VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI, KARNATAKA

A Mini-Project Report on

"MEDICAL EQUIPMENT RENTAL SYSTEM"

Submitted in partial fulfilment for the award of degree of
Bachelor of Engineering In

INFORMATION SCIENCE AND ENGINEERING
By

MOHAMED IDRIS SHUJA (4NN20IS024)

Under the Guidance of

Sri. PRASANNA KUMAR G

Assistant Professor
Department of Information Science and Engineering
NIEIT, Mysuru-18

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING NIE
INSTITUTE OF TECHNOLOGY
MYSURU-570018
2022-2023

NIE INSTITUTE OF TECHNOLOGY

#50(part), Hootagalli Industrial area, Koorgalli Village Mysuru-18

ESTD-2008

DEPARTMENT
OF
INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

Certified that the **FILE STRUCTURES** mini-project work entitled **"MEDICAL EQUIPMENT RENTAL SYSTEM"** is a bonafide work carried out by **MOHAMED IDRIS SHUJA (4NN20IS024),** in partial fulfilment for the award of the degree of Bachelor of Engineering in Information Science and Engineering of the Visvesvaraya Technological University, Belagavi during the year 2022-2023. The project report has been approved as it satisfies the academic requirements with respect to the Project work prescribed for Bachelor of Engineering Degree.

Signature of the Guide	**Signature of the HOD**
Sri. PRASANNA KUMAR G	Dr. NANDINI M S
Assistant Professor	Associate Professor & Head
NIEIT, Mysuru	NIEIT, Mysuru

External Viva

Name of the examiners	**Signature with Date**
1……………………………..	1……………………………..
2……………………………..	2……………………………..

ACKNOWLEDGEMENT

We Sincerely owe our gratitude to all people who helped and guided us in completing this mini-project work.

We are thankful to Dr. ROHINI NAGAPADMA, Principal, NIEIT, Mysore, for having supported us in academic endeavours.

We are thankful to Dr. NANDINI M S, Head, Department of Information Science and Engineering, NIEIT for providing us timely suggestion, encouragement and support to complete this mini-project.

We would like to sincerely thank our mini-project guide Sri. PRASANNA KUMAR G for providing relevant information, valuable guidance and encouragement to complete this mini-project.

We would also like to thank all our teaching and non-teaching staff members of the Department. We are grateful to the college for keeping labs open whenever required and providing us Systems and required software.

We are always thankful to our parents for their valuable support and guidance in every step. We express our deepest gratitude and indebted thanks to NIEIT which has provided us an opportunity in fulfilling our most cherished desire of reaching our goal.

Yours Sincerely,

MOHAMED IDRIS SHUJA (4NN20IS024)

ABSTRACT

In today's healthcare landscape, the availability and accessibility of medical equipment play a crucial role in delivering effective patient care. However, procuring and maintaining a diverse range of medical equipment can be a significant challenge for healthcare facilities, particularly smaller clinics, temporary medical setups, or under-resourced regions. The concept of a Medical Equipment Rental System has emerged as a viable solution to address these challenges and enhance access to healthcare resources.

Medical Equipment Rental System offers a wide range of medical equipment for rent, making healthcare more accessible and affordable.

The goal is to provide various medical equipment on rent at your doorstep so your loved ones can stay in the comfort of home with the best equipment by their side.

A substantial amount of equipment is out-of-service in the developing world. This limits access to important medical procedures. Given the considerable out-of-service rates, the most sustainable solution is the renting (rather than donating or purchasing) equipment in order to increase access to medical procedures.

The project has an admin section and a customer section. Admin will have the preinstalled admin id and password through which medical equipment can be added or deleted in the catalogue which is visible in the customer section.

Admin has the authority of updating the medical equipment in both home care and hospital care sections and the customers can view the catalogue of the available equipment and place an order for the desired medical equipment.

Overall, the Medical Equipment Rental System presents a promising solution to bridge the gap between healthcare facilities' equipment needs and the availability of resources. By leveraging technology and streamlining the rental process, this system contributes to improving patient care, reducing costs, and promoting efficient resource allocation in the healthcare industry. As healthcare demands continue to evolve, the Medical Equipment Rental System offers a flexible and adaptable approach to meet the equipment requirements of diverse healthcare settings.

ABSTRACT

In today's healthcare landscape, the availability and accessibility of medical equipment play a crucial role in delivering effective patient care. However, procuring and maintaining a diverse range of medical equipment can be a significant challenge for healthcare facilities, particularly smaller clinics, temporary medical setups, or under-resourced regions. The concept of a Medical Equipment Rental System has emerged as a viable solution to address these challenges and enhance access to healthcare resources.

Medical Equipment Rental System offers a wide range of medical equipment for rent, making healthcare more accessible and affordable.

The goal is to provide various medical equipment on rent at your doorstep so your loved ones can stay in the comfort of home with the best equipment by their side.

A substantial amount of equipment is out-of-service in the developing world. This limits access to important medical procedures. Given the considerable out-of-service rates, the most sustainable solution is the renting (rather than donating or purchasing) equipment in order to increase access to medical procedures.

The project has an admin section and a customer section. Admin will have the preinstalled admin id and password through which medical equipment can be added or deleted in the catalogue which is visible in the customer section.

Admin has the authority of updating the medical equipment in both home care and hospital care sections and the customers can view the catalogue of the available equipment and place an order for the desired medical equipment.

Overall, the Medical Equipment Rental System presents a promising solution to bridge the gap between healthcare facilities' equipment needs and the availability of resources. By leveraging technology and streamlining the rental process, this system contributes to improving patient care, reducing costs, and promoting efficient resource allocation in the healthcare industry. As healthcare demands continue to evolve, the Medical Equipment Rental System offers a flexible and adaptable approach to meet the equipment requirements of diverse healthcare settings.

LIST OF CONTENTS

Chapter 1

INTRODUCTION

File structures are an essential component of computer systems that organize and manage data stored in files. A file structure defines the logical organization, access methods, and storage formats for files, enabling efficient storage, retrieval, and manipulation of data. Different file structures are designed to cater to specific requirements and optimize data access based on the application's needs.
"File structure" refers to the format of the label and data blocks and of any logical record control information. The organization of a given file may be sequential, relative, or indexed. 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.

1.1. DESCRIPTION TO FILE STRUCTURE

Early Work assumed that files were on tape. Access was sequential and the cost of access grew in direct proportion to the size of the file. As files grew very large, unaided sequential access was not a good solution. Disks allowed for direct access. Indexes made it possible to keep a list of keys and pointers in a small file that could be searched very quickly. With the key and pointer, the user had direct access to the large, primary file. As indexes also have a sequential flavor, when they grew too much, they also became difficult to manage.
The idea of using tree structures to manage the index emerged in the early 60's. However, trees can grow very unevenly as records are added and deleted, resulting in long searches requiring many disk accesses to find a record.
In 1963, researchers came up with the idea of AVL trees for data in memory. AVL trees, however, did not apply to files because they work well when tree nodes are composed of single records rather than dozens or hundreds of them. In the 1970's came the idea of B-Trees which require an O(logk N) access time where N is the number of entries in the file and k, the number of entries indexed in a single block of the B-Tree structure --> B-Trees can guarantee that one can find one file entry among millions of others with only 3 or 4 trips to the disk. Retrieving entries in 3 or 4 accesses is good, but it does not reach the goal of accessing data with a single request.

Chapter 1

INTRODUCTION

File structures are an essential component of computer systems that organize and manage data stored in files. A file structure defines the logical organization, access methods, and storage formats for files, enabling efficient storage, retrieval, and manipulation of data. Different file structures are designed to cater to specific requirements and optimize data access based on the application's needs.

"File structure" refers to the format of the label and data blocks and of any logical record control information. The organization of a given file may be sequential, relative, or indexed. 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.

1.1. Description to file structure

Early Work assumed that files were on tape. Access was sequential and the cost of access grew in direct proportion to the size of the file. As files grew very large, unaided sequential access was not a good solution. Disks allowed for direct access. Indexes made it possible to keep a list of keys and pointers in a small file that could be searched very quickly. With the key and pointer, the user had direct access to the large, primary file. As indexes also have a sequential flavor, when they grew too much, they also became difficult to manage.

The idea of using tree structures to manage the index emerged in the early 60’s. However, trees can grow very unevenly as records are added and deleted, resulting in long searches requiring many disk accesses to find a record.

