



DEPARTMENT OF
COMPUTER SCIENCE AND ENGINEERING

PROJECT REPORT ON
PRODUCT RECORD MANAGEMENT SYSTEM

DATA STRUCTURES (LAB) || CSE 106 || SEC: 213 D1

DATE OF SUBMISSION: 09 SEPT, 2022

SUBMITTED BY

SUBMITTED TO

MD. SAIFUL ISLAM RIMON

TAMIM AL MAHMUD

STU. ID: 213002039

ASST. PROF, DEPT OF CSE

Remarks (if any):	
Marks	Signature of Instructor



Green University of Bangladesh

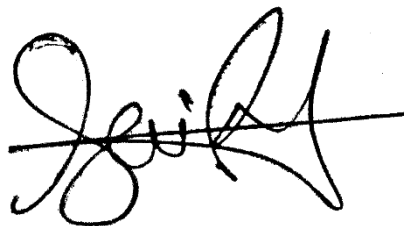
Dhaka-1216, Bangladesh

DECLARATION

GREEN UNIVERSITY OF BANGLADESH

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Here, I am **MD SAIFUL ISLAM RIMON** (213002039) confirm that this report and the work presented in it are my own achievement. I have read and understood the penalties associated with plagiarism.



MD SAIFUL ISLAM RIMON

09-09-2022

Date:

CERTIFICATION

GREEN UNIVERSITY OF BANGLADESH

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

This is to certify that this project is fully adequate in scope and quality as an undergraduate project work.

.....
TAMIM AL MAHMUD
ASST. PROF,
DEPT OF CSE

DATE

ABSTRACT

PRODUCT RECORD MANAGEMENT SYSTEM

The idea of developing a product record management system is to improve work efficiency for wholesale business partners. A product record management is something where a business holders store information of larger amount of product to manage and work with them effectively. This system exists to simplify information tracking for wholesale business partner. It aids in the tracking of information. Many industrial institutions currently using this type of software for their management. The software assists the vendor department in keeping track of product's information.

TABLE OF CONTENT

CONTEXT	PAGES
DECLARATION -----	02
CERTIFICATION -----	03
ABSTRACT -----	04
LIST OF FIGURES -----	07
CHAPTER 1. INTRODUCTION -----	08
1.1 WHAT IS PRODUCT RECORD MANAGEMENT SYSTEM -----	08
1.2 WHAT DOES PRODUCT RECORD MANAGEMENT SHOULD DO -----	08
1.3 WHAT DOES PRODUCT RECORD MANAGEMENT SHOULD INCLUDE -----	09
1.4 AIM AND OBJECTIVES/GOALS -----	10
CHAPTER 2. PROJECT IMPLEMENTATION -----	11
2.1 C++ PROGRAM -----	11
2.2 FLOWCHART -----	13
2.3 CODE -----	14
2.4 OUTPUT -----	23
CHAPTER 3. MATERIALS AND PROCEDURES -----	28
3.1 MATERIALS -----	28
3.2 PROCEDURES -----	28
CHAPTER 4. TERMINOLOGY -----	30
4.1 IDE (INTEGRATER DEVELOPMENT ENVIRONMENT)	30
4.2 CODE -----	30
4.3 BUILD AND RUN -----	31
4.3 DATABASE -----	31
4.3 ENCRYPTION -----	32
CHAPTER 5. FEATURES AND LIMITATIONS -----	33
5.1 FEATURES -----	33

5.2 LIMITATIONS -----	33
CHAPTER 6. PERFORMANCE EVALUATION -----	34
6.1 PERFORMANCE EVALUATION -----	34
6.2 RESULT AND DISCUSSION -----	34
CHAPTER 7. CONCLUSION -----	35
7.1 CONCLUSION-----	35
7.2 RECOMMENDATION AND FUTURE PLAN-----	35
REFERENCES -----	36

LIST OF FIGURES

1. *Figure 2.1: IEEE-the best 10 top programming language in 2018.*
2. *Figure 2.3.1: Starting interface asking password to give access in it.*
3. *Figure 2.3.2: Landing page of the app after logging*
4. *Figure 2.3.3: Option choosing page*
5. *Figure 2.3.4: Product record adding page*
6. *Figure 2.3.5: Product record information modifying page*
7. *Figure 2.3.2: Landing page of the app after logging*
8. *Figure 2.3.3: Option choosing page*
9. *Figure 2.3.5: Product record information modifying page*
10. *Figure 2.3.2: Landing page of the app after logging*
11. *Figure 2.3.4: Product record adding page*
12. *Figure 2.3.6: Showing all product record information in one page*
13. *Figure 2.3.7: Showing individual view of product record's information*
14. *Figure 2.3.8: Product record information deleting page*
15. *Figure 2.3.9: Password changing page*
16. *Figure 2.3.10: Encrypted database*

CHAPTER 1: INTRODUCTION

1.1 WHAT IS PRODUCT RECORD MANAGEMENT SYSTEM?

A product record Management System is something where a business holders store information of larger amount of product to manage and work with them effectively. This system exists to simplify information tracking for wholesale business partner.

1.2 WHAT DOES PRODUCT RECORD MANAGEMENT SHOULD DO?

Business-centric record management systems work to make the management of information more accessible. Below is a list of tracking pieces you might have on your information management system:

Basic Requires:

- Add Product Basic Information
- Add Category
- Add Prices
- Add Stock / Available Product
- Add Automated Time Stamp
- Security / Login System
- Landing Page

Operations:

- INSERTION
- DELETION
- SEARCH
- MODIFY / UPDATE PERTICULAR INFO

The focus of this sort of management system is that it is business-centric. This software differs from your other type of management system.

1.3 WHAT SHOULD A PRODUCT RECORD MANAGEMENT SHOULD INCLUDE?

For an ideal *Product Record Management System* there is something need which are obvious.

- Product information
- Price
- Stock / Available Product
- Time Info
- An Easy Interface
- Easy adding system
- Easy modifying system
- Easy deleting system
- Viewing option
- Security
- Database system

1.4 Aim and Objectives/Goals

The Aim of this project is to design and implement a product record management system that is able to perform the tasks which are included to the system.

