**Employee Management System**

*A Minor Project Report*

*Submitted in partial fulfillment of the requirements for the*

*Award of degree of Bachelor of Computer Applications*

**2019 – 2022**

**Submitted by Guided by**

**Ms. Neha Sabharwal**

**Name: Yukteshwar Bidhat**

**ERP ID: 0191BCA058**

**Name: Sanchit Talreja**

**ERP ID: 0191BCA045**





BHARATI VIDYAPEETH

(DEEMED TO BE UNIVERSITY)

INSTITUTE OF MANAGEMENT & RESEARCH, NEW DELHI

A-4, Paschim Vihar, New Rohtak Road, New Delhi-110063

**2021**

**STUDENT UNDERTAKING**

This is to certify that the project titled “**Employee Management System**” submitted to Bharati Vidyapeeth (Deemed to be University), Pune in partial fulfillment of the requirement for the award of the degree of Bachelor of Computer Applications is an original work carried out under the guidance of **Ms. Neha Sabharwal**. The matter embodied in this project is a genuine work done by me and has been submitted neither to this University nor to any other University for the fulfillment of the requirement of the course of study.

Signature

Name of the student: Yukteshwar Bidhat

ERPID: 0191BCA058

PRN No: 1920100593

**CERTIFICATE FROM INTERNAL GUIDE**

This is to certify that the Project titled **Employee Management System** is an academic work done by **Sanchit Talreja and Yukteshwar Bidhat** submitted in the partial fulfillment of the requirement for the award of the Degree of **Bachelor of Computer Applications** from **Bharati Vidyapeeth (Deemed to be University), Pune** under my guidance.

To the best of my knowledge and belief the data & information presented by him/her in the project has not been submitted earlier whether to this University or to any other University / Institute for the fulfillment of the requirement of any course of study.

E:\DATA BACKUP\Desktop\dc0f2f6e-e1ec-4a68-ae10-feea82de6839.jpg

Signature

Neha Sabharwal

**CERTIFICATE FROM THE DIRECTOR**

This is to certify that the Project titled “**Employee Management System”** is an academic work done by **Sanchit Talreja and Yukteshwar Bidhat** submitted in partial fulfillment of the requirement for the award of the Degree of Bachelor of Computer Applications from **Bharati Vidyapeeth (Deemed to be University), Pune.**

The authenticity of the project work will be examined by the viva examiner which includes data verification, checking duplicity of information etc. and it may be rejected due to non-fulfillment of quality standards set by the institute.

**Dr. A. K. Srivastav**

**Director-Incharge**

**ACKNOWLEDGEMENT**

I would like to express my gratitude and appreciation to all those who gave me the possibility to complete this report. A special thanks to our Major project Guide Ms. Neha Sabharwal whose help, stimulating suggestions and encouragement, helped me to coordinate our project especially in writing this report topic and achieving the goal as well as his encouragement to maintain our progress in track. I would to appreciate the guidance given by other supervisor as well as the panels especially in our project presentation that has improved our presentation skills by their comment and tips.

Yukteshwar Bidhat

0191BCA058

**Index**

|  |  |
| --- | --- |
| **Chapter 1 INTRODUCTION** | **Page no.** |
| * 1. Introduction About the project |  |
| * 1. Need Of Computerization Of System / Problem In Existing System |  |
| * 1. Proposed Software |  |
| * 1. Importance Of The Work / Project |  |
|  |  |
|  |  |
| **Chapter 2 SYSTEM ANALYSIS** |  |
| 2.1 Analysis Methodology /requirement gathering techniques |  |
| 2.2 Feasibility Study |  |
| 2.3 Choice of Platforms |  |
| 2.3.1 Software used |  |
| 2.3.2 Hardware used |  |
|  |  |
|  |  |
| **CHAPTER 3 SYSTEM DESIGN** |  |
| 3.1 Process Model used |  |
| 3.2 Database Design (ERD/Data Dictionary/ Table Design) |  |
| 3.3 Functional Design (DFD) |  |
| 3.4 Interface Design (Screen shots of forms) |  |
| 3.5 Output Design (Screen Shots of Report) |  |
|  |  |
|  |  |
| **CHAPTER 4 TESTING AND IMPLEMENTATION** |  |
| 4.1 Testing Methodology |  |
| 4.2 Testing Methodology applied |  |
| 4.3 Test Cases |  |
| 4.4 Test Data |  |
| 4.5 Gap Analysis |  |
| 4.6 Rework/ Retest |  |
| 4.7 Hardware & Software Requirement |  |
| 4.7.1 Hardware Requirement |  |
| 4.7.2 Software Requirement |  |
|  |  |
|  |  |
| **CHAPTER 5 CONCLUSION AND REFERENCES** |  |
| 5.1 Conclusion |  |
| 5.2 Limitation of System |  |
| 5.3 Future Scope for Modification |  |
| 5.4 References/ Bibliography |  |
|  |  |
| **ANNEXURES** |  |
| A-1 Menu Flow Diagram |  |
| A-2 Sample Input |  |
| A-3 Sample Output |  |
| A-4 Program Code |  |
| A -5 Mentee Feedback |  |
|  |  |
|  |  |

**CHAPTER 1**

1.1 Introduction about Project

1.2 Need of Computerization of System/ Problem in existing system

1.3 Proposed Software

1.4 Importance of the Work/project

**Introduction about Project**

**Employee Management System** is an application developed to maintain the details of employees working in an organization. It maintains information about the personal details as well as professional details of employees. This application is actually made by C programming with File Handling.

This application is helpful to the organization which maintains data of employees related to it. It is simple to understand and can be used by anyone, therefore it is user friendly.

The project is developed in C language with File Handling. However, the **main objective** behind developing this software is to convert all the manual work into an automated system which helps to eliminate paper work, improves employee services, make work error free and is less time consuming.

Ems has four modules

* User/Admin Login
* Employee Profile
* Employee Attendance
* Employee Payroll or Salary

**Problem in Existing System**

* **Lack of Security:** The paper document is less secure compared to an electronic system/computerized system. Documents which are misplaced can easily get into wrong hands. The leak of confidential documents can cost big amount to an organization as the clients expect their information to be secure in the hands of organization.
* **Time Consuming:** Manually managing is very tough and time-consuming process. Handling of documents and storing them safe is not easy. It’s not only about security but also carrying out different documents at different places.
* **Insufficient storage spaces:** Paperwork can take up a significant amount of space, and the need of space will increase as the number of the document increases. Because of this the paperwork also increases and records will also increase leads to insufficient space.
* **Increase Cost:** One of the biggest drawbacks of paper-based document management system is the associated costs. Documentation, record keeping, storage, etc. is a very costly process.
* **Difficulty in a modification of data:** When working with paper documents, it is much harder to make changes. You have to make a new copy of the employee rather than destroying the original copy and it is difficult to maintain many copies of each employee.
* **Employee productivity:** Payroll systems, employee data, overtime, submissions and many more are typically entered manually. Even a tiny change in any of the above will lead to incorrect data entered/written. Then you have to make changes, therefore it’s a huge wastage of time which can lower their productivity and morale.
* **Efficiency and Reliability:** It is difficult to achieve efficiency and reliability using manual work (counting the number of employees, remembering their salary, and many more). A human can make mistakes in keeping the record and also it can be misplaced.
* **Lack of Security:** The paper document is less secure compared to an electronic system/computerized system. Documents which are misplaced can easily get into wrong hands. The leak of confidential documents can cost big amount to an organization as the clients expect their information to be secure in the hands of organization.

**Proposed Software**

* **Less possibility of damage:** There is very less chances of damage as it isn’t done manually, it is done on computer and there is no need to keep the copies of employees as computerized system is safe and secure.
* **More secure:** Details of employees are secure and the software is also secure. Software is only functioned by the employees who have the correct username and password. However, the professional and personal details of all employees within the organization is saved.
* **Reduce paper work and time:** This software reduces time and enter data generally faster than humans, which takes less time of employees so that they can focus on their other work too. However, if software is developed it also reduce the paper work of humans.
* **Easily modified:** In computerized system the data is modified easily and we can easily update, delete, modify, etc. the records of the employees.
* **Eliminates error and reduce the cost:** Computer generally record data and there is very less chances of errors and it saves human resources. It is more accurate to use computers rather than to write in the pages. Human errors may cost the organization the unnecessary costs and manager has to check twice.

