

ACKNOWLEDGEMENT

We would like to express our profound gratitude to His Divine Soul **Padmabhushan Sri Sri Sri Dr. Balagangadharanatha MahaSwamiji** and His Holiness **Jagadguru Sri Sri Sri Dr. Nirmalanandanatha MahaSwamiji** for providing us an opportunity to complete our academics in this esteemed institution.

We would also like to express our profound thanks to **Revered Sri Sri Dr. Prakashnath Swamiji**, **Managing Director**, SJB Institute of Technology, for his continuous support in providing amenities to carry out this project in this admired institution.

We express our gratitude to **Dr. K V Mahendra Prashanth**, **Principal**, SJB Institute of Technology, for providing us an excellent facilities and academic ambience; which have helped us in satisfactory completion of project work.

We extend our sincere thanks to **Dr. Rekha B**, **Professor & Head**, Department of Information Science and Engineering; for providing us an invaluable support throughout the period of our project work.

We wish to express our heartfelt gratitude to the **mini-project coordinator, Mrs. Sridevi G M, Asst. Prof,** Department of Information Science and Engineering for her valuable guidance, suggestions and cheerful encouragement during the entire period of our project work.

Finally, we take this opportunity to extend our earnest gratitude and respect to our parents, Teaching & Non-teaching staff of the department, and all our friends, who have directly or indirectly supported us during the period of our project work.

GAURI R(1JB19IS028)
KARTHIK S(1JB19IS041)

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi-590018

A File Structures Mini-Project Report

On

MUSIC STORE MANAGEMENT SYSTEM

SUBMITTED IN PARTIAL FULFILLMENT FOR THE AWARD OF DEGREE

BACHELOR OF ENGINEERING

IN

INFORMATION SCIENCE AND ENGINEERING

SUBMITTED BY

GAURI R(1JB19IS028)
KARTHIK S(1JB19IS041)

Under the Guidance of
Mrs. Sridevi G.M.
Assistant Professor
Dept. of ISE, SJBIT

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

SJB INSTITUTE OF TECHNOLOGY

BGS HEALTH AND EDUCATION CITY,
KENGERI, BENGALURU-560060, KARNATAKA, INDIA.

2021 – 2022

|| Jai Sri Gurudev ||
Sri Adichunchanagiri Shikshana Trust ®

SJB INSTITUTE OF TECHNOLOGY

BGS Health & Education City, Kengeri, Bengaluru – 560 060

Department of Information Science & Engineering

CERTIFICATE

Certified that the Mini-project work entitled **"Music Store Management System",** is a bonafide work carried out by **GAURI R(1JB19IS028)** and **KARTHIK S(1JB19IS041),** students of **SJB Institute of Technology,** in partial fulfilment for 6th semester **File Structures Laboratory with mini project** in **INFORMATION SCIENCE AND ENGINEERING** of the **VISVESVARAYA TECHNOLOGICAL UNIVERSITY**, **BELAGAVI** during the academic year **2021-22.** It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of mini project prescribed for the said degree.

Mrs. SRIDEVI G M **Asst. Professor** **Dept. of ISE, SJBIT**	**Dr. REKHA B** **Professor & Head** **Dept. of ISE, SJBIT**

EXTERNAL VIVA

NAME OF THE EXAMINERS	SIGNATURE WITH DATE
1.	
2.	

ABSTRACT

Music and music items has become an inevitable part of our life. Music is one of the greatest so others and healers of an afflicted heart. Customers obtain these music items from music stores. An existing system is nothing but a manual Music store. A manually run Music store has so many drawbacks. Music Store Management System have to manage various musical items comes with number of models and variety. Maintaining all musical records such as create order, calculate bill, add new music in database, edit the music description, and delete any item from large music library was not an easy task on regular basis.This product has been mainly designed to overcome some of the problems faced with the manual system. The main problem faced was unnecessary delay. The previous system in use was also expensive and time consuming. In order to avoid unnecessary delay and minimize the flaws that existed in the previous system a follow up module for the existing system has been designed called the **'Music Store Management System'**.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

The aim of this project is the development of a sample centralized relational Music store application. This application has to store information of customers and artists with their products. In this context the functionality is to update, remove and insert records for the different entities. The database is built for the clerks and the managers from a Music store. Customers are ordering by phone or by email. The clerk of the databases must be able to fulfill the wishes of the customer. These wishes include finding the right album and ordering this album. This project team decided to implement the core functionality first and later to attach additional functions. The Core functionality is:

• Add, delete and update Customer information

• Add, delete, and update information about the album, artists and songs.

