

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belgaum-590 014, Karnataka.



A

Mini Project Report

On

“TELECOM BILLING SYSTEM”

Submitted in the partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF ENGINEERING IN INFORMATION SCIENCE AND ENGINEERING

Submitted by

KUMMARA PAVAN KALYAN 1EW20IS0140

VISHWAS HOLLA 1EW20IS088

Under the Guidance of

Ms. Sanjitha
Assistant Professor,
Dept of ISE,EWIT



DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

EAST WEST INSTITUTE OF TECHNOLOGY

BANGALORE - 560 091

2022-2023

EAST WEST INSTITUTE OF TECHNOLOGY

Sy. No.63, Off. Magadi Road, Vishwaneedam Post, Bangalore - 560 091
(Affiliated to Visvesvaraya Technological University, Belgaum)

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the mini project work entitled “**TELECOME BILLING SYSTEM**” presented by (**Kummara Pavan Kalyan 1EW20IS040, Vishwas Holla 1EW20IS088**), bonafied students of **EAST WEST INSTITUTE OF TECHNOLOGY**, Bangalore in partial fulfillment for the award of **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University**, Belgaum during the year **2022-2023**. It is certified that all corrections/suggestions indicated have been in corporated in the report. The mini project has been approved as it satisfies the academic requirements in respect of mini project prescribed for the said degree.

Signature of Guide

Ms.Sanjitha

Asst Prof, Dept. of ISE
EWIT, Bangalore

Signature of HOD

Dr Suresh M B

Prof & Head, Dept. of ISE
EWIT, Bangalore

Signature of Principal

Dr. K Channakeshavalu

Principal
EWIT, Bangalore

External Viva

Name of the Examiners

Signature with date

1. _____

2. _____

ABSTRACT

The project is based on “ INDEXING” of Primary Index. The database used in this project is “**TELECOM BILLING SYSTEM**” has been done by CodeBlocks IDE using “C++”. The project main intension is to build indexing using one lakh records, the records is given in the form of “TXT” file format . In primary Indexing we use primary key has Index number. We are performing Insertion, Deletion, Modify, Search, Payment and Build index for a dataset. The data file is ordered on a key field. The key field is generally the primary Key of the relation

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped me in carrying out this project work. We would like to take this opportunity to thank them all.

First and foremost, we would like to thank **Dr. K Channakeshavalu**, Principal, EWIT, Bangalore, for his moral support towards completing our project work.

We would like to thank, **Dr. Suresh M B**, Professor and Head of Department of ISE, EWIT, Bangalore, for his valuable suggestions and expert advice.

We deeply express our sincere gratitude to our guide **Mrs. Sanjitha**, Department of ISE, EWIT, Bangalore for her able guidance throughout the project work and guiding us to organize the report in a systematic manner.

We thank our Parents, and all the faculty members of Department of Information science & Engineering for their constant support and encouragement.

Last, but not the least, we would like to thank our peers and friends who provided us with valuable suggestions to improve our mini project.

Kummara Pavan Kalyan 1EW20IS088
Vishwas Holla 1EW20IS023

LIST OF CONTENTS

CONTENTS	PAGENO
ABSTRACT	i
ACKNOWLEDGEMENT	ii
CHAPTER 1: INTRODUCTION	1-3
1.1 FILE STRUCTURES	1
1.2 OVERVIEW	2
1.3 OBJECTIVES	2
CHAPTER 2: LITERATURE SURVEY	4-7
2.1 EXISTING SYSTEM	6
2.2 PROPOSED SYSTEM	6
2.3OBJECTIVEOFTHEPROPOSEDSYSTEM	7
CHAPTER 3: EXISTING SYSTEM AND DRAWBACK	7-8
3.1 HARDWARE REQUIREMENTS	8
3.2 SOFTWARE REQUIREMENTS	8
CHAPTER 4: SYSTEM IMPLEMENTATION	9-12
4.1 ABOUT VISUAL BASIC 6.0	9
4.2 ABOUT MICROSOFT ACCESS	11
CHAPTER 5: PROPOSED SYSTEM	12-15
5.1 INTRODUCTION	12
5.2 DESIGN STRATEGY	13
5.3 PROJECT DESCRIPTION	14
5.4 DATAFLOW DAIGRAM	15
CHAPTER 6: DEVELOPMENT AND METHODOLOGY	9-19

6.1 TECHNIQUES FOR IMPLEMENTATION	9
6.2 MODULES DESCRIPTION AND CODE	18-46
CHAPTER 7: TESTING	46-49
CHAPTER 8: RESULTS	50-53
CONCLUSION	54
REFERENCE	55

LIST OF FIGURES

FIG NO	CAPTION/TITLE	PAGE NO
4.1	DATA FLOW DIAGRAM	6
7.1	HOME SCREEN	12
7.2	DATA INSERTION SCREEN	12
7.3	VIEW SCREEN	13
7.4	VIEW SINGLE DATA	13
7.5	MODIFY SCREEN	14
7.6	DELETE SCREEN	14
7.7	DELETE ALL SCREEN	14
7.8	TEXT FILE	15
7.9	INDEX FILE	15

CHAPTER 1

INTRODUCTION

1.1 FILE STRUCTURES

Telephone billing system project is to present the requirement of the Computerization of Billing System. The project thus calculates the telephone bills automatically. It does almost every work which is related to automatic telephone billing connection system via- new connection, customer record modification, viewing customer records & all works related to rate of bills, meter readings in addition to bill calculation and bill generation.

“Telephone Billing System” is developed as per seeing the increasing requirement to speed up the work and incorporate a new work culture. Thus a new software has been proposed to reduce manual work, improving work efficiency, saving time and to provide greater flexibility and user-friendliness as the system previously followed was totally manual one with lots of errors. Since it is directly associated with the database, so there is very little maintainability problem with this tool. Since there is very limited usage of separate forms, this tool is very much portable. This tool uses several canvases on the same form. This tool is very much flexible for future enhancements. The main objective while implementing the project

Telephone Billing System were to minimize the work and at the same time increase the speed of the work done and also the information retrieval will become easy. In this project the maintenance of database as well as overall project will become easy. The purpose of the project is to develop a system which is user friendly, easy to use maintain and satisfies all the requirements of the user of the specified system. Security measure will be adopted, by maintaining the login of username and the password. Data redundancy will be greatly reduced because this new system is built using Visual Basic 6.0 as front-end. It entails looking into duplication of efforts bottlenecks and inefficient existing procedures.

1.2 MODULE DESIGN

Our project is implemented using the following modules.

- Input Design.
- Output Design.
- Table Design.

1.2.1 INPUT DESIGN

Input design is the process of converting user-originate inputs to a computer-based format. The goal of design input data is to make data entry as easy, logical and free. The most common source of data processing errors is inactive input data. Effective design of the input data minimizes the error made by data entry operators. Verification and validation is the most important in input design. User-friendly input design enables quick error detecting and correction. We can prevent the user entering invalid data into the databases by warning, neglecting or messaging appropriately. The user is then allowed to input correct data. Some help provisions may aid the user to point out the error. In this system inputs are collected from terminals through keyboard.