The Objectives of the project are as follows:

- Implement the security first.
- Implement all the features/Basic Requires (*Chapter 1.2*) as far as can possible.
- Implement all operations (*Chapter 1.2*)
- Implement an easy operating user interface.
- Encrypted Database system.

CHAPTER 2: PROJECT IMPLEMENTATION

For developing this project, I have used C++ as programming language. Let's discuss about it.

2.1 C++ PROGRAM

- C++ is a cross-platform language that can be used to create high-performance applications.
- C++ was developed by Bjarne Stroustrup, as an extension to the [C language](#).
- C++ gives programmers a high level of control over system resources and memory.
- C++ is one of the world's most popular programming languages.
- C++ can be found in today's operating systems, Graphical User Interfaces, and embedded systems.
- C++ is an object-oriented programming language which gives a clear structure to programs and allows code to be reused, lowering development costs.
- C++ is portable and can be used to develop applications that can be adapted to multiple platforms.
- C++ is fun and easy to learn!
- As C++ is close to [C#](#) and [Java](#), it makes it easy for programmers to switch to C++ or vice versa.

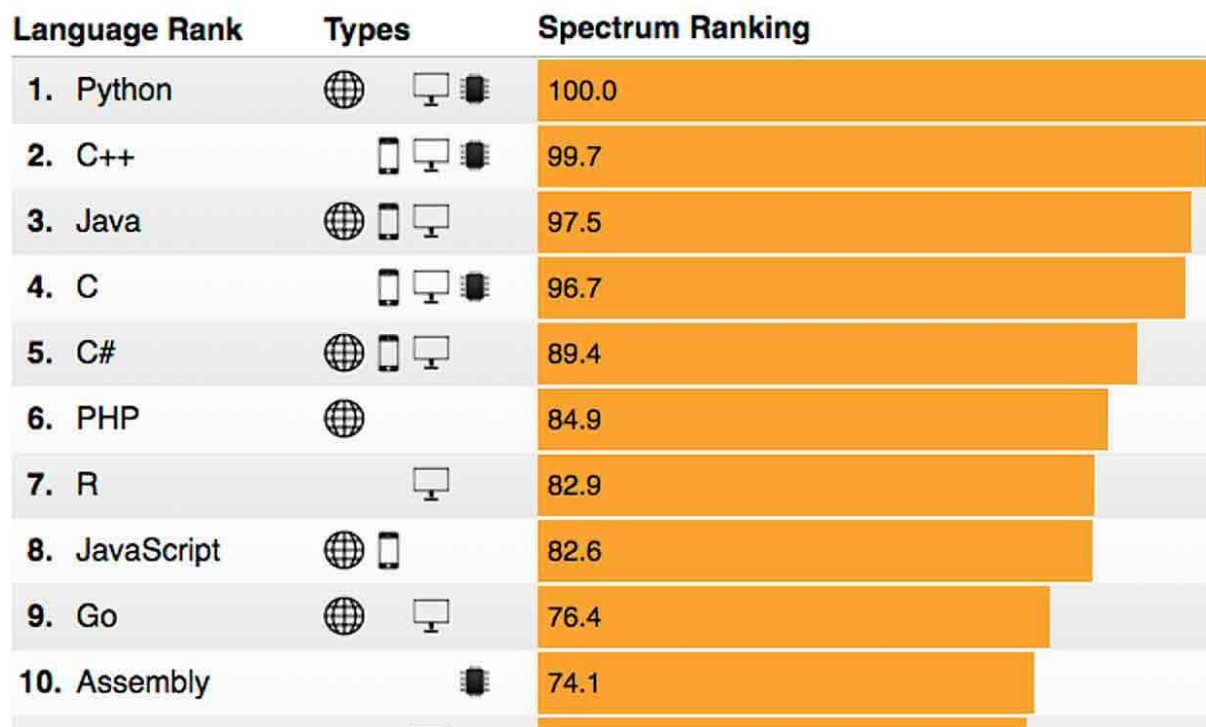
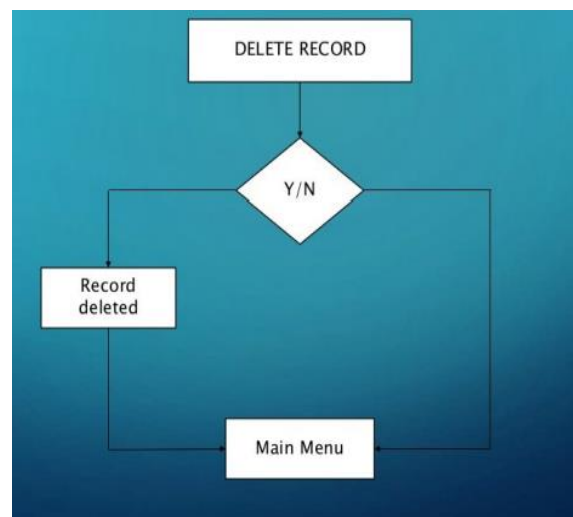
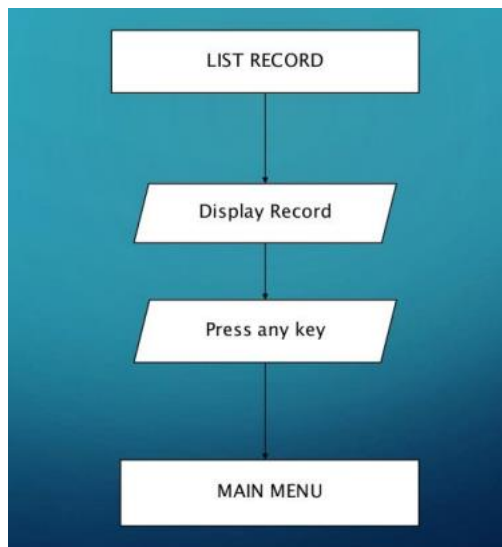
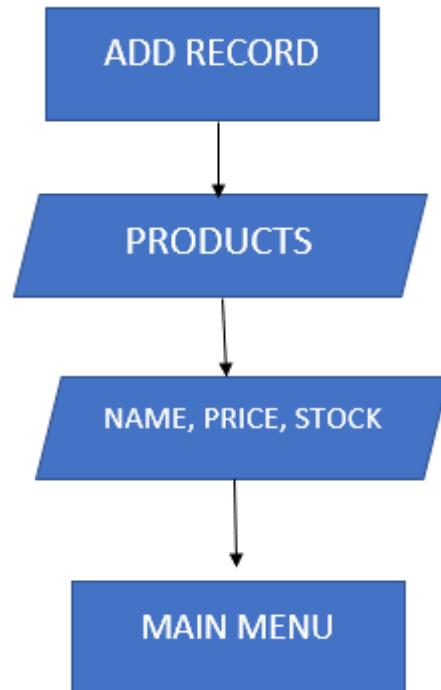
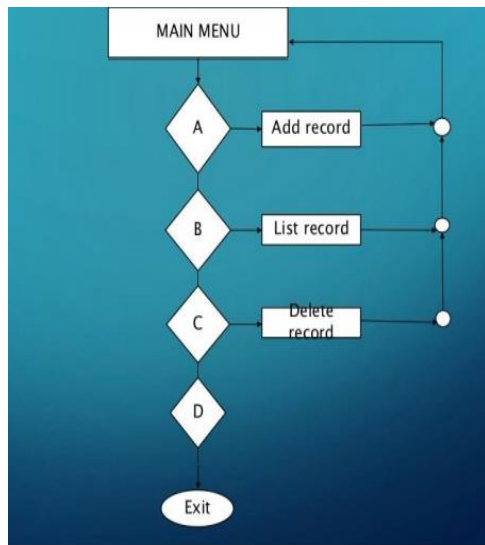


