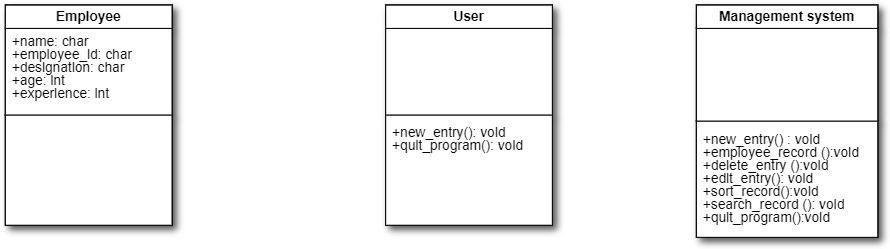
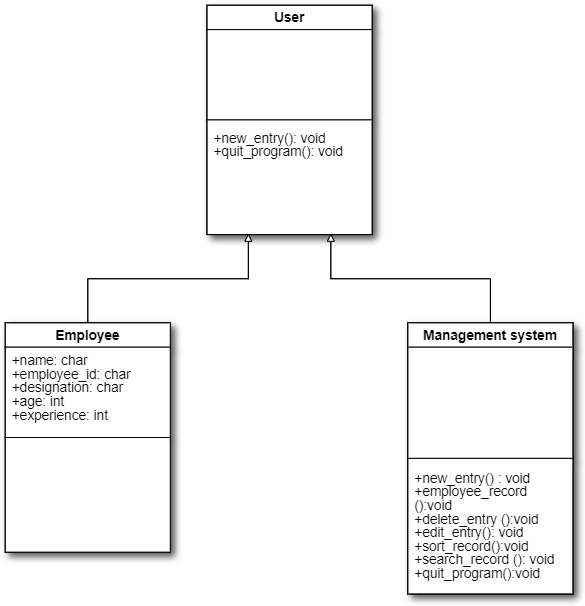
ABSTRACT

**Problem Discussion**: The primary objective of this project was to create a program for an Employee management system using the computer language C++. The program handles and stores all the data of employees in a file. The management system performs features like inserting, deleting and editing entries, searching records and sorting table. This is an academic group project by the team ‘Ultimate Coders’, for the course COMP 150, in the University of the Fraser Valley, instructed and guided by Professor Lili Saghafi.