**Importance of the Work/project**

* Storing of data is computerized and therefore in future there is no loss of data.
* Chances of error in computerized software are very less and therefore the errors are minimized.
* Computerized Software reduces the cost and time of the employees and organization.
* Maintain a proper database for all the employees or for an organization.
* Setting the goals for employees, developing skills and motivating them.
* Track the progress of the organization, track the progress of employees within the organization.
* There is a better accessibility and communication of employees within or out of the organization.

**CHAPTER 2**

**SYSTEM ANALYSIS**

2.1 Analysis Methodology /requirement gathering techniques

2.2 Feasibility Study

2.3 Choice of Platforms

2.3.1 Software used

2.3.2 Hardware used

## ANALYSIS METHODOLOGY/ REQUIREMENTS GATHERING

**TECHNIQUES**

* There are numerous ways to gather information from the employees but the best way is to conduct the interviews to gather employee’s information.
* Also, we can conduct the surveys or questionnaires which allow analysts to gather or collect information from employees. Asking the same questions in different manner, so that customer do not get irritate.
* Also, we can conduct group interview, the advantage of having the group interview of employees is we can gather more information at once else interviewing the employee’s different ways.
* We should also remove the communication barrier. The manager and employee should be comfortable in sharing the thoughts with each other else the requirements or information is not gathered properly.
* Also, user observation is necessary as after interviewing the employees we have to see the feedback of them based on all the surveys and questionnaires or the face to face interview. Feedback of each employee is necessary in an organization, so that information is gathered.

## FEASIBILITY STUDY

A feasibility study aims to uncover the strengths and weakness of a project, opportunities and threats present in environment. It is an analysis that takes the factors of the project into account.

There are 3 types of Feasibility Study that is performed: -

#### Economic Feasibility Study

1. **Technical Feasibility Study**
2. **Operational Feasibility Study**
3. **Economic Feasibility Study:**

It is used to indicate the benefit of a project for distributing funds and taking benefits into consideration. As name says “economic”, however it means things related to funds or costs.

With the help of Employee Management System Software, the organization saves a lot of funds which is used in present and as well as in future too.

#### Technical Feasibility Study:

It is used to evaluate the details of how and why you are delivering the services to the employees. It also looks that if the software is successfully satisfying the employee’s need/requirements.

With the help of Employee Management System Software, the organization is satisfying the employees with their fulfillments of the requirements and information they asked for.

#### Operational Feasibility Study:

It is used to measure how a system solves the problems and identifies how the system is fulfilling the employee’s requirements during the development of the project.

With the help of Employee Management System Software, the organization checks and focuses on level on which the proposed system stands with the existing environment.

## CHOICE OF PLATFORM

* OS: **“Operating System”** An interface between the hardware and programs used or run by employees. We have used **WINDOWS 10 HOME.**
* Front End: It is a **GUI** of a computer that makes it to use easily.
* Back end: It is a **File Handling** which acts like a Server or application server.
* Turbo C: It is an application in which **C programming** is done. It is a compiler used to compile the C language code.

## HARDWARE USED

* Processor: It is a machine which processes something known as CPU. We have used i5- 8250u.
* Clock Speed: It records the speed of functioning of computer

* RAM: **“Random Access Memory”** used to store the data or the information in running programs. We have used 8gb RAM.
* Monitor: It is an electronic display that includes the **screen** in which everything is shown in a form of picture.
* Mouse: It is a device which works as a **cursor or pointer** on the screen. It is very much helpful.
* Keyboard: It is a device that allows user to **enter the characters** they want to, like: numbers, alphabets, symbols, etc.

**CHAPTER 3 SYSTEM DESIGN**

Process Model used

Database Design (ERD/Data Dictionary/ Table Design)

Functional Design (DFD)

Interface Design (Screen shots of forms)

Output Design (Screen Shots of Report)

**Process Model Used**

A traditional process model that is “Waterfall Model” is being used in the software “Employee Management System”.

Structure of the Waterfall Model” diagram: -



It is called **“Waterfall Diagram”** because all the process is in sequential manner no process can skip in between the model. The work done by first process is used as a base line for second process.

* **Analyzing: -** In the very first phase we have to analyze all the requirements related to the software and put it into the specific document.
* **Designing: -** After analyzing phase, the next phase is to design the software by preparing the proper design, ERD diagram, DFD diagram are made.
* **Coding: -** When design is prepared then the code has to done for the software in specific programming language either Java, C or on any other platform
* **Testing: -** After coding phase is completed the testing phase comes. When the testing is done for the small modules then this technique is known as unit testing. The combination of the modules of all required modules is called system testing and testing is done on the basis of code.
* **Implementation and Maintenance: -** After testing are over implementation the software is done and maintenance of the software is done before and after the software is delivered.

Every software which is developed is different from another software in any one way and it also requires a suitable model of Software Development Life Cycle (SDLC).

The use of Waterfall Model is going to be most appropriate where:

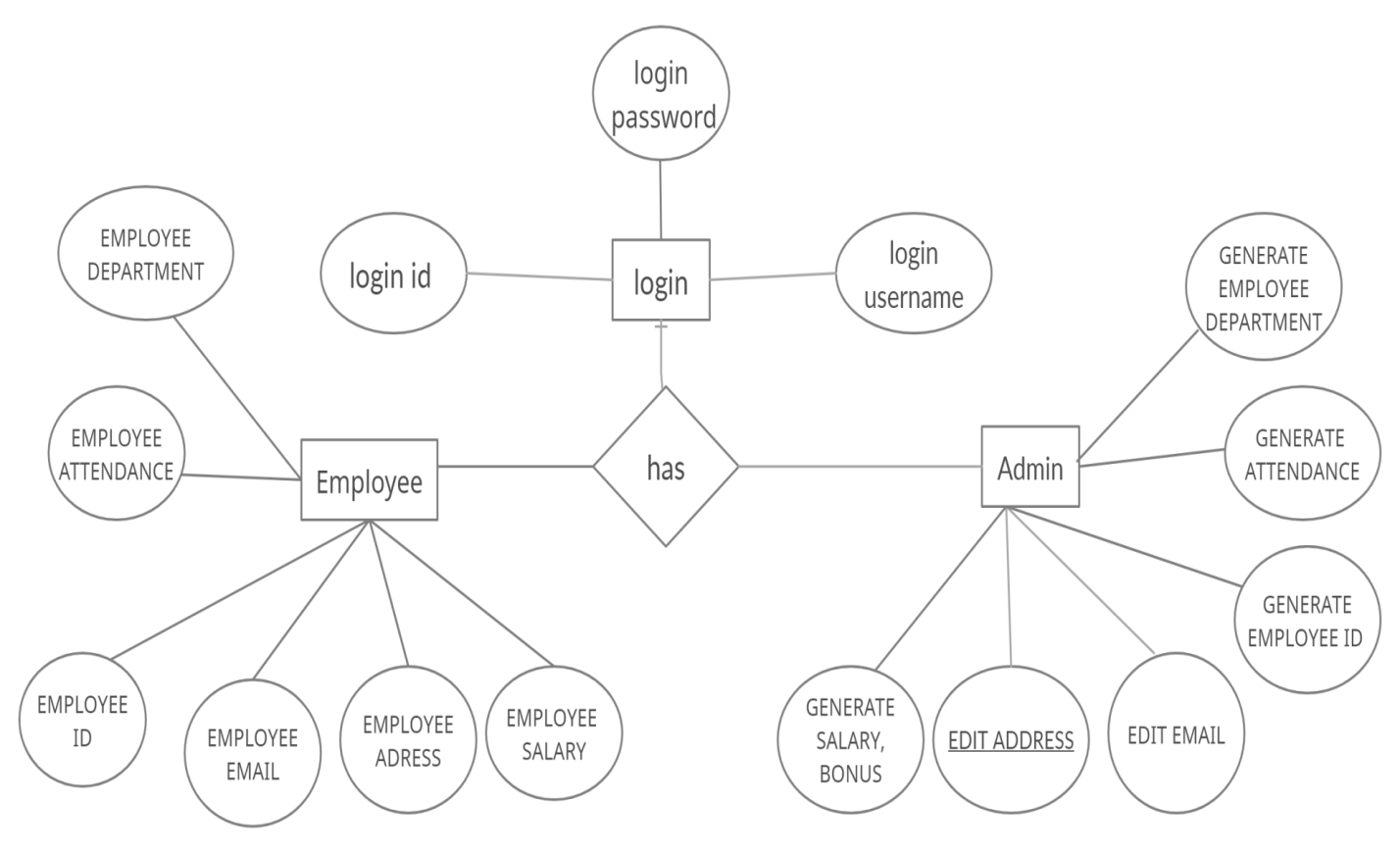
* Requirements of the software are documented and are clear
* Technology is easily understood
* The project is short and free of ambiguity