In 1963, researchers came up with the idea of AVL trees for data in memory. AVL trees, however, did not apply to files because they work well when tree nodes are composed of single records rather than dozens or hundreds of them. In the 1970’s came the idea of B-Trees which require an O(logk N) access time where N is the number of entries in the file and k, the number of entries indexed in a single block of the B-Tree structure --> B-Trees can guarantee that one can find one file entry among millions of others with only 3 or 4 trips to the disk. Retrieving entries in 3 or 4 accesses is good, but it does not reach the goal of accessing data with a single request.

From early on, Hashing was a good way to reach this goal with files that do not change size greatly over time.

Recently, Extendible Dynamic Hashing guarantees one or at most two disk accesses no matter how big a file becomes.

1.2. Advantages of File Structure

File structures offer several advantages in organizing and managing data within computer systems. Some of the key advantages include:

- Data Organization: File structures provide a systematic way to organize and structure data within files. This allows for logical grouping and categorization of related data elements, making it easier to manage and access the information.
- Efficient Data Access: Different file structures offer various access methods that optimize data retrieval based on specific application requirements. For example, sequential file structures are efficient for accessing data sequentially, while random access file structures allow direct and quick access to specific records. By utilizing appropriate file structures, data access can be performed efficiently, leading to faster processing times and improved system performance.
- Flexibility: File structures offer flexibility in terms of data storage and retrieval methods. They can adapt to different data types, sizes, and access patterns. File structures can be customized and optimized based on specific application needs, allowing for efficient and tailored data management.
- Scalability: File structures provide scalability options, allowing for the management of large volumes of data. As the amount of data increases, file structures can be designed to handle the growing data sets, ensuring efficient storage and retrieval operations without sacrificing performance.
- Data Integrity: File structures often incorporate mechanisms to ensure data integrity and consistency. For example, indexed file structures maintain index structures that help maintain data integrity by enforcing unique key constraints and providing efficient search capabilities. This helps prevent data duplication and maintains data integrity throughout the file.
- Data Sharing and Collaboration: File structures facilitate data sharing and collaboration among multiple users or applications. By providing controlled access and sharing mechanisms, file structures enable concurrent access to files, allowing multiple users to read and modify data simultaneously while ensuring data integrity and consistency.

Compatibility: File structures are widely supported by various operating systems and software applications, making them compatible across different platforms and environments. This compatibility ensures that data stored in file structures can be easily accessed, shared, and processed by different systems or applications.

1.3. Application on File Structure

File structures are widely used in various applications to organize and manage data efficiently. Here are some common applications that benefit from different file structures:

- Databases: Databases employ file structures to store and retrieve large volumes of structured data. Different file structures such as B-trees, hash indexes, and sequential files are used to optimize data access, ensure data integrity, and support efficient querying and retrieval operations.
- File Systems: File systems in operating systems use file structures to manage files and directories. Hierarchical file structures, such as tree structures, allow for easy navigation and organization of files, while indexing structures enable fast file retrieval based on names, metadata, or attributes.
- Web Applications: Web applications rely on file structures to store web pages, multimedia content, and user data. Content Management Systems (CMS) use file structures to organize and manage website assets, including HTML, CSS, JavaScript files, images, and documents.
- Media Libraries: Media libraries, such as music or video libraries, utilize file structures to categorize and manage media files. Hierarchical folder structures, combined with metadata indexing, allow for efficient searching, sorting, and browsing of media content.
- Document Management Systems: Document management systems utilize file structures to organize and store documents. Hierarchical structures and metadata indexing enable users to search, retrieve, and version documents efficiently, promoting collaboration and document control within organizations.
- Computer-Aided Design (CAD) Systems: CAD systems use file structures to store and manage design files, including blueprints, drawings, and 3D models. Specialized file structures optimize data retrieval, revision control, and collaboration among multiple designers working on the same project.
- Scientific Data Analysis: Scientific applications dealing with large datasets, such as weather forecasting or genetic research, rely on file structures to manage and analyze

data. File structures, such as NetCDF (Network Common Data Form) or HDF5 (Hierarchical Data Format), offer efficient storage, retrieval, and manipulation of multidimensional data arrays.

- Version Control Systems: Version control systems, like Git, utilize file structures to track changes and manage different versions of source code files. Branching, merging, and history tracking mechanisms are built on top of file structures to enable collaboration and code management.

1.4. Introduction to Medical Equipment Rental System

The Medical Equipment Rental System is a software-based solution designed to facilitate the rental and management of medical equipment in healthcare settings. This system provides a platform that connects healthcare providers in need of temporary access to medical devices with equipment suppliers or rental agencies. By streamlining the rental process, the Medical Equipment Rental System aims to enhance access to vital healthcare resources and improve patient care.

Medical Equipment Rental System offers a wide range of medical equipment on rent at your doorstep so your loved ones can stay in the comfort of home with the best equipment by their side. Since health care needs are higher in senior age brackets, a corresponding growth in demand for this type of services can be expected.

From the perspective of hospital management, the health-care equipment provided includes maintenance activities that aim at ensuring operational quality of service with the objective of minimizing costs.

The system offers several features and functionalities that simplify the rental process. Healthcare providers can browse through a catalog of available medical equipment, which can range from simple diagnostic tools to complex life support systems. The system provides detailed information about each equipment, including specifications, availability, and rental terms.

1.5. Objectives

The objective of the Medical Equipment Rental System is to provide healthcare facilities with a comprehensive and efficient solution for accessing medical equipment on a rental basis. The system aims to achieve the following objectives:

- Enhance Access to Medical Equipment: The primary objective of the system is to improve access to medical equipment for healthcare providers, particularly for those who may face challenges in acquiring and maintaining a wide range of equipment. By offering a centralized platform for equipment rental, the system ensures that healthcare facilities can easily access the necessary medical devices on a temporary basis.
- Optimize Resource Utilization: The system aims to optimize the utilization of medical equipment by enabling multiple healthcare providers to share and rent the same equipment when it is not in use. This helps to maximize the efficiency and utilization of resources, reducing idle time and unnecessary duplication of equipment.
- Increase Flexibility and Scalability: The system provides healthcare facilities with the flexibility to scale their equipment needs based on demand. It allows healthcare providers to quickly rent additional equipment during peak periods, expand service capabilities, or replace malfunctioning devices without the need for significant upfront investments.
- Simplify Rental Process: The system aims to streamline the rental process by providing a user-friendly interface for healthcare providers and equipment suppliers. It simplifies equipment selection, order placement, and rental agreement management, reducing administrative burdens and saving time for both parties involved.
- Ensure Transparency and Accountability: The system promotes transparency and accountability by providing detailed information about equipment availability and rental terms. It allows healthcare providers to track the status of their rental orders, ensuring transparency in the rental process and facilitating effective communication between healthcare providers and equipment suppliers.

Chapter 2

REQUIREMENT SPECIFICATION

A System requirements specification is a document or set of documentation that describes the features and behaviour of a system of software application. It includes a variety element that attempts to define the intended functionality required by the customer to satisfy their different uses. There are two types of requirements: Hardware and Software requirements.

In addition to specifying how the system should behave, the specification also defines at a high level the main business processes that will be supported, what simplifying assumptions have been made and what key performance parameters will need to be met by the system . This document describes the nature of a project, software or application. This includes the purpose, scope, functional and non-functional requirements, software and hardware requirements of the project.

2.1. Hardware REQUIREMENTS

It captures the complete hardware requirements for the system or a portion of the system. These requirements include the minimum processor speed, memory, and disc space required to install windows. In almost all cases, you will want to make sure that your hardware exceeds these requirements to provide adequate performance for the services and applications running on the server.

Processor: Ryzen 5

RAM: 8 GB and above

Hard-disc: 194 MB and above

2.2 Software Requirements

The software requirements are description of features and functionalities of the target system. Requirements convey the expectation of the users from the software product. The requirements can be obvious or hidden, known or unknown, expected or unexpected from client's point of view.

It is a document created by system analyst after the requirements are collected from various stake holders. It defines how the intended software will interact with hardware external interface, speed of operation, response time of the system, portability of the software across various platforms, maintainability, speed of recovery after crashing, security, quality, limitations etc.

Operation system: Windows 11

Software used: Code Blocks

Programming language: C++

Chapter 3

SYSTEM ANALYSIS

System analysis is the process of studying and understanding a system in order to identify its goals, requirements, components, and interactions. It involves gathering and analyzing information about the current system, identifying problems or inefficiencies, and proposing solutions for improvement. System analysis is a crucial phase in the development or enhancement of any system, whether it's a software application, a business process, or an organizational structure.

3.1 Analysis Of The Application

Med Street offers a wide range of medical equipment for rent, making healthcare more accessible and affordable.

The goal is to provide various medical equipment on rent at your doorstep so your loved ones can stay in the comfort of home with the best equipment by their side. The computerization system does the job monitoring the record in easy and effective manner as stated below:

Effectively handle medical equipment related data.

