

**High Level Design & Low-Level Design**

|  |  |  |
| --- | --- | --- |
| **Index** | |  |
| 1. Introduction | | ------------------------------------------------ 3 |
| 1.1 Intended audience | | ------------------------------------------------ 3 |
| 1.2 Project purpose | | ------------------------------------------------ 3 |
| 1.3 Key project objective | | ------------------------------------------------ 3 |
| 1.4 Project scope and limitation | | ------------------------------------------------ 3 |
| 1.5 Functional overview | | ------------------------------------------------ 3 |
| 1.5.1Header files | | ------------------------------------------------ 3 |
| 1.5.2 Functions | | ------------------------------------------------ 4 |
| 2. Design overview | | ------------------------------------------------ 5 |
| 2.1 Design objective | | ------------------------------------------------ 5 |
| 2.2 Design alternative | | ------------------------------------------------ 5 |
| 2.3 User interface paradigms | | ------------------------------------------------ 5 |
| 2.4 Error detection/ Exceptional Handling | | ------------------------------------------------ 5 |
| 2.5 Performance | | ------------------------------------------------ 6 |
| 2.6 Maintenance | | ------------------------------------------------ 6 |
| 3. System architecture | | ------------------------------------------------ 6 |
| 3.1 ER Diagram  3.2 Flowchart | | ------------------------------------------------ 7  ------------------------------------------------ 8 |
| 4. Environment description | | -------------------------------------------------7 |
| 4.1 Time zone support | | -------------------------------------------------7 |
| 4.2 Language support | | -------------------------------------------------7 |
| 4.3 User desktop requirement | | -------------------------------------------------7 |
| 4.4 Server-side requirement | | -------------------------------------------------7 |
| 4.4.1Deployment consideration | | 7 |
| 4.4.2 Application server disk space | | 7 |
| 4.4.3 Database server disk space | | 7 |
| 4.4.4 Integration requirements | | 7 |
| 4.4.5 Network | | 7 |
| 4.5.1 Operating system | | 7 |
| 5. Configuration | |  |
| 5.1 Operating System  5.2 Database  5.3 Desktop | | -------------------------------------------------  -------------------------------------------------  ------------------------------------------------- |
| 6. Reference |
|  |
|  |
|  |
|  |
|  |
|  |
|  | |  |

**1. Introduction: -**

# 1.1 Intended Audience: -

The audience set for this project would include company and applicants.

# 1.2 Project Purpose: -

The online interview management system is a simple console application without graphics, specifically developed for managing interview process with facilities of adding, deleting, modifying details. This system will replace the paper system into a digital system which would not require any hard copies and physical space for record keeping. The system will serve as the centralized database for all the records in the entire interview process. The system is easy and convenient to use.

# 1.3 Key Project Objectives: -

* Allow admin to view company and applicant details.
* Allow user to create account.
* Allow user to add, delete, modify and display details.

# 1.4 Project scope and limitation: -

Primarily, the scope of the online interview management system features to ensure smooth interview process. The main aim of this application is to automate records on the system. It provides primary functions which are required by the interview process in order to run a stable system. In addition to that it also helps to manually check the records of the pre-existing system. The user can also change or manipulates the new data that is being added. The proposed solution is to design and develop an interview management system. The system will contain an admin that will manage the entire interview process. The limitations of Online Management System is Technology issues and high volume of responses.

# Functional Overview: -

* + 1. **Header Files**

Following header files are included in the program:

* #include <stdio.h>
* #include <string.h>
* #include <stdlib.h>
* #include <unistd.h>
  + 1. **Functions**

1.5.2.1 Admin Module

The functions that are used for admin module is mentioned below:

* admin\_login(): This function is used by admin to login.
* admin\_task(): This function is a main menu of admin.
* display\_company(): In this function, admin can view the details of company.
* applicant\_menu(): In this function, admin can view the details of applicant.

1.5.2.2 Company Module

The functions that are used for company part is mentioned below:

* display\_company\_menu(): This function displays the company menu.
* company\_login(): This function is used by company to login.
* company\_register(): This function is used by company to register.
* addnode(): Company can add the details in this function.
* modifynode(): Company can modify the details in this function.
* deletenode(): Company can delete the details in this function.
* updateNodeFile(): Company can update the details in this function.
* display(): Company can display the details in this function.
* recent\_openings\_menu(): This function displays the recent opening menu.
* add\_openings(): This function is used by company to add recent openings.
* display\_openings(): This function is used by company to display recent openings.
* delete\_openings(): This function is used by company to delete recent openings.
* updateOpFile(): This function is used by company to update recent openings.
* modify\_openings(): This function is used by company to modify recent openings.

