**Micro Project**

**Computer programming**

**Banking application**



Department of basic science

Annasaheb Dange College of Engineering and technology Ashta

**Submitted by:**

Unmesh Suhas Mali 237

# Akshat Vijay Rasane 230

**Abstract**

The Banking application In C is developed in C programming language, This program allows you to Store account deposited, withdrawl can be checked.

This abstract describes a banking application program implemented in the C programming language. The program aims to provide a secure and efficient platform for managing various banking operations, including account creation, funds transfer, balance inquiry, and transaction history retrieval.

**Introduction**

Our application is designed to provide you with a seamless and secure banking experience, all within the comfort of your computer or mobile device. With a user-friendly interface and robust features, you can manage your finances, perform transactions, and stay in complete control of your accounts.

Utilizing the power of C programming, we have prioritized efficiency and reliability, ensuring that your banking needs are met with utmost precision. Whether you're a tech-savvy individual or new to the world of online banking, our program is tailored to cater to your requirements and safeguard your financial data. Get ready to embark on a journey of convenience and trust as we redefine the way you bank, one line of code at a time.

**Objectives**

Following are the objectives of the project,

* + 1. Efficient account management.
    2. user friendly interface
    3. Reduce paperwork.
    4. The system is user friendly.

# Implementation

#incIude <stdio.h> #incIude<conio.h> struct Account

int accountNumber;

float balance;

void deposit(struct Account” account, float amount) account->balance -I-= amount;

printf("Amount %.f deposited successfully.\n", amount);

void withdraw(struct Account” account, float amount) { if (account->balance >= amount) (

float balance;

account->balance -= amount;

printf("Amount %.f withdrawn successfully.\n", amount);

else

printf("InsuPicient balance.\n");

void baIanceEnquiry(struct Account” account)

printf("Account Number: %d\n", account->accountNumber); printf("Balance: O/o.f\n", account->balance);

int main()

struct account myaccount ; myaccount. accountnumber=12345; myaccount.balance=1000;

int choice; float amount;

float my account; do

printf("\nBanking Application\n"); printf("1. Deposit\n");

printf("2. Withdraw\n"); printf("3. Balance Enquiry\n"); printf("4. Exit\n"); printf("Enter your choice: "); scanf("%d", &choice);

switch (choice)

camel:

printf("Enter the amount to deposit: ");

Scanf(!!Oo !!, &amount);

deposit(&myAccount, amount);

break;

case 2:

case 3:

printf("Enter the amount to withdraw: "); scanf("%f", &amount); withdraw(&myAccount, amount);

break;

baIanceEnquiry(&myAccount); break;

case 4:

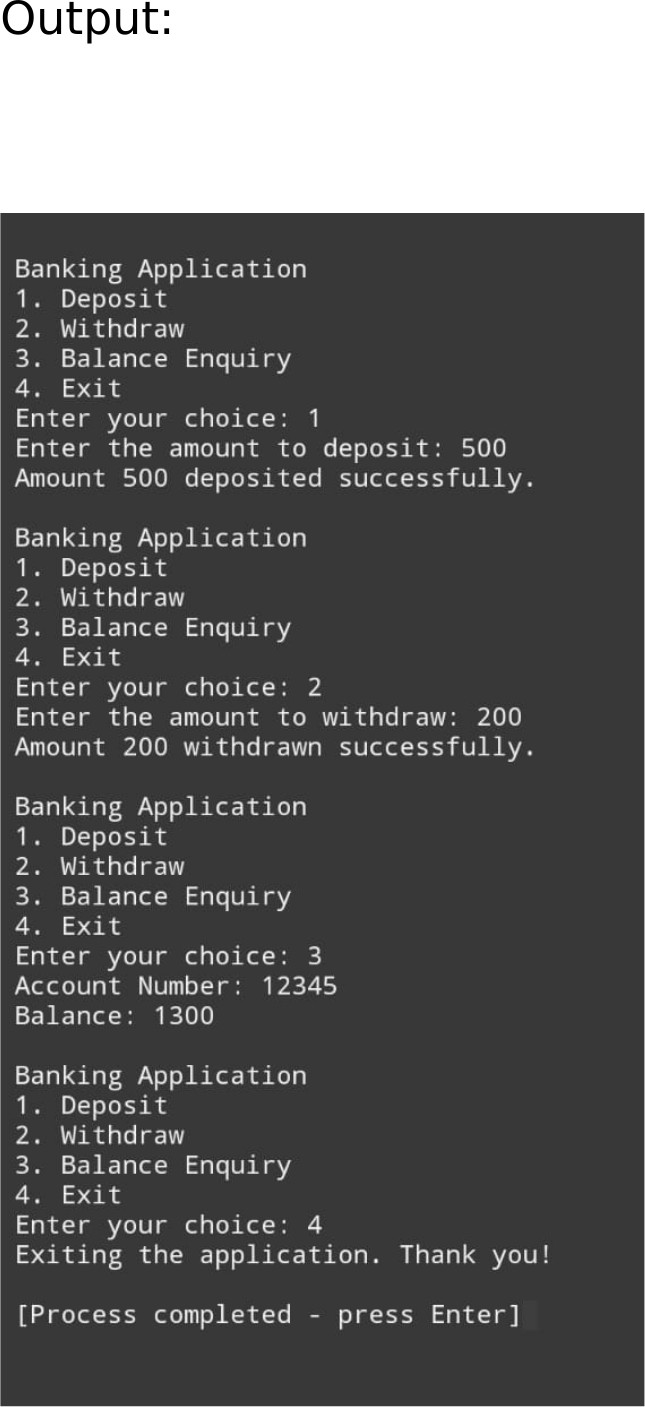
printf("Exiting the application. Thank you!\n");

break; default:

printf("Invalid choice. Please try again.\n"); break;

while (choice != 4); getch() ;

return 0;



### **Conclusion**

In conclusion, the banking application prog ram in C represents a efficient solution for manag ing various banking operations. Through its user-friend Iy interface and secure authentication mechanisms, customers can easily access their accounts, conduc transactions, and view their financial information with confidence.

The program's well-structured code ensures reliability and maintainability, while its incorporation of error handling and data validation enhances security and prevents potential issues. As a result, this banking application in C stands as a testament to the power of programming in simplifying complex financial processes, fostering trust, and providing convenience for both customers and banking professionals alike.