Keep records of home care as well as hospital care equipment.

Generate a bill.

3.2. Operations Performed On A File

- **ADDING NEW EQUIPMENT:** This feature allows admin to add a new equipment in file. Information such as name of the medical equipment, ID, type, other details and price should be provided, and the data is stored in file.
- **DISPLAY EQUIPMENT :** This option shows the list of as name of the medical equipment, ID, type, other details and price stored in file.
- **DELETE EQUIPMENT :** In this operation we can perform deletion in the current section (home care of hospital care). You can delete the equipment just by providing the ID number.
- **VIEW INDEX LIST :** In this operation we can check the indexing of the file .

Chapter 4

IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus, it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective. The implementation stage involves careful planning, investigation of the existing system and its constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

4.1 Code Snippet

```
#include<stdio.h>
#include<stdlib.h>
#include<iostream>
#include<fstream>
#include<string>
#include<sstream>
#include<ctime>
using namespace std;

class homecare_a
{
  string        buffer,aname,adetails,aid,aprice;
string aidlist[100];      int count,addlist[100];
public:

     void   createindex();                  string
extractaid(string);
            void sortindex();
void insert();              void
display_equipment();
```

```
 void print();              void
search(string);                        int
searchindex(string);
                        void remove(string);

                         /* index list */
void index_create();                string
aid_extract(string);
                        void index_sort();
} a;

class hospitalcare_b
{            public:                    string
bid,bprice,bdetails,bname;
string  bidlist[100];                   int
count,addlist[100];                   void
createindex();          void insert();
        void
display_b_record();
void  print();                  void
remove(string);          int
searchindex(string);
void                search(string);
void sortindex();          string extractbid(string);

        /* index list */
                         void index_create();
string bid_extract(string);
void index_sort();
}b;
```

```
void
homecare_a::index_create()   {
fstream file;   int pos;

            string   buffer,aid;        count=-1;
file.open("homecare_equipment.txt",ios::in);
 while(!file.eof())
            {
           pos=file.tellg();
buffer.erase();            getline(file,buffer);
           if(buffer[0]!='*')
           {
if(buffer.empty())
break;
aid=aid_extract(buffer);
aidlist[++count]=aid;
                     addlist[count]=pos;            }
}      file.close();      index_sort();
buffer.erase();  }   string
homecare_a::aid_extract(string buffer)
 {      string aid;      int
i=0;      aid.erase();
while(buffer[i]!='|')
aid+=buffer[i++];
return aid;
 }
 void homecare_a::index_sort()
{      int i,j,tempadd;      string
tempid;
for(i=0;i<=count;i++)
     {
           for(j=i+1;j<=count;j++)
```

```
          {
if(aidlist[i]>aidlist[j])

              {                    tempid=aidlist[i];
                                      aidlist[i]=aidlist[j];
                   aidlist[j]=tempid;
tempadd=addlist[i];

addlist[i]=addlist[j];
                   addlist[j]=tempadd;
              }
          }
     }

     cout<<"\n-------INDEXLIST--------\n";
cout<<"\n                              ID\tINDEX";
for(i=0;i<=count;i++)
cout<<"\n"<<aidlist[i]<<"\t"<<addlist[i]<<endl;

 }

 void hospitalcare_b::index_create() {
    fstream     file;
int pos;
         string buffer,bid;
    count=-1;
    file.open("hospitalcare_equipment.txt",ios::in);
    while(!file.eof())
           {
      pos=file.tellg();
buffer.erase();
getline(file,buffer);
if(buffer[0]!='*')
```

```
        {
if(buffer.empty())
             break;           bid=bid_extract(buffer);
bidlist[++count]=bid;
addlist[count]=pos;         }     }     file.close();
index_sort();     buffer.erase();  }   string
hospitalcare_b::bid_extract(string buffer)
 {      string bid;      int
i=0;      bid.erase();
while(buffer[i]!='|')
bid+=buffer[i++];
return bid;
 }
 void hospitalcare_b::index_sort()
 {      int i,j,tempadd;
string tempid;
for(i=0;i<=count;i++)
     {
           for(j=i+1;j<=count;j++)
           {
if(bidlist[i]>bidlist[j])

                {
tempid=bidlist[i];
                                             bidlist[i]=bidlist[j];
                      bidlist[j]=tempid;
                      tempadd=addlist[i];                     addlist[i]=addlist[j];
                      addlist[j]=tempadd;
                }
           }
     }

     cout<<"\n-------INDEXLIST--------\n";
```

```
cout<<"\n ID\tINDEX";

    for(i=0;i<=count;i++)
cout<<"\n"<<bidlist[i]<<"\t"<<addlist[i]<<endl<<endl;

 }

void homecare_a::createindex()
          {
fstream file;   int pos;
          string   buffer,aid;       count=-1;
file.open("homecare_equipment.txt",ios::in);
 while(!file.eof())
          {
         pos=file.tellg();
buffer.erase();          getline(file,buffer);
         if(buffer[0]!='*')
         {
if(buffer.empty())
break;
aid=extractaid(buffer);
aidlist[++count]=aid;
addlist[count]=pos;
}      }      file.close();
sortindex();
buffer.erase();
 }

 void homecare_a::insert()
 {
    string buffer,aname,adetails,aid,aprice;
    int pos;
    fstream fp;
```

```
          cout << "\nEnter Home Care Equipment ID\n";
     cin >> aid;          cout << "\nEnter Home Care
Equipment Name\n";     cin >> aname;
     cout << "\nEnter Home Care Equipment Price\n";     cin
>> aprice;
     cout << "\nEnter Home Care Equipment Details\n";
cin >> adetails;

     buffer=aid+"|"+aname+"|"+adetails+"|"+aprice+"          $\n";
fp.open("homecare_equipment.txt",ios::out|ios::app);     pos=fp.tellp();
fp<<buffer;
fp.close();
aidlist[++count]=aid;
addlist[count]=pos;
     sortindex();
     cout<<"\nHome Care Equipment Added Successfully....\n";
 }

 string homecare_a::extractaid(string buffer)
 {     string aid;     int
i=0;     aid.erase();
while(buffer[i]!='|')
aid+=buffer[i++];
return aid;
 }

 void homecare_a::sortindex()
{     int i,j,tempadd;     string
tempid;
for(i=0;i<=count;i++)
     {
          for(j=i+1;j<=count;j++)
          {
if(aidlist[i]>aidlist[j])
               {
                    tempid=aidlist[i];

aidlist[i]=aidlist[j];                    aidlist[j]=tempid;                    tempadd=addlist[i];

addlist[i]=addlist[j];
addlist[j]=tempadd;                 }
          }
     }

 }

 void  homecare_a::search(string  key)
{         int  pos=0,t;         string  buffer;
fstream   file;                   buffer.erase();
pos=searchindex(key);       if(pos==-1)
cout<<"Record not found\n";
else if(pos>=0)
     {
          file.open("homecare_equipment.txt");
t=addlist[pos];            file.seekp(t,ios::beg);
getline(file,buffer);
cout<<"Record                     found\n";
//cout<<"\nRecord:\n"<<buffer;
file.close();
     }
 }

 int homecare_a::searchindex(string key)
 {
    int                       low=0,high=count,mid=0,flag=0;
```

```
while(low<=high)
    {
        mid=(low+high)/2;
        if(aidlist[mid]==key)
        {               flag=1;
break;
        }
else
if(aidlist[mid]<key)               low=mid+1;
else
if(aidlist[mid]>key)               high=mid-1;
    }      if(flag)           return
mid;
else          return
-1;
 }

 void homecare_a::remove(string key)
 {
         fstream           fp;
char delch='*';            int
pos=0,i,address;
         string   buffer,aid,aname,aprice,adetails;       fstream
file;
         pos=searchindex(key);    if(pos>=0)
         {
     file.open("homecare_equipment.txt");
     address=addlist[pos];
file.seekp(address,ios::beg);
     file.put(delch);
file.close();
     cout<<"Record    has     been    deleted\n";      for(i=pos;i<count;i++)
     {
        aidlist[i]=aidlist[i+1];
        addlist[i]=addlist[i+1];
        }
count--;   }
else
        cout<<"Record not found\n";
  }
  void homecare_a::display_equipment(){
     cout<<"The Home Care Equipment Details are\n";

      std::ifstream f("homecare_equipment.txt");

    if (f.is_open())
         std::cout << f.rdbuf();
 }
 void homecare_a::print()
 {
 cout<<"Equipment    ID:"<<aid<<"    Equipment    Name:"<<aname<<"    Details:"<<adetails<<"
Price:"<<aprice<<endl;
 }

void hospitalcare_b::createindex() {
    fstream file;
int pos;           string
buffer,bid;      count=-
1;
file.open("hospitalcare
_equipment.txt",ios::in)
;
while(!file.eof())
            {
        pos=file.tellg();
buffer.erase();       getline(file,buffer);
```

```
        if(buffer[0]!='*')
        {
          if(buffer.empty())
            break;
bid=extractbid(buffer);
bidlist[++count]=bid;
addlist[count]=pos;
}    }    file.close();
sortindex();
buffer.erase(
);
 }

 void hospitalcare_b::insert()
 {
   string buffer,bname,bdetails,bid,bprice;
int pos;     fstream fp;

  cout  <<  "\nEnter  Hospital  Care  Equipment  ID\n";
cin  >>  bid;           cout  <<  "\nEnter  Hospital  Care
Equipment Name\n";    cin >> bname;
   cout << "\nEnter Hospital Care Equipment Details\n";
cin >> bdetails;
   cout  <<  "\nEnter  Hospital  Care  Equipment  Price\n";
cin >> bprice;

   buffer=bid+"|"+bname+"|"+bdetails+"|"+bprice+"          $\n";
fp.open("hospitalcare_equipment.txt",ios::out|ios::app);
   pos=fp.tellp();
fp<<buffer;    fp.close();    bidlist[++count]=bid;
addlist[count]=pos;    sortindex();    cout<<"\nHospital Care
Equipment Added Successfully....\n";   }
```

```
 string hospitalcare_b::extractbid(string buffer)
 {      string bid;      int
i=0;      bid.erase();
while(buffer[i]!='|')
bid+=buffer[i++];
return bid;  }
 void hospitalcare_b::sortindex()
{      int i,j,tempadd;      string
tempid;
for(i=0;i<=count;i++)
     {
           for(j=i+1;j<=count;j++)
           {
if(bidlist[i]>bidlist[j])

                {
tempid=bidlist[i];
                                             bidlist[i]=bidlist[j];
                      bidlist[j]=tempid;                      tempadd=addlist[i];

addlist[i]=addlist[j];
                      addlist[j]=tempadd;
                }
           }
     }

 }

 void hospitalcare_b::search(string key)
 {      int pos=0,t;      string buffer;
fstream file;      buffer.erase();
pos=searchindex(key);      if(pos==-
1)        cout<<"Record not
found\n";
    else if(pos>=0)
    {
         file.open("hospitalcare_equipment.txt");
         t=addlist[pos];          file.seekp(t,ios::beg);
getline(file,buffer);
cout<<"Record          found\n";          //cout<<"\nRecord:"<<buffer;