**Advantages** of Waterfall Model are: -

* Used easily and in a simple way
* Easy to manage if rigidity is there
* If one phase is not completed, the other phase will not start
* Stages are defined clearly
* The model is understood easily
* Tasks are arranged easily
* Results are documented in a systematic way

**Database Design**

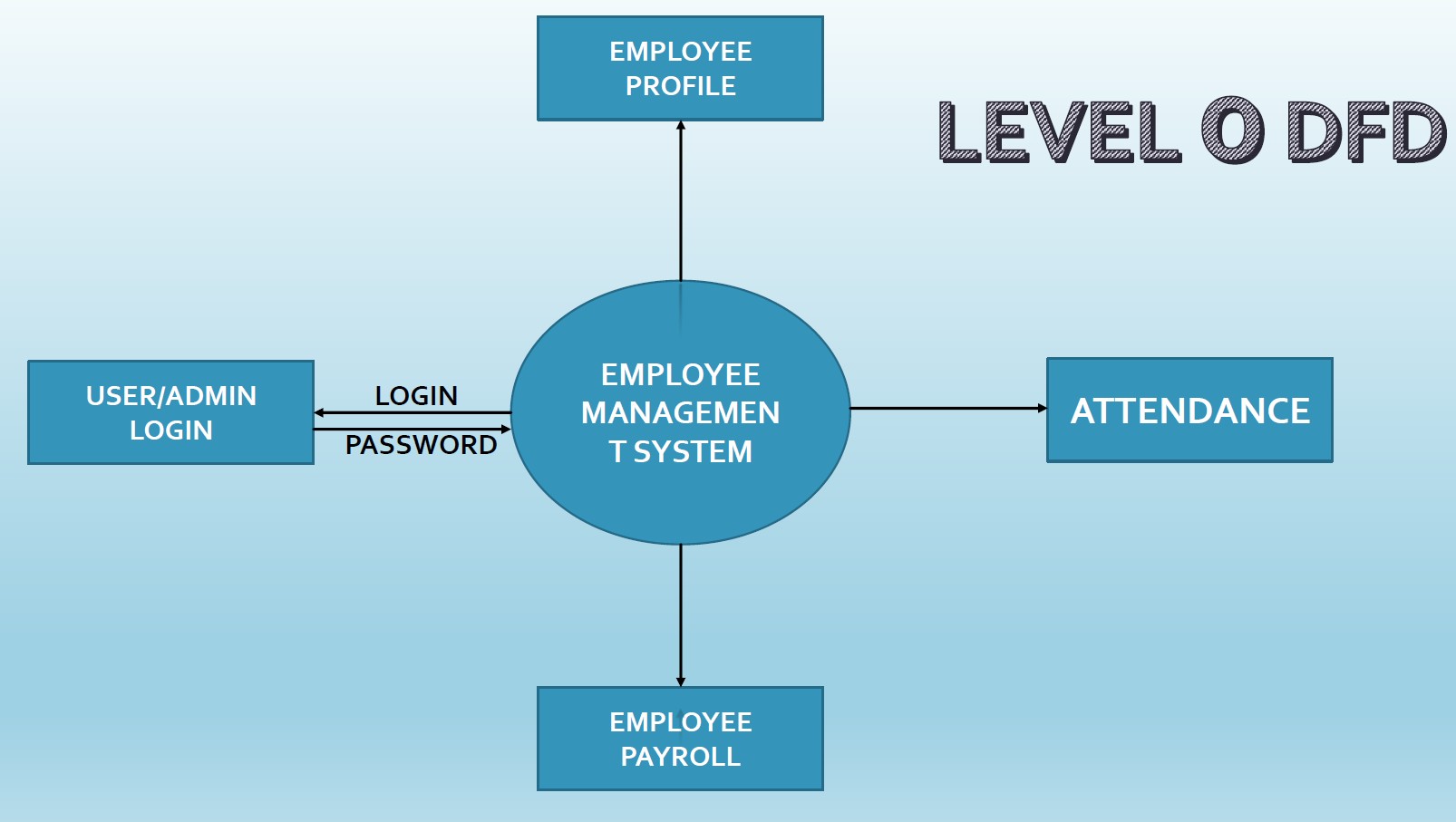
**ERD**

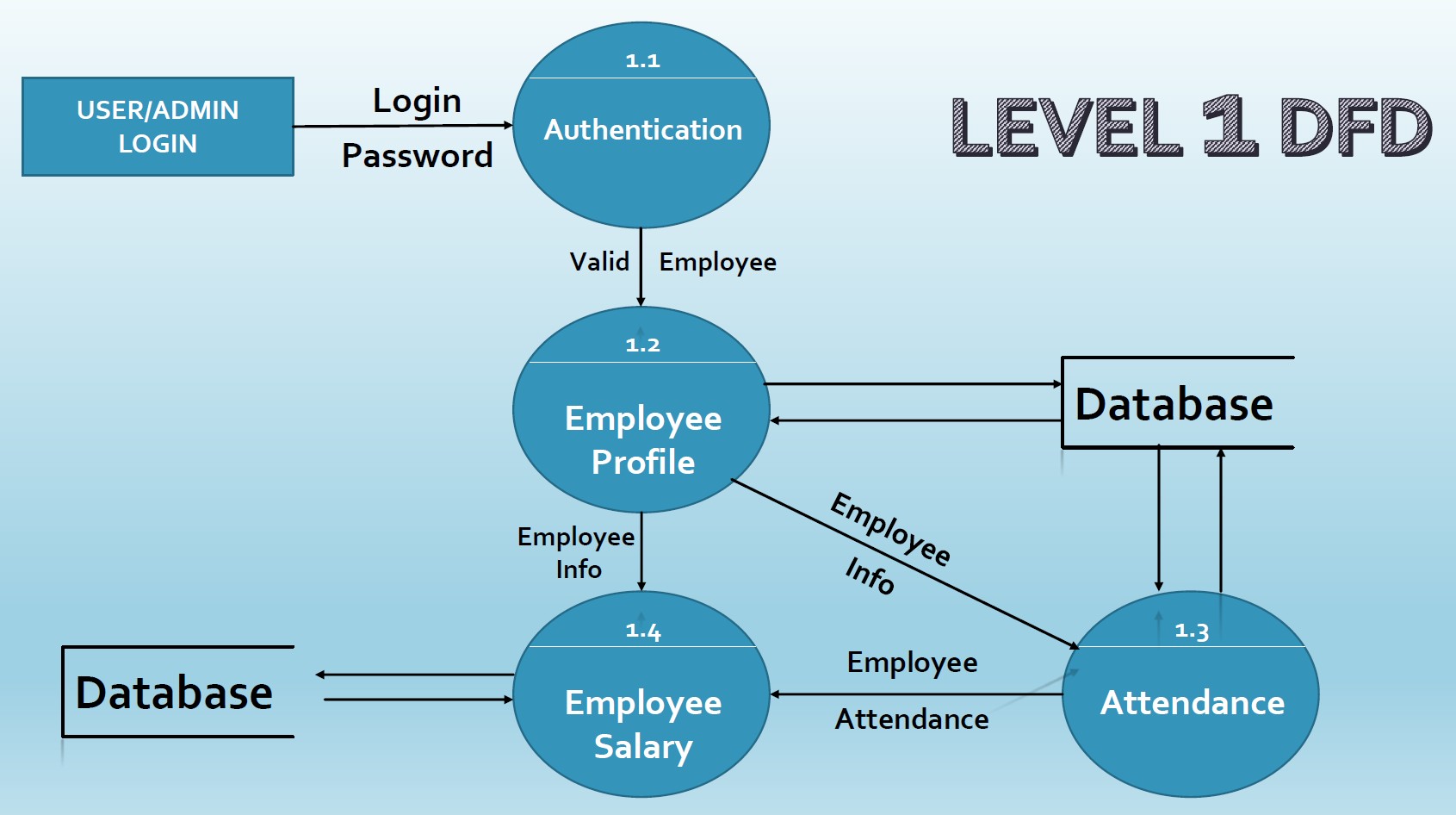
****

**Functional Design**

**DFD mentioned in Employee Management System**

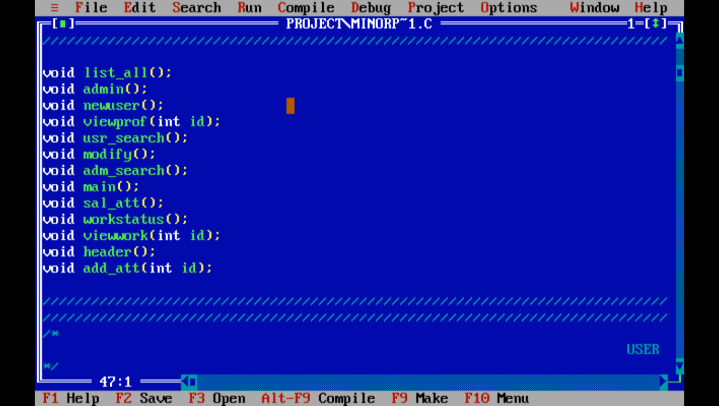
**LEVEL 0 DFD**

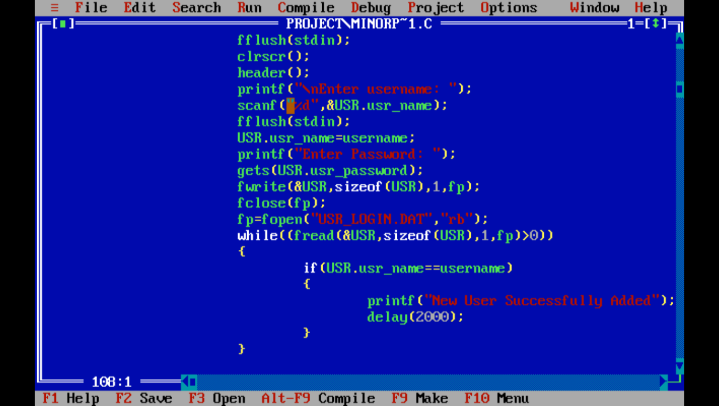


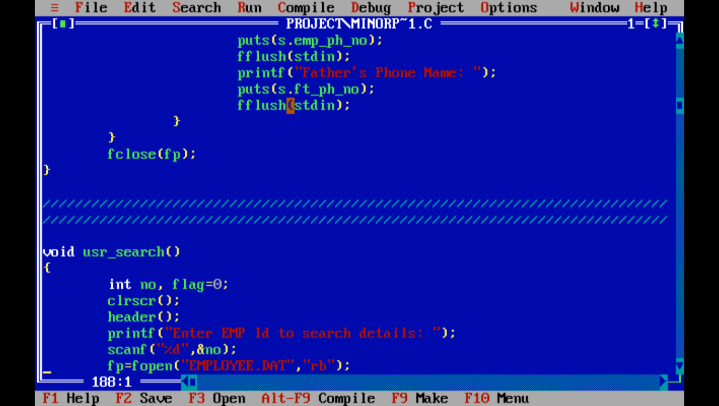


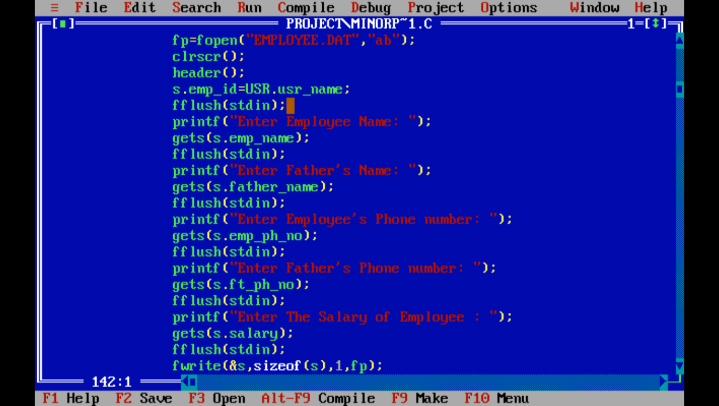
**LEVEL 1 DFD**

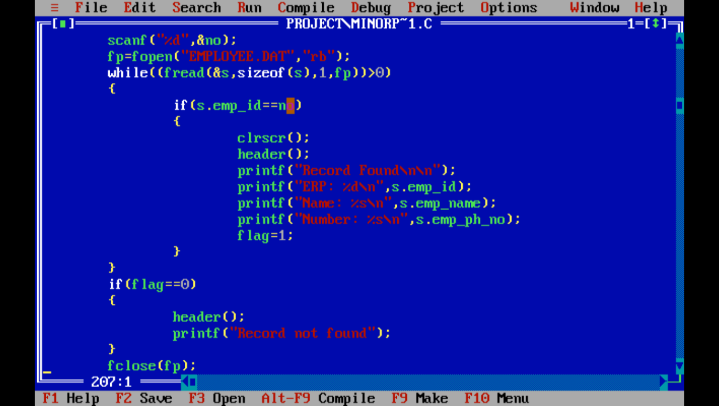
**INTERFACE DESIGN**

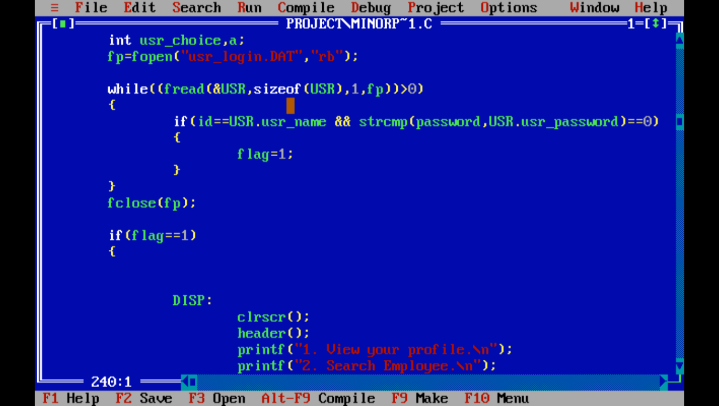


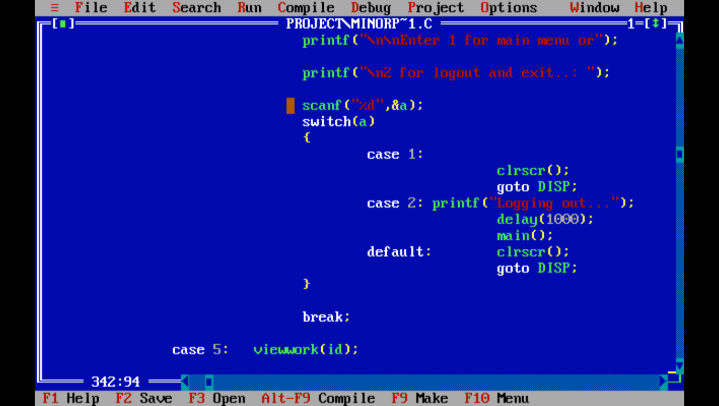


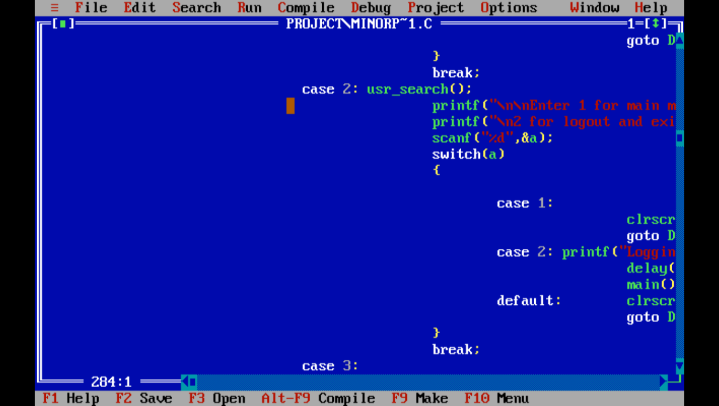


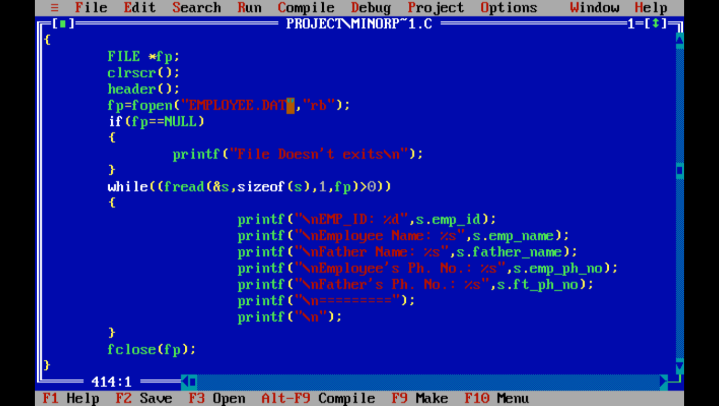


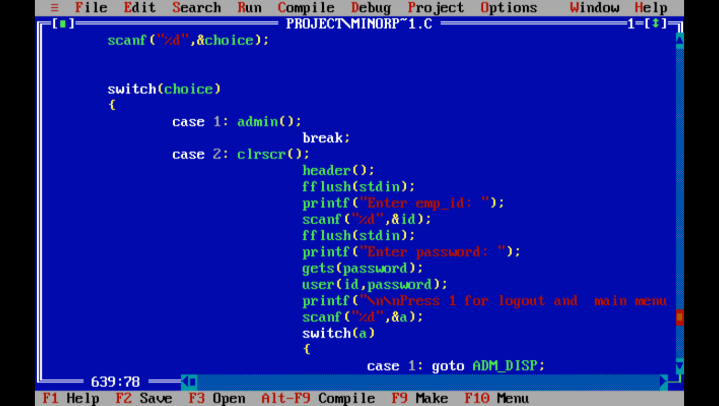






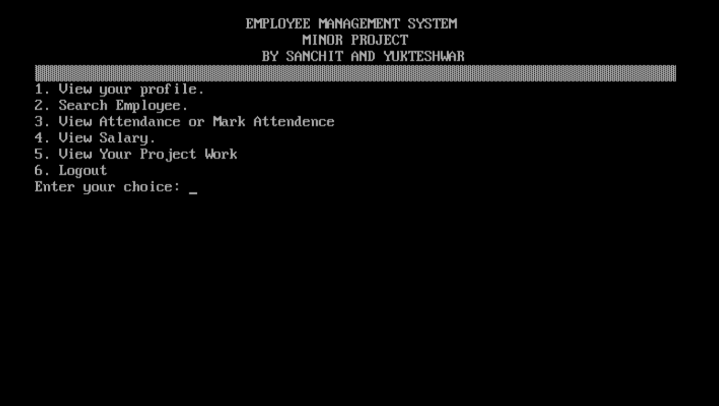






**OUTPUT DESIGN**





**CHAPTER 4 TESTING AND IMPLEMENTATION**

Testing Methodology

Testing Methodology applied

Test Cases

Test Data

Gap Analysis

Rework/ Retest

Hardware & Software Requirement

Hardware Requirement

Software Requirement

**Testing Methodology**

**UNIT TESTING:**

It is the lowest level of checking the software or the primary step for testing the software. In this testing, each unit is checked or tested thoroughly. It has many inputs but a single output. Further all units are integrated and goes further for testing.

#### INTEGRATION TESTING:

It is the second level of checking or testing the software. All the units are integrated and comes for integration testing that is it will be tested in a group. The main purpose of testing this is to find the faults between the integrated units.