1.2.2 OUTPUT DESIGN

Output design has been an ongoing activity from the very beginning of the project. The objective of the output design is to convey the information of all past activities, current status and to emphasize important events. The output generally refers to the results and information that is generated from the system. In the output design phase one or more output media can be selected. Out of which the most common ones are CRT displays and print out. Here only CRT display has been attempted. A rapid enquiry is obtained from CRT displays. From design is made interesting and attractive.

1.2.3 TABLE DESIGN

In this table design we are created tables such as customer records table, call rates table, customer metre reading table, bill record table and the user login table for the easy and effective understanding. The purpose of the project is to develop a system which is user friendly, easy to use, maintain and satisfies all the requirements of the user of the specified system

1. CUSTOMER RECORDS

Cust name	Text
Cust adds	Text
Cust phone	Number

2. CALL RATES

Local	Number
Mobile	Number
STD	Number
ISD	Number
Monthly Rental	Number

3. CUSTOMER METER READING

Cust phone	Text
M Local	Number
M mobile	Number
M std	Number
M Isd	Number

4. BILL RECORD

Cust name	Text
Cust phone	Text
Cust add	Text
Local	Number
Mobil emt	Number
STD mt	Number
ISD mt	Number

CHAPTER 2

LITERATURE SURVEY

This literature survey explores the topic of telecom billing systems, which are crucial for managing the complex process of charging and invoicing customers in the telecommunications industry. The survey provides an overview of existing research and industry practices related to telecom billing systems, including their architecture, functionality, challenges, and advancements. By examining the literature, this survey aims to provide a comprehensive understanding of the current state of telecom billing systems and identify potential areas for future research and improvement.

1. Introduction

- Overview of telecom billing systems
- Importance of efficient billing systems in the telecommunications industry
- Purpose and scope of the literature survey

2. Telecom Billing System Architecture

- Components of a typical telecom billing system
- Data flow and interaction between different system modules
- Integration with other systems (customer relationship management, network management, etc.)

3. Billing System Functionality

- Subscriber management and provisioning
- Rating and charging mechanisms
- Invoice generation and distribution
- Payment processing and revenue assurance
- Dispute management and customer support

4. Challenges in Telecom Billing Systems

- Scalability and performance issues
- Billing accuracy and dispute resolution
- Security and fraud prevention
- Regulatory compliance
- Integration with diverse network technologies (fixed-line, mobile, broadband, etc.)

5. Telecom Billing System Advancements

- Real-time billing and charging
- Convergent billing for multiple services (voice, data, video, etc.)
- Personalized and flexible pricing models
- Usage analytics and predictive billing
- Billing system automation and artificial intelligence

6. Case Studies and Industry Practices

- Overview of telecom billing systems deployed by major service providers
- Success stories and lessons learned from billing system implementations
- Best practices in telecom billing system design and management

7. Future Research Directions

- Emerging trends and technologies in telecom billing
- Potential areas for innovation and improvement
- Regulatory implications and policy considerations
- Impact of 5G and IoT on billing systems

8. Conclusion

- Summary of key findings from the literature survey
- Recommendations for further research and development in telecom billing systems

By conducting a literature survey on telecom billing systems, this study provides valuable insights into the current state of the field and highlights areas that require further research and improvement. The findings can be used by researchers, industry professionals, and policymakers to enhance the efficiency, accuracy, and customer satisfaction of telecom billing systems.

2.1 Existing System

The existing system was a manual one. Whatever be the process involved in the system were done through register (files) . There were lots of complexities involved in the system. When any customer takes new connections then separate files were maintained. Updating of data was very tedious job. It was not easy to do several administrative works like managing rates of calls, addition or modification of metered calls & customer entries

2.2 Proposed System

The new system titled “TELEPHONE BILLING SYSTEM” was hence proposed to remove all the drawbacks discussed above. Information is a vital ingredient for the operation and management of any organization. Thus, any system should have the ability to provide error free filtered information after processing the required data. This system has been taken up with a view for developing a more sophisticated system that can be easily handled by any kind of users. The proposed system aims at efficient and timely information for decision-making, integrate with other functions, and reduce redundant work.

Important features of this proposed system

- Consistent user interface with high economic features built into it.
- System design in modular and structured way so as to make the integration with other subsystems easier.
- User has complete control as it provides and accept only appropriate and valid data.
- User-friendly error messages are provided wherever necessary.
- Addition, deletion, modification of records as when needed.
- Providing connections to new customers.
- Bill generation for customers.

2.3 OBJECTIVES OF THE PROPOSED SYSTEM

- To reduce workload of staff.
- To reduce the delay in processing time.
- To reduce the delay in bill generation.
- To provide the user-friendliness in all possible ways.
- To provide greater flexibility.
- To store data in a centralized location to reduce redundancy and increase consistency.

CHAPTER 3

3. EXISTING SYSTEM AND DRAWBACK

In the existing system all the office works was done manually. The manual work processes were time consuming and hence slow. Following are the main drawbacks of the existing system:

- The existing system is totally manual thus there are chances of error in Processing.
- The basic and major drawbacks in the existing system are the speed of retrieval of data from files, which leads to delay.
- Maintenance of data is very cumbersome and laborious job.
- The manual jobs such as calculation are more error prone.
- There are plenty of chances of duplicity of data and information.
- Updating is very tedious job.
- There is no central database from where one can get different statistical data at one place.

Since existing system was totally manual which has lots of complexities, shortcomings in itself and all the data was being stored in registers, files .Thus to overcome the limitations of existing system, the new computerized system needed, so that information can be provided to the user more quickly and more accurately.

3.1 Hardware Requirements

The software should run on any sort of desktop or laptop environment, regardless of the operating system. Essential input/output devices are keyboard, mouse, and printers; nothing else is required but can be recommended if desired.

- Processor: Intel(R) Core (TM) i3-5005U CPU
- Hard Disk: 64-bit Operating System
- RAM: 2GB
- Other standard physical devices like keyboard, mouse etc.

3.2 Software Requirements

The Software Requirements deal with defining software resources requirements and prerequisites that needs to be installed on a computer to provide optimal functioning of an application. In this part we are going to study the functional requirements used in the project and the nonfunctional requirements needed for our project and we also discuss the external requirements which plays important role for the software to be developed. This software requirement is the phase where we diagnose all the data to prescribe in the project and needed to be analyzed for the later phases here analyzing is nothing but what type of steps you need to carry out for the implementation of project.

- Operating System: Windows 11
- Application: CodeBlocks
- Text File: Notepad
- Language: C++

CHAPTER 4

SYSTEM IMPLEMENTATION

4.1 ABOUT VISUAL BASIC 6.0

Visual Basic 6.0 is an enjoyable language due to its visual environment. Building a windows program in Visual Basic requires dragging and dropping graphic objects on to the screen from a toolbox. Thus, Visual Basic is the efficient GUI tools to develop some exciting windows-based application.

Visual Basic 6.0 is much more than just a programming language. The programming language forms the background of all that takes place in a running Visual Basic program. The language is a secondary consideration to the user interface. A windows program offers a high degree of user interaction using the graphical elements that forms the objects on the window the user sees. If the user interface is not correct, user will not like the program.

Visual Basic lets one add menus, text boxes, command buttons, option buttons (for making exclusive choice), check boxes, list boxes, scroll bars and file and directory boxes to black windows. One can use Visual Basic to communicate with other applications, running under windows. Visual Basic offers: More Internet features, better support for data base development, more language feature to make programming job easier.