file.close();
    }
 }

 int hospitalcare_b::searchindex(string key)
 {
    int                    low=0,high=count,mid=0,flag=0;
while(low<=high)
    {
         mid=(low+high)/2;
if(bidlist[mid]==key)
         {              flag=1;
break;
         }
else
if(bidlist[mid]<key)                low=mid+1;
else
if(bidlist[mid]>key)                high=mid-1;
    }      if(flag)            return
mid;
else          return
-1;
 }

 void hospitalcare_b::remove(string key)
 {
           fstream           fp;    char
delch='*';        int
pos=0,i,address;
string
buffer,bid,bname,bprice,
bdetails;
           fstream           file;
pos=searchindex(key);     if(pos>=0)
           {
      file.open("hospitalcare_equipment.txt");
address=addlist[pos];       file.seekp(address,ios::beg);
      file.put(delch);
file.close();
cout<<"Record
has       been
deleted\n";
for(i=pos;i<count;i++)
      {
      bidlist[i]=bidlist[i+1];
addlist[i]=addlist[i+1];
      }
count--;  }  else
      cout<<"Record not found\n";

  }
  void               hospitalcare_b::display_b_record(){
cout<<"The   Hospital   Care   Equipment   Details   are\n";
                                                   std::ifstream
f("hospitalcare_equipment.txt");

   if (f.is_open())
       std::cout << f.rdbuf();
 }   void
hospitalcare_b::print()
 {
            cout<<"Equipment ID:"<<bid<<" Equipment Name:"<<bname<<" Equipment
Details: "<<bdetails<<" Equipment Price:" <<bprice<<endl;
 }

/*------------------Main()----*/



 int main()  {      int
          ch;
string key;
    string custname, custphno, custaddr, aprice,
bprice;      char ans;      homecare_a a;
    hospitalcare_b b;



    a.createindex();
    b.createindex();

    mainpage:

         while(1)
         {
         int ach,bch,cch;
         string username="admin",password="admin",u,p;             cout<<"\n\t*******
MEDICAL EQUIPMENT RENTAL SYSTEM *******\n";             cout<<"\n\t\tEnter
\n\t\t1. Admin\n\t\t2. Customers for Home Care\n\t\t3. Customers for Hospital Care\n\t\t0.
Exit\n";
```

```
cout<<"\n\t**************************************************************\n";
cout<<"\nEnter your choice\n";
            cin>>ch;
            switch(ch)
            {
case 1: cout<<"Enter your username\n";
                            cin>>u;
cout<<"Enter  your    password\n";                       cin>>p;
if(u==username && p==password)
{                           int adch;                          cout<<"\nWelcome
Admin\n";                           admin:
while(1)                                                 {
cout<<"----------------------\n"; cout<<"|
ADMIN SECTION
|\n";
cout<<"----------------------\n"; cout<<"\n1.
Home Care Equipment
\n2. Hospital Care Equipment\n(0) <<--
BACK\n"; cout<<"\nEnter your choice\n";
cin>>adch; switch(adch)
{        case        1:
while(1)
{                                                                                                                                                         int
a_ch;
cout<<"----------------------** HOME CARE EQUIPMENT **-------------------------\n";
 cout<<"\n1. Add Equipment \t\t\t2. Display Equipment\n3. Delete Equipment\t\t4. View index
List\n\n\t\t\t(0) <<-- BACK\n";
cout<<"------------------------------------------------------------\n";
 cout<<"\nEnter your choice\n";
cin>>a_ch; switch(a_ch)
{
case 1:a.insert();
```

```
break;
        case 2:a.display_equipment();
               break;
        case 3:cout<<"Enter Equipment ID :";
cin>>key;
               a.remove(key);
               cout<<"File is Updated\n";

break;
case
4:a.index_create();
break;            case  0:goto
admin;              break;
        default: cout<<"Invalid choice";
            }         }
break;            case 2:
while(1)
            {
int b_ch;
            cout<<"-----------------------** HOSPITAL CARE EQUIPMENT **--------------------------\n";
cout<<"\n1.    Add    Equipment     \t\t\t2.  Display         Equipment\n3.          Delete
Equipment\t\t4. View index list\n\n\t\t\t(0) <<-- BACK\n";            cout<<"-------------------------------
-------------------------------\n";
 cout<<"\nEnter your choice\n";
            cin>>b_ch;      switch(b_ch)
                        {
                                    case 1: b.insert();
            break;          case      2:
b.display_b_record();
break;
          case 3: cout<<"Enter Equipment ID :";                    cin>>key;
                              b.remove(key);
                             cout<<"File is Updated :\n";
```

```
                              break;
          case 4: b.index_create();
                              break;
           case 0: goto admin;
                              break;
default:            cout<<"Invalid
choice";
                    }
           }
           break;

           case 3: while(1)
           {

           break;



           case 0: goto mainpage;
                              break;
           default:            cout<<"Invalid
           choice";
                              }
                    }
 }}
else
           cout<<"Invalid
           Credentials\n";
break;
```

```
        case 2:cout<<"Welcome Customer for Home Care\n";
                    cout<<"----------------------------\n";
                                    cout<<"|      CUSTOMER CREDENTIALS      |\n";
                            cout<<"----------------------------\n";


                cout<<"1. Enter Customer Details \n2. Display Catalogue\n\n\t\t(0) <<-BACK\n";
cout<<"------------------------\n";                cout<<"Enter your choice \n";
                cin>>ach;                switch(ach)
                {
                  case 1: cout<<"Enter Name\n";
                        cin>>custname;
                        cout<<"Enter Phone Number\n";
                        cin>>custphno;                        cout<<"Enter
Address\n";
                        cin>>custaddr;                        cout<<"---------\n";
                        cout<<"Do you want to see the catalogue? Press Y or N \n";
cin>>ans;
                        if  (ans   =='Y'||ans=='y')
{

a.display_equipment();                          }
                        cout<<"\n\n";
                        cout<<"-------------------------------------------------\n";
cout<<"----------------PLACING       THE    ORDER-----------------\n";
cout<<"\n\n";
                        cout<<"Enter the ID of the Equipment you want to order\n";
cin>>key;

a.search(key);
                        cout<<"\n\n\n";                        cout<<"------------------------------
------------------\n";                    cout<<"------------------BILL GENERATED------
------------\n";
```

```
                    cout<<"---------------------------------------------
---\n";                 cout<<"NAME: ";                    cout<<custname;
cout<<"\n";                 cout<<"PHONE: ";                    cout<<custphno;
cout<<"\n";                 cout<<"ADDRESS: ";                    cout<<custaddr;
                    cout<<"\n";
                    cout<<"EQUIPMENT ID: ";
                    cout<<key;
                    cout<<"\n";
                    cout<<"EQUIPMENT DETAILS: ";
                    a.search(key);                    cout<<"\n";
                    cout<<"--------------------------------------------------------------\n";


break;

                case 2: a.display_equipment();
                cout<<"------------------------------------------------------------------\n";
cout<<"Enter  the     ID     of     the     Equipment     you     want    to     order\n";
cin>>key;

a.search(key);                    break;
case 0: goto
mainpage;
                default: cout<<"Invalid choice";
                }           break;

                  case 3: cout<<"Welcome Customer for Hospital Care\n";
                  cout<<"----------------------\n";
                                   cout<<"|   CUSTOMER CREDENTIALS     |\n";
                              cout<<"----------------------\n";

                 cout<<"1. Enter Customer Details \n2. Display Catalogue\n\n\t\t(0) <<-BACK\n";
cout<<"------------------------\n";                cout<<"Enter your choice \n";
```

```
                    cin>>bch;                   switch(bch)
                    {
                      case 1: cout<<"Enter Name\n";
                              cin>>custname;
cout<<"Enter Phone Number\n";
                              cin>>custphno;                          cout<<"Enter
Address\n";
                              cin>>custaddr;                          cout<<"---------\n";
                              cout<<"Do you want to see the catalogue? Press Y or N \n";
cin>>ans;
                              if (ans =='Y'||ans=='y')
                              {
b.display_b_record();                              }
                              cout<<"\n\n";
                              cout<<"------------------------------------------------\n";
cout<<"----------------PLACING        THE     ORDER----------------\n";
cout<<"\n\n";
                              cout<<"Enter the ID of the Equipment you want to order\n";
cin>>key;


b.search(key);
                              cout<<"\n\n\n";                          cout<<"-----------------------------
------------------\n";                          cout<<"------------------BILL GENERATED------
------------\n";
                              cout<<"---------------------------------------------
---\n";                          cout<<"NAME:  ";                          cout<<custname;
cout<<"\n";
                              cout<<"PHONE:   ";                          cout<<custphno;
cout<<"\n";                          cout<<"ADDRESS:  ";
cout<<custaddr;
                              cout<<"\n";
                              cout<<"EQUIPMENT ID: ";
                              cout<<key;                          cout<<"\n";
```

```
                        cout<<"EQUIPMENT DETAILS: ";
                        b.search(key);                    cout<<"\n";
                        cout<<"--------------------------------------------------------------\n";

break;
                                case 2: b.display_b_record();                              cout<<"--------
------------------------------------------------
\n";                                    cout<<"Enter the ID of the Equipment you want to order\n";
cin>>key;
                        b.search(key);
                                         break;
     case 0: goto mainpage;
default: cout<<"Invalid choice";
                  }            break;

          case 0:exit(0);                default:
cout<<"Invalid Choice";
          }
          }
}
```