Figure 2.1: IEEE-the best 10 top programming language in 2018

We can see that, C++ has a ranking of 99.7 (out of 100) from the perspective of it's uses.

2.2 FLOW CHART



2.3 CODE

```
#include <iostream>
#include<string>
#include<stdlib.h>
#include<math.h>
#include <windows.h>
#include<conio.h> //CONSOLE INPUT OUTPPUT. FOR WORKING SCREEN CLEAR
AND GETCH
#include<iomanip> //FOR WORKING SETW - SET WIDTH
#define Product struct Prod

void add(FILE * fp);
void modify(FILE * fp);
void display(FILE * fp);
void Indivisual(FILE *fp);
void password();
FILE * del(FILE * fp);
void printChar(char ch,int n);
void title();
FILE *tp;

void gotoxy(int x,int y)
{
    COORD CRD;
    CRD.X = x;
    CRD.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),CRD);
}

struct pass
{
    char pass[25];
} pa;

struct Prod
{
    char name[100];
    char category[50];
    int product_id;
    //float sgpa[12];
    float price;
    int stock;
    char time[30];
};

using namespace std;

int main()
{
    cout << fixed << setprecision(2); //HOW MANY NUMBERS WILL SHOW
AFTER POINT
    int ch,id,k,i;
    char c,pas[50];
    SetConsoleTitle("Product Record Management System | GREEN
UNIVERSITY OF BANGLADESH | 213002039");
```



```

        }
    }
    else
    {
        cout << "Wrong Password . Get Out";
        getch();
    }
    return 1;
}

void password()
{
    char c;
    cout << "\nEnter new password :";
    fflush(stdin);
    gets(pa.pass);
    cout << "\nSave password (y/n) :";
    fflush(stdin);
    cin >> c;
    if(c=='y' || c=='Y')
    {
        tp=fopen("F:/Password.txt","w+");
        fwrite(&pa,sizeof(pa),1,tp);
        fclose(tp);
        cout << "\n\tPassword Saved\n";
    }
    else
    {
        cout << "Password not saved :\n";
        cout << "Press any key to continue >>>";
        getch();
    }
}

void printChar(char ch,int n)
{
    while(n--)
    {
        putchar(ch);
    }
}

void title()
{
    system("cls");
    system("COLOR 03");
    cout << "\n\n\t";
    printChar('=',19);
    cout << " Product Record Management System ";
    printChar('=',19);
    cout << "\n";
}

void add(FILE * fp)
{
    title();

```

```

char another='y';
Product s;
int i;
float price;

fseek(fp,0,SEEK_END);
while(another=='y' || another=='Y')
{
    cout << "\n\n\t\tEnter Full Name of Product: ";
    fflush(stdin);
    fgets(s.name,100,stdin);
    s.name[strlen(s.name)-1]='\0';

    cout << "\n\n\t\tEnter Category (Hint: Mouse/Keyboard/Monitor
etc.): ";
    fflush(stdin);
    fgets(s.category,50,stdin);
    s.category[strlen(s.category)-1]='\0';

    cout << "\n\n\t\tEnter Product ID number: ";
    cin >> s.product_id;

    cout << "\n\n\t\tEnter Product price: ";
    cin >> s.price;

    cout << "\n\n\t\tStock available: ";
    cin >> s.stock;

    time_t now = time(0); //current date and time based on system

    char* dt = ctime(&now); //convert now to string form
    fflush(stdin);
    strcpy(s.time,dt);
    s.time[strlen(s.time)-1]='\0';

    fwrite(&s,sizeof(s),1,fp);

    cout << "\n\n\t\tAdd another Product?(Y/N)?";
    fflush(stdin);
    another=getchar();
}
}

FILE * del(FILE * fp)
{
    title();

    Product s;
    int flag=0,tempproduct_id,siz=sizeof(s);
    FILE *ft;

    if((ft=fopen("temp.txt","wb+"))==NULL)
    {
        cout << "\n\n\t\t\t\t\t!!! ERROR !!!\n\n\t\t";
        system("pause");
        return fp;
    }
}