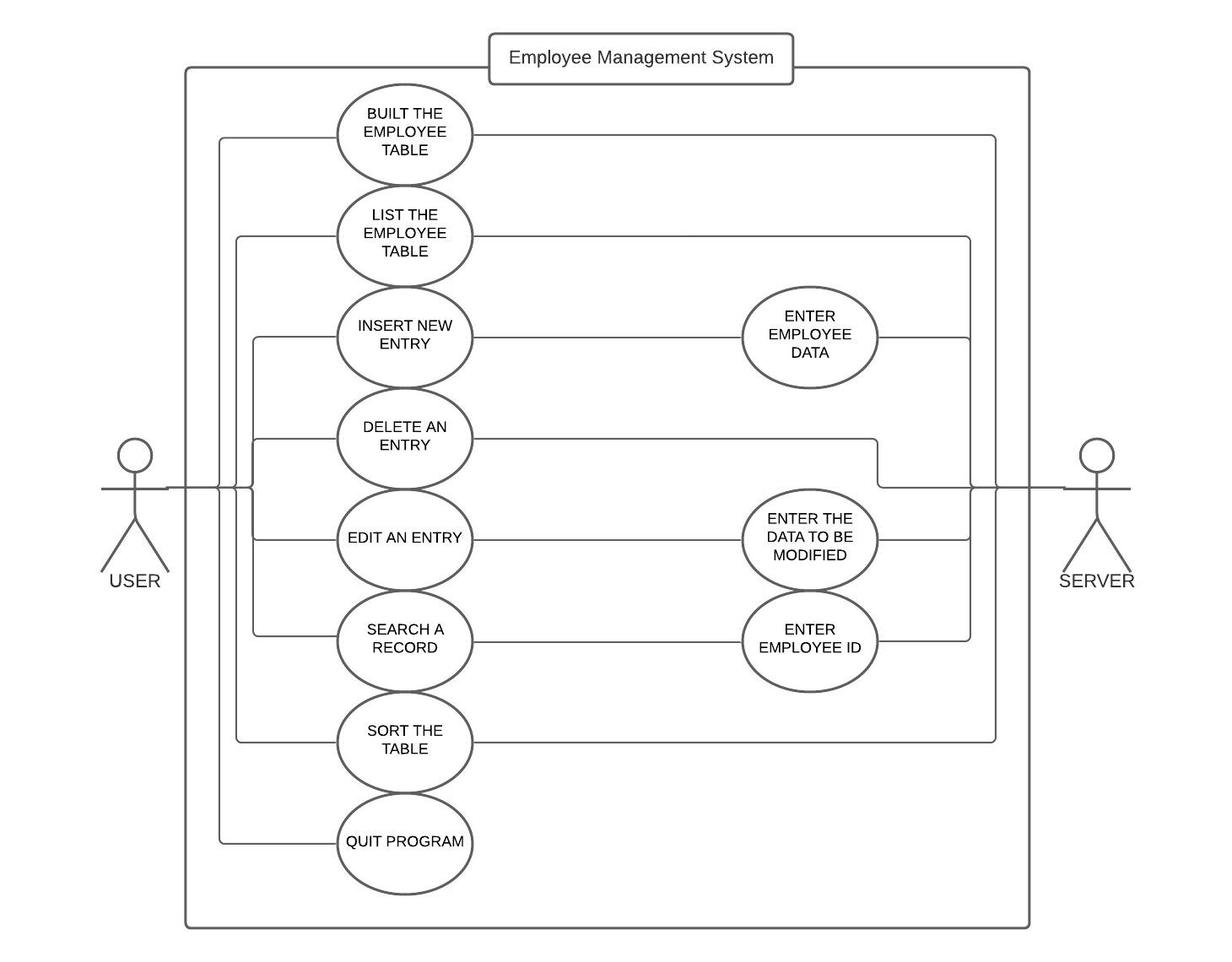
While developing this management system we went through various phases of the software development life cycle. The first step was the planning phase in which we discussed the purpose and the scope of this system. In the second phase which was the analysis phase, various charts such as problem analysis chart, etc. were made so that the members of the team would be aware of the problems to be solved for the successful completion of the project. The next step was to evaluate the design aspect of the program which would outline the working of the program. This came under the design phase. Various charts and diagrams such as the IPO chart, flowchart, CRC diagrams and use case diagrams were made during this phase. The algorithm and pseudocode required to write the actual program were also made during this phase. Now, it was time to write the actual code for the program which came under the implementation phase. During this phase, the code for the program was written and tested multiple times on Code Blocks. This process of writing, testing, and editing the code continued till a fully functional program was prepared. In the final phase, which is the maintenance phase, the fully functional program was tested again, and a training manual was created. (Paridhimathur95, 2021)

CRC Diagram and Class Diagram





Use case diagram

****

Problem Analysis Chart

|  |  |
| --- | --- |
| Given Data | Required Results |
| Section 1:  To create a basic program for Employee Management System that get the details of the employees from the file. | Section 2:   1. View employee records based on what details are required to be viewed. 2. Create new entries of the employees. 3. Delete an entry of the employee. 4. Edit an entry of the employee. 5. Search an entry of the employee. 6. Sort the record of the employees. 7. Quit the program |
| Processing Required | Solution Alternatives |
| Section 3:   1. In a function named main, create a main menu containing member function consisting of seven switch cases. 2. Case 1: build the employee table. 3. Case 2: enlist the employee table. 4. Case 3: insert new entry. 5. Case 4: delete an entry. 6. Case 5: edit an entry. 7. Case 6: search a record. 8. Case 7: sort a record. 9. Else quit the program. | Section 4:   1. Give manager the ability to view employee records based on what details are required to be viewed. 2. Give the ability to create new records of the employee. 3. Give manager the ability to delete entries of the employees. 4. Give manager the ability to edit entries of the employees. 5. Give manager the ability to search entries of the employees. 6. Give manager the ability to sort entries of the employees. 7. Give the ability to quit the program. |