SOME TOOLS OF VISUAL BASIC 6.0

- Data access features allow creating databases, front-end applications and scalable server side components for most popular databases formats including Microsoft SQL server and other enterprise level databases.
- Active X technology allows using functionality provided by other applications such as Microsoft Word, Microsoft Excel and other applications and objects could be created using the Professional Enterprise editions of Visual Basic.
- Internet capabilities make it easy to provide access to documents and applications across the Internet or Intranet server applications.

collection grows. Proper indexing and relationships between tables optimize the performance of data operations within the system.

The finished applications are a true .exe files that uses a Visual Basic virtual machine that can be freely distributed.

SIGNIFICANCE FEATURES OF VISUAL BASIC 6.0

Toolbox: The Tool Box window differs from the tool bar. The Tool Box is a collection of tools that acts as a repository of controls we can place on forms. Some tools are Selection Pointer, Picture Box, Label, Text Box, Frame Button, Command Botany, Check Box, Option Button etc.

Text Boxes:

It is used to display text or to accept user input. Most of the code is written to process the information users enter into them. Several properties of text boxes are as follows: -

Text:

The text property in text box is the analog of the caption property for a command button or a form; it controls text the users see. It determines whether text on the control such as label or command button, is left justified, cantered, or right justified on the control. The Alignment property take one of the three values: 0-Left justify, 1-Right justify, 2-Center.

Max Length:

This property specifies the maximum number of characters that the text box will accept. A value of 0 indicates that the user can enter a value of any length.

Locked:

This property determines whether the user can enter a value or change the default value of the text box. If true, the user cannot change the text box value until the program, at run time assigns a false to this property.

Use Labels to display information programmer does not want the user to be able to change. Most common use for Labels is to identify a text box or other control by describing its contents. Another common use is to display help information.

Message Boxes:

Message boxes display information in the dialog box superimposed on the form. They want for the user to choose a button before return to the application box.

4.2 ABOUT MICROSOFT ACCESS**Database:**

A database is a set of data, organized for easy access. The database is the actual data. It is the database that you will be accessing when you need to retrieve data.

Data Dictionary:

The data dictionary is a set of tables Access uses to maintain information about the database. The data dictionary contains information about tables, indexes, clusters, and so on.

DBA (Database Administrator):

The DBA is the person responsible for the operation, configuration, and performance of the database. The DBA is charged with keeping the database operating smoothly, ensuring that backups are done on a regular basis and installing new software. Other responsibilities might include planning for future expansion and disk space needs, creating databases and tablespaces, adding users and maintaining security, and monitoring the database and retuning it as necessary. Large installations might have teams of DBAs to keep the system running smoothly; alternatively, the tasks might be segmented among the DBAs.

DBMS or RDBMS:

The Database Management System is the software and collection of tools that manages the database. A Relational Database Management System is a DBMS that is relational in nature. This means that the internal workings access data in a relational manner. Access is an RDBMS.

CHAPTER 5

SYSTEM DESIGN

5.1 INTRODUCTION

System design is the second step in the system life cycle, in which overall design of the system is achieved. The functionalities of the system is designed and studied in this phase. The first step is designing of program specification. This determines the various data inputs to the system, data flow and the format in which output is to be obtained. Design phase is a transmission phase because it is a transition from user oriented document to computer data. The activity in the design phase is the allocation of functions to manual operations, equipment and computer programs. Flow charts prepared in the study time received and decomposed until all functions in the system perform evidently.

Design is a multistep process that focuses on data structures, software architecture, procedural details(algorithms etc) and links between the modules. The design process goes through logical and physical stages. In logical design reviews are made linking existing system and specification gathered. The physical plan specifies any hardware and software requirement, which satisfies the local design.

Modularization of task is made in the mode. The success of any integrated system depends on the planning of each and every fundamental module. Usually a project is revised in step by step sequence. Inter phase management of such module is also important. Software design methodology changes continually as new methods, better analysis and broader understanding evolve.

Various techniques for software design do exit with the availability of criteria for design quality. Software design leads three technical activities-design, code and test. The techniques for software design do exit with the availability of criteria for design quality. Software design leads three technical activities-design, code and test that are required to build and verify software. Each activity transforms information, which validates the software. The design system converts theoretical solution introduced by the feasibility study into a logical reality.

5.2 DESIGN STRATEGY

The design strategy is a vital aspect of the system to be developed. The design of the software reflects the basic understanding of the problem. For designing a good system what we have to be is to get correct definition of the problem and analyze the problem thoroughly.

The design of a system should be such that if a small portion is changed. The rest of the system should be unaffected. This is the flexibility of the system. Greater the system flexibility greater will be the system reliability. While carrying out the job of designing of a new system one has to consider many factors. These factors include the drawbacks and limitations of the present manual system as well as of the features and advantages of the proposed system. It should be designed in such a manner that even a layman can run it without any difficulty.

An important quality of a software must enjoy is “user friendliness”. It can be achieved in many ways like providing menu, giving context sensitive help, doing automatic validation to input data, etc. Another main factor is speed efficiency. In order to achieve speed efficiency, the program should be designed accordingly and the user is provided with a compiled copy of the software package with necessary data file format rather than source code.

Design of input and output formats is equally important for any design. The output format should be designed in such a way that it must reflect all the required information in detail. The design of the database itself such as type of data stored, size of data etc. Some of the decisions made during database design are:

- Which data items are to be recorded and in which database.
- Length of each record, based on the characteristics of the data items on which it is based.
- Data who's unauthorized change must be prevented.
- Data, which must be avoided from redundancy.
- Maintenance of data integrity etc.