```

```

    }

    cout << "\n\n\tEnter ID number of the product to Delete the
Record";
    cout << "\n\n\t\t\tProduct ID No. : ";
    cin >> tempproduct_id;

    rewind(fp);

    while((fread(&s,siz,1,fp))==1)
    {
        if(s.product_id==tempproduct_id)
        {
            flag=1;
            cout << "\n\tRecord Deleted for";
            cout << "\n\n\t\t" << s.name << "\n\n\t\t" << s.category
<< "\n\n\t\t" << s.product_id << "\n\t";
            continue;
        }

        fwrite(&s,siz,1,ft);
    }

    fclose(fp);
    fclose(ft);

    remove("db.txt");
    rename("temp.txt","db.txt");

    if((fp=fopen("db.txt","rb+"))==NULL)
    {
        cout << "ERROR";
        return NULL;
    }

    if(flag==0)
        cout << "\n\n\t\tNO Product FOUND WITH THE INFORMATION\n\t";

    printChar('-',65);
    cout << "\n\t";
    system("pause");
    return fp;
}

void modify(FILE * fp)
{
    title();

    Product s;
    int i,flag=0,tempproduct_id,siz=sizeof(s);
    float price;

    cout << "\n\n\tEnter ID Number of Product to MODIFY the Record :
";
    cin >> tempproduct_id;

```

```

rewind(fp);

while( (fread(&s,siz,1,fp))==1)
{
    if(s.product_id==tempproduct_id)
    {
        flag=1;
        break;
    }
}

if(flag==1)
{
    fseek(fp,-siz,SEEK_CUR);
    cout << "\n\n\t\t\tRecord Found\n\t\t\t";
    printChar('=',38);
    cout << "\n\n\t\t\tProduct Name: " << s.name;
    cout << "\n\n\t\t\tProduct ID: " << s.product_id <<
"\n\t\t\t";
    printChar('=',38);
    cout<<"\n\nWhich option you want to MODIFY?\n1. NAME\n2.
CATEGORY\n3. PRODUCT ID\n4. PRICE\n5. STOCK\n\n";
    int choice;
    cin >> choice;
    switch(choice)
    {
    case 1:
    {
        cout << "\n\n\t\t\tEnter Full Name of Product: ";
        fflush(stdin);
        fgets(s.name,100,stdin);
        s.name[strlen(s.name)-1]='\0';
        break;
    }
    case 2:
    {
        cout << "\n\n\t\t\tEnter Category: ";
        fflush(stdin);
        fgets(s.category,50,stdin);
        s.category[strlen(s.category)-1]='\0';
        break;
    }
    case 3:
    {
        cout << "\n\n\t\t\tEnter Product ID number: ";
        cin >> s.product_id;

        break;
    }
    case 4:
    {
        cout << "\n\n\t\t\tEnter Product price: ";
        cin >> s.price;
        break;
    }
    case 5:
    {
        cout << "\n\n\t\t\tStock available: ";

```

```

        cin>> s.stock;
        break;
    }
    default:
        cout << "Wrong Option Chosen!";
        break;
    }

    fwrite(&s,sizeof(s),1,fp);
}

else
    cout << "\n\n\t!!!! ERROR !!!! RECORD NOT FOUND";

    cout << "\n\n\t";
    system("pause");
}

void display(FILE * fp)
{
    title();
    Product s;
    int i,siz=sizeof(s);

    rewind(fp);

    while((fread(&s,siz,1,fp))==1)
    {
        cout << "\n\t\tNAME : " << s.name;
        cout << "\n\n\t\tCategory : " << s.category;
        cout << "\n\n\t\tProduct ID : " << s.product_id;
        cout << "\n\n\t\tProduct price: " << s.price << " BDT";
        cout << "\n\n\t\tStock Available: " <<s.stock << " PCS.";
        cout << "\n\n\t\tDate added: " <<s.time ;

        cout << "\n\t";
        printChar('-',65);
    }
    cout << "\n\n\n\t";
    printChar('*',65);
    cout << "\n\n\t";
    system("pause");
}

void Individual(FILE *fp)
{
    title();

    int tempproduct_id,flag,siz,i;
    Product s;
    char another='y';

    siz=sizeof(s);

    while(another=='y' || another=='Y')
    {

```

```

        cout << "\n\n\tEnter Product ID Number: ";
        cin >> tempproduct_id;

        rewind(fp);

        while((fread(&s,siz,1,fp))==1)
        {
            if(s.product_id==tempproduct_id)
            {
                flag=1;
                break;
            }
        }

        if(flag==1)
        {
            cout << "\n\t\tNAME : " << s.name;
            cout << "\n\n\t\tCategory : " << s.category;
            cout << "\n\n\t\tProduct ID : " << s.product_id;
            cout << "\n\n\t\tProduct price: " << s.price << " BDT";
            cout << "\n\n\t\tStock Available: " <<s.stock << " PCS.";
            cout << "\n\n\t\tDate added: " <<s.time ;

            cout << "\n\t";
            printChar('-',65);

        }
        else
            cout << "\n\n\t\t!!!! ERROR RECORD NOT FOUND !!!!!";

        cout << "\n\n\t\tShow another Product information? (Y/N)?";
        fflush(stdin);
        another=getchar();
    }
}

