VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi-590 018

A Mini -Project Work on

Leave Management System

A Dissertation work submitted in partial fulfillment of the requirement
for the award of the degree

Bachelor of Engineering
In
Information Science & Engineering

Submitted by

Pruthvi Patil	**1AY18IS084**
Malipeddu Harshavardhan	**1AY18IS063**

Under the guidance of
Prof. Arun KH
Assistant Professor

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING
ACHARYA INSTITUTE OF TECHNOLOGY
(AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI.APPROVED BY AICTE, NEW DELHI)
Acharya Dr. Sarvepalli Radhakrishnan Road, Soldevanahalli, Bengaluru-560107

2020-21

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING
ACHARYA INSTITUTE OF TECHNOLOGY
(AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI.APPROVED BY AICTE, NEW DELHI)
Acharya Dr. Sarvepalli Radhakrishnan Road, Soldevanahalli, Bengaluru-560107

Certificate

This is to certify that the Mini-Project work entitled **Leave Management System** is a bonafide work carried out by **Pruthvi Patil(1AY18IS084)** and **Malipeddu Harshavardhan(1AY18IS063)** in partial fulfillment for the award of the degree of **Bachelor of Engineering** in **Information Science and Engineering** of the **Visvesvaraya Technological University**, Belagavi during the year 2020-21. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library. The Project has been approved as it satisfies the academic requirements in respect of Project work prescribed for the Bachelor of Engineering Degree.

Prof. Arun KH Guide	**Prof. C K Marigowda** HOD

Name of the Examiner's	**Signature with date**

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of this mini-project would be incomplete without the mention of the people who made it possible through constant guidance and encouragement.

We would take this opportunity to express our heart-felt gratitude to **Sri. B. Premnath Reddy**, Chairman, Acharya Institutes and **Dr. Prakash M R**, Principal, Acharya Institute of Technology for providing the necessary infrastructure to complete this mini-project.

We wish to express our deepest gratitude and thanks to **Prof. C K Marigowda**, Head of the Department, Information Science and Engineering and the project coordinators **Prof.Pakruddin B and Prof. Raushan Kashypa** for their constant support.

We wish to express sincere thanks to our guide **Prof. Arun KH,** Assistant Professor, Department of Information Science and Engineering for helping us throughout and guiding us from time to time.

A warm thanks to all the faculty of Department of Information Science and Engineering, who have helped us with their views and encouraging ideas.

Pruthvi Patil (1ay18is084)

Malipeddu Harshavardhan (1AY17IS063)

ABSTRACT

This system is all about Leave application which makes ease the work of users by this computerized software. By this application a user can store employee details, retrieve contact details, maintain notes, to-do lists, personal data, set events for important meetings of works on single platform. The Leave Management System is based on Internet based Application that can This project is aimed at developing an online leave management system that is of importance to an organization be accessed throughout the organization or a specified group or department.

A user who is working on system can set events for the important work while doing some another work. Events will remind him about that work. So, this application is convenient platform for a user to manage, contacts, daily work schedules and to enhance the punctuality of the user.

As manual computing system becomes more numerous, complex the need for the systematic approaches development becomes increasingly apparent. This system can be used to automate the workflow of leave applications and their approvals. The periodic crediting of leave is also automated. There are features like email notifications, automatic approval of leave, report generations etc., in this system. Leave Management application will reduce paper work .

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction to File Structures

File Structures is the Organization of Data in Secondary Storage Device in such a way that minimize 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.

"File organization" refers to the logical relationship among the various records that constitute the file, particular with respect to means of identification access to any specific record. "File structure" refers to the format of label and data blocks and of any logical record control, information. The organization of the given file maybe sequential relative, or indexed.

Co-Sequential Processing

Co-sequential Processing is the coordinated processing of two or more sequential lists to produce single output list. Sometimes the processing results in a merging, or union, sometimes it results in a matching, or intersection and other times the operation is a combination of matching and merging of the items in the lists.