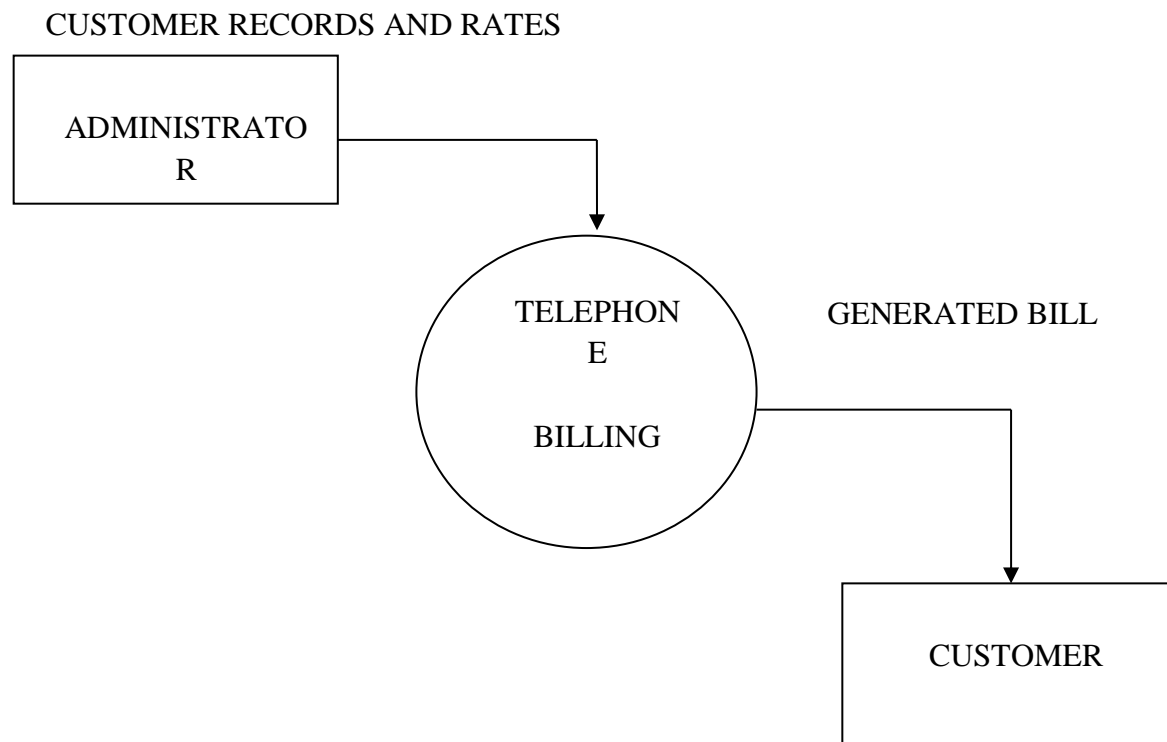
5.3 PROJECT DESCRIPTION

The telecom billing system project aims to develop a comprehensive and efficient system for managing the billing and invoicing process in the telecommunications industry. The system will handle the complex tasks of subscriber management, service provisioning, usage tracking, rating, invoicing, payment processing, dispute management, and reporting. By implementing this system, telecom service providers can streamline their billing operations, improve accuracy, enhance customer satisfaction, and ensure compliance with regulatory requirements.

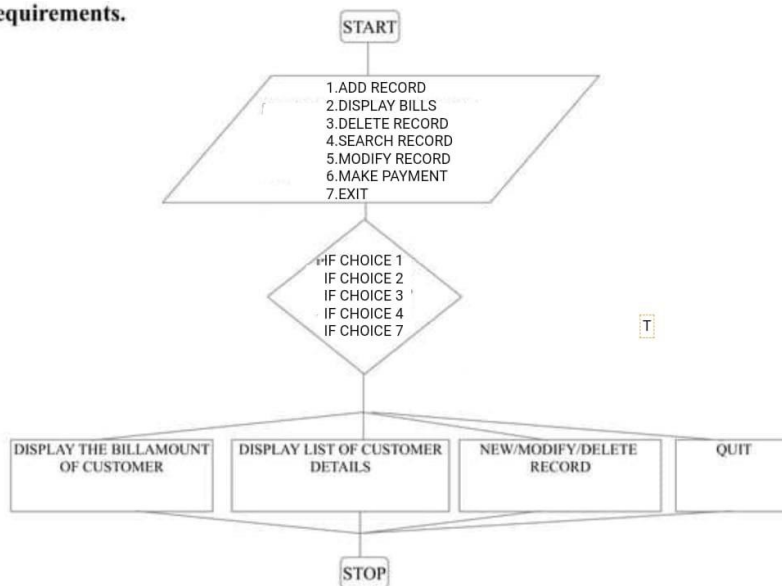
Key Features and Functionality:

1. **Subscriber Management:** The system will allow the registration and management of subscribers, including capturing their personal details, contact information, and service preferences. It will handle subscriber provisioning and activation of services.
2. **Service Plan Configuration:** The system will provide the ability to define and configure different service plans, including voice, data, SMS, and value-added services. Service plans can include various features, pricing models, and discounts.
3. **Usage Tracking and Rating:** The system will track and record usage details for each subscriber, including call duration, data usage, and SMS volume. It will apply appropriate rating algorithms to calculate charges based on the usage and service plan.
4. **Invoice Generation and Distribution:** The system will generate accurate and detailed invoices for each subscriber based on their service usage during a billing cycle. Invoices will include charges for different services, taxes, discounts, and any additional fees. Invoices can be delivered electronically or through traditional mail.

5.4 DATAFLOW DIAGRAM



requirements.



CHAPTER 6

DEVELOPMENT AND METHODOLOGY

This paper presents an overview of the development and methodology of a telecom billing system. It explores the key steps involved in designing and implementing an efficient billing system for the telecommunications industry. The paper discusses the various methodologies used in the development process and highlights the important considerations, challenges, and best practices associated with each stage. By understanding the development and methodology of telecom billing systems, organizations can successfully implement robust systems that streamline billing processes and improve customer satisfaction.

6.1 Introduction

- Overview of telecom billing systems and their significance
- Importance of a well-designed billing system for telecom operators
- Objectives and scope of the paper

6.2 Requirements Analysis

- Gathering and documenting functional and non-functional requirements
- Identifying business processes and workflows to be supported
- Analyzing billing-related data and system integration requirements

6.3 System Design

- High-level architectural design of the billing system
- Selection of appropriate technologies and platforms
- Database design for efficient data storage and retrieval
- Integration with external systems (network elements, CRM, etc.)

6.4 Development Methodologies

- Waterfall methodology: Sequential development process with distinct phases
- Agile methodology: Iterative and collaborative approach for flexibility and adaptability
- Hybrid methodologies: Combining elements of waterfall and agile approaches
- Selection of the most suitable methodology based on project requirements

6.5 Implementation and Testing

- Coding and configuration of billing system modules
- Unit testing, integration testing, and system testing
- Test data preparation and validation
- Performance testing to ensure scalability and efficiency

6.6 Data Migration and System Deployment

- Data extraction, transformation, and loading (ETL) processes
- Migration of existing customer and billing data to the new system
- Configuration and deployment of the billing system infrastructure
- User training and documentation preparation

6.7 Post-Deployment Activities

- System monitoring and maintenance
- Continuous improvement and enhancements
- Bug fixing and troubleshooting
- User feedback and system optimization

6.8 Challenges and Best Practices

- Addressing scalability and performance issues
- Ensuring billing accuracy and data integrity
- Security measures and fraud prevention
- Compliance with regulatory requirements
- User acceptance testing and change management

6.9 Case Studies

- Examples of successful telecom billing system implementations
- Lessons learned and best practices from real-world projects

6.10 Conclusion

- Summary of the development and methodology of telecom billing systems
- Importance of a well-planned and executed development process
- Future directions and potential advancements in telecom billing system development

By following a systematic development methodology and considering the various stages outlined in this paper, organizations can develop robust and efficient telecom billing systems. This will enable them to effectively manage billing processes, improve revenue management, and enhance customer satisfaction in the dynamic telecommunications industry

Front end:

The programming has been done using the language C++ and run in CodeBlocks application. C++ is a powerful and expressive programming language that emphasizes code readability and simplicity. It has a vast ecosystem of libraries and frameworks that enable developers to accomplish various tasks efficiently.

Back end:

Text file: Because of their simplicity, text files are commonly used for storage of information. They avoid some of the problems encountered with other file formats, such as

endianness, padding bytes, or differences in the number of bytes in a machine word.

Further, when data corruption occurs in a text file, it is often easier to recover and continue processing the remaining contents. The "Subscribe.txt" file serves as the primary data store, while the "index.txt" file aids in quick access to specific dance records by providing the corresponding offset positions.