•Insert new and change a customer order. A customer order exists out of the customer information and the product information's.

This product has been mainly designed to overcome some of the problems faced with the manual system. The main problem faced was unnecessary delay. The previous system in use was also expensive and time consuming. In order to avoid unnecessary delay and minimize the flaws that existed in the previous system a follow up module for the existing system has been designedcalled the 'Music Store Management System'.

1.1 File structures

File Structures is the organization of data in secondary storage device in such a way that minimizes the access time and the storage space. A file structure is a combination of representations for data in files and of operations for accessing the data. A file structure allows applications to read, write and modify data. It might also support finding the data that matches some search criteria or reading through the data in some particular order.

Since the details of the representation of the data and the implementation of the operations determine the efficiency of the file structure for particular applications, improving these details can help improve secondary storage access time.

There is one important distinction in file structures and that is the difference between the logical structure of the data that is the relationships that will exist between data items independently of the way in which these relationships may actually be realized within any computer. It is this logical aspect that we will concentrate on.The physical organization is much more concerned with optimizing the use of the storage medium when a particular logical structure is stored on, or in it. Typically for every unit of physical store there will be a number of units of the logical structure (probably records) to be stored in it.

A file system consists of two or three layers. Sometimes the layers are explicitly separated, and sometimes the functions are combined. The logical file system is responsible for interaction with the user application. It provides the application program interface (API) for file operations-open, close, read etc., and passes the requested operation to the layer below it for processing. The logical file system manages open file table entries and per-process file descriptors. This layer provides file access, directory operations, and security and protection.

The second optional layer is the virtual file system. “This interface allows support for multiple concurrent instances of physical file systems, each of which is called a file system implementation”.

The third layer is the physical file system. This layer is concerned with the physical operation of the storage device (e.g. disk). It processes physical blocks being read or written. It handles buffering and memory management and is responsible for the physical placement of blocks in specific locations on the storage medium. The physical file system interacts with the device drivers or with the channel to drive the storage device.

1.1.1 Applications of File Structure

Relative to other parts of a computer, disks are slow. One can pack thousands of megabytes on a disk that fits into a notebook computer. The time it takes to get information from even relatively slow electronic RAM is about 120 ns. Getting the same information from a typical disk takes 30ms. So, the disk access is quarter of a million times longer than a memory access. Hence, disks are very slow compared to memory.

On the other hand, disks provide enormous capacity at much less cost than memory. They also keep the information stored on them when they are turned off. Tension between a disk's relatively slow access time and its enormous, non-volatile capacity is the driving force behind file structure design. Good file structure design will give us access to all the capacity without making our applications spend a lot of time waiting for the disk.

Chapter 2

SOFTWARE REQUIREMENT SPECIFICATIONS

Systems analysis is the process of observing systems for troubleshooting or development purposes. It is applied to information technology, where computer-based systems require defined analysis according to their makeup and design.

1. Software requirements

- Operating System – Windows OS
- Front End – Sublime Text (or any other text editors)
- Back end – Turbo C++ Compiler

2. Hardware Requirements

- CPU-Intel Core i3
- RAM - 8GB
- Peripherals - Keyboard, Mouse

3. Technology Used

- Front End - C++ Programming
- Controller – C++
- Backend – C++ Compiler

C++ - C++ is a general-purpose programming language created by Bjarne Stroustrup as an extension of the C programming language, or "C with Classes". The language has expanded significantly over time, and modern C++ now has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation. It is almost always implemented as a compiled language, and many vendors provide C++ compilers,
including the

Free Software Foundation, LLVM, Microsoft, Intel, Oracle, and IBM, so it is available on many platforms.

C++ was designed with a bias toward system programming and embedded, resource-constrained software and large systems, with performance, efficiency, and flexibility of use as its design highlights. C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications, including desktop applications, video games, servers (e.g. e-commerce, Web search, or SQL servers), and performance-critical applications (e.g. telephone switches or space probes).