Aim of the File Structure

The need for file structures

- Ideally, we would like to get the information we need once access to the disk
- If it is impossible to get what we need in one access, we want structures that allow us to find the target information with as few accesses as possible.
- We want our file structures to group information so we are likely to get thing
  We need with only trip to the disk.

1.2 Fundamental Operations on File

Open: A function or system call that makes a file ready for use. It may also bind a logical filename to a physical file. Its arguments include the logical filename, the physical filename and may also include information on how the file is to be accessed.

Close: A function or system call that breaks the link between the logical filename and the corresponding physical filename.

Create: A function or system call that causes a file to be created on secondary storage and may also binds a logical filename to the file's physical filename.

Read: A function or a system call used to obtain input from a file or device. When viewed at the lowest level, it requires three arguments: a source file logical name corresponding to an open file; the destination address for the address and the size or amount of data to be read.

Write: A function or system call used to provide output capabilities. When viewed at the lowest level, it requires three arguments: a destination filename corresponding to an open file; the source address of the bytes that are to be written and the size or amount of data to be written.

Seek: A function or a system call that sets the read/write pointer to a specified position in a file. Languages that provide seeking functions allow programs to access specific.

1.3 Services provided to the user

The functions that the Source Code Storage System provides are as follows:

1. **INSERT:** This operation is performed when new data needs to be added to the system, for e.g. when developer creates a new source code, the source code entry is inserted in the files. This option has choices:

a) Source code: This choice allows entering newly created source code into the files.

b) Sign up: This will create new developer account. This option is selected when a developer is enrolled in the Source Code Storage System.

2. **DELETE:** This operation clears the existing records in the various files. It is used when for e.g. a member leaves Source Code Storage System or when source code file is to be deleted is disposed of from system.. But care must be taken while performing this operation and permission taken from the author of files because the system could lose any important data.

3. **SEARCH:** This function is used to search particular data from the System. This function can search for data related in the entities:

a) Source code: To search for a particular source code, to know whether it is currently available in System or not. This can be done by entering name of the file.

b) Login id: This will find out the particular developer who possesses the particular Source Code.

4. **STORAGE:** This operation is used for storing the source code done by the developers. For this operation to be successful the member must meet some criteria like she developer should have account with unique name and password. All these checks are done by software. If then operation is successful, then the system automatically stores the codes developed.

5. **DISPLAY:** This is used to display each and every record from the system.

a) Source code: Record of every source code, done by the developer

CHAPTER 2

HARDWARE AND SOFTWARE REQUIREMENTS

2.1 Software requirements

- Documentation Tool: MS Word
- Development Tool: Dev C++

2.2 Hardware requirements

- Windows 8/8.1/10
- 4GB RAM
- 64 bit operating system
- 16 GB Hard disk

CHAPTER 3

SYSTEM OVERVIEW

3.1 Problem Definition

Manual system involves a lot of paper work" so it becomes time3consuming and costly .The chances of errors in calculation of delivery of Books are more in the current manual system. The calculation of total collection for day or month or year is very difficult. Currently no security is provided to the large amount of data of the every book details. It becomes very difficult to maintain details of every Book as records increases day by day.

3.1.1 Solution on problem

Creating new software we should analyze what is the basic need of the software. Analysis is nothing but a planning of creation of software to get proper output from it. Analysis is detail study of protects that you want to show in your software solving problems. The basic need of the software is to save the time of the user with the help of all useful information. And also to maintain the collection of data in your computer systematically" so that it's easy to understand. The proposed system provides lot of facility to the user to store information of the Books and it provides information in quick time in a systematic manner.

The processing time on the data is very fast. It provides required data quickly to the user and also in specified manner to the user. All the information of Books changes is given to the user and also the reports are also generated according to the requirement of the user. Today it is becoming very difficult to maintain record manually. This software system easily does the job of maintaining daily records as well as the transaction according to the user requirements.

3.2 The Sequential Search

Data items are stored in a collection such as a list, we say that they have a linear or sequential relationship. Each data item is stored in a position relative to the others. In Python lists, these relative positions are the index values of the individual items. Since these index values are ordered, it is possible for us to visit them in sequence. This process gives rise to our first searching technique, the **sequential search**.