#### SYSTEM TESTING:

It is the third level and also the last level of checking or testing the software. It is done after the integration testing by the team. System Testing tests the complete software for the errors, defects, bugs and also checks that system meets the user requirements. After this testing, the acceptance testing takes place but it is done by end user before accepting the software for his/her satisfaction.

**Testing Methodology Applied**

#### UNIT TESTING:

Unit testing tests each and every unit of the code, even the smallest unit is checked or tested thoroughly. However, there are very less chances of error in the code.

#### INTEGRATION TESTING:

Integration testing is done after unit testing completes successfully. It checks the error that can occur due to integration of different modules.

#### SYSTEM TESTING:

It is the third level and also the last level of checking or testing the software. It is done by the developing team before the delivery of the software to the end user. It thoroughly checks the software on various parameters so that user/end user may not find any defect in it.

## TEST CASES

### It is a set of variables under which the main leader of test team will decide that the system satisfies the requirements and work correctly or not. It also finds problems in the design of the software.

#### CHARACTERISTICS OF GOOD TEST CASE

* Easily identified, obviously by its name
* Simple and easily tested, no need of any senior’s help
* It traces all the requirements
* Quality is assured of the software
* Avoid doing unnecessary things
* Should be consistent, complete and clear
* Should be written without any ambiguity

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Sample input | Sample output | Interpretations |
| 1. | To login:  If **ID and Password**  both are CORRECT | User will be logged in. | Pass |
| 1.1 | If only **ID** is  CORRECT | User will not be  logged in. | Pass |