```

2.4 OUTPUT:

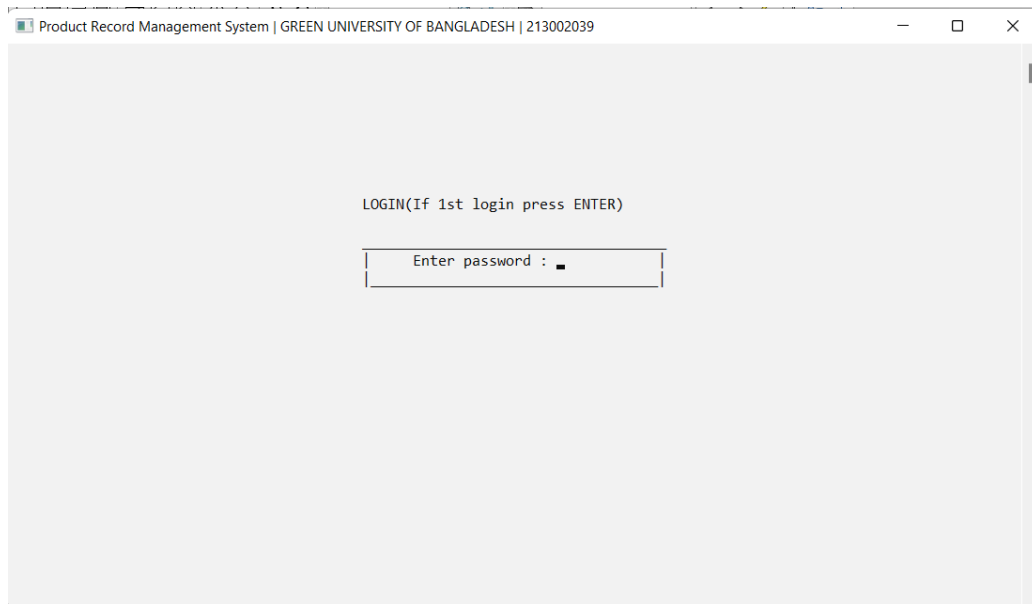


Figure 2.3.1: Starting interface asking password to give access in it.

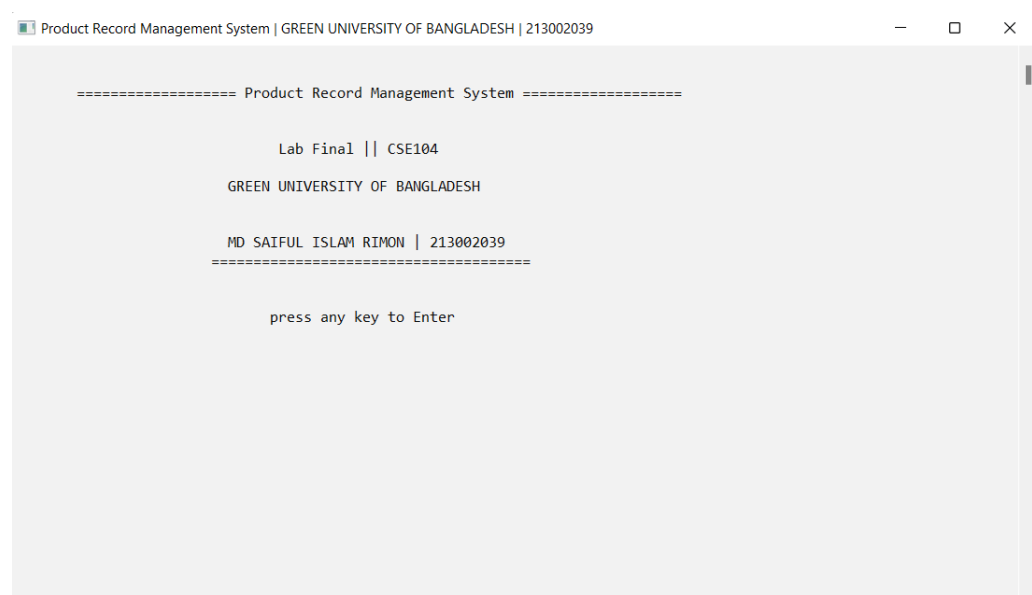


Figure 2.3.2: Landing page of the app after logging

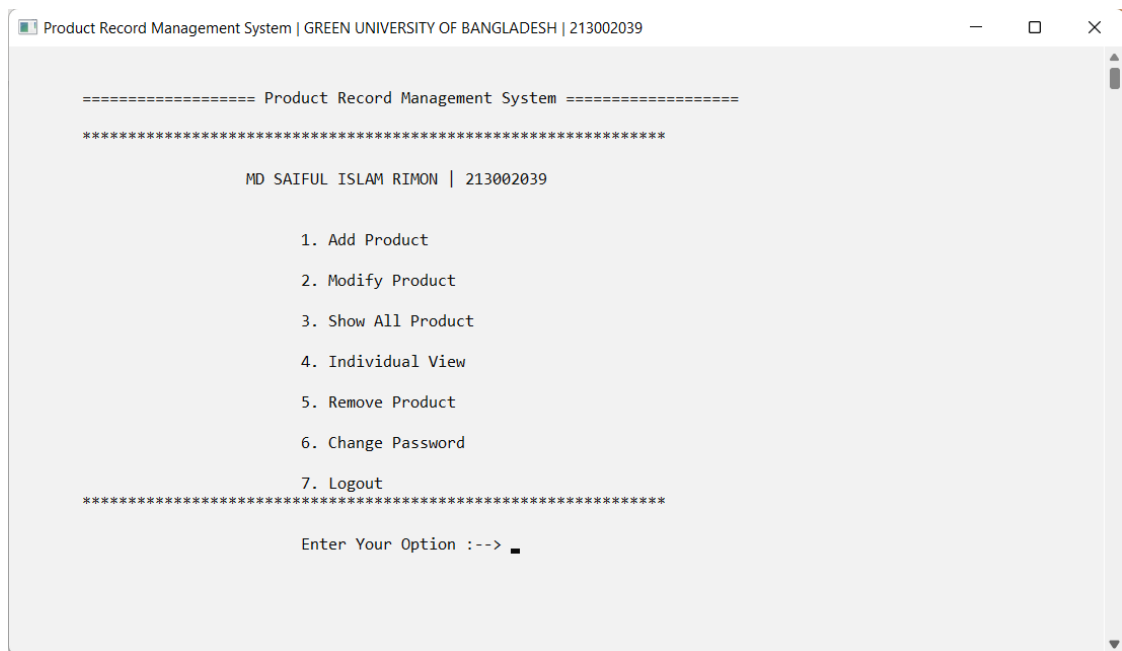


Figure 2.3.3: Main Menu / Option choosing page

A screenshot of a web browser window titled "Product Record Management System | GREEN UNIVERSITY OF BANGLADESH | 213002039". The page content is a form for adding a product, with a light gray background. It features a header "==== Product Record Management System =====", a separator line "*****", and a series of input prompts: "Enter Full Name of Product: HP VISION 4K", "Enter Category (Hint: Mouse/Keyboard/Monitor etc.): MONITOR", "Enter Product ID number: 951", "Enter Product price: 10000", "Stock available: 10", and "Add another Product?(Y/N)?".