Chapter 5

TESTING

5.1 Introduction To Testing:

Software testing is an investigation conducted to provide stake holders with information about the quality of the software product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test technique include the process of executing a program or application with the intent of finding software bugs (errors and defects) and verifying that the software product is fit for use.

5.2 Stages In The Implementation Of Testing:

Unit Testing:

During this first round of resting, the program is submitted to assessments that focus on specific units or components of the software to determine whether each one is fully functional. Unit testing, a testing technique using which individual modules are tested to determine if there are any issues by the developer himself.
The main aim is to isolate each unit of the system to identify, analyze and fix the defects. Advantages such as improves design and allows better refactoring of code, reduces cost of testing as defects are captured in very early phase.
Techniques such as: Black box testing- using which the user interface are tested,

White box testing-used to test each one of those functions' behavior is tested, gray box testingused to execute tests.

Integration Testing:

Integration testing allows individuals the opportunity to combine all the units within a program and test them as a group. This testing level is designed to find interfaces defects between the modules. This is beneficial because it determines how efficiently the units are running together. Keep in mind that no matter how efficiently each unit is running, if they aren't properly integrated, it will affect the functionality of the software program. The purpose of integration testing is to verify the functional, performance, and reliability between the modules that are integrated.

System testing:

System testing is the first level in which the complete application is tested as a whole. The goal at this level is to evaluate whether the system has complied with all of the outlined requirements and to see that it meets quality standards. System testing is undertaken by independent testers who haven't played a role in developing the program. This testing is performed in an environment that closely mirrors production. System testing is very important because it verifies that the application meets the technical, functional and business requirements that were set by the customer.

Acceptance Testing:

The final level, acceptance testing is conducted to determine whether the system is ready for release. During the software development life cycle, requirements changes can sometimes be misinterpreted in a fashion that does not meet the intended needs of the users. During this final phase, the user will test the system to find out whether the application meets their business needs. Once this process has been completed and the software has passed, the program will then be delivered to production.

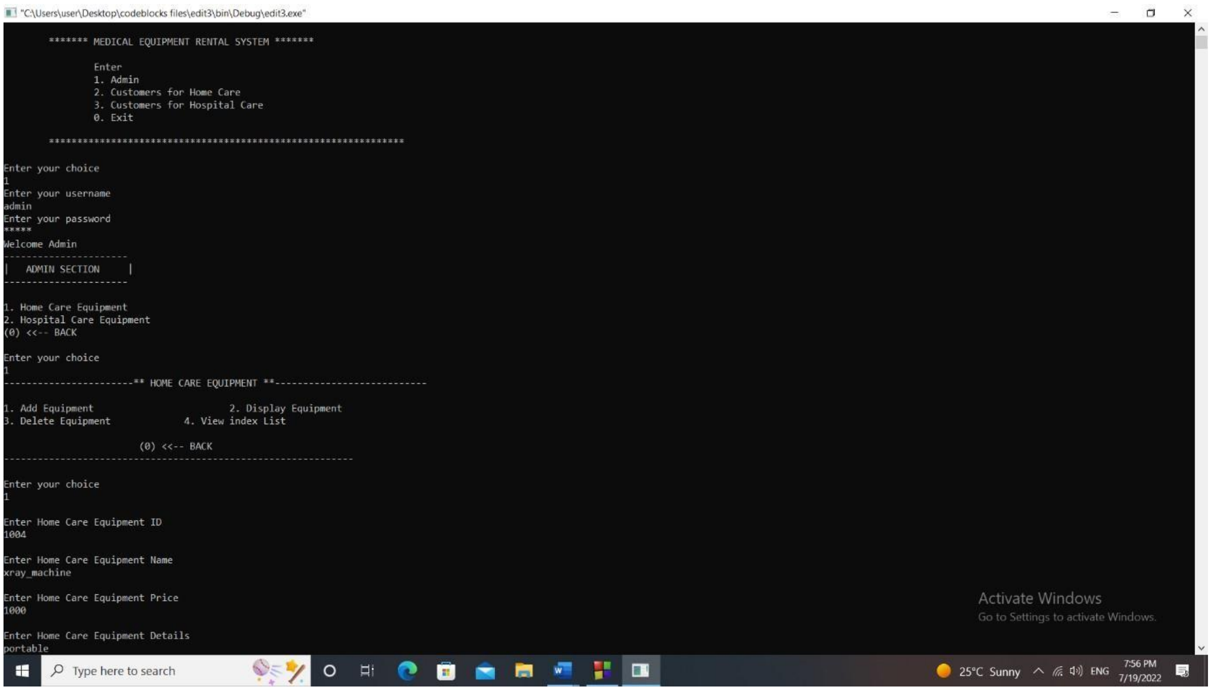
5.3 Testing Result

Sl	Test case	Input Data	Steps to executed test case	Expected result	Pas s / Fail
1	Correct username or password	After entering the username and password	After successful login, the user should see the main menu	The user sees the main menu	pass
2	Any of the option chosen	Enter the option	The user should enter the respective option working	The user enters the Respective option working	pass