IPO Chart

|  |  |  |  |
| --- | --- | --- | --- |
| Input | Process | Module reference | Output |
| Employee details | Define the function named option. | Define | View employee records based on what details are required to be viewed. |
|  | Create a main menu consisting of seven switch cases. |  | enlist the employee table |
|  | In case 1, a function named build is declared which takes the data from an external file named empdata.txt. In this, employee table is built based on what details are required. | Read | insert new entry. |
|  | Print the record of the employees. | Print | delete an entry. |
|  | In case 2, a function named view is declared which takes the data from an external file named empdata.txt. | Read | edit an entry. |
|  | Print the record of the employees. | Print | search a record. |
|  | In case 3, a function named add is declared where the user enters details of the employee. | Read | sort the table |
|  | In case 4, a function named deletee is declared which deletes the data from an external file named empdata.txt when the manager specifies the employee id. | Read |  |
|  | Print the confirmation for the deletion of employee details. | Print |  |
|  | In case 5, a function named edit is declared which edits the data from an external file named empdata.txt when the manager specifies the employee id. | Read |  |
|  | In case 6, a function named show is declared which enables the manager to search the data from an external file named empdata.txt when the manager specifies the employee id. | Read |  |
|  | Print the employee details based on the employee id. | Print |  |
|  | In case 7, a function named sort is declared which enables the manager to sort the data from an external file named empdata.txt. | Print |  |
|  | Else quit the program |  |  |
|  | End |  |  |

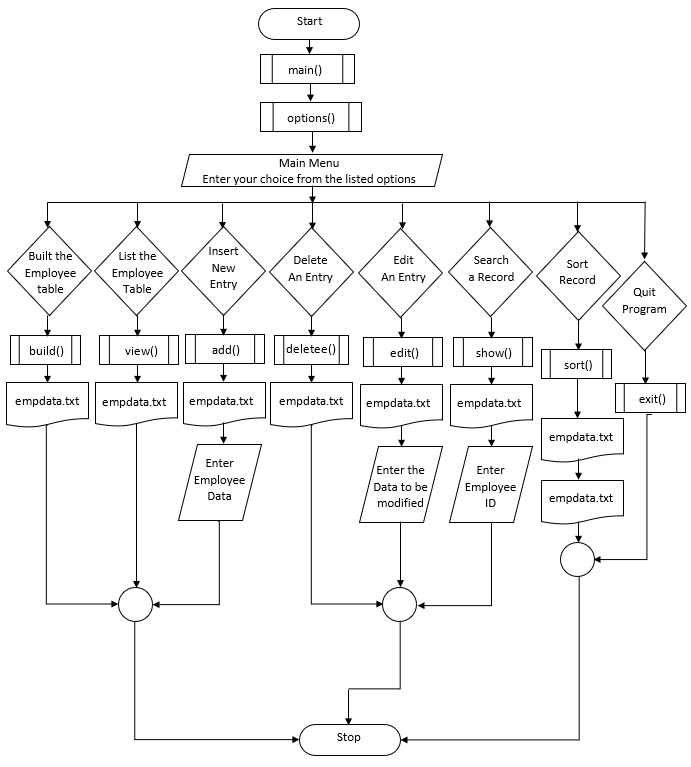
Algorithm

Main Function

Steps

1. Start
2. Instantiate an object e of the class employee.
3. Call the member function named option of the object e which includes:
   * Display the Main Menu of the Employee Management System as shown below:
     + Press 1---->Built The Employee Table
     + Press 2---->List The Employee Table
     + Press 3---->Insert New Entry
     + Press 4---->Delete An Entry
     + Press 5---->Edit An Entry
     + Press 6---->Search a record
     + Press 7---->Sort Record
     + Press 8---------->Quit Program
   * Get input from the user using a switch case:
     + Case 1: call function build.
     + Case 2: call function view.
     + Case 3: call function add.
     + Case 4: call function delete.
     + Case 5: call function edit.
     + Case 6: call function show.
     + Case 7: call function sort.
     + Default: call function exit.
4. Stop.

Flowchart



Pseudocode

Program starts.

Declare and define a Class employee with access specifier set to public.

Declare the member functions empname[40], empcode[5], designation[20], age, experience, emp[max], tempemp[max], sortemp[max] and sortemp1[max].

