Ex. No. 2 **IMPLEMENTATION OF SYSTEM CALLS IN UNIX**

Date : 05.08.2024

AIM:

To write a program for implementing process management using the following system callsof UNIX operating system: fork, exec, getpid, exit, wait, close.

ALGORITHM:

1. Start the program.
2. Read the input from the command line.
3. Use fork() system call to create process, getppid() system call used to get the parent process ID and getpid() system call used to get the current process ID
4. execvp() system call used to execute that command given on that command line argument
5. execlp() system call used to execute specified command.
6. Open the directory at specified in command line input.
7. Display the directory contents. 8. Stop the program.

PROGRAM:

```
#include<stdio.h> main(int arc,char*ar[])

{ int pid; char s[100];

pid=fork(); if(pid<0)

printf("error"); else

if(pid>0)

{

wait(NULL);

 printf("\n Parent Process:\n"); printf("\n\tParent Process

 id:%d\t\n",getpid()); execlp("cat","cat",ar[1],(char*)0);

 error("can't executecat %s,",ar[1]);

 } else {

 printf("\

 nChild

 process:

 ");
```

```
printf("\n\tChildprocess parent id:\t %d",getppid()); sprintf(s,"\n\tChild process id
:\t%d",getpid()); wsrme(1,s,strlen(s)); printf(" "); printf(" ");

printf(" "); execvp(ar[2],&ar[2]); error("can't execute %s",ar[2]);

}
}
```

SAMPLE OUTPUT:

```
[root@localhost ~]# ./a.out tst date Child process:
Child process id :
3137 Sat Apr 10 02:45:32 IST 2010
Parent Process:
Parent Process id:3136 sd dsaASD[root@localhost ~]# cat tst sddsaASD
```

RESULT:

Thus the program for process management was written and successfully executed.