**Schedular Reminder Daemon**

**High Level Design Document**

**History :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Schedular Reminder Daemon** | | | | |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | **Approver** |
| 21/10/22 | 1.0 |  | Initial draft |  |
|  |  |  |  |  |
|  |  |  |  |  |

Introduction

Purpose:

The purpose of this project is to implement and checks for monthly and annual dates and sends/prints them on console every time the user logs in terminal. Once the obligation is met user can dismiss the reminders by calling the program with a command line to dismiss further notifications.

Intended Audience:

<There is no such specific audience, it could be used by Anyone who forgets to pay bills on time and thus suffers with penalty.>

Acronyms/Abbreviations:

|  |  |
| --- | --- |
| CLIENT | USER |
| SERVER | TO ESTABLISH CONNECTION BETWEEN  CLIENTS |
| TCP | TRANSMISSION CONTROL PROTOCOL |

Project Purpose:

The purpose of this project is to implement and checks for monthly and annual dates and sends/prints them on console every time the user logs in terminal. Once the obligation is met user can dismiss the reminders by calling the program with a command line to dismiss further notifications.

Key Project Objectives:

.

1 Add/Remove an obligation to a system configuration file.

2 Modify an obligation to a system configuration file.

3 compare the task reminder with current date

4 Obligation is met can dismiss the reminders

In Scope:

It provides a general architecture for reminder application, and individual can use it as the basis for getting reminder . The application is written in c language and used tools like valgrind splint , gdb. there are multiple options are there so that user can

add/remove/modify reminders and having options like enable or disable options.

Functional Overview

The following functions are included in the program:

a. void modify(int line, char \*sid, char \*des, char \*date, char \*status);

DESCRIPTION : This function modifies the reminder by SID in the file.

Parameters : int line - Line number of schedule in the file.

char \*sid - SID which you want to modify.

char \*des - new description entered by user.

char \*date - new date entered by user.

char \*status - new status entered by user.

b. void remove\_func(int line);

DESCRIPTION : This function removes the reminder by SID in the file.

Parameters : int line - int line - Line number of schedule in the file.

c. void add\_reminder(char \*sid, char \*des, char \*date, char \*status);

DESCRIPTION : This function adds new reminder in the file.

Parameters : char \*sid - SID for new reminder entered by user.

char \*des - Description for the new reminder.

char \*date - Date for the new reminder.

char \*status - Status for the new reminder.

d. int read\_schedule(schedule\_t \*Schedules)

DESCRIPTION : This function reads all reminders from schedule file.

Parameters : schedule\_t \*Schedules - pointer reference for array of structure of structure schedule\_t.

Return Value : This function return the total no. of reminders from the file.

e. void show\_reminders(schedule\_t \*Schedules, int tnum, char

\*user\_date);

DESCRIPTION : This function shows all the reminders from the csv file to the command prompt after matching some constraints

Parameters : schedule\_t \*Schedules - pointer reference for array of structure of structure schedule\_t int tnum will provide the total number of reminders

Design Overview:

Instant Chatters comprises of the following modules:

|  |  |
| --- | --- |
| Name of the Module | Display set date |
| Handled by | sidharth sharma |
| Description | function displays all reminders on console |

|  |  |
| --- | --- |
| Name of the Module | Add reminder |
| Handled by | sahil koul |
| Description | this function will allow user to add new reminder |

|  |  |
| --- | --- |
| Name of the Module | Remove function |
| Handled by | gyandeep kushwaha |
| Description | function will remove the reminder |

|  |  |
| --- | --- |
| Name of the Module | read schedule |
| Handled by | ankit chakrabarti |
| Description | function read schedules from csv file |

|  |  |
| --- | --- |
| Name of the Module | set date |
| Handled by | aditya khajuria |
| Description | function allows to user to get reminder or perticular date |

Design Objectives:

Schedule Reminder service that runs in background and checks for monthly and annual dates and sends/prints them on console every time the user logs in terminal. The dates and obligations are entered manually by the user using a separate program. Once the obligation is met user can dismiss the reminders by calling the program with a command line to dismiss further notifications.

**2.2 Design Alternative:**

We have used structure instead of Linkedlist beacuse linkedlist we can acess only one side and its takes lot of time to traverse from first node to

end node thats consume lots of time to modify the csv file thats where structure make it easy to use.

**2.2.1 User Interface Paradigms:**

It displays multiple options in queue to user by pressing appropriate number user can perform multiple operations like adding / removing the

reminder modifying it and can also search for todays reminders which help user to remind their task .

**2.2.2 Error Detection / Exceptional Handling:**

User should should follow all the constarints like date format and all

user should add data with proper details and configuration before searching for reminder or removing or modifying it .

**2.2.3 Performance:**

The system will work on the client terminal. The performance depends on the hardware component of the user’s system.

**2.2.4 Maintenance:**

Very little maintenance should be required for this setup. An initial configuration

will be the system/virtual 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 csv file need to be changed.

No Physical maintenance on the system’s parts may be required

**3.Environment Description:**

**3.1 Time Zone Support:** IST- Kolkata

**3.2 Language Support:** US English

**3.3.1 Deployment Considerations:**

a. Easy setup: no session storage daemon, use tmpfs and memory caching to enhance performance.

b. Local storage is used.

c. No network latency to consider.

d. To scale buys a bigger CPU, more memory, larger hard drive, or additional hardware.

**3.3.2 Application Server Disk Space:**

a. No such disk space is required as the program is fully functional on online IDE(s)/cloud as well. the google cloud used where we store our file as well as csv file to store records.

**3.3.3 Database Server Disk Space:**

a. No such disk space is required as the program is fully functional on online IDE(s) as well. The Local Operating System is required and one CSV file to store the records of processes.

**3.3.4 Integration Requirements:**

1. Language: C

2. Tools: Splint ,Valgrind, gdb

3. Compiler: gcc

4. Linux Environment

**3.4 Configuration:**

**3.4.1: Operating System**: Linux environment Ubuntu