CSE 2005 - Operating System

J-component Project

Graphic User Interface for System Calls.

Team Members:

NAME	Reg. No.
1. Hrishikesh Bharadwaj C	16BCE0722
2. Anish Saha	16BCE0277
3. Ayush Rout	16BCE0930

SLOT: A1

Prof. Sendhilkumar K. S.

Keywords: Graphic User Interface, System Calls, Process Management, File System Management, CRUD operations.

Abstract:

In this project we plan to make a user interface on the Windows platform. This interface will basically invoke the different system calls that are an integral part of any OS's functionality. System calls would include hardware-related services such as, accessing a hard disk drive, creation and execution of new processes, and communication with integral kernel services such as process scheduling. This interface would emulate a basic button based UI in order to accomplish tasks that vary from opening a directory, accessing data, running and executing programs (already present in the device), creating processes/threads and many more such options. The advantage of such a UI in comparison to the already present Command line Interface would be its ease of access and better understanding of the various calls that one might use. In addition to the above such a UI may be further developed in order to make it more human like and thus tend towards the creation of a more sophisticated Interface that is an urgent need.

Introduction:

The **graphical user interface (GUI)** is a type of user interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, instead of text-based user interfaces, typed command labels or text navigation. GUIs were introduced in reaction to the perceived steep learning curve of command-line interfaces (CLIs),which require commands to be typed on a computer keyboard.

Large widgets, such as **windows**, usually provide a frame or container for the main presentation content such as a web page, email message or drawing. Smaller ones usually act as a user-input tool., usually provide a frame or container for the main presentation content such as a web page, email message or drawing. Smaller ones usually act as a user-input tool.

The actions in a GUI are usually performed through direct manipulation of the graphical elements. Beyond computers, GUIs are used in many handheld mobile devices such as MP3 players, portable media players, gaming devices, smartphones and smaller household, office and industrial controls. The term GUI tends not to be applied to other lower-display resolution types of interfaces, such as video games (where head-up display (HUD) is preferred), or not including flat screens, like volumetric displays because the term is restricted to the scope of two-dimensional display screens able to describe generic information, in the tradition of the computer science research at the Xerox Palo Alto Research Center.

In computing, a **system call** is the programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on. This may include hardware-related services (for example, accessing a hard disk drive), creation and execution of new processes, and communication with integral kernel services such as process scheduling. System calls provide an essential interface between a process and the operating system.

In most systems, system calls can only be made from userspace processes, while in some systems, OS/360 and successors for example, privileged system code also issues system calls.

The system call would emulate a basic button based UI in order to accomplish tasks that vary from opening a directory, accessing data, running and executing programs (already present in the device), creating processes/threads and many more such options.

The system calls that we have tried to emulate in this project include:

- Process Management:
    1. Creating a new Process
    2. Terminating a new Process
    3. Getting Details of the Process.
- File Management:
    1. Creating a File
    2. Reading a file.
    3. Updating a file
    4. Displaying The list of Files in the Given Directory.
- Information Management:
    1. Displaying Time Continuously
    2. Getting System Hardware Information

Literature review:

1. SYSTEM CALLS:

A *system call*, sometimes referred to as a *kernel call*, is a request in a Unix-like operating system made via a *software interrupt* by an *active process* for a *service* performed by the *kernel*. System calls can also be viewed as clearly-defined, direct entry points into the kernel through which programs request services from the kernel. They allow programs to perform tasks that would not normally be permitted.

Examples of the services performed by the kernel include as input/output (I/O) and *process creation*. The former can be defined as any movement of data to or from the combination of the CPU and *main memory* (