Declaring functions exit, build, view, show, edit, add, delete and options.

Defining the member function of class employee called exit outside the class.

Output to the screen " Press enter to go back " and read the screen.

Defining the member function of class employee called build outside the class.

Read the name, emp id, experience, age or designation from the screen and store it in the file “empdata.txt”. Switch

In case 1 output to the screen a table with name, code and designation.

In the case 2 output to the screen the name and code in the form of a table.

In case 3 output to the screen a table with name, code and experience.

In the case 4 output to the screen the name, code and age in the form of a table.

In case 3 output to the screen a table with name, code, designation, year and age.

Defining the member function of class employee called view outside the class.

Output to the screen a table with the name, code, designation, years and age from the file “empdata.txt”.

Defining the member function of class employee called sort outside the class.

Open the file “empdata.txt” using open function. Now, read the file and store the data in string variable by\_Line. Close the file using close function.

Declare an extreme variable named inflow2. Open “empdata.txt”. read the file and store the data in a string variable named lock. Compare the data.

Declare an ofstream variable named offflow. Open the file “empdata.txt”. write the data in this file and close the file using close function. Close the file “empdata.txt”. rename “empdata2.txt” file and rename it to “empdata.txt”.

Defining the member function of class employee called show outside the class.

Output “Enter employee ID” and then read the ID from the screen

While scan the file “empdata.txt” for the employee details for the read ID.

If the checked and empcode are equal to zero then output employee details.

Defining the member function of class employee called edit outside the class.

Read new employee details from the screen and save the data to "temp.txt" and remove data from the previous file "empdata.txt".

Then print out to the screen “updated”.

Defining the member function of class employee called add outside the class.

Read employee details from the screen for a new record and save the record to "empdata.txt".

Defining the member function of class employee called deletee outside the class.

Read the employee ID from the screen to be removed from "empdata.txt".

Defining the member function of class employee called options outside the class.

Output the options for the main menu of the employee management system. Switch

In case 1 call the build function.

In case 2 call the view function.

In case 3 call the add function.

In case 4 call the delete function.

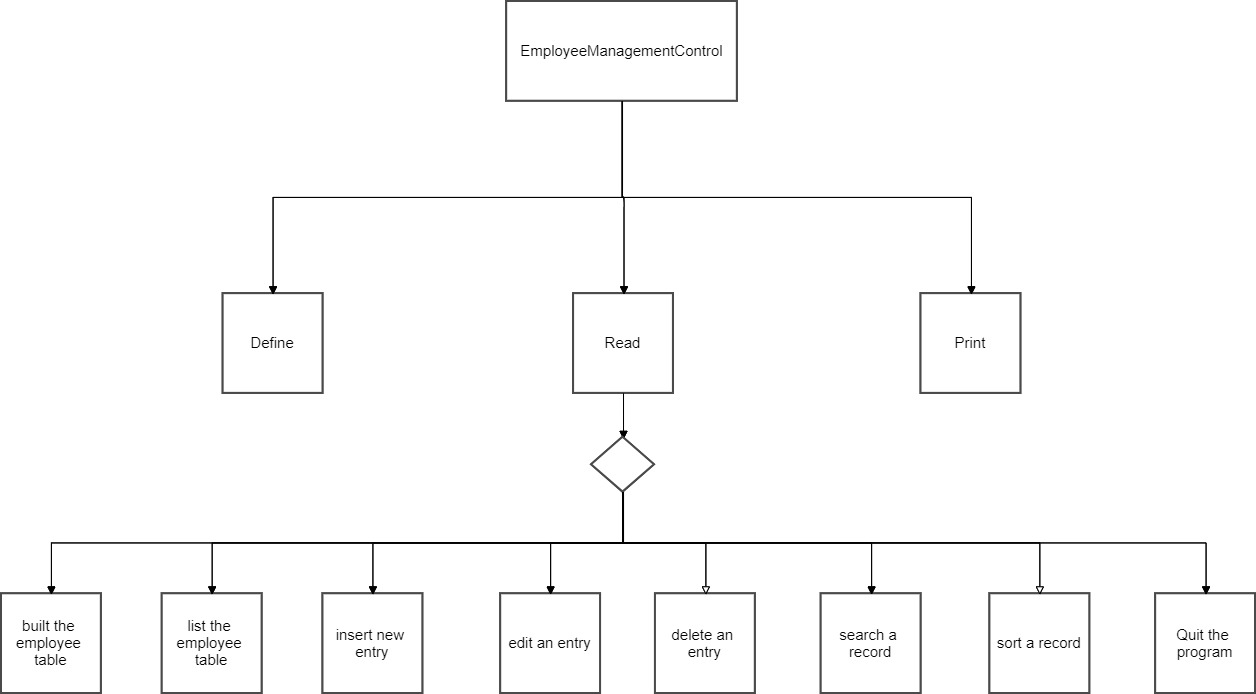
In case 5 call the edit function.