3.2.1 Analysis of Sequential Search

To analyze searching algorithms, we need to decide on a basic unit of computation. This is typically the common step that must be repeated in order to solve the problem. For searching, it makes sense to count the number of comparisons performed. Each comparison may or may not discover the item we are looking for.

We make another assumption here. The list of items is not ordered in any way. The items have been placed randomly into the list. In other words, the probability that the item we are looking for is in any particular position is exactly the same for each position of the list.

To check the item is in the list, the only way to know it is to compare it against every item present. If there are n items, then the sequential search requires n comparisons to discover that the item is not there. In the case where the item is in the list, the analysis is not so straightforward. There are actually three different scenarios that can occur. In the best case we will find the item in the first place we look, at the beginning of the list. We will need only one comparison. In the worst case, we will not discover the item until the very last comparison, the nth comparison.

On average, we will find the item about halfway into the list; that is, we will compare against n/2 items. However, that as *n* gets large, the coefficients, no matter what they are, become insignificant in our approximation, so the complexity of the sequential search, is O(n).

CHAPTER 4

DESIGN

4.1 DATA FLOW DESIGN

A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

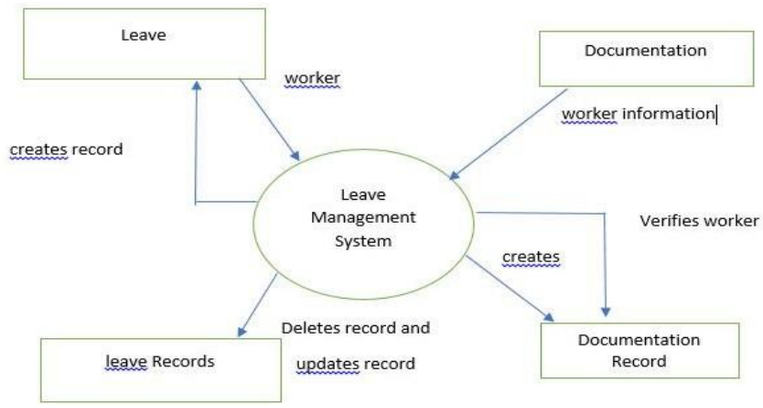

Fig 4.1: Data Flow Design

4.2 CLASS DIAGRAM

A class diagram is an illustration of the relationships and source code dependencies among classes in the Unified Modeling Language (UML). In this context, a class defines the methods and variables in an object, which is a specific entity in a program or the unit of code representing that entity. Class diagrams are useful in all forms of object-oriented programming (OOP). The concept is several years old but has been refined as OOP modeling paradigms have evolved.

CHAPTER 5

IMPLEMENTATION

Introduction:

We have used the object oriented concept in the implementation of design. The object oriented programming language used was C++. The concept of inheritance is used to a large extent.

Functions and the members involved:

Design the class cbi. Each object of this class represents information about each worker.

Members in cbi class includes c_id, name, address, department, phone number, designation, salary and number of cases handled by the worker.

Methods are unpack, pack, disp and assign.

Design the class btree.

Members include insert.

Methods include create and dis.

Design the class block. Here each worker is given a key.

Members include key, disp, cnt.

Methods include split and merge.

CHAPTER 6

TESTING

The integral part of any system's development life cycle is testing without which the system developed is sure to fail and result in loss of economic and manpower investments besides user's dissatisfaction and downfall of reputation. System testing is the stage of implementation, which aims at ensuring that the system works accurately and efficiently before actual operation commences. No program or system design is perfect, communication between the user and the designer is not always complete or clear. All this can result in errors.

Another reason for system testing is its utility as a user oriented vehicle before implementation. The application system is worthless if does not meet user needs, thus the system should be tested to see whether it meets the user requirements.

Testing here is conducted in bottom up approach as follows:

- Module testing: Here testing is done at each module level. Each case has been thoroughly tested to discover pitfalls.
- System testing: Here testing is done after all the modules have been integrated.