| 1.2 | If both are  INCORRECT | User will not be  logged in. | Pass |
| 2. | To ADD RECORD  if same ID already exist. | User will get error that same ID  already exist. | Pass |

|  |  |  |  |
| --- | --- | --- | --- |
| 2.1 | To ADD RECORD  if same ID doesn’t exist already. | Record will be added successfully. | Pass |
| 3. | To DELETE  RECORD if record exist. | Record will be  removed successfully. | Pass |

|  |  |  |  |
| --- | --- | --- | --- |
| 3.1 | To DELETE  RECORD if record doesn’t exist. | User will get error  that ID doesn’t exist. | Pass |
| 4. | To MODIFY  RECORD if record exist. | User will be asked  to enter new details for the ID. | Pass |
| 4.1 | To MODIFY  RECORD if record doesn’t exist. | User will get error that record doesn’t  exist. | Pass |
| 5. | To SEARCH  RECORD if already exist. | User will find the  details of the record. | Pass |
| 5.1 | To SEARCH  RECORD if it doesn’t exist. | User will get error that record not  found. | Pass |
| 6. | To VIEW all the records if record  exist. | User will see all the records . | Pass |
| 6.1 | To VIEW all the records if no record exists. | User will see a blank screen with option to exit or move to main  menu. | Pass |
| 7. | To SEARCH  RECORD if already  exit. (in case of USER login) | Users will find brief details of the record. (in case of  USER login) | Pass |
| 7.1 | To SEARCH  RECORD if it  doesn’t exist. (in case of USER login) | User will get error that record not found. (in case of  USER login) | Pass |

**GAP ANAYSIS**

GAP Analysis is the process of comparing actual performance with the expected performance to decide whether it is meeting the expectations of the employees or not.

GAP analysis also looks at the progress of the current project and plans about the future of the organization. Also, it is important to do the GAP Analysis of the project so that we are aware about our current state of the project.