1.5.2.3 Applicant Module

The functions that are used for company part is mentioned below:

* applicant\_menu(): This function displays the applicant menu.
* applicant\_login(): This function is used by applicant to login.
* applicant\_register(): This function is used by applicant to register.
* add\_applicant\_node(): Applicant can add the details in this function.
* modify\_applicant\_node(): Applicant can modify the details in this function.
* delete\_applicant\_node(): Applicant can delete the details in this function.
* updateNodeFile(): Applicant can update the details in this function.
* display\_applicant(): Applicant can display the details in this function.
* recent\_openings\_menu(): Applicant can view the recent openings uploaded by the company.

1.5.2.4 DataValidation Module

The functions that are used in data validation module is mentioned below:

* username\_validation(): This function is used to validate username i.e., username should contain at east 1uppercase, 1lowercase and length less than or equal to 10.
* password\_validation(): This function is used to validate password i.e., password should contain at least 1 uppercase, 1 lowercase, 1 digit, 1 special character and length equal to 8.
* email\_validation(): This function is used to validate email id i.e., email id should contain ‘@’ and ‘.’ Symbol and length less than or equal to 30.
* phone\_number\_validation(): This function is used to validate phone number i.e., phone number should contain 10 digits and can have 9 or 8, or 7 or 6 as a first digit.
* company\_id\_validation(): This function is used to validate company id i.e., company id should contain at least 1 uppercase and 1 number and have length equal to 6.
* opening\_id\_validation(): This function is used to validate opening id i.e., opening id should contain 6 digits.
* int date\_validation(): This function is used to validate date.

# Design Overview: -

Online Interview Management System comprises of the following modules:

# Design Objectives: -

* Allow users to create a new account
* Add Details
* Delete Details
* Modify Details
* Display Details

# Design Alternative: -

We have used linked list instead of stack and queue as Insertion and Deletions operations are fast and easier in linked list. Memory allocation is done during run-time. (i.e., no need to allocate any fixed memory).

# User Interface Paradigms: -

The Online Interview Management System gives a user an option to have interview management system stored on a system file. A system always works faster than a person can. User is given an interface to login to the system, an option to add, delete, modify and display the details.

# Error Detection / Exception Handling: -

* If the user doesn’t have any pre-existing account, the user has to create one else it won’t perform any functions and would give “Incorrect Username or Password”.
* While registering to the system, user should provide username and password. We check the validity of the username and password entered. If the length of username entered is greater than 10 or the length of password is greater than 8 then it prints error message “Incorrect Username or Password”.

# Performance: -

The system will work on the user’s terminal. The performance shall depend upon hardware components of the user and the internet connection.

# Maintenance: -

Very little maintenance should be required for this setup. An initial configuration will be the only system required interaction after system is put together. The only other user maintenance would be any changes to settings after setup, and any specified special cases where user settings or history need to be changed. Physical maintenance on the system’s parts may be required and would result in temporary loss of data or Internet. Upgrades of hardware and software should have little effect on this project but may result in downtime.

# SYSTEM ARCHITECTURE: -

* 1. **ER Diagram**

**![Diagram

Description automatically generated]()**

* 1. **Flowchart**

**Diagram

Description automatically generated**

# Environment Description: -

Linux is a Unix-like, open source and community-developed operating system (OS) for computers, servers, mainframes, mobile devices and embedded devices. It is supported on almost every major computer platform, making it one of the most widely supported operating systems. GCC compiler is used to compile and run a C program on the Linux operating system.

* 1. **Time Zone Support**

Supported on any given time (preferred Indian Standard time).

**4.2 Language Support**

C programming is a general-purpose, procedural, imperative computer programming language developed in 1972 by Dennis M. Ritchie at the Bell Telephone Laboratories to develop the UNIX operating system. C is easy to get started with, especially if you're running Linux. You can already run C code because Linux systems include the GNU C library. To write and build it, all you need to do is install a compiler, open a text editor, and start coding.

# User Desktop Requirements: -

Windows system with minimum 4GB of RAM is required.

# 5. Configuration

### 

### **5.1. Operating System**

Linux is an open-source operating system (OS). The OS sits between applications and hardware and makes the connections between all your software and the physical resources that do the work.

# 5.2. Database

File handling is used, which refers to the method of storing data in the C program in the form

of an output or input that might have been generated while running a C program in a data file

## **5.3 Desktop**

## 1GB for 32-bit Windows 7 or later desktops and 2GB for 64-bit Windows 7 or later desktops.

# 6. Reference:

The references are:

* All training classes conducted by trainer.
* All lab-sessions conducted during training.
* https://[www.programiz.com/dsa/linked-list](http://www.programiz.com/dsa/linked-list)
* https://[www.javatpoint.com/file-handling-in-c](http://www.javatpoint.com/file-handling-in-c)