Test cases

Sl.No	Module	Test Case Description	Input	Expected Output	Actual Output	Status
1	Welcome Page	Verify whether Menu page displays.	Run the program.	Displays Menu page	Displays Menu Page	Pass
2	Menu Display	Displays the menu list.	Enter your choice.	Goes to the part of code you entered for to execute.	Goes to the part of code you entered for to execute.	Pass
3	Display all leave records using hash index.	Displays the leave records of all the workers.	Enter choice 5 to display	Displays the leave records one after the other	Displays the leave records one after the other	Pass
4	Apply leave record into the file	Takes data to insert into the file	Input values of the worker	Reads the values one by one	Reads the values one by one	Pass

5	Search for leave record using hash leave.	Searches for the worker details of the leave applied.	Enter workers emp_id number to search	Displays the workers details of the leave applied.	Displays the worker details of the leave applied.	Pass
6	Delete leave record	Deletes the leave record of the worker	Asks for the emp_id to be deleted	Deletes the leave record	Delete the leave record	Pass
7	Modifies leave record	Updates the leave record of the worker	Enter for the leave_id to be modified for the worker	Modifies the leave record of the worker	Modifies the leave record of the worker.	Pass
8	Quit	Exits from the program	Select the choice to quit	Quits from the program	Quits from the program	Pass