PROGRAM CODE

```
#include <iostream>
#include<fstream>
#include<stdlib.h>
#include<conio.h>
#include<stdio.h>
#include<string.h>
#include<cstring>
#include<ctype.h>
#include<windows.h>
#include<ctime>
#include<limits>
#include<iomanip>
#include<ctype.h>
#include<sstream>
using namespace std;

const int BLUE = 1;
const int GREEN = 2;
const int RED = 4;
const int WHITE = 15;

void addrecords();
void displayrecord();
void deleterecords();
```

```
void searchrecord();
void modifyrecord();
void payment();

void addrecords()
{
    fstream file, indexFile;
    char test;
    int primaryKey;
    indexFile.open("index.txt", ios::out | ios::app);
    file.open("subscribe.txt", ios::app | ios::binary);
    if (!file || !indexFile)
    {
        cout << "Error opening file! \n";
        exit(1);
    }
    while (1)
    {
        changeColour(WHITE);
        cout << "Enter the Phone Number: ";
        s.phonenumber = getValidIntegerInput();
        primaryKey = s.phonenumber;
        cout << "Enter the Name: ";
        getline(cin >> ws, s.name);
        cout << "Select the Subscription Term: ";
        plans();
        s.status = "PENDING";
        file.seekp(0,ios::end);
        streampos pos = file.tellp();
        file.write(reinterpret_cast<const char*>(&s), sizeof(s));
```

```
    cout<<"position::"<<pos<<endl;
    getch();
    indexFile << primaryKey << ' ' << pos << endl;

    cout << "\n";
    for (int i = 0; i < 10; i++)
    {
        cout << char(176);
    }
    cout << " RECORD ENTERED SUCCESSFULLY ";
    for (int i = 0; i < 10; i++)
    {
        cout << char(176);
    }
    changeColour(GREEN);
    cout << "\n\n Press Y/y key to add another record or any other key to GO BACK.";
    test = getche();
    system("cls");
    if (!(test == 89 || test == 121))
        break;
}
file.close();
indexFile.close();
}

void displayrecord()
{
    fstream file;
    int counter=1;
```

```

int a=5,b=3,num=0;

file.open("subscribe.txt",ios::in);

if(!file)

{

    cout<<"\n";for(int i=0; i<10; i++){cout << char(176);}cout<<" NO RECORD FOUND
";for(int i=0; i<10; i++){cout << char(176);}

    cout<<"\n\n";

}

cout << char(201);for(int i=0; i<10; i++){cout << char(205);} cout<< char(203); for(int
i=0; i<18; i++){cout << char(205);} cout<< char(203);for(int i=0; i<24; i++){cout <<
char(205);} cout<< char(203);for(int i=0; i<12; i++){cout << char(205);} cout<<
char(203);for(int i=0; i<16; i++){cout << char(205);} cout<< char(203);for(int i=0; i<22;
i++){cout << char(205);} cout<<char(187)<< endl;

cout << char(186)<<" Sl.no "<<" "<<char(186)<<" Phone No. "<< char(186)<<"
Name"<<" "<< char(186)<<" Plan"<<" "<<char(186)<<" Amount"<<" "<<
char(186)<<" Payment Status"<<" "<< char(186)<< endl;

cout << char(200); for(int i=0; i<10; i++){cout << char(205);} cout<< char(202); for(int
i=0; i<18; i++){cout << char(205);} cout<< char(202);for(int i=0; i<24; i++){cout<<
char(205);} cout<< char(202);for(int i=0; i<12; i++){cout << char(205);} cout<<
char(202);for(int i=0; i<16; i++){cout << char(205);} cout<< char(202);for(int i=0; i<22;
i++){cout << char(205);}cout<< char(188);

cout<<"\n";

while((file.read((char*)&s,sizeof(s))))

{

    gotoxy(a,b);

    cout<<counter<<".";

    gotoxy(a+11,b);

    cout<<s.phonenumber;

    //a=a+10;

    gotoxy(a+28,b);

    cout<<s.name;

    //a=a+10;

    gotoxy(a+54,b);