## 

## RETEST

Retest is necessary after rework. Rework is done to remove the errors but rework can lead to different error or it may not remove all the error. Retest is done to ensure that errors which were detected in the last build were removed in the rework.

**REQUIREMENTS OF THE SOFTWARE**

|  |  |  |
| --- | --- | --- |
| * Processor: |  | Intel(R) Pentium (R) CPU N3540 @ 2.16 GHz |
| * RAM: |  | 2 GB |
| * HDD: |  | 10 GB |
| * OS: |  | Microsoft Windows |
| * Keyboard: |  | Yes |
| * Mouse: |  | No |

**CHAPTER 5 CONCLUSION AND REFERENCES**

Conclusion

Limitation of System

Future Scope for Modification

References/ Bibliography

## CONCLUSION

The project “Employee Management System” is made for the employees and to ease the work of humans. It is a computer-based application, humans can do the work on computer rather than doing the work manually which will save their time and efforts. Rather than recoding the data of employee manually, humans can use this software to record the data.

C with File Handling is used to develop this software and this software is user-friendly, however it is very simple to use. Personal and Professional data of the employees are secure and modified easily without making any new copies of the employees.

However, this project conclude that the records of the employees and organizations should be within this software so that humans can save their time and efforts, which they can put in their other works.

**LIMITATION OF SYSTEM**

* It cannot provide any data without human interaction
* Previous data (which is not needed) is deleted manually
* Only applicable in maintain employees record of a specific organization
* It can’t be access from anywhere in the world as it is not an online system
* The system does not generate automatic alerts about low attendance or alerts about duplicate or missing information.

## FUTURE SCOPE OF MODIFICATION

As for future, some additions can be implemented into the application of code, however, code will be more flexible and reliable. The role of such systems is the basic need for the organization and employees within it and there’s a good control and records or data are maintained very well.

Some of the features that will be added to the software are

* Marinating the system for a macro enterprise.
* Including features like clock checking in, auto generated taxation system.
* A larger work flow system.
* Attendance system connected to the salary system and a calendar.
* Attendance system that alerts the employee for low attendance.

## 

## REFERENCE / BIBLIOGRAPHY

**ANNEXURES**

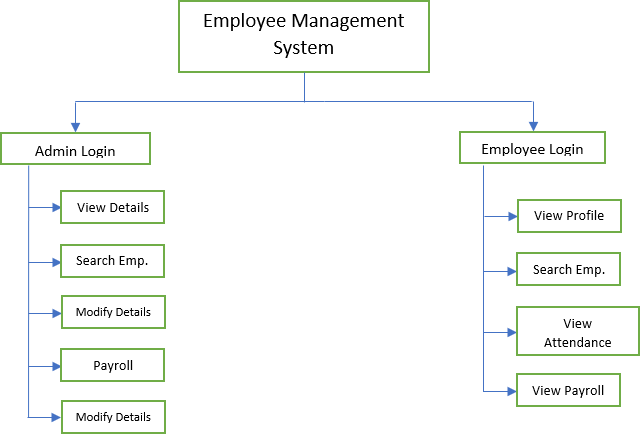
**Menu Flow Diagram**

**Sample Input**

**Sample Output**

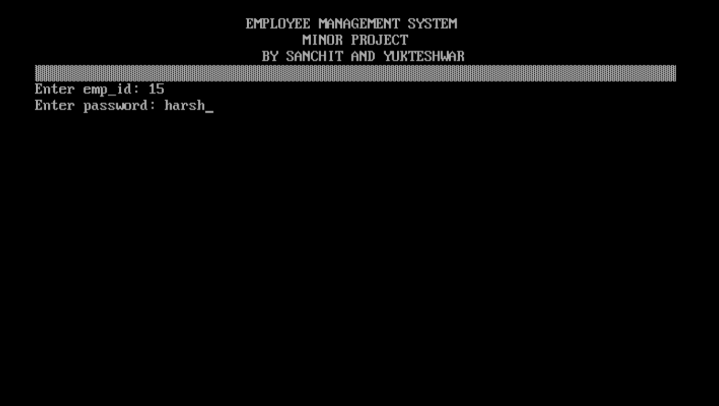
**Program Code**

**MENU FLOW DIAGRAM**

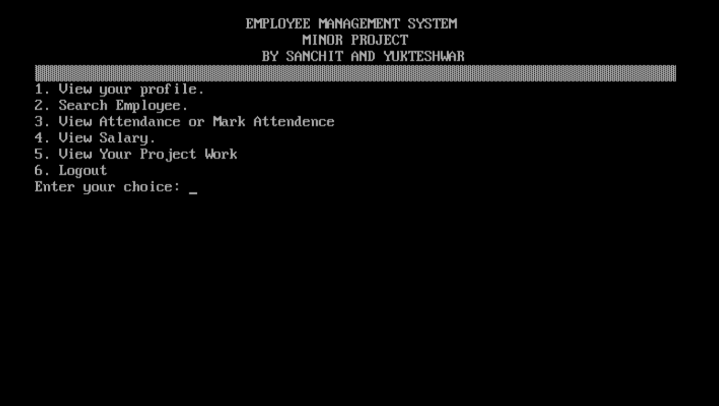
**v**

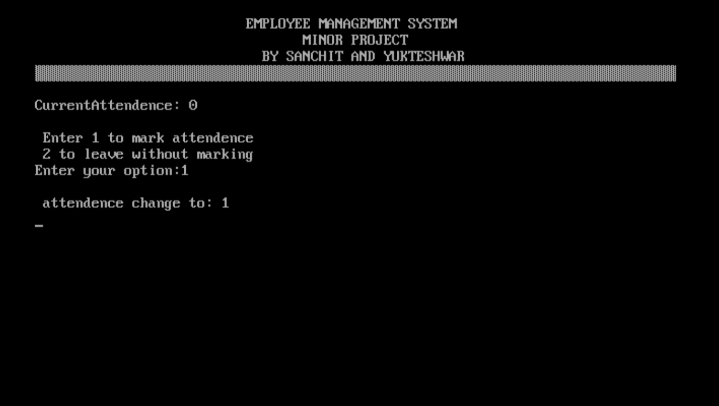
**Sample Input**

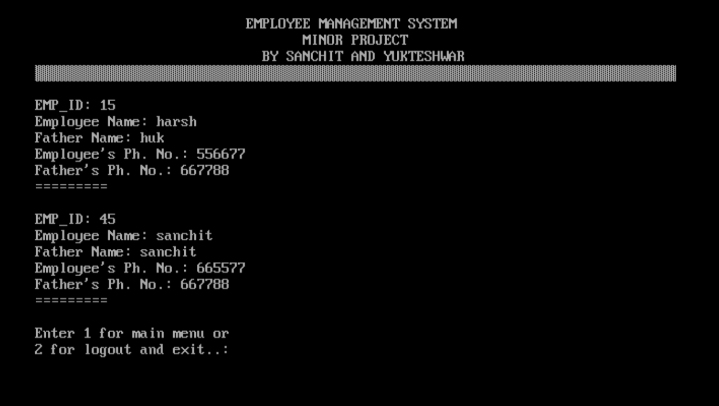




**Sample Output**







**Program Code**

#include <dos.h>

#include <stdio.h>

#include <conio.h>

#include <string.h>

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

struct info

{

char emp\_name[30], father\_name[30];

int emp\_id, attendence;

char emp\_ph\_no[15], ft\_ph\_no[10];

char salary[10];

char work[15], status[15];

} s;

struct user

{

int usr\_name;

char usr\_password[10];

} USR;

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

FILE \* fp;

int flag;

int add = 0;

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void list\_all ();

void admin ();

void newuser ();

void viewprof (int id);

void usr\_search ();

void modify ();

void adm\_search ();

void main ();

void sal\_att ();

void workstatus ();

void viewwork (int id);

void header ();

void add\_att (int id);

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

/\*

USER

\*/

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

newuser ()

{

char masterkey[10], password[20];

int username;

printf ("Enter masterkey: ");

fflush (stdin);

gets (masterkey);

if (strcmp (masterkey, "sanchit") == 0)