3	Add records	Data to be filled in each field	When clicked enter all the data is stored in record	All data is stored in record	pass
4	Modify record	After entering the DID	Fetch the DID entered and show the old details and collect new data to that.	After collecting data, all details is stored in record.	pass
5	Delete record	Enter the DID	If the entered DID is valid the respective record with that DID gets deleted.	Deleted	pass
6	Search record	Enter the DID	If the entered DID is valid the respective record with that DID gets fetched	Displays the concerned record details.	pass

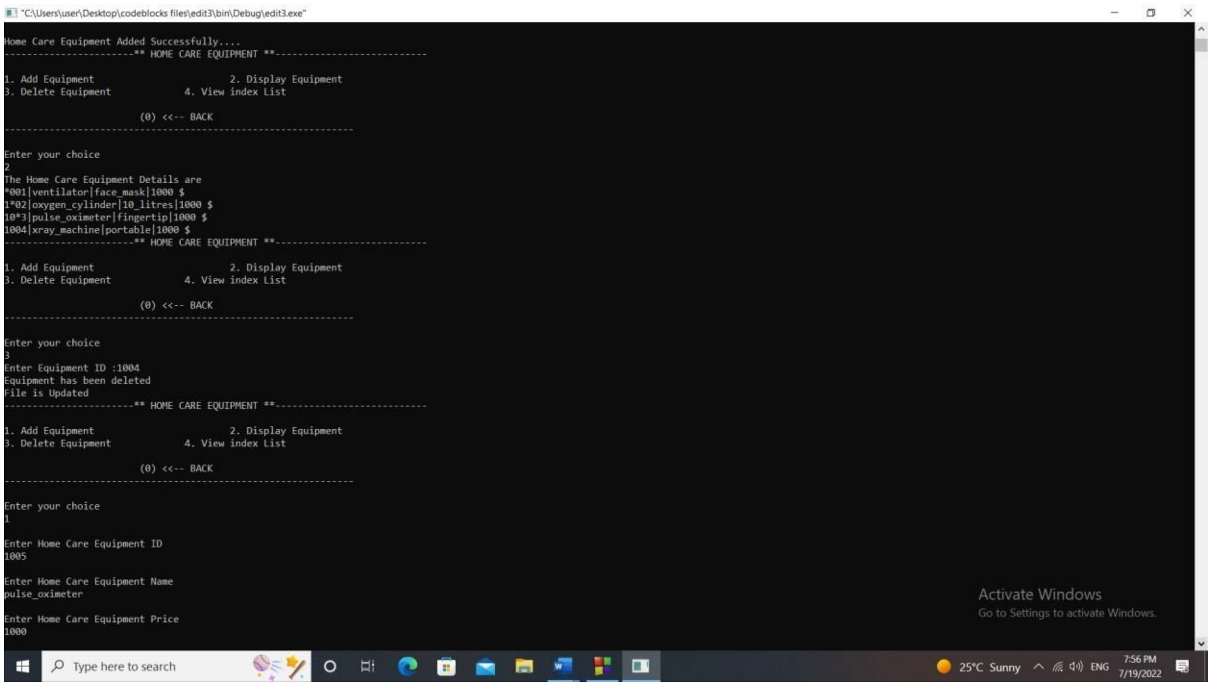
Several errors were detected and rectified and the whole project is working as it should give proper output and high efficiency.

SNAPSHOTS



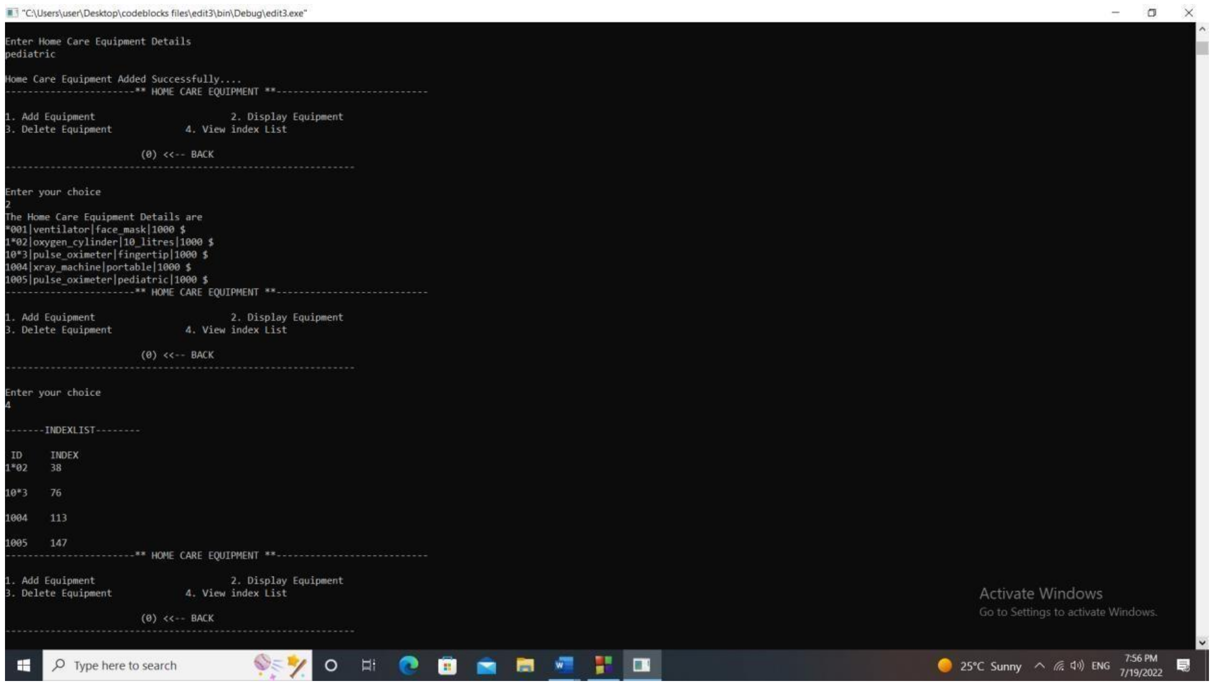
```
"C:\Users\user\Desktop\codeblocks files\edit3\bin\Debug\edit3.exe"

      ******* MEDICAL EQUIPMENT RENTAL SYSTEM *******

              Enter
              1. Admin
              2. Customers for Home Care
              3. Customers for Hospital Care
              0. Exit

      ****************************************************************

Enter your choice
1
Enter your username
admin
Enter your password
*****
Welcome Admin
----------------------
|   ADMIN SECTION     |
----------------------

1. Home Care Equipment
2. Hospital Care Equipment
(0) <<-- BACK

Enter your choice
1
----------------------** HOME CARE EQUIPMENT **--------------------------

1. Add Equipment                       2. Display Equipment
3. Delete Equipment            4. View index List

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
1

Enter Home Care Equipment ID
1004

Enter Home Care Equipment Name
xray_machine

Enter Home Care Equipment Price
1000

Enter Home Care Equipment Details
portable
```