In case 6 call the show function.

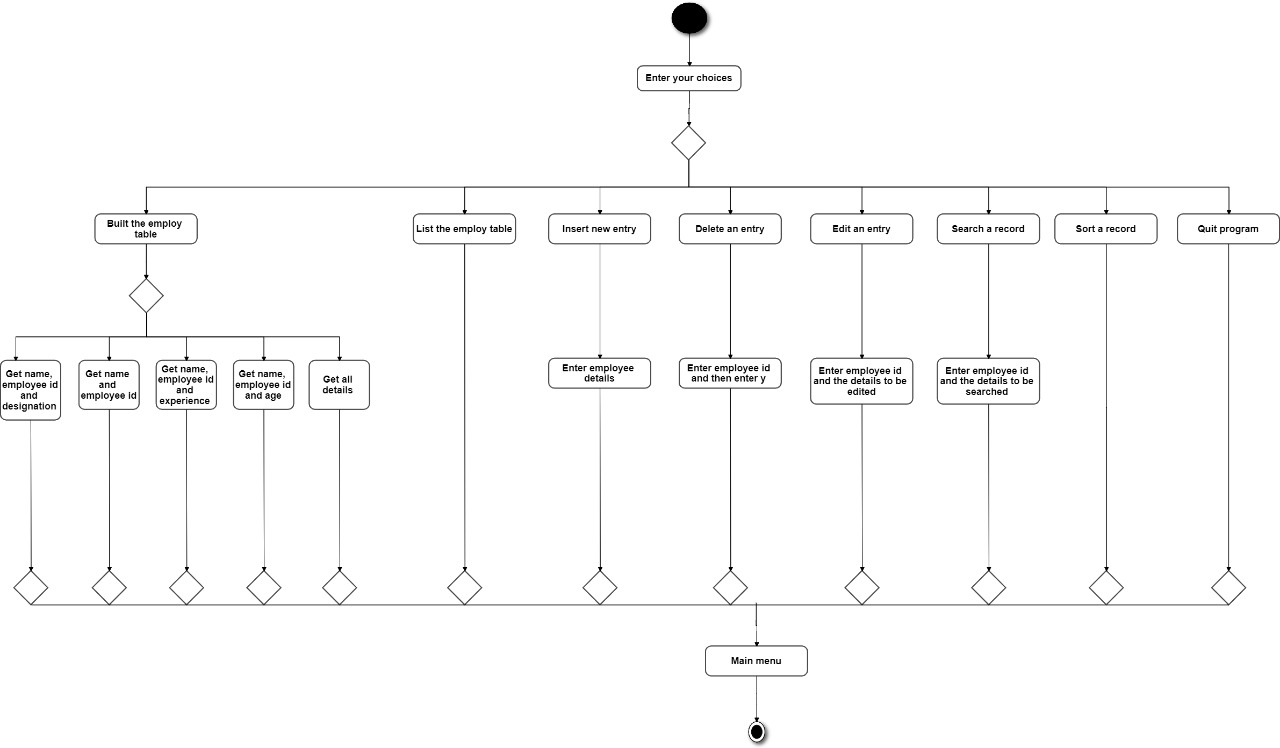
In case 7 call the sort function.

In default case call the exit function.

Stop.