Figure 2.3.4: Product adding page

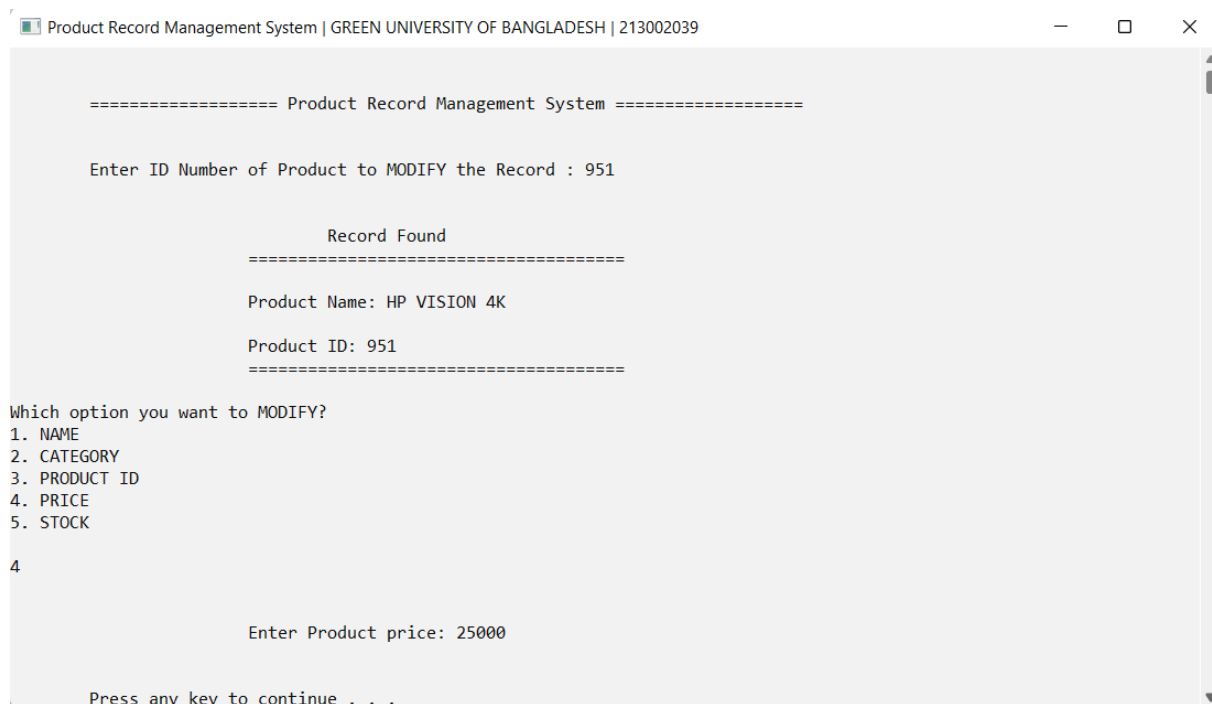


Figure 2.3.5: Product information modifying page

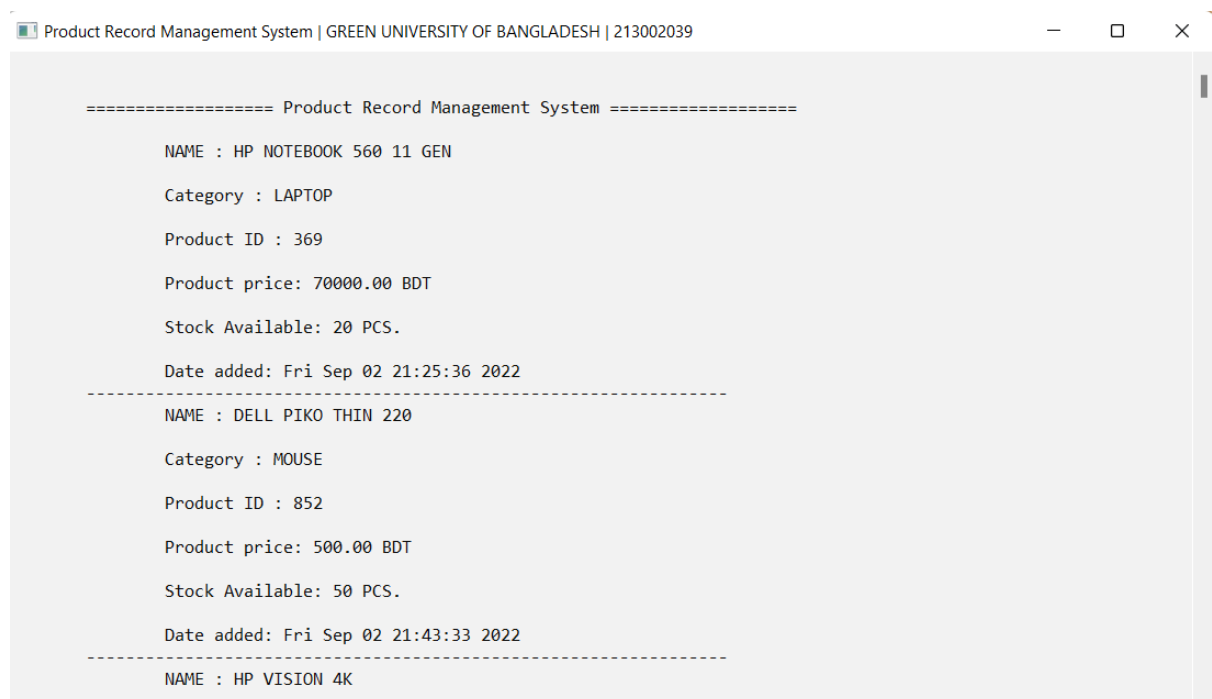


Figure 2.3.6: Showing all product information in one page

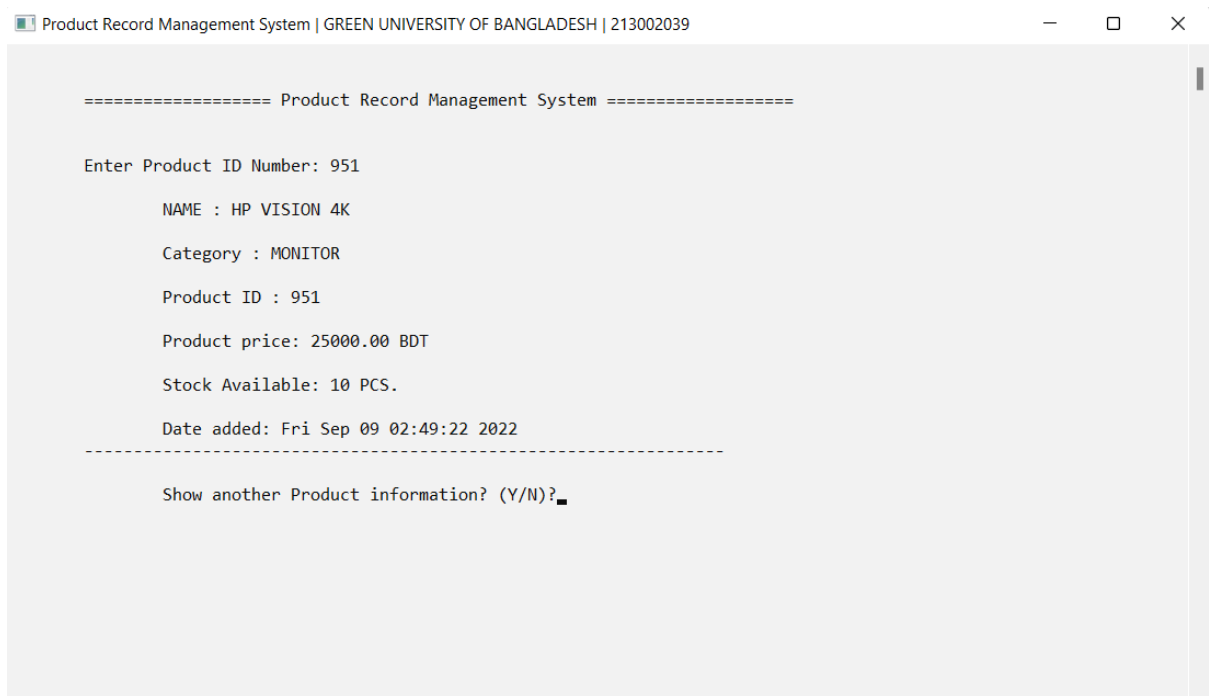


Figure 2.3.7: Showing individual view of product's information

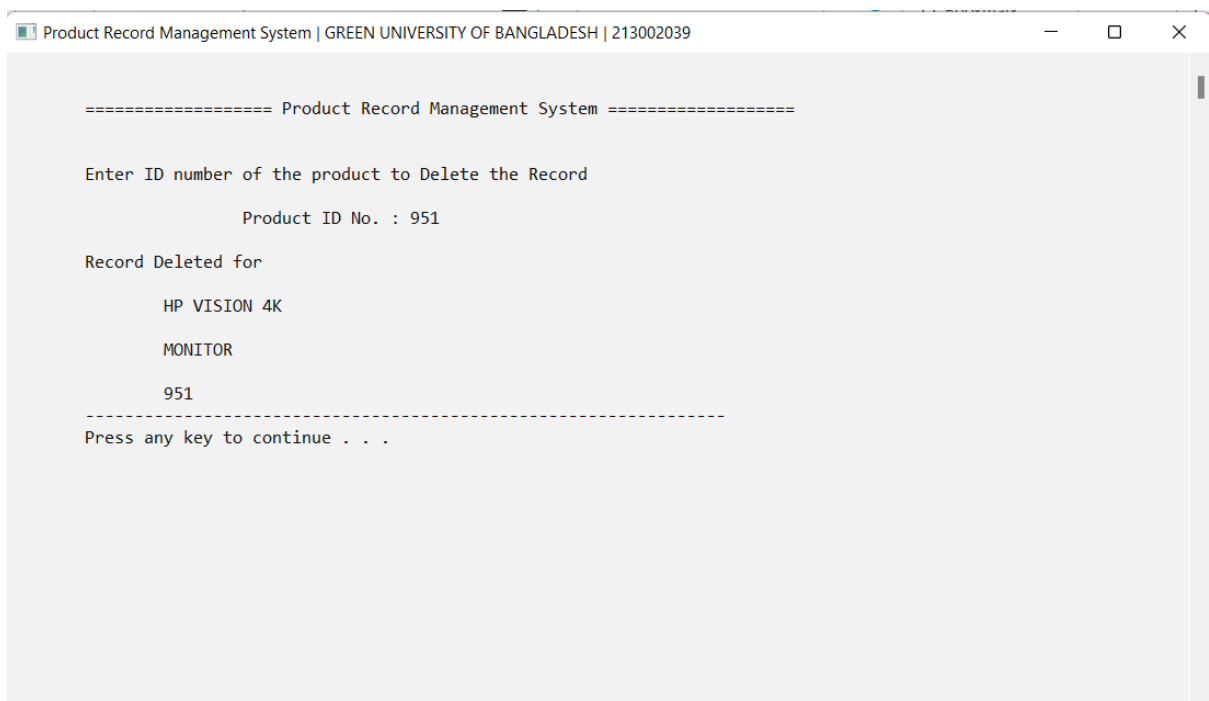


Figure 2.3.8: Product information deleting page

CHAPTER 3: MATERIALS AND PROCEDURES

3.1 MATERIALS

The following components were used for the implementation of the project:

- Computer
- IDE (Integrated Development Environment)

Like: Code Blocks, Visual Studio etc.

- Code

3.2 PROCEDURES:

The following procedures need to be followed step by step-

- Install any IDE (like: Code Blocks, Visual Studio etc.)
- Create a project in **code blocks**
- Write down the code which is given in ***chapter 2.2.***
- Compile and run the code
- Enter the password: **213002039**. If it is first time, then **press ENTER** button to get into the app and change the password as you wish.
- After logging into app, user will see a landing page with some basic information of the organization. Press ENTER button again.