Features

C++ Is Object Oriented Programming Language and It Is A Very Simple and Easy Language; It Is He Enhanced Form of C Programming Language, This Language Has Following Features and Here We Discuss Some Important Features of C++.

1. OOP (Object-Oriented Programming)

C++ is an object-oriented language, unlike C which is a procedural language. This is one of the most important features of C++. It employs the use of objects while programming. These objects help you implement real-time problems based on data abstraction, data encapsulation, data hiding, and polymorphism. We have briefly discussed all the 5 main concepts of object- oriented programming.

2. Platform or Machine Independent/ Portable

In simple terms, portability refers to using the same piece of code in varied environments. Let us understand this C++ feature with the help of an example. Suppose you write a piece of code to find the name, age, and salary of an employee in Microsoft Windows and for some apparent reason you want to switch your operating system to LINUX. This code will work in a similar fashion as it did in Windows.

3. Simple

When we start off with a new language, we expect to understand in depth. The simple context of C++ gives an appeal to programmers, who are eager to learn a new programming language.If you are already familiar with C, then you don't need to worry about facing any trouble while working in C++. The syntax of C++ is almost similar to that of C. Afterall C++ is referred toas "C with classes".

4. High-level programming language

It is important to note that C++ is a high-level programming language, unlike C which is a mid-level programming language. It makes it easier for the user to work in C++ as a high- level language as we can closely associate it with the human-comprehensible language, that is, English.

5. Popular

After learning C, it is the base language for many other popular programming languages which supports the feature of object-oriented programming. Bjarne Stroustrup found Simula 67, the first object-oriented language ever, lacking simulations and decided to develop C++.

6. Case sensitive

Just like C, it is pretty clear and lowercase characters keyword **'cout'** changes that the C++ programming language treats the uppercase in a different manner. For instance, the meaning of the if we write it as **'Cout'** or **"COUT"**. Other programming languages like HTML and MySQL are not case sensitive.

7. Compiler-Based

Unlike Java and Python that are interpreter-based, C++ is a compiler based language and hence it a relatively much faster than Python and Java.

8. DMA (Dynamic MemoryAllocation)

Since C++ supports the use of pointers, it allows us to allocate memory dynamically. We may even use constructors and destructors while working with classes and objects in C++.

9. Existence of Libraries

The C++ programming language offers a library full of in-built functions that make things easy for the programmer. These functions can be accessed by including suitable header files.

10. Speed

The Visual Studio IDE is a creative launching pad that you can use to edit, debug and build code , and then publish an app. An Integrated development environment (IDE) is a feature rich program that supports many aspects of software development

Chapter 3

SYSTEM DESIGN

The system design document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

Systems design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development.

A purpose statement affects the design process by explaining what the developer wants the project to do, rather than describing the project itself. The Design Document will verify that the current design meets all the explicit requirements contained in the system model as well as the implicit requirements desired by the customer.

3.1 Data Flow Diagram

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually "say" things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO.

A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish. Data flow diagrams were popularized in the late 1970s, arising from the book *Structured Design*, by computing pioneers Ed Yourdon and Larry Constantine. They based it on the "data flow graph" computation models by David Martin and Gerald Estrin. The structured design concept took off in the software engineering field, and the DFD method took off with it. It became more popular in business circles, as it was applied to business analysis, than in academic circles.

Also contributing were two related concepts:

- Object Oriented Analysis and Design (OOAD), put forth by Yourdon and Peter Coad to analyze and design an application or system.

- Structured Systems Analysis and Design Method (SSADM), a waterfall method to analyze and design information systems. This rigorous documentation approach contrasts with modern agile approaches such as Scrum and Dynamic Systems Development Method (DSDM.)

Three other experts contributing to this rise in DFD methodology were Tom DeMarco, Chris Gane and Trish Sarson. They teamed up in different combinations to be the main definers of the symbols and notations used for a data flow diagram. A data flow diagram can dive into progressively more detail by using levels and layers, zeroing in on a particular piece. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. The necessary level of detail depends on the scope of what you are trying to accomplish.