{

fp = fopen ("usr\_login.DAT", "rb");

fflush (stdin);

clrscr ();

header ();

printf ("Enter erp\_id: ");

scanf ("%d", &username);

while ((fread (&USR, sizeof (USR), 1, fp)) > 0)

{

if (username == USR.usr\_name)

{

flag = 1;

}

}

fclose (fp);

if (flag == 1)

{

printf ("Cannot add same employee again...!!!");

}

else

{

fp = fopen ("USR\_LOGIN.DAT", "ab");

fflush (stdin);

clrscr ();

header ();

printf ("\nEnter username: ");

scanf ("%d", &USR.usr\_name);

fflush (stdin);

USR.usr\_name = username;

printf ("Enter Password: ");

gets (USR.usr\_password);

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

fclose (fp);

fp = fopen ("USR\_LOGIN.DAT", "rb");

while ((fread (&USR, sizeof (USR), 1, fp) > 0))

{

if (USR.usr\_name == username)

{

printf ("New User Successfully Added");

delay (2000);

}

}

add = 1;

}

}

else

{

header ();

printf ("Unauthorized Acess Denied...!!!");

delay (2000);

exit ();

}

if (add == 1)

{

clrscr ();

fp = fopen ("EMPLOYEE.DAT", "ab");

clrscr ();

header ();

s.emp\_id = USR.usr\_name;

fflush (stdin);

printf ("Enter Employee Name: ");

gets (s.emp\_name);

fflush (stdin);

printf ("Enter Father's Name: ");

gets (s.father\_name);

fflush (stdin);

printf ("Enter Employee's Phone number: ");

gets (s.emp\_ph\_no);

fflush (stdin);

printf ("Enter Father's Phone number: ");

gets (s.ft\_ph\_no);

fflush (stdin);

printf ("Enter The Salary of Employee : ");

gets (s.salary);

fflush (stdin);

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

fclose (fp);

printf ("\n");

printf ("Employee record has been created");

printf ("\n");

delay (2000);

}

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

viewprof (int id)

{

clrscr ();

header ();

fp = fopen ("EMPLOYEE.DAT", "rb");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (id == s.emp\_id)

{

printf ("\t\tProfile Information\t\t\n");

printf ("\nERP: %d", s.emp\_id);

printf ("\nEmployee Name: %s", s.emp\_name);

printf ("\nFather Name: %s", s.father\_name);

printf ("\nEmployee's Phone number:");

puts (s.emp\_ph\_no);

fflush (stdin);

printf ("Father's Phone Name: ");

puts (s.ft\_ph\_no);

fflush (stdin);

}

}

fclose (fp);

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

usr\_search ()

{

int no, flag = 0;

clrscr ();

header ();

printf ("Enter EMP Id to search details: ");

scanf ("%d", &no);

fp = fopen ("EMPLOYEE.DAT", "rb");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (s.emp\_id == no)

{

clrscr ();

header ();

printf ("Record Found\n\n");

printf ("ERP: %d\n", s.emp\_id);

printf ("Name: %s\n", s.emp\_name);

printf ("Number: %s\n", s.emp\_ph\_no);

flag = 1;

}

}

if (flag == 0)

{

header ();

printf ("Record not found");

}

fclose (fp);

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

/\*

USER

\*/

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

user (int id, char password[10])

{

int usr\_choice, a;

fp = fopen ("usr\_login.DAT", "rb");

while ((fread (&USR, sizeof (USR), 1, fp)) > 0)

{

if (id == USR.usr\_name && strcmp (password, USR.usr\_password) == 0)

{

flag = 1;

}

}

fclose (fp);

if (flag == 1)

{

DISP:

clrscr ();

header ();

printf ("1. View your profile.\n");

printf ("2. Search Employee.\n");

printf ("3. View Attendance or Mark Attendence\n");

printf ("4. View Salary.\n");

printf ("5. View Your Project Work\n");

printf ("6. Logout\n");

printf ("Enter your choice: ");

scanf ("%d", &usr\_choice);

switch (usr\_choice)

{

case 1:

viewprof (id);

printf ("\n\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

main ();

default:

clrscr ();

goto DISP;

}

break;

case 2:

usr\_search ();

printf ("\n\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

main ();

default:

clrscr ();

goto DISP;

}

break;

case 3:

add\_att (id);

printf ("\n\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

main ();

default:

clrscr ();

goto DISP;

}

break;

case 4:

clrscr ();

header ();

fp = fopen ("EMPLOYEE.DAT", "rb");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (id == s.emp\_id)

{

printf ("\nSalary of this Month: %s", s.salary);

}

}

fclose (fp);

printf ("\n\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

main ();

default:

clrscr ();

goto DISP;

}

break;

case 5:

viewwork (id);

printf ("\n\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

main ();

default:

clrscr ();

goto DISP;

}

break;

case 6:

printf ("Logging out ");

delay (2000);

clrscr ();

main ();

default:

printf ("\ninvalid number selected\n");

delay (3000);

goto DISP;

}

}

else

{

header ();

printf ("Invalid login...");

printf ("\n\nLogging out...");

delay (1000);

exit ();

}

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

/\*

ADMIN

\*/

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

list\_all ()

{

FILE \* fp;

clrscr ();

header ();

fp = fopen ("EMPLOYEE.DAT", "rb");

if (fp == NULL)

{

printf ("File Doesn't exits\n");

}

while ((fread (&s, sizeof (s), 1, fp) > 0))

{

printf ("\nEMP\_ID: %d", s.emp\_id);

printf ("\nEmployee Name: %s", s.emp\_name);

printf ("\nFather Name: %s", s.father\_name);

printf ("\nEmployee's Ph. No.: %s", s.emp\_ph\_no);

printf ("\nFather's Ph. No.: %s", s.ft\_ph\_no);

printf ("\n=========");

printf ("\n");

}

fclose (fp);

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

/\*

ADMIN

\*/

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

admin ()

{

int adm\_choice, a, flag;

char masterkey[5], username[10], name[10], password[10];

clrscr ();

header ();

fflush (stdin);

printf ("Enter username: ");

gets (name);

fflush (stdin);

printf ("Enter password: ");

gets (password);

if (strcmp (name, "sanchit") == 0)

{

if (strcmp (password, "sanchit") == 0)

{

DISP:

clrscr ();

header ();

printf ("\n1. View Employee details.");

printf ("\n2. Search Employee.");

printf ("\n3. Modify Employee details.");

printf ("\n4. Edit Employee salary or attendence.");

printf ("\n5. Add New User login.");

printf ("\n6. Add Work for Employee. ");

printf ("\n7. Logout");

printf ("\nEnter your choice: ");

scanf ("%d", &adm\_choice);

switch (adm\_choice)

{

case 1:

list\_all ();

printf ("\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

main ();

default:

clrscr ();

goto DISP;

}

break;

case 2:

adm\_search ();

printf ("\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

exit ();

default:

clrscr ();

goto DISP;

}

break;

case 3:

modify ();

printf ("\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

exit ();

default:

clrscr ();

goto DISP;

}

break;

case 4:

sal\_att ();

printf ("\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

exit ();

default:

clrscr ();

goto DISP;

}

break;

case 5:

newuser ();

printf ("\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

exit ();

default:

clrscr ();

goto DISP;

}

break;

case 6:

workstatus ();

printf ("\nEnter 1 for main menu or");

printf ("\n2 for logout and exit..: ");

scanf ("%d", &a);

switch (a)

{

case 1:

clrscr ();

goto DISP;

case 2:

printf ("Logging out...");

delay (1000);

exit ();

default:

clrscr ();

goto DISP;

}

break;

logout:

case 7:

printf ("Logging out ");

delay (2000);

clrscr ();

main ();

default:

printf ("invalid number selected");

delay (3000);

goto DISP;

}

}

else

{

header ();

printf ("Invalid login...");

delay (1000);

main ();

}

}

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

add\_att (int id)

{

int b;

clrscr ();

header ();

fp = fopen ("EMPLOYEE.DAT", "rb+");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (id == s.emp\_id)