- Now, a user will see some options for performing tasks (like: ADD product information, MODIFY product information etc..). User have to choose one to perform.
- Once an operation has complete, app will say “*Press any key*”. If user press any key, then app will give all the options again.
- This is all how a user can perform with this app.
- At last, a user can logout from the app.

[DON'T WORRY! YOUR DATA HAS BEEN SAVED ENCRYPTED IN PROJECT FOLDER IN A TEXT FILE NAMED “*db.txt*”.]

CHAPTER 4: TERMINOLOGIES

4.1 IDE (Integrated Development Environment)

Integrated development environments (IDE) are applications that facilitates the development of other applications. Designed to encompass all programming tasks in one application, one of the main benefits of an IDE is that they offer a central interface with all the tools a developer needs, including:

- **Code editor:** Designed for writing and editing source code, these editors are distinguished from text editors because work to either simplify or enhance the process of writing and editing of code for developers
- **Compiler:** Compilers transform source code that is written in a human readable/writable language in a form that computers can execute.
- **Debugger:** Debuggers are used during testing and can help developers debug their application programs.
- **Build automation tools:** These can help automate developer tasks that are more common to save time.

In addition, some IDEs may also include:

- **Class browser:** Used to study and reference properties of an object-oriented class hierarchy.
- **Object browser:** Used to inspect objects instantiated in a running application program.
- **Class hierarchy diagram:** Allows developers to visualize the structure of object-oriented programming code.

The IDE may be a stand-alone application, though it might also be included as part of one or more compatible applications.

4.2 WHAT IS CODE?

In computer programming, *computer code* refers to the set of instructions, or a system of rules, written in a particular programming language (i.e., the source code).

It is also the term used for the source code after it has been processed by a compiler and made ready to run on the computer (i.e., the object code).

4.3 BUILD AND RUN

- *Running* is getting some binary executable (or a script, for interpreted languages) to be, well... *executed* as a new *process* on the computer;
- *Compiling* is the process of parsing a program written in some high level language (higher if compared to machine code), checking it's syntax, semantics, linking libraries, maybe doing some optimization, then creating a binary *executable* program as an output. This executable may be in the form of machine code, or some kind of byte code — that is, instructions targeting some kind of *virtual machine*;
- *Building* usually involves checking and providing dependencies, inspecting code, compiling the code into binary, running automated tests and packaging the resulting binary[ies] and other assets (images, configuration files, libraries, etc.) into some specific format of deployable file. Note that most processes are optional and some depend on the targeted platform you are building for. As an example, packaging a Java application for Tomcat will output a .war file. Building a Win32 executable out of C code could just output the .exe program, or could also package it inside a .msi installer.

4.4 DATABASE

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized.

4.5 ENCRYPTION

Encryption is a way of scrambling data so that only authorized parties can understand the information. In technical terms, it is the process of converting human-readable plaintext to incomprehensible text, also known as ciphertext. In simpler terms, encryption takes readable data and alters it so that it appears random. Encryption requires the use of a cryptographic key: a set of mathematical values that both the sender and the recipient of an encrypted message agree on.



Although encrypted data appears random, encryption proceeds in a logical, predictable way, allowing a party that receives the encrypted data and possesses the right key to decrypt the data, turning it back into plaintext. Truly secure encryption will use keys complex enough that a third party is highly unlikely to decrypt or break the ciphertext by brute force — in other words, by guessing the key.

Data can be encrypted "at rest," when it is stored, or "in transit," while it is being transmitted somewhere else.

3.6 DATA REDUNDANCY

Data redundancy is a condition created within a database or data storage technology in which the same piece of data is held in two separate places.

This can mean two different fields within a single database, or two different spots in multiple software environments or platforms. Whenever data is repeated, it basically constitutes data redundancy.

Data redundancy can occur by accident but is also done deliberately for backup and recovery purposes.

CHAPTER 5: FEATURES AND LIMITATIONS

5.1 FEATURES

- Security (Includes: login, logout)
- Landing page
- Easy option choosing
- Adding product information
- Modifying product information
- Display all product information
- Display Individual information
- Removing product record
- Encrypted database
- Stock Calculator
- Password changing

5.2 LIMITATION

- Can only store few information
- Interface is not smart
- No reduction system existing for data redundancy
- No product record's account system
- No invoice system

CHAPTER 6: PERFORMANCE EVALUATION

6.1 PERFORMANCE EVALUATION

✓ *Successfully Completed:*

- ✓ Security successfully implemented, especially the login and logout system.
- ✓ Option choosing perfectly working
- ✓ Adding product record perfectly working
- ✓ Modifying product record perfectly working
- ✓ Displaying all product record together perfectly working
- ✓ Removing product record perfectly working
- ✓ Database is encrypted successfully
- ✓ Stock calculation is giving correctly
- ✓ Password changing is successfully working

○ *Due:*

- Storing big amount of data is unavailable
- Interface is not so smart
- There is no data redundancy system for database
- There is no account's system
- There is no invoice generating system

6.2 RESULT AND DISCUSSION

The product record management system is working fine. For implementing this project I have to use codeblocks as IDE, write codes, setup all security systems, setup database system, setup all tasks which will be performed while using, setup encryption system etc.. There are so many limitations exist in this program, but still it's fine to work.

CHAPTER 7: CONCLUSION

7.1 CONCLUSION

Our project is only a humble venture to satisfy the needs in an industry. Several user-friendly coding has also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

7.2 RECOMMENDATION AND FUTURE PLAN

1. Add more field to store information
2. making the interface smart
3. add data redundancy detecting system
4. adding account's system
5. adding invoice system

REFERENCES

- [1] IDE from <https://www.veracode.com>
- [2] Code from <https://www.techtarget.com/whatis/definition/code>
- [3] Database from <https://www.oracle.com/database/what-is-database/>
- [4] Encryption from <https://www.cloudflare.com/learning/ssl/what-is-encryption/>
- [5] Data Redundancy from <https://www.techopedia.com/definition/18707/data-redundancy>

WORD COUNT: 3480

[END]