Snapshots 1: Admin Section- Home Care: Adding the Equipment



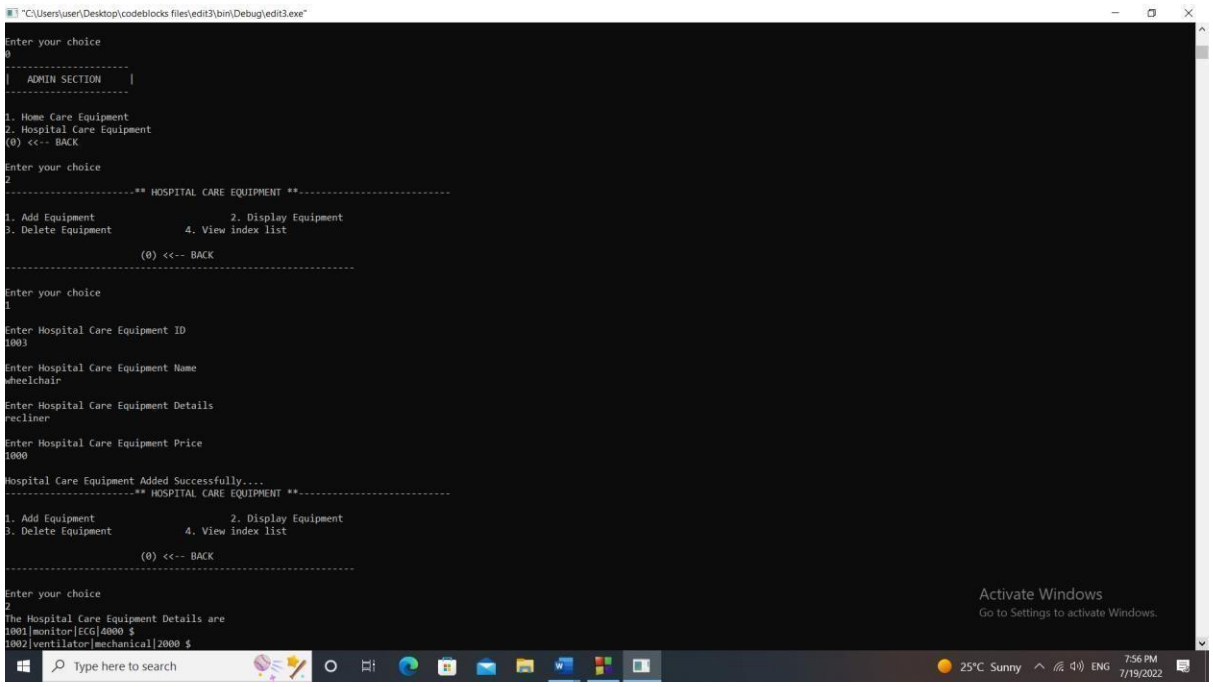
```
"C:\Users\user\Desktop\codeblocks files\edit3\bin\Debug\edit3.exe"
Home Care Equipment Added Successfully....
----------------------** HOME CARE EQUIPMENT **--------------------------

1. Add Equipment                       2. Display Equipment
3. Delete Equipment            4. View index List

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
2
The Home Care Equipment Details are
*001|ventilator|face_mask|1000 $
1*02|oxygen_cylinder|10_litres|1000 $
10*3|pulse_oximeter|fingertip|1000 $
1004|xray_machine|portable|1000 $
----------------------** HOME CARE EQUIPMENT **--------------------------

1. Add Equipment                       2. Display Equipment
3. Delete Equipment            4. View index List

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
3
Enter Equipment ID :1004
Equipment has been deleted
File is Updated
----------------------** HOME CARE EQUIPMENT **--------------------------

1. Add Equipment                       2. Display Equipment
3. Delete Equipment            4. View index List

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
1

Enter Home Care Equipment ID
1005

Enter Home Care Equipment Name
pulse_oximeter

Enter Home Care Equipment Price
1000
```

Snapshots 2: Admin Section- Home Care: Deleting the Equipment



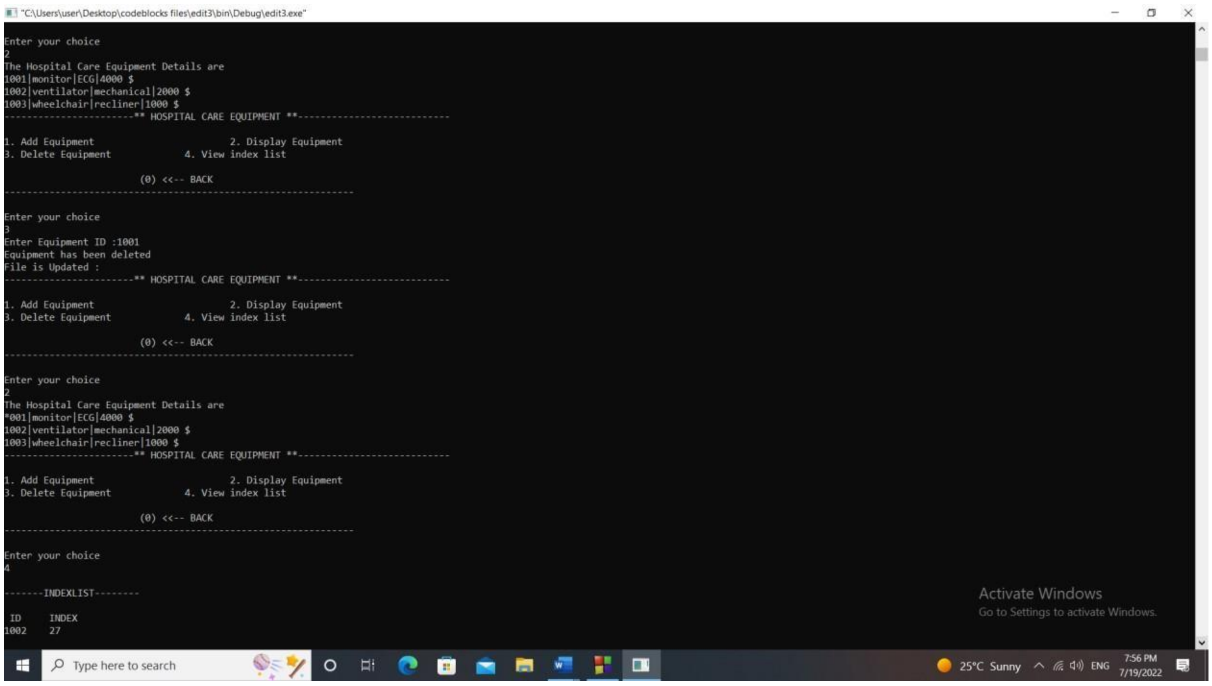
```
"C:\Users\user\Desktop\codeblocks files\edit3\bin\Debug\edit3.exe"

Enter Home Care Equipment Details
pediatric

Home Care Equipment Added Successfully....
----------------------** HOME CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index List

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
2
The Home Care Equipment Details are
*001|ventilator|face_mask|1000 $
1*02|oxygen_cylinder|10_litres|1000 $
10*3|pulse_oximeter|fingertip|1000 $
1004|xray_machine|portable|1000 $
1005|pulse_oximeter|pediatric|1000 $
----------------------** HOME CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index List

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
4

-------INDEXLIST--------

 ID     INDEX
1*02    38

10*3    76

1004    113

1005    147
----------------------** HOME CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index List

                        (0) <<-- BACK
-------------------------------------------------------------
```

Snapshots 3: Admin Section- Home Care: View Index List



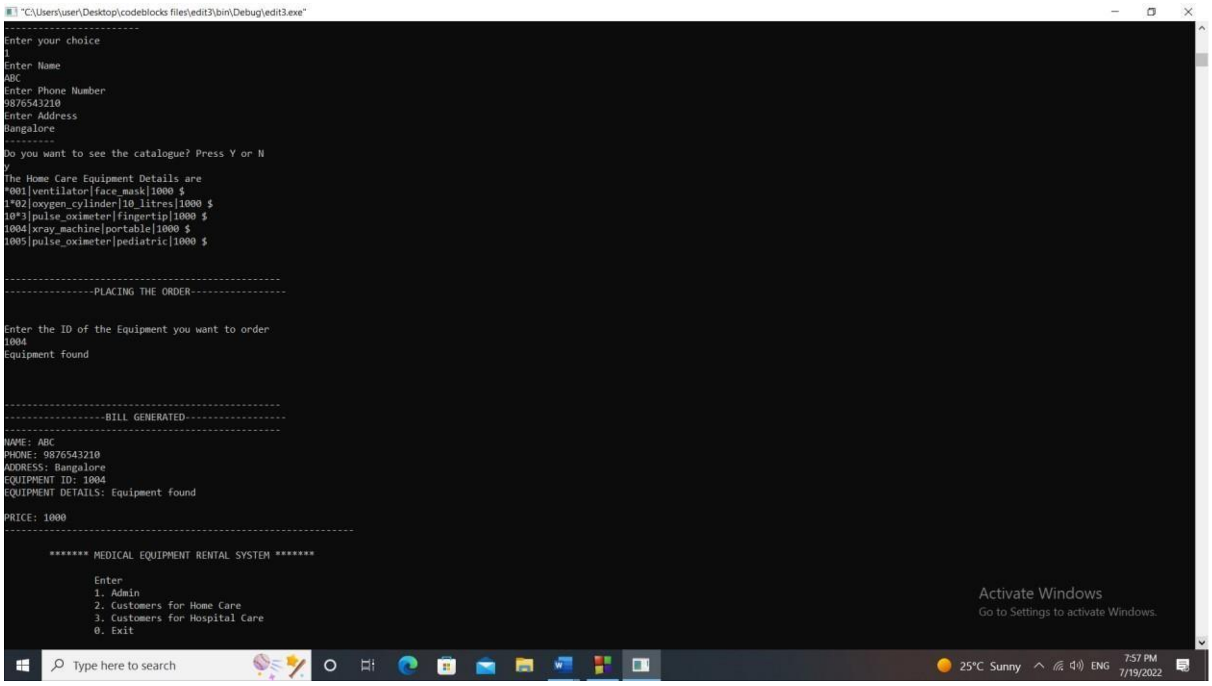
```
"C:\Users\user\Desktop\codeblocks files\edit3\bin\Debug\edit3.exe"

Enter your choice
0
----------------------
|   ADMIN SECTION     |
----------------------

1. Home Care Equipment
2. Hospital Care Equipment
(0) <<-- BACK

Enter your choice
2
----------------------** HOSPITAL CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index list

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
1

Enter Hospital Care Equipment ID
1003

Enter Hospital Care Equipment Name
wheelchair

Enter Hospital Care Equipment Details
recliner

Enter Hospital Care Equipment Price
1000

Hospital Care Equipment Added Successfully....
----------------------** HOSPITAL CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index list

                        (0) <<-- BACK
-------------------------------------------------------------

Enter your choice
2
The Hospital Care Equipment Details are
1001|monitor|ECG|4000 $
1002|ventilator|mechanical|2000 $
```

