesley Professional, 2013.

King, K. N. "C Programming: A Modern Approach." W. W. Norton & Company, 2008.

Reema Thareja. "Programming in C." Oxford University Press, 2014.

Microsoft Corporation. "C Language Reference." Microsoft Developer Network (MSDN).

The GNU Project. "The GNU C Library Reference Manual." GNU Press.

Kernighan, Brian W., and Dennis M. Ritchie. "The C Programming Language." Prentice Hall, 1988.

Balagurusamy, E. "Programming in ANSI C." Tata McGraw-Hill Education, 2008.

Prata, Stephen. "C Primer Plus." Addison-Wesley Professional, 2013.

King, K. N. "C Programming: A Modern Approach." W. W. Norton & Company, 2008.