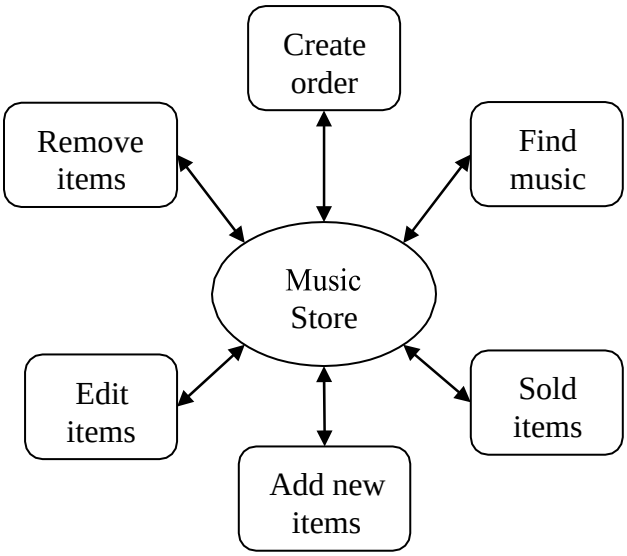


Figure 3.1 ZERO LEVEL DFD

The figure 3.1 shows zero level data flow diagram of Music Store management system. Zero level data flow diagrams concentrate mainly on overview of the whole system or process being analyzed or modelled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. It should be easily understood by a wide audience, including stakeholders, business analysts, data analysts and developers.

Chapter 4

IMPLEMENTATION

Implementation is the stage in the project where the theoretical design is turned into a working system and is giving confidence on the new system for the users that it will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover, an evaluation of change over methods. Implementation is the most important phase. The most critical stage in achieving a successful new system is giving the users confidence that the new system will work and be effective. Any system developed should be secured and protected against possible hazards.

4.1 Music Store Manager Features

The program can create order, find music, sold items, item in stock, all items, add new Item, remove Item.

1. Create Order

Here the user can create an order, choose items, delete items and buy the item and finally show the price of total items. The user can also choose same item.

2. Find Music

User can find their song with four different categories Name, Category, Type, and Artist.

3. Sold Items

It shows how many item and which items were sold.

4. Item in Stock

It shows the item that is in the stock that means the quantity of the item is more then zero.

5. All Items

Shows all the items in the database.

6. Add New Item

Add new item in the database.

7. Edit Item

Edit any item content.

8. Remove Item

Can delete any item from database.

9. Exit

Exit the program.

4.2 Main Function of Program

The switch case is used for switching between this functions

- AddNewItem();
- AllItems();
- ItemInStock();
- FindMusic();
- EditItem();
- RemoveItem();
- CreateOrder();
- SoldItems();

4.3 Structure of Record

```
struct record
{
  char id[50];
   char name[50];
   char artist[50];
   char genre[50];
```

```
    char album[50];
    char price[20];
    char quantity[20];
    char nosold[20];
};
```

4.4 Storing Records

- Array Data Structure is used to store details of records like id, Artist name, genre, album, price, quantity etc.
- Array: buff[50] is used to store all the information.
- All the information are separated by inserting '|' character at the end of each detail.

```
char buff[50];
                strcpy(buff,r.id);
                strcat(buff,"|");
                strcat(buff,r.name);
                strcat(buff,"|");
                strcat(buff,r.artist);
                strcat(buff,"|");
                strcat(buff,r.genre);
                strcat(buff,"|");
                strcat(buff,r.album);
                strcat(buff,"|");
                strcat(buff,r.price);
                strcat(buff,"|");
                strcat(buff,r.quantity);
                strcat(buff,"|");
```

```
strcat(buff,r.nosold);
                strcat(buff,"|");
                for(int i=0;i<60-strlen(buff);i++)
                    strcat(buff,"|");
                fp<<buff<<endl;
                fp.close();


record store_data::unpack(char buff[])
{
                record r;
                int i=0,j=0;
                while(buff[j]!='|')
                {
                                r.id[i++]=buff[j++];
                }
                r.id[i]='\0';
                i=0;
                j++;
                while(buff[j]!='|')
                {
                                r.name[i++]=buff[j++];
                }
                r.name[i]='\0';
                i=0;
                j++;
```

```
        while(buff[j]!='|')
        {
                r.album[i++]=buff[j++];
        }
        r.album[i]='\0';
        i=0;
        j++;
        while(buff[j]!='|')
        {
                r.price[i++]=buff[j++];
        }
        p.rice[i]='\0';
        i=0;
        j++;
        while(buff[j]!='|')
        {
                q. uantity[i++]=buff[j++];
        }
        r.quantity[i]='\0';
        i=0;
        j++;
        while(buff[j]!='|')
        {
                r.nosold[i++]=buff[j++];
        }
        r.nosold[i]='\0';
        return(r);
}
```