Interactivity Diagram

Activity Diagram

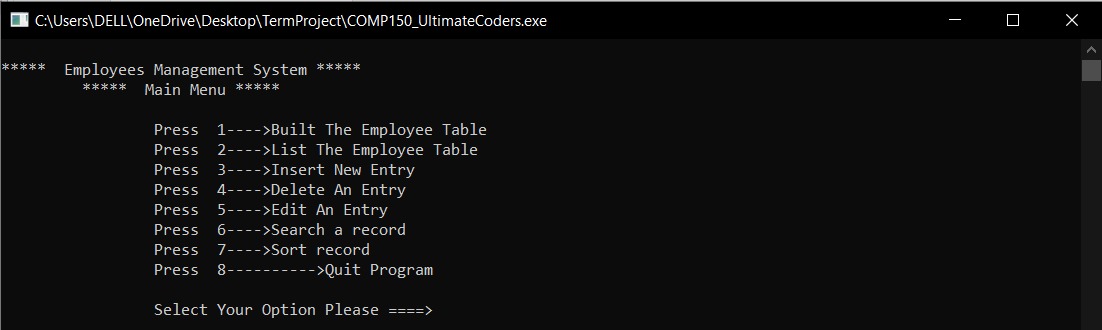


Data Dictionary

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Item** | **Variable Name** | **Data Type** | **Module** | **Scope** | **Pseudonym /Module** | **Error check** |
| Built the employee table | ch | int | Read | Local | NONE | NONE |
| List the employee table | checkId[5]  empname[40]  empcode[5]  designation[20]  age  experience | char  char  char  char  int  int | Read  Print  Print  Print  Print  Print | Local  Global  Global  Global  Global  Global | NONE | NONE |
| Insert New Entry | empname  empcode  designation  age | char  char  char  int | Read  Read  Read  Read | Global  Global  Global  Global | NONE | NONE |
| Delete an entry | checkId[5]  ch | char  char | Read  Read | Local  Local | NONE | NONE |
| Edit an entry | checkId[5]  newdesignation[10]  newname[20]  newexperience | char  char  char  int | Read  Read  Read  Read | Local  Local  Local  Local | NONE | NONE |
| Search a record | checkId[5]  empname  empcode  designation  age  experience | char  char  char  char  int  int | Read  Print  Print  Print  Print  Print | Local  Global  Global  Global  Global  Global | NONE | NONE |
| Sort the table | inflow  num  by\_Line  inflow2  lock  offlow | instream  int  string  instream  string  outstream | Read  Read  Read  Read  Read  Print | Local  Local  Local  Local  Local  Local | NONE | NONE |

Screenshots of Build and Output



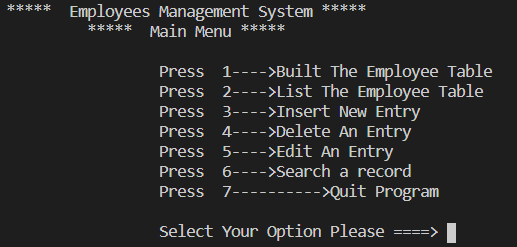


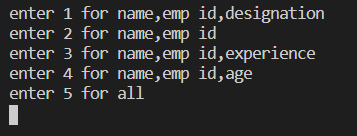
**Testing Data**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Subfield | Testing Data 1 | Testing Data 2 | Testing Data 3 |
| Insert New Entry | Full Name | SteveJobs | BarryLewis | LunaAdams |
| Employee ID | 3301 | 3266 | 3299 |
| Designation | Technician | Chairman | President |
| Employee Age | 45 | 35 | 50 |
| Employee Experience | 15 | 10 | 21 |
| Delete An Entry | Enter Employee ID to Remove | 3301 | 3266 | 3299 |
| Press y(lowercase) to confirm removing | y | y | y |
| Else | Enter anything except y(lowercase) | Enter anything except y(lowercase) | Enter anything except y(lowercase) |
| Edit An Entry | Employee ID | 3301 | 3266 | 3299 |
| Enter new Fullname | SteveLewis | BarryJobs | LunaJames |
| Enter new age | 42 | 50 | 35 |
| Enter new experience | 12 | 19 | 10 |
| Enter new designation | Chairman | President | Technician |
| Search a record | Employee ID | 3301 | 3266 | 3299 |