Snapshots 4: Admin Section- Hospital Care: Adding the Equipment



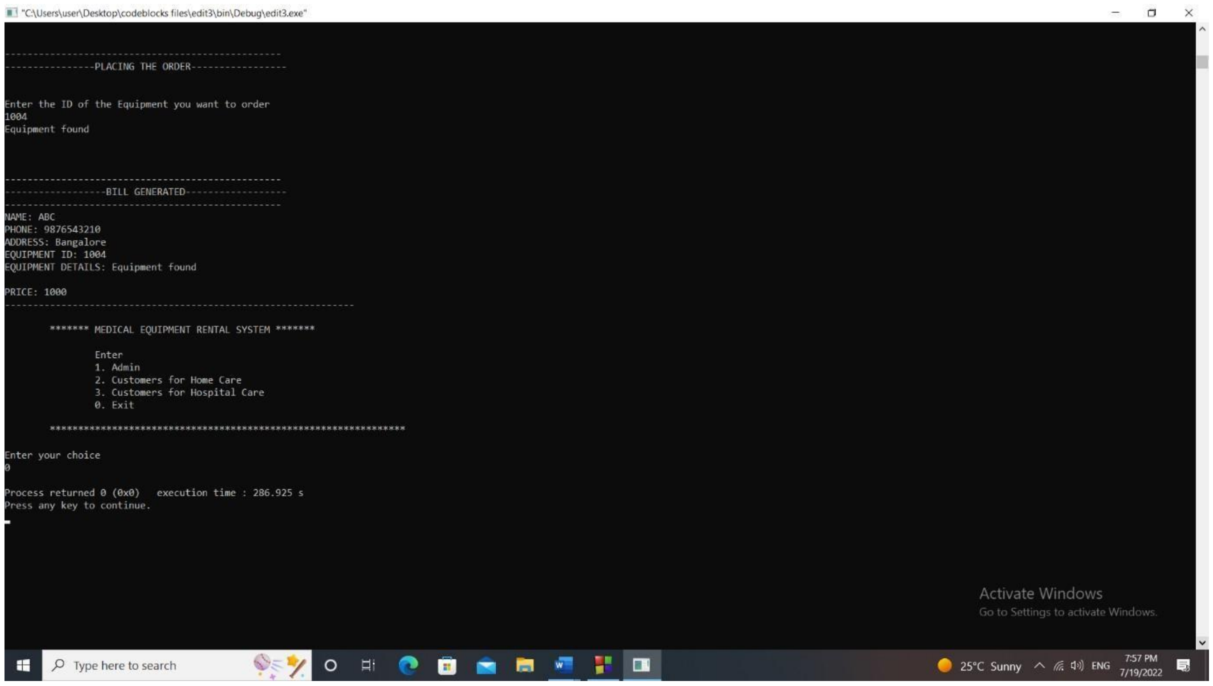
```
"C:\Users\user\Desktop\codeblocks files\edit3\bin\Debug\edit3.exe"
Enter your choice
2
The Hospital Care Equipment Details are
1001|monitor|ECG|4000 $
1002|ventilator|mechanical|2000 $
1003|wheelchair|recliner|1000 $
----------------------** HOSPITAL CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index list

                        (0) <<-- BACK
--------------------------------------------------------------

Enter your choice
3
Enter Equipment ID :1001
Equipment has been deleted
File is Updated :
----------------------** HOSPITAL CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index list

                        (0) <<-- BACK
--------------------------------------------------------------

Enter your choice
2
The Hospital Care Equipment Details are
*001|monitor|ECG|4000 $
1002|ventilator|mechanical|2000 $
1003|wheelchair|recliner|1000 $
----------------------** HOSPITAL CARE EQUIPMENT **--------------------------

1. Add Equipment                        2. Display Equipment
3. Delete Equipment             4. View index list

                        (0) <<-- BACK
--------------------------------------------------------------

Enter your choice
4

-------INDEXLIST--------

 ID     INDEX
1002    27
```

Snapshots 5: Admin Section- Hospital Care: Displaying, Deleting and Viewing the Index List
Snapshots 6: Customer Section Initial Part



```
"C:\Users\user\Desktop\codeblocks files\edit3\bin\Debug\edit3.exe"
------------------------
Enter your choice
1
Enter Name
ABC
Enter Phone Number
9876543210
Enter Address
Bangalore
---------
Do you want to see the catalogue? Press Y or N
y
The Home Care Equipment Details are
*001|ventilator|face_mask|1000 $
1*02|oxygen_cylinder|10_litres|1000 $
10*3|pulse_oximeter|fingertip|1000 $
1004|xray_machine|portable|1000 $
1005|pulse_oximeter|pediatric|1000 $


-------------------------------------------------
----------------PLACING THE ORDER-----------------


Enter the ID of the Equipment you want to order
1004
Equipment found



-------------------------------------------------
------------------BILL GENERATED------------------
-------------------------------------------------
NAME: ABC
PHONE: 9876543210
ADDRESS: Bangalore
EQUIPMENT ID: 1004
EQUIPMENT DETAILS: Equipment found

PRICE: 1000
--------------------------------------------------------------

        ******* MEDICAL EQUIPMENT RENTAL SYSTEM *******

                Enter
                1. Admin
                2. Customers for Home Care
                3. Customers for Hospital Care
                0. Exit
```

Snapshots 7: Customer Placing Order


"C:\Users\user\Desktop\codeblocks files\edit3\bin\Debug\edit3.exe"

```
-------------------------------------------------
----------------PLACING THE ORDER-----------------


Enter the ID of the Equipment you want to order
1004
Equipment found



-------------------------------------------------
------------------BILL GENERATED------------------
-------------------------------------------------
NAME: ABC
PHONE: 9876543210
ADDRESS: Bangalore
EQUIPMENT ID: 1004
EQUIPMENT DETAILS: Equipment found

PRICE: 1000
--------------------------------------------------------------

        ******* MEDICAL EQUIPMENT RENTAL SYSTEM *******

                Enter
                1. Admin
                2. Customers for Home Care
                3. Customers for Hospital Care
                0. Exit

        ***************************************************************

Enter your choice
0

Process returned 0 (0x0)   execution time : 286.925 s
Press any key to continue.
```

Snapshots 8: Bill Generated

CONCLUSION

In conclusion, the Medical Equipment Rental System offers a comprehensive solution to address the challenges associated with accessing and managing medical equipment in healthcare facilities. By providing a centralized platform for equipment rental, the system enhances access to vital healthcare resources, improves operational efficiency, and ultimately contributes to the delivery of quality patient care.

Since health care needs are higher in senior age brackets, a corresponding growth in demand for this type of services can be expected.

Transparency and accountability are ensured through the system's tracking and monitoring capabilities, providing healthcare providers with real-time visibility into the status of their rental orders and facilitating accurate billing and payment processes.

By implementing the Medical Equipment Rental System, healthcare facilities can achieve operational efficiency, cost-effectiveness, and improved patient care. It eliminates the burden of equipment ownership and maintenance, enabling healthcare providers to focus on their core competencies while ensuring access to the necessary medical devices.

Overall, the Medical Equipment Rental System streamlines the rental process, optimizes resource utilization, enhances flexibility, and promotes transparency in accessing and managing medical equipment. It is a valuable tool that contributes to the efficient and effective delivery of healthcare services.

FUTURE ENHANCEMENT

The project has been developed in a very short period of time and all efforts have been taken so that this project is very efficient in its execution there still exists some scope of improvement in our project. We will add the modification feature for more convenience of customer. We will improve our system for clearer view.

REFERENCE

- Michael J. Folk, Bill Zoellick, Greg Riccardi: File Structures-An Object OrientedApproach with C++, 3rd Edition, Pearson Education, 1998.
- K.R. Venugopal, K.G. Srinivas, P.M. Krishnaraj: File Structures Using C++, TataMcGraw-Hill, 2008.
- Raghu Ramakrishan and Johannes Gehrke: Database Management Systems, 3rd Edition, McGraw Hill, 200