4.5 Function FindMusic()

- Find items from stored data.
- Four ways to find items based on : ID, Music name, Artist name, Albumname.
- By choosing one item from the list the user **can search** that particular item
- Finally the Exit Code terminates the program or go to main function or repeat the same function.

4.6 Function EditItem()

- User can edit any item.
- Firstly all the items name and ids are displayed in the console.
- Then the user can choose only one item and can edit the item.
- The user can change the item property or remain the item property same.
- Finally the Exit Code terminates the program or go to main function or repeat the same function.

Chapter 5

SNAPSHOTS

This chapter contains the execution snapshots of the project Music Management System.

1. **Main menu:** This page displays the start screen of Musicmanagement system.

Figure 5.1 Main Menu Options

2. Add new item: Here item added by specifying music ID, music name, Artist name, album name, price and quantity.

Figure 5.2 Add new item

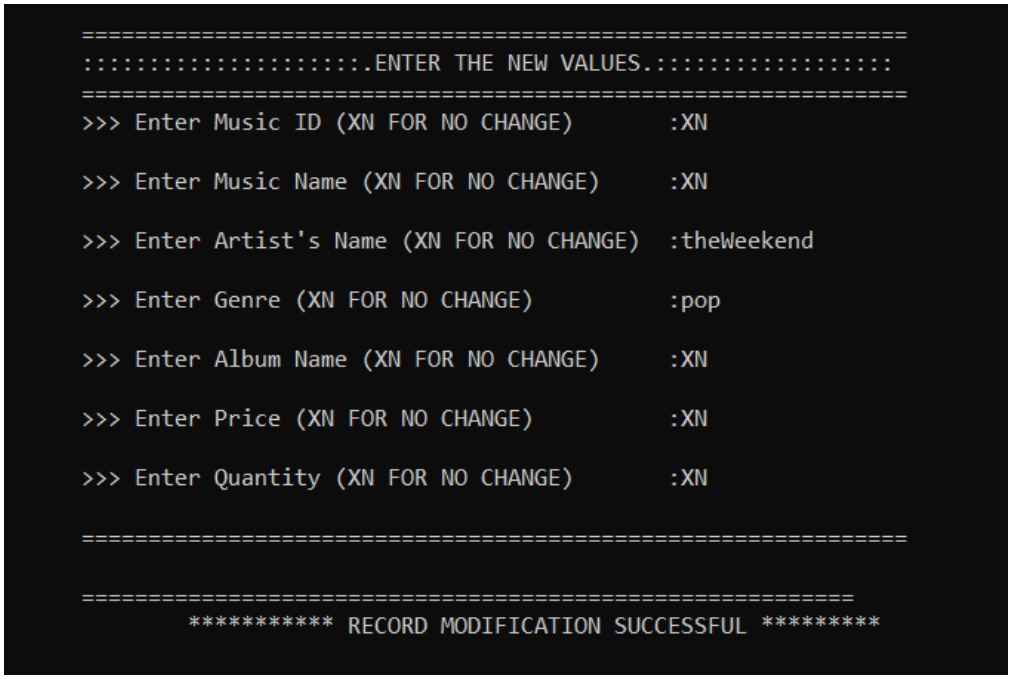
3. All Items: This page displays all the items present.