{

printf ("\nCurrentAttendence: %d\n", s.attendence);

printf ("\n Enter 1 to mark attendence");

printf ("\n 2 to leave without marking");

printf ("\nEnter your option:");

scanf ("%d", &b);

switch (b)

{

case 1:

s.attendence++;

printf ("\n attendence change to: %d\n", s.attendence);

fseek (fp, -(long) sizeof (s), 1);

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

delay (3000);

break;

case 2:

printf ("\nLogging out...\n");

delay (1000);

break;

default:

printf ("without marking attendence");

delay (2000);

break;

}

break;

}

}

fclose (fp);

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

adm\_search ()

{

int no;

clrscr ();

header ();

printf ("Enter EMP Id to search details: ");

scanf ("%d", &no);

fp = fopen ("EMPLOYEE.DAT", "rb");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (s.emp\_id == no)

{

clrscr ();

header ();

printf ("Record Found");

printf ("\nERP: %d", s.emp\_id);

printf ("\nEmployee Name: %s", s.emp\_name);

printf ("\nFather's Name: %s", s.father\_name);

printf ("\nEmployee Number: %s", s.emp\_ph\_no);

printf ("\nFather's Number: %s", s.ft\_ph\_no);

printf ("\nEmployee Salary: %s", s.salary);

printf ("\nEmployee Attendence: %d", s.attendence);

printf ("\nWork Assigned: %s", s.work);

printf ("\nWork Status: %s", s.status);

flag = 1;

}

}

if (flag == 0)

{

gotoxy (40, 15);

printf ("\n\nRecord not found\n\n");

}

fclose (fp);

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

modify ()

{

FILE \* fp;

int no, flag = 0;

printf ("Enter EMP\_ID to modify: ");

scanf ("%d", &no);

fp = fopen ("EMPLOYEE.DAT", "rb+");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (s.emp\_id == no)

{

clrscr ();

header ();

printf ("\nEMP\_ID: %d", s.emp\_id);

printf ("\nEmployee Name: %s", s.emp\_name);

printf ("\nPress any key ...");

getch ();

clrscr ();

header ();

printf ("\nEnter new Detials");

fflush (stdin);

printf ("\nEnter Employee Name: ");

gets (s.emp\_name);

fflush (stdin);

printf ("\nEnter Father's Name: ");

gets (s.father\_name);

fflush (stdin);

printf ("\nEnter Employee Number: ");

gets (s.emp\_ph\_no);

fflush (stdin);

printf ("\nEnter Father's Number: ");

gets (s.ft\_ph\_no);

fflush (stdin);

fseek (fp, -(long) sizeof (s), 1);

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

printf ("Record Updated");

delay (2000);

}

if (s.emp\_id == no)

{

flag = 1;

}

}

fclose (fp);

if (flag == 0)

{

header ();

printf ("Record doesn't exist");

}

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

workstatus ()

{

FILE \* fp;

int no, flag = 0;

clrscr ();

header ();

printf ("Enter EMP\_ID to assign work: ");

scanf ("%d", &no);

fp = fopen ("EMPLOYEE.DAT", "rb+");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (s.emp\_id == no)

{

clrscr ();

header ();

printf ("\nEMP\_ID: %d", s.emp\_id);

printf ("\nEmployee Name: %s", s.emp\_name);

printf ("\nCurrent work Assigned: %s", s.work);

printf ("\nPress any key ...");

getch ();

clrscr ();

header ();

printf ("\nEnter new Detials");

fflush (stdin);

printf ("\nEnter Work to be Assigned: ");

gets (s.work);

fflush (stdin);

printf ("\nEnter Work Status: ");

gets (s.status);

fflush (stdin);

fseek (fp, -(long) sizeof (s), 1);

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

printf ("\n\nWork Assigned");

delay (2000);

}

if (s.emp\_id == no)

{

flag = 1;

}

}

fclose (fp);

if (flag == 0)

{

header ();

printf ("Record doesn't exist");

}

}

////////////////////////////////////////////////////////////////////////////////////////////////

////////////////////////////////////////////////////////////////////////////////////////////////

void

viewwork (int id)

{

clrscr ();

header ();

fp = fopen ("EMPLOYEE.DAT", "rb");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (id == s.emp\_id)

{

printf ("\nProject you are working on: %s", s.work);

printf ("\nYour Project Status: %s", s.status);

}

}

fclose (fp);

}

/////////////////////////////////////////////////////////////////////////////

/////////////////////////////////////////////////////////////////////////////

void

main ()

{

int choice;

int a, id;

char password[10];

ADM\_DISP:clrscr ();

gotoxy (20, 8);

printf

("\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2");

gotoxy (20, 9);

printf ("\xb2\xb2 LOGIN \xb2\xb2");

gotoxy (20, 10);

printf ("\xb2\xb2 ~~~~~~~ \xb2\xb2");

gotoxy (20, 11);

printf ("\xb2\xb2 1. ADMIN \xb2\xb2");

gotoxy (20, 12);

printf ("\xb2\xb2 \xb2\xb2");

gotoxy (20, 13);

printf ("\xb2\xb2 2. USER \xb2\xb2");

gotoxy (20, 14);

printf ("\xb2\xb2 \xb2\xb2");

gotoxy (20, 15);

printf ("\xb2\xb2 3. EXIT \xb2\xb2");

gotoxy (20, 16);

printf ("\xb2\xb2 \xb2\xb2");

gotoxy (20, 17);

printf ("\xb2\xb2 Enter your choice: \xb2\xb2");

gotoxy (20, 18);

printf ("\xb2\xb2 \xb2\xb2");

gotoxy (20, 19);

printf

("\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2\xb2");

gotoxy (43, 17);

scanf ("%d", &choice);

switch (choice)

{

case 1:

admin ();

break;

case 2:

clrscr ();

header ();

fflush (stdin);

printf ("Enter emp\_id: ");

scanf ("%d", &id);

fflush (stdin);

printf ("Enter password: ");

gets (password);

user (id, password);

printf

("\n\nPress 1 for logout and main menu or 2 for logout and exit...");

scanf ("%d", &a);

switch (a)

{

case 1:

goto ADM\_DISP;

case 2:

exit ();

}

break;

case 3:

exit ();

default:

goto ADM\_DISP;

}

getch ();

}

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////

void

sal\_att ()

{

FILE \* fp;

int no, flag = 0;

printf ("Enter EMP\_ID to edit salary or attendence of Employee: ");

scanf ("%d", &no);

fp = fopen ("EMPLOYEE.DAT", "rb+");

while ((fread (&s, sizeof (s), 1, fp)) > 0)

{

if (s.emp\_id == no)

{

clrscr ();

header ();

printf ("\nEMP\_ID: %d", s.emp\_id);

printf ("\nEmployee Name: %s", s.emp\_name);

printf ("\nEmployee Attendence: %d", s.attendence);

printf ("\nEmployee Monthly Salary: %s", s.salary);

printf ("\nPress any key ...");

getch ();

clrscr ();

header ();

printf ("\nModify Detials");

fflush (stdin);

printf ("\nEnter Employee Salary: ");

gets (s.salary);

fflush (stdin);

printf ("\nEnter Employees Attendence:");

scanf ("%d", &s.attendence);

fseek (fp, -(long) sizeof (s), 1);

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

printf ("\nSucessfully Modified");

}

if (s.emp\_id == no)

{

flag = 1;

}

}

fclose (fp);

if (flag == 0)

{

header ();

printf ("Record doesn't exist\n");

}

}

///////////////////////////////////////////////////////////////////////////////////////

///////////////////////////////////////////////////////////////////////////////////////

void

header ()

{

int i;

clrscr ();

gotoxy (27, 2);

printf ("EMPLOYEE MANAGEMENT SYSTEM");

gotoxy (34, 3);

printf ("MINOR PROJECT");

gotoxy (29, 4);

printf ("BY SANCHIT AND YUKTESHWAR\n");

for (i = 1; i <= 79; i++)

{

gotoxy (i, 5);

printf ("\xb2");

}

printf ("\n");

}

**MENTEE FEEDBACK REPORT**