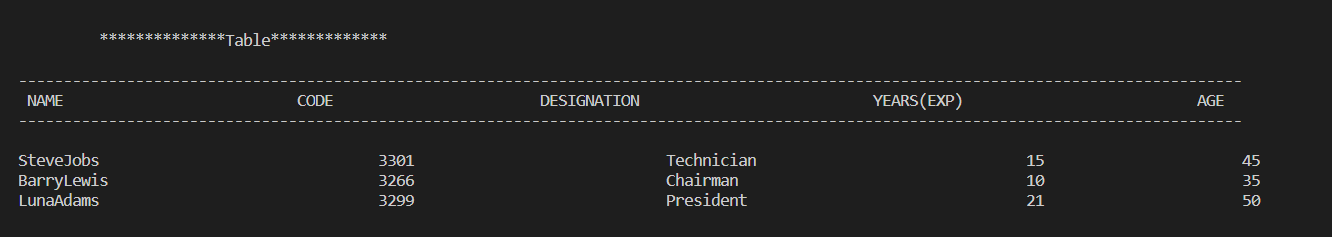
Kindly Follow the instructions given below to use the Employee Management System:

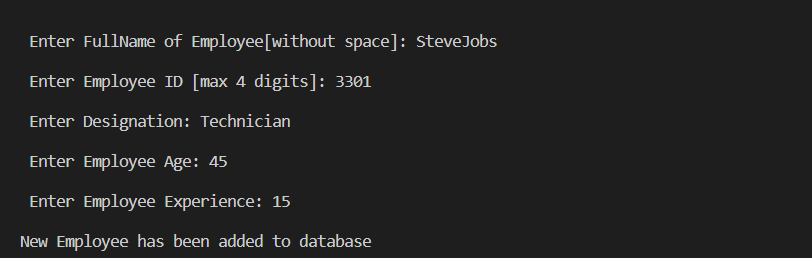
Training Manual

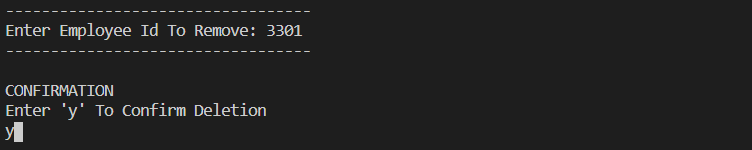
* The system initiates with a main menu providing seven options:
* Now enter your choice:

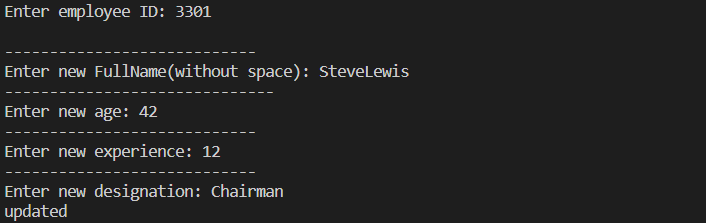
1. If you want to Built the Employee table enter 1. This will further provide you with five different options:

Enter the option according to your interest of how you want to see the table.

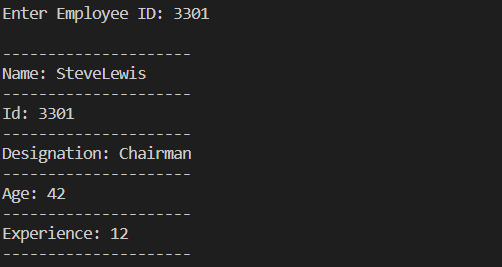
1. If you want to see the list of Employee enter 2.
2. If you want to Insert new entry enter 3. Then Enter the requested data as shown below:



1. If you want to delete an entry enter 4. Now enter the employee ID of the record you want to delete and then enter ‘y’ (lowercase) to confirm deletion. It is shown below:
2. If you want to edit an entry enter 5. Now enter the employee ID for the record you want to delete and then enter the new data of the employee as shown below:



1. If you want to search a record enter 6. Then enter the Employee ID of the employee you want to search the record of (This will show the updated data):



1. If you want to sort the records enter 7. If the terminal shows data sorted successfully then confirm it by checking the list of employees by entering 2 in the main menu.
2. If you want to quit the program enter 8.

Paridhimathur95. (2021). SDLC. Retrieved from https://www.coursehero.com/file/43591533/7-SDLCdocx/.

**References**