:::::::::::::::::::::::::.VH1 MUSIC RECORDS.:::::::::::::::::::::::::::::

Music ID	Music Name	Artist Name	Genre	Album Name	Price
100	OCEAN_EYES	BILLIE_EILISH	POP	DONT_SMILE	120
101	NO_PLACE	BACKSTREET_BOYS	ROCK	NO_PLACE	150
103	HEY_JUDE	THE_BEATLES	ROCK	HEY_JUDE	1000
201	LOVELY	CN	ROCK	DAA	120
500	lofi	the	weekend	bestofWeekend	99

Figure 5.3 All items

4. Edit item: user can edit any item by entering music ID**.**



```
============================================================
:::::::::::::::::::::.ENTER THE NEW VALUES.::::::::::::::::::
============================================================
>>> Enter Music ID (XN FOR NO CHANGE)       :XN

>>> Enter Music Name (XN FOR NO CHANGE)     :XN

>>> Enter Artist's Name (XN FOR NO CHANGE)  :theWeekend

>>> Enter Genre (XN FOR NO CHANGE)          :pop

>>> Enter Album Name (XN FOR NO CHANGE)     :XN

>>> Enter Price (XN FOR NO CHANGE)          :XN

>>> Enter Quantity (XN FOR NO CHANGE)       :XN

============================================================

============================================================
        ********** RECORD MODIFICATION SUCCESSFUL *********
```

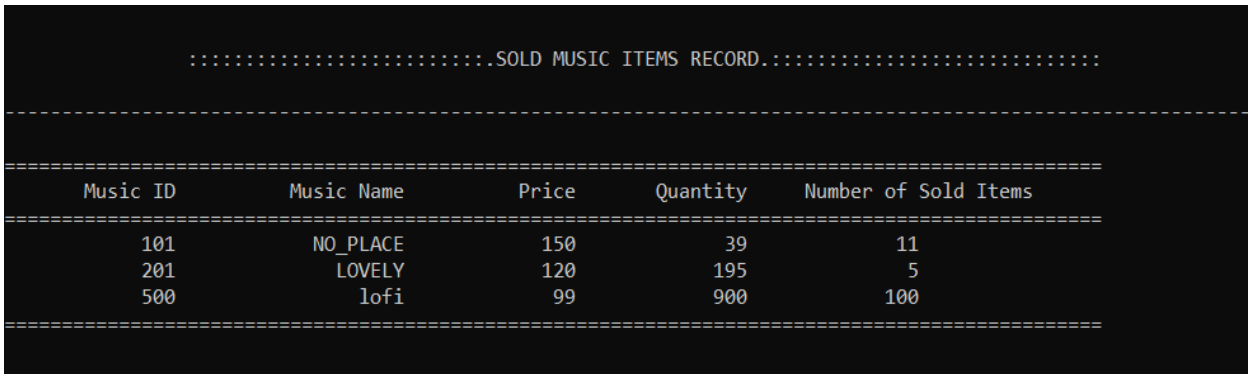
Figure 5.4 Edit item

5. Create Order: user can create order by entering music id and other details.

```
::::::::::::::::::::::::: CREATE ORDER ::::::::::::::::::::::::::
-------------------------------------------------------------------------------------------------


>>> ENTER THE MUSIC ID: 500

==============================================
::::::::::::::::::.DETAILS.::::::::::::::::::
==============================================
: *Music ID               :500
: *Music Name             :lofi
: *Price                  :99
: *Quantity               :1000
: *Number of Sold Items   :NA
==============================================


============================================================
:::::::::::::::::::::.CREATING ORDER.:::::::::::::::::::::::
============================================================
>>>> *Enter Quantity       :100

>>>> *Price            :Rs. 9900
============================================================


>>> Enter Customer's Name         :Ankit Ekka
>>> Enter Customer's Phone Number :                    >>> Enter Customer's Address      :Bangalore

=============================================================================================
:::::::::::::::::::::::::::::.VH1   MUSIC   STORE.::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::.BILLING RECEIPT.::::::::::::::::::::::::::::::::::::::::::::
=============================================================================================
:.Customer's Name :Ankit                          Phone Number:Ekka
:.                    Address :Bangalore
---------------------------------------------------------------------------------------------
 Music Name                Quantity         Price (in Rs.)       Total Price(in Rs.)
---------------------------------------------------------------------------------------------
lofi                     100                      99                    9900
=============================================================================================

=========================================================

        ***** ORDER CREATED SUCCESSFUL *****
```