```

```
        cout<<"Rs."<<s.planamt;

        //balance

        gotoxy(a+68,b);

        cout<<"Rs."<<s.amount;

        //status

        gotoxy(a+89,b);

        cout<<s.status;

        a=5;

        b=b+2;

        counter++;

        num=b;

    }

    if(counter>1)

    {

        for(int i=0;i<num-2;i++){ gotoxy(0,i+3);cout<<char(186);}

        for(int i=0;i<num-2;i++){ gotoxy(11,i+3);cout<<char(186);}

        for(int i=0;i<num-2;i++){ gotoxy(30,i+3);cout<<char(186);}

        for(int i=0;i<num-2;i++){ gotoxy(55,i+3);cout<<char(186);}

        for(int i=0;i<num-2;i++){ gotoxy(68,i+3);cout<<char(186);}

        for(int i=0;i<num-2;i++){ gotoxy(85,i+3);cout<<char(186);}

        for(int i=0;i<num-2;i++){ gotoxy(108,i+3);cout<<char(186);}

        gotoxy(0,num);

        cout << char(200); for(int i=0; i<10; i++){ cout << char(205);} cout<< char(202);    for(int
i=0; i<18; i++){ cout << char(205);}    cout<< char(202);for(int i=0; i<24; i++){ cout<<
char(205);} cout<< char(202);for(int i=0; i<12; i++){ cout << char(205);} cout<<
char(202);for(int i=0; i<16; i++){ cout << char(205);} cout<< char(202);for(int i=0; i<22;
i++){ cout << char(205);} cout<< char(188);

    }

    file.close();
```

```
changeColour(GREEN);
cout<<"\n"<<char(174)<<" Press any key to go back";
_getch();
system("cls");
}

void deleterecords()
{
    fstream file, dumpFile, dumpIndex, indexFile;
    int i = 0, count_file=1;
    int phone;
    bool flag=false;

    file.open("subscribe.txt", std::ios::binary | std::ios::in | std::ios::out|ios::app);
    indexFile.open("index.txt", std::ios::in | std::ios::out|ios::app);

    if (!file || !indexFile) {
        cout << "Can't open file\n";
        exit(0);
    }

    dumpFile.open("dumpFile.txt", ios::binary | ios::out|ios::app);
    dumpIndex.open("dumpIndex.txt", std::ios::out|ios::app);

    if (!dumpFile || !dumpIndex) {
        cout << "Can't open file\n";
        exit(0);
    }
    int newIndex;
```

```
cout << "Enter the phone number to be deleted: ";
phone=getValidIntegerInput();
if (file.is_open() && indexFile.is_open()) {
    int primaryKey,offset;
    while (indexFile >> primaryKey >> offset) {
        if (primaryKey != phone) {
            cout<<"primary key:-----"<<primaryKey<<"phine::-----"
"<<phone<<"offset::"<<offset<<endl;
            file.seekg(offset);
            cout<<"locatoin::"<<file.tellg();
            getch();
            file.read((char*)&s, sizeof(s));
            dumpFile.seekp(0,ios::end);
            dumpFile.write((char*)&s, sizeof(s));
            dumpIndex << primaryKey << ' ' << dumpFile.tellp()-sizeof(s) << endl;
        }
        else {
            flag = true;
        }
    }
}

if (flag) {
    cout<<"inside flag"<<endl;
    file.close();
    indexFile.close();
    dumpIndex.close();
    dumpFile.close();
    remove("subscribe.txt");
    rename("dumpFile.txt", "subscribe.txt");
}
```



```
remove("index.txt");
rename("dumpIndex.txt", "index.txt");

std::cout << "\n";
for (int i = 0; i < 10; i++) {
    std::cout << char(176);
}
std::cout << " RECORD FOUND ";
for (int i = 0; i < 10; i++) {
    std::cout << char(176);
}
std::cout << "\n\nRECORD DELETED SUCCESSFULLY\n\n";
std::cout << "Press any key to continue...";
std::getchar();
system("cls");
}
else {
    file.close();
    indexFile.close();
    dumpIndex.close();
    dumpFile.close();
    std::cout << "The Entered Phone Number is NOT FOUND\n";
    remove("dumpFile.txt");
    remove("dumpIndex.txt");
    std::cout << "Press any key to continue...";
    std::getchar();
    system("cls");
}
}
```

```
void searchrecord()
{
    fstream file,indexFile;
    bool flag;
    int i = 0,count_file=1;
    int phone;
    int animate=0;
    file.open("subscribe.txt",ios::out|ios::in);
    indexFile.open("index.txt", std::ios::in | std::ios::out);
    if(!file || !indexFile)
    {
        cout<<"Add Record before searching\n";
        exit(0);
    }
    cout<<"Enter the phone number to be searched: ";
    phone=getValidIntegerInput();

    while(animate<2)
    {
        system("cls");
        cout<<"\n"<<char(179);
        Sleep(500);
        system("cls");
        cout<<char(196)<<char(196);
        Sleep(500);
        system("cls");
        cout<<"\n"<<" "<<char(179);
        Sleep(500);
        system("cls");
```

```
        cout<<"\n\n"<<char(196)<<char(196);
        Sleep(500);
        system("cls");
        animate=animate+1;
    }
    if (file.is_open() && indexFile.is_open()) {
        int primaryKey,offset;
        while (indexFile >> primaryKey >> offset) {
            if(primaryKey==phone){
                flag = true;
            }
        }
    }

    if(flag)
    {
        cout<<"\n";for(int i=0; i<10; i++){cout << char(176);}cout<<" RECORD FOUND
";for(int i=0; i<10; i++){cout << char(176);}
    }
    else
        cout<<"Record NOT FOUND.\n";
    changeColour(GREEN);
    cout<<"\n\nPress any key to continue...";
    _getch();
    system("cls");
}
```

```
void modifyrecord()
{
    fstream file, dumpFile, indexFile;
```

```
int i = 0, count_file = 1;
int primaryKey, index, phone, offset;
bool flag = false;
char edit;
file.open("subscribe.txt", ios::out | ios::in | ios::app);
indexFile.open("index.txt", ios::out | ios::in | ios::app);
if (!file || !indexFile)
{
    cout << "can not able to open file\n";
    exit(0);
}
dumpFile.open("dumpFile.txt", ios::app);
if (!dumpFile)
{
    cout << "can not able to open file\n";
    exit(0);
}
cout << "Enter the phone number to edit details: ";
cin >> phone;
system("cls");
while (indexFile >> primaryKey >> offset) {
    if (primaryKey != phone) {
        file.seekg(offset);
        file.read((char*)&s, sizeof(s));
        dumpFile.seekp(0, ios::end);
        dumpFile.write((char*)&s, sizeof(s));
    }
    else {
        file.seekp(offset);
        file.read((char*)&s, sizeof(s));
    }
}
```

```
    userdetails();

    cout<<"EDIT DETAILS: \n";

    cout<<"1. Change Name\n 2. Change Plan\n 3. Change both Name and Plan\n 4.
Exit\n";

    cout<<"Enter your choice: ";

    a:cin>>edit;

    switch(edit)
    {
        case '1':cout<<"\nEnter new name: ";

            getline(cin >> ws, s.name);

            dumpFile.write(reinterpret_cast<char*>(&s), sizeof(s));

            break;

        case '2':cout<<"\nSelect New Plan: ";

            plans();

            dumpFile.write(reinterpret_cast<char*>(&s), sizeof(s));

            break;

        case '3':cout<<"\nEnter new name: ";

            getline(cin >> ws, s.name);

            cout<<"\nSelect New Plan: ";

            plans();

            dumpFile.write(reinterpret_cast<char*>(&s), sizeof(s));

            break;

        case '4':file.close();

            system("cls");

            return;

        default:cout<<"Enter valid option\n";

            goto a;
    }

    flag = true;
}

}
```

```
if(flag)
{
    file.close();
    indexFile.close();
    dumpFile.close();
    remove("subscribe.txt");
    rename("dumpFile.txt","subscribe.txt");
    remove("dumpFile.txt");
    cout<<"\n\nThe entered phone number is FOUND and record MODIFIED
SUCCESSFULLY\n";
    changeColour(GREEN);
    cout<<"Press any key to continue...";
    _getch();
    system("cls");
}
else
{
    cout<<"The Entered Phone Number is NOT FOUND\n";
    remove("dumpFile.txt");
    changeColour(GREEN);
    cout<<"Press any key to continue...";
    _getch();
    system("cls");
}
}
```

```
void payment()
```

```
{
    fstream file,indexFile;

    int i = 0,count_file=1;
    bool flag=false;
    int index,phone,primaryKey,offset,amt;
    file.open("subscribe.txt",ios::out|ios::in);
    indexFile.open("index.txt",ios::out|ios::in);
    if(!file)
    {
        cout<<"Unable to open the file\n";
        exit(0);
    }
    cout<<"Enter phone number of the subscriber for payment : ";
    cin>>phone;
    while (indexFile >> primaryKey >> offset)
    {
        if (primaryKey == phone)
        {
            file.seekg(offset);
            file.read((char*)&s, sizeof(s));
            if(s.amount > 0)
            {
                flag = true;
                billmod();
                cout<<"\nEnter amount of payment : ";
                cin>>amt;
                s.amount=s.amount-amt;
                if(s.amount == 0)
                {
                    s.status = "PAID";
```

```
    }
    file.seekg(-sizeof(s),ios_base::cur);
    file.write((char*)&s,sizeof(s));
    file.close();
    break;
}
break;
}
}
indexFile.close();
if(flag)
{
    cout<<"\n";for(int i=0; i<10; i++){cout << char(176);}cout<<" PAYMENT
SUCCESSFUL ";for(int i=0; i<10; i++){cout << char(176);}
    cout<<"\n\n";
}
else if(i==2)
{
    billmod();
    cout<<"\n\n";for(int i=0; i<10; i++){cout << char(176);}cout<<" PAYMENT
ALREADY DONE! ";for(int i=0; i<10; i++){cout << char(176);}
    cout<<"\n\n";
}
else
{
    cout<<"PHONE NUMBER NOT FOUND\n";
    cout<<"Please make sure you have a subscription plan.\n";
}
changeColour(GREEN);
cout<<"Press any key to continue...";
_getch();
```



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

CHAPTER 7

TESTING

The completion of a system will be achieved only after it has been thoroughly tested. Though this gives a feel the project is completed, there cannot be any project without going through this stage. The importance of testing is that the system is expected to run according to member's requirement before delivering it to the user. The system is tested on basis of specifications so that it does not fail on user site.