CHAPTER 7

SNAPSHOTS

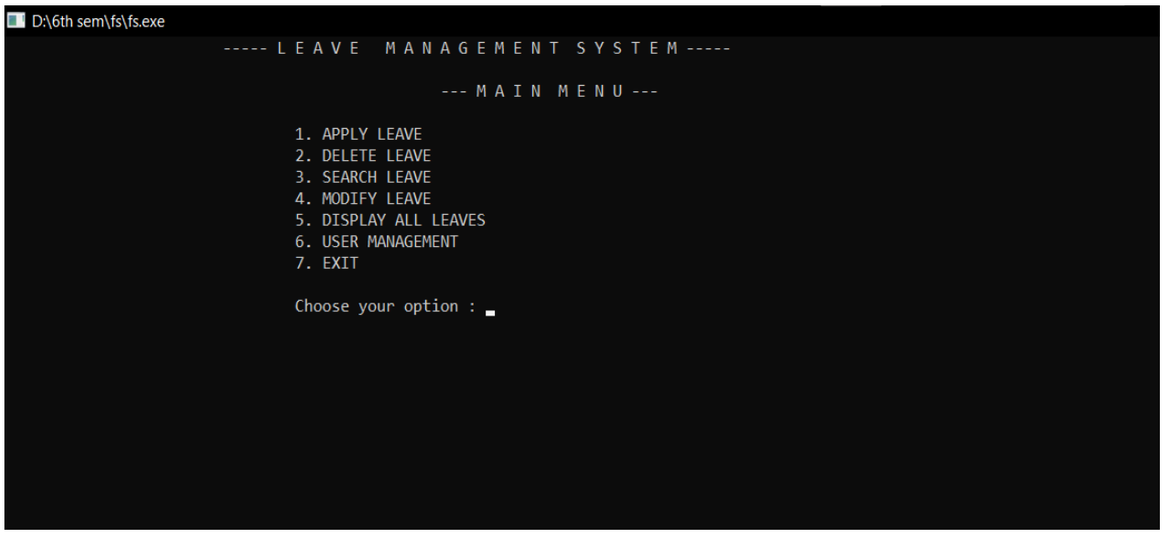

Fig 7.1: Introductory page

Fig 7.2:page showing user information

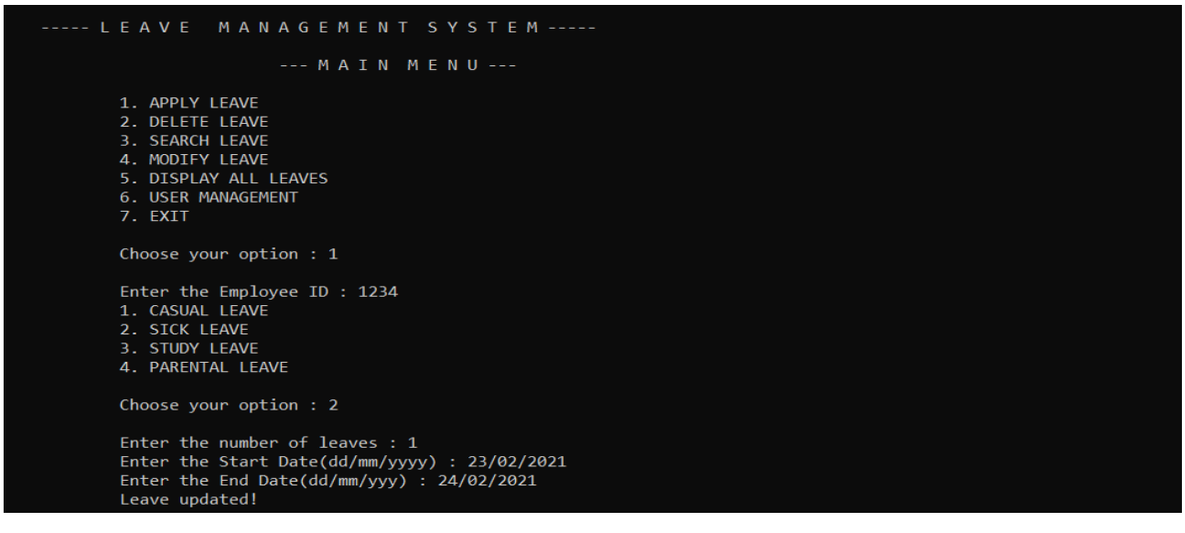

Fig 7.3: Apply Leave page

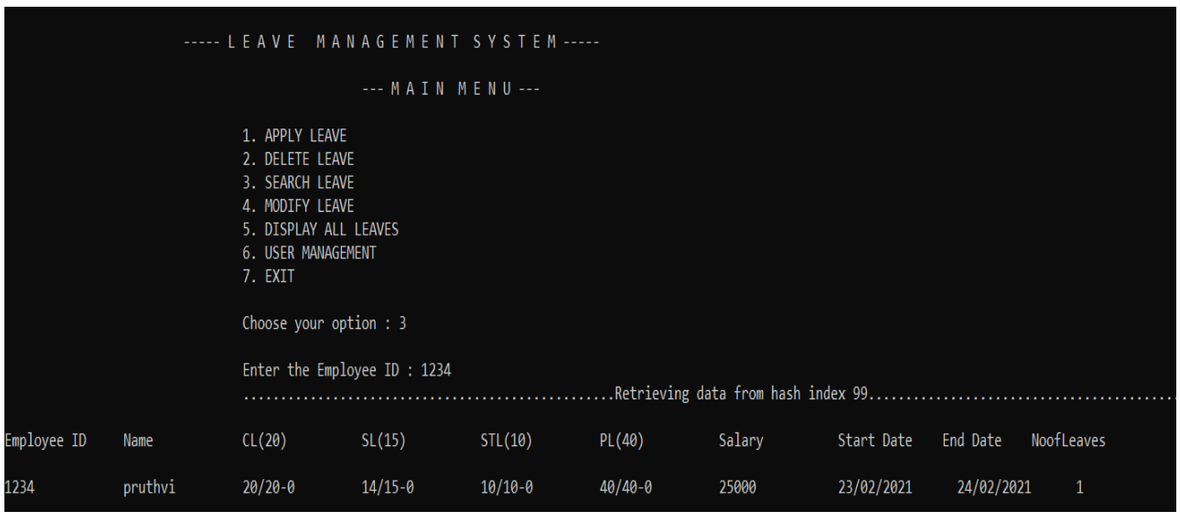

Fig 7.4:Search leave record page

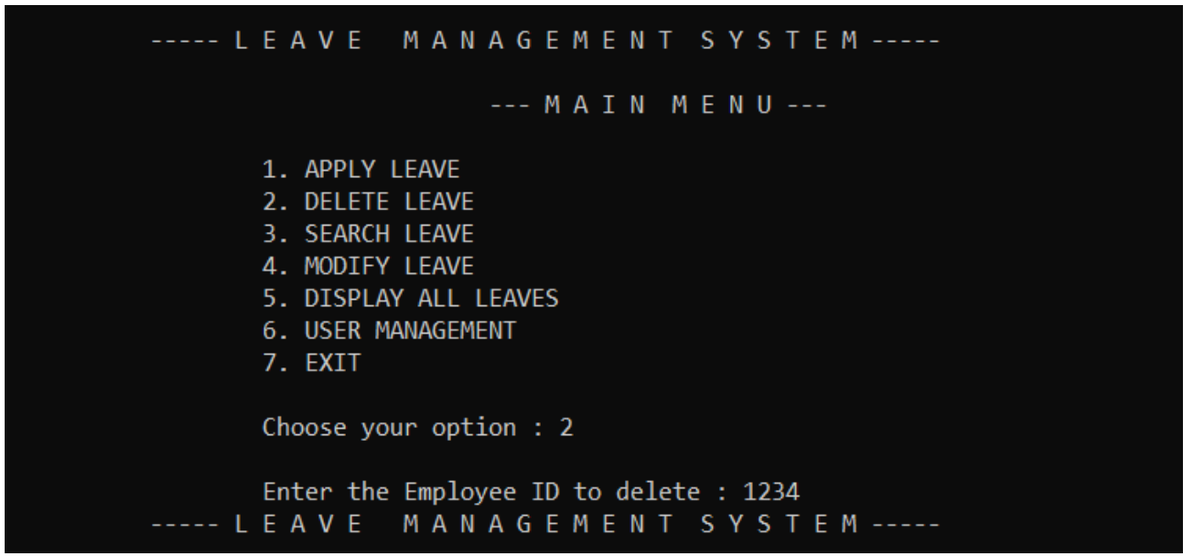

Fig 7.5: Deletes leave record page

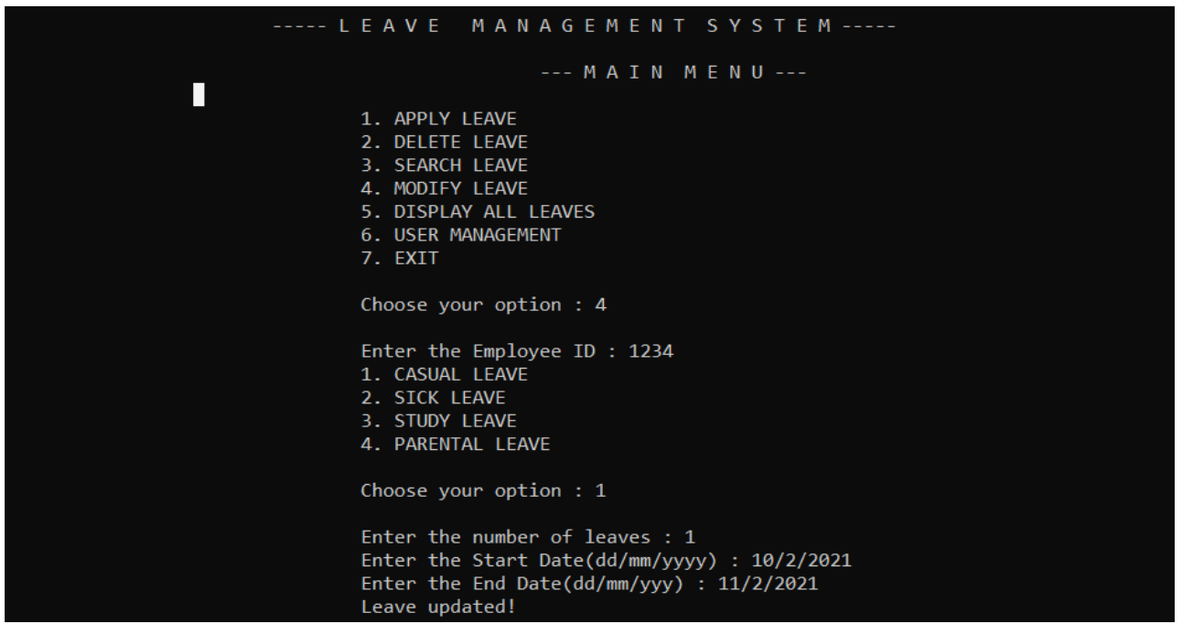

Fig 7.6: Modify leave record page

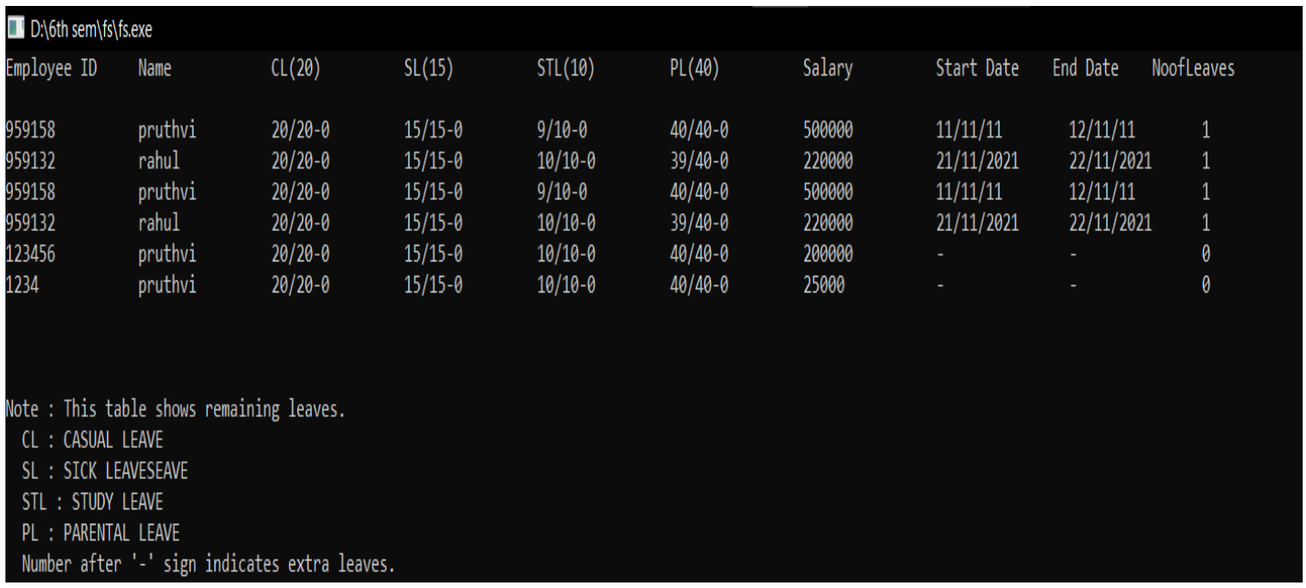

Fig 7.7: display all leave record page.

CONCLUSIONS AND FUTURE ENHANCEMENT

Conclusions

The concept of Hash Index, where we create do the operations like matching, merging for maintaining records. We have used buffer hierarchy in order to achieve this. Here a sorted file is maintained. These files are loaded into main memory. Since we are processing two files at a time we can hence reduce the number of disk accesses.

Limitation

As the number of records grows comparison processes also grows hence maintaining the large files becomes very difficult as we cannot read.

Future Enhancements

Given the File Structure design that we have built, we foresee a lot of areas that need enhancements.

1. First of all searching of records based on secondary key indexes is the major void that needs to be filled. Any database would require such facilities of record modifications and deletions.

2. Faster retrieval is one goal that the designers constantly try to achieve. One way of accomplishing the goal by using index structures itself is by creating multiple levels of these structures. The improvement will definitely show up as the database increases in size.

BIBILOGRAPHY

Book references

1. File Structures An Object-Oriented Approach with C++ by Michael J.Folk, Bill Zoellick, Greg Riccardi
2. Software Engineering-by Ian Sommerville
3. Let Us C++ by Yashavant .P.Kanetkar
4. The C Programming Language by Kernighan and Ritchie

Web references

1. www.stanford.edu
2. www.awl.com
3. www.codeproject.com
4. www.sourceforge.com