Figure 5.5 Create Order

6. Sold music: user can view all the sold music items.

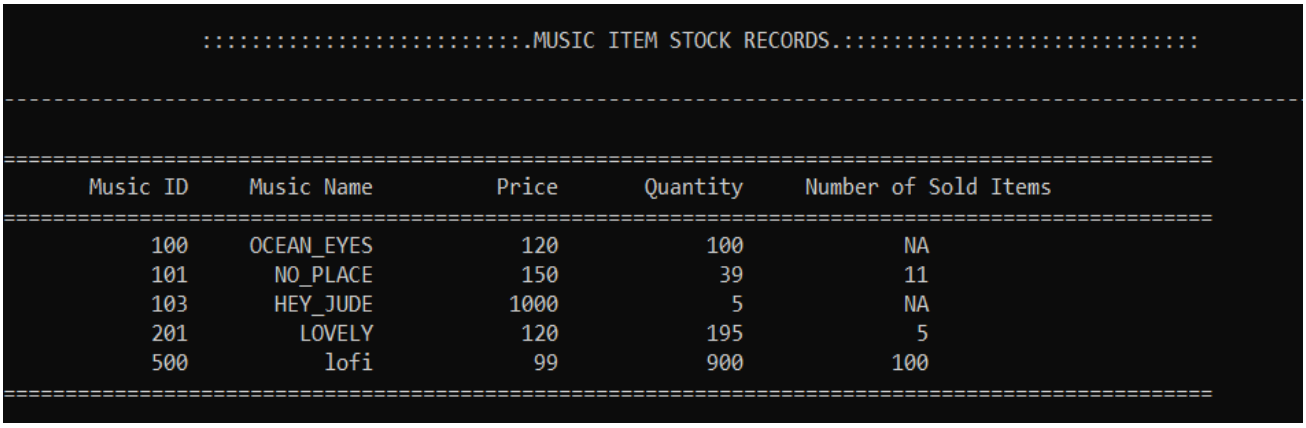


:::::::::::::::::::::::::.SOLD MUSIC ITEMS RECORD.:::::::::::::::::::::::::::

Music ID	Music Name	Price	Quantity	Number of Sold Items
101	NO_PLACE	150	39	11
201	LOVELY	120	195	5
500	lofi	99	900	100

Figure 5.6 Sold Music

7. Items in Stock: user can view all the music items that are in stock.



:::::::::::::::::::::::::.MUSIC ITEM STOCK RECORDS.:::::::::::::::::::::::::::

Music ID	Music Name	Price	Quantity	Number of Sold Items
100	OCEAN_EYES	120	100	NA
101	NO_PLACE	150	39	11
103	HEY_JUDE	1000	5	NA
201	LOVELY	120	195	5
500	lofi	99	900	100

Figure 5.7 Items in Stock

8. Find music: user can search records based on music id, music name, album name, artist name.

Figure 5.8 Find music

9. Remove Item: user can delete music record.

Figure 5.8 Remove item

Chapter 6

CONCLUSION AND FUTURE ENHANCEMENTS

Conclusion

The software “Music store management system” reduces the considerable drawbacks like burden of human labor, portable defect and errors. This software saves time and provides 24 hour accessibility even from a remote place. Programs are menu driven which help even a newcomer to use the system with little training. Testing has been done with actual data and system is much better than the existing one. The system is done with an insight into the necessary modification that may require in the future. Hence the system can be maintained successfully, without much rework.

Future Enhancements

Future enhancements can be made by implementing the project using B+ tree structure. It allows not only direct access but also sequential access which makes it even more efficient than B-tree. The project can further be enhanced by using Hashing technique to further reduce the number of accesses and improve the performance. Hashing technique allows record access in only one or very few runs with the help of a hashing function. Using such advanced techniques for better indexing, we can implement complex functions such as comprehensive balance sheets and budgeting.

REFERENCES

[1] Michael J Folk, Bill Zoellick, Greg Riccardi: File Structures – An Object Oriented Approach with C++, 3 rd. Edition, Pearson Education, 1998

[2] https://www.geeksforgeeks.com/

[3] https://www.stackoverflow.com/

[4] https://sanfoundry.com/