Performance Testing:

Performance testing for sorting in the Dance Club Management System can be done to evaluate the efficiency of the sorting algorithm used in the code. Here's a brief outline of how you can conduct performance testing for sorting:

- 1. Test Data Generation:** Generate a set of test data that simulates realistic scenarios. This data should represent Dance records with various attributes such as ID, name, address, phone number, plan, and payment status. Create a range of data sizes, starting from a small dataset to a large dataset, to assess the sorting algorithm's scalability.
- 2. Performance Metrics:** Define the performance metrics you want to measure, such as execution time, memory usage, and CPU utilization. Execution time is particularly important in sorting algorithms, as it directly impacts the efficiency of the sorting process.
- 3. Test Execution:** Run the sorting algorithm on different sizes of test data. Measure and record the execution time for each dataset. Additionally, monitor the memory usage and CPU utilization during the sorting process.
- 4. Analysis and Optimization:** Analyze the performance test results to identify any performance bottlenecks or areas for optimization. If the sorting algorithm is not performing well, consider implementing a more efficient sorting algorithm or optimizing the existing algorithm. Techniques such as parallel processing or using more efficient data structures can often improve sorting performance.

5. **Iterative Testing:** Repeat the performance testing process after implementing optimizations or changes to the sorting algorithm. Compare the new results with the previous ones to evaluate the effectiveness of the optimizations.
6. **Performance Test Reporting:** Document the performance test results, including the dataset sizes, execution times, and any observations or insights. Summarize the performance characteristics of the sorting algorithm and provide recommendations for improvements, if necessary.

Performance testing for the dance club management system involves evaluating its ability to handle a significant workload and maintain acceptable response times. The test focuses on assessing the system's performance under normal and peak usage scenarios.

Performance Testing:

Performance testing for sorting in the Dance Club Management System can be done to evaluate the efficiency of the sorting algorithm used in the code. Here's a brief outline of how you can conduct performance testing for sorting:

1. **Test Data Generation:** Generate a set of test data that simulates realistic scenarios. This data should represent Dance records with various attributes such as ID, name, address, phone number, plan, and payment status. Create a range of data sizes, starting from a small dataset to a large dataset, to assess the sorting algorithm's scalability.
2. **Performance Metrics:** Define the performance metrics you want to measure, such as execution time, memory usage, and CPU utilization. Execution time is particularly important in sorting algorithms, as it directly impacts the efficiency of the sorting process.
3. **Test Execution:** Run the sorting algorithm on different sizes of test data. Measure and record the execution time for each dataset. Additionally, monitor the memory usage and CPU utilization during the sorting process.
4. **Analysis and Optimization:** Analyze the performance test results to identify any performance bottlenecks or areas for optimization. If the sorting algorithm is not performing well, consider implementing a more efficient sorting algorithm or optimizing

the existing algorithm. Techniques such as parallel processing or using more efficient data structures can often improve sorting performance.

5. Iterative Testing: Repeat the performance testing process after implementing optimizations or changes to the sorting algorithm. Compare the new results with the previous ones to evaluate the effectiveness of the optimizations.

6. Performance Test Reporting: Document the performance test results, including the dataset sizes, execution times, and any observations or insights. Summarize the performance characteristics of the sorting algorithm and provide recommendations for improvements, if necessary.

Performance testing for the dance club management system involves evaluating its ability to handle a significant workload and maintain acceptable response times. The test focuses on assessing the system's performance under normal and peak usage scenarios.

CHAPTER 8

RESULTS

The fig 7.1 shows the main menu of the project. The main menu contains the various options displayed to the users. Here user can choose any option to perform various operations

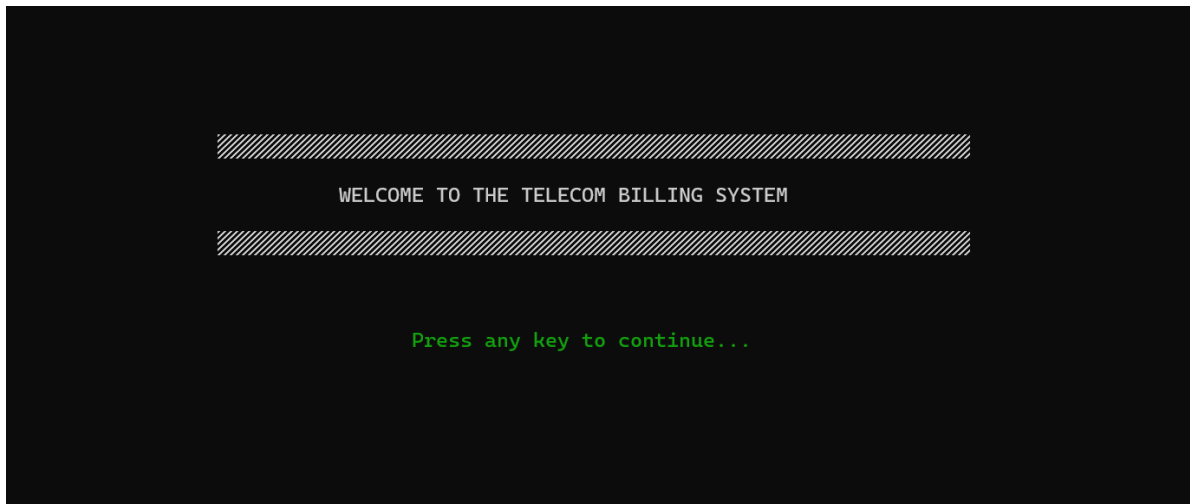
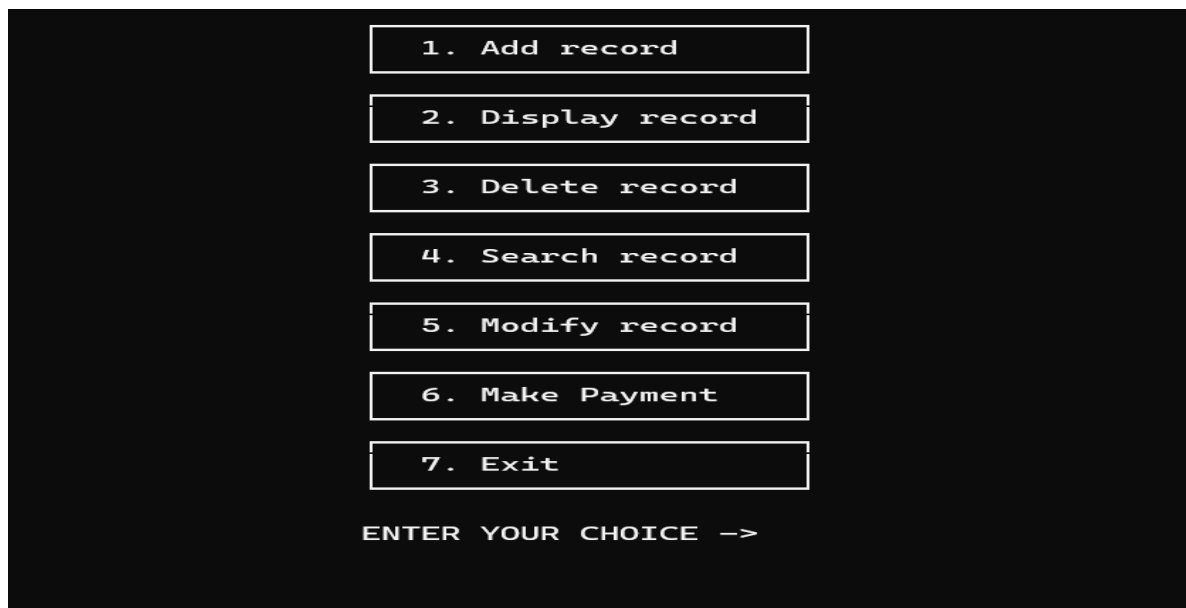


Fig 8.1: Welcome Page

The fig 7.2 shows insertion page, that allows user to add new record. the user should enter details such as id, name, instructor, timings, price.



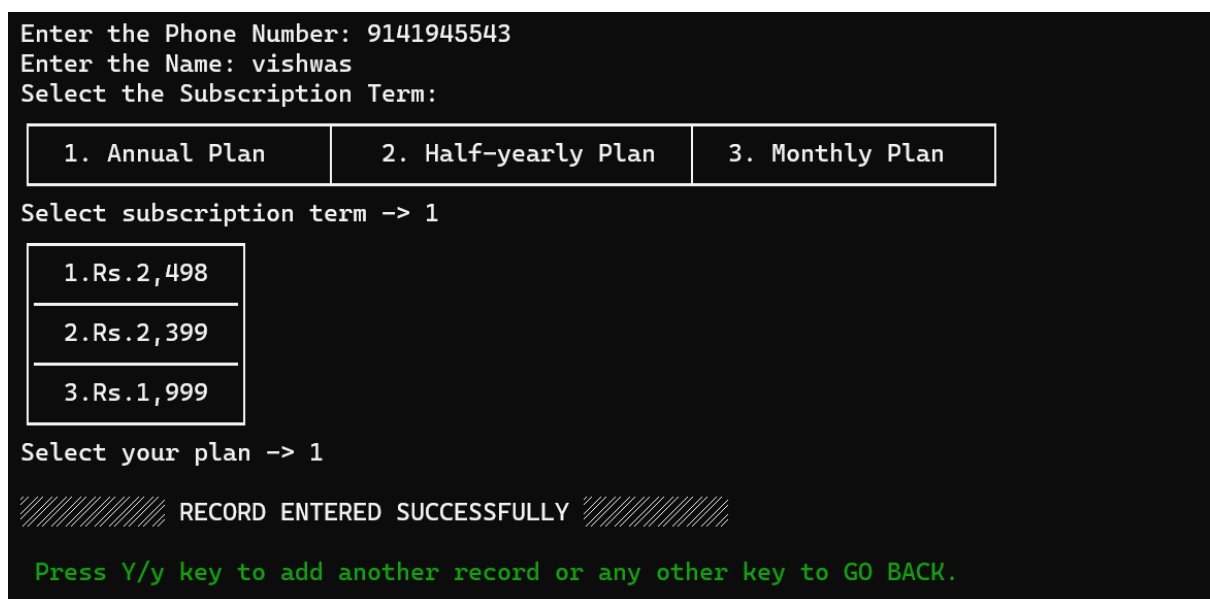
A screenshot of a terminal window showing a menu with seven options, each in a rectangular box. The options are: 1. Add record, 2. Display record, 3. Delete record, 4. Search record, 5. Modify record, 6. Make Payment, and 7. Exit. Below the menu, the text 'ENTER YOUR CHOICE ->' is displayed.

1. Add record
2. Display record
3. Delete record
4. Search record
5. Modify record
6. Make Payment
7. Exit

ENTER YOUR CHOICE ->

Fig 8.2:Home page

The display page in fig 7.3,it displays all the details of record inserted in the project.



A screenshot of a terminal window showing the process of adding a record. It starts with prompts for phone number (9141945543) and name (vishwas). Then it asks to select a subscription term, showing three options: 1. Annual Plan, 2. Half-yearly Plan, and 3. Monthly Plan. The user selects 1. Then it asks to select a plan, showing three options: 1. Rs.2,498, 2. Rs.2,399, and 3. Rs.1,999. The user selects 1. Finally, it displays 'RECORD ENTERED SUCCESSFULLY' and a green prompt to press Y/y to add another record or any other key to GO BACK.

Enter the Phone Number: 9141945543
Enter the Name: vishwas
Select the Subscription Term:

1. Annual Plan	2. Half-yearly Plan	3. Monthly Plan
----------------	---------------------	-----------------

Select subscription term -> 1

1. Rs.2,498
2. Rs.2,399
3. Rs.1,999

Select your plan -> 1

RECORD ENTERED SUCCESSFULLY

Press Y/y key to add another record or any other key to GO BACK.

Fig 8.3: Display Plans

The fig 7.4 shows the search page,here user can search any record which is inserted.

Enter the Phone Number: 8328503332
 Enter the Name: pavan
 Select the Subscription Term:

1. Annual Plan	2. Half-yearly Plan	3. Monthly Plan
----------------	---------------------	-----------------

Select subscription term -> 2

1.Rs.1,295
2.Rs.999
3.Rs.850

Select your plan -> 1

RECORD ENTERED SUCCESSFULLY

Press Y/y key to add another record or any other key to GO BACK.

Fig 8.4: To view Plan with details

The fig 7.5 shows the modified and updated record.

Sl.no	Phone No.	Name	Plan	Amount	Payment Status
1.	8328503332	pavan	Rs.1295	Rs.1528	PENDING
2.	9141946634	vishwas	Rs.1295	Rs.1528	PENDING

« Press any key to go back

Fig 8.5: Modify Screen

The fig 7.6 shows deletion page ,here user can delete any record which is inserted.

Enter the phone number to be deleted: 9141946634

RECORD DELETED SUCCESSFULLY

Press any key to continue...|

Fig 8.6:Delete screen

The fig 7.7 shows delete all option ,here user can delete all the record.

Sl.no	Phone No.	Name	Plan	Amount	Payment Status
1.	9141946634	vishwas	Rs.1295	Rs.0	PAID
2.	8431663456	pavanEWIT	Rs.259	Rs.305	PENDING

« Press any key to go back|

Fig 8.7: Payment History

The fig 7.8 shows that backend content of the project .All records inserted are stored in data file.

CONCLUSION

The telecom billing system plays a critical role in the telecommunications industry by managing the complex process of charging and invoicing customers. Through the literature survey conducted on telecom billing systems, several key findings and insights have emerged.

Firstly, the architecture of a telecom billing system consists of various components that work together to ensure accurate and efficient billing operations. These components include subscriber management, rating and charging mechanisms, invoice generation, payment processing, and dispute management. Integration with other systems, such as customer relationship management and network management, is also important for seamless operations.

Secondly, the survey identified several challenges faced by telecom billing systems. Scalability and performance issues can arise due to the large volume of transactions and increasing subscriber base. Ensuring billing accuracy and resolving disputes in a timely manner are crucial for maintaining customer satisfaction. Security and fraud prevention measures are necessary to protect sensitive customer data and prevent unauthorized usage. Regulatory compliance is another challenge, as telecom billing systems need to adhere to legal and industry standards. Integration with diverse network technologies adds complexity to the billing process.

REFERENCES

Reference URL's:

- www.sourcecodesworld.com
- www.chat.openai.com
- www.youtube.com
- <https://github.com/topics/filestructure>
- https://www.kashipara.com/project/python-project_12
- <https://www.freeprojectz.com/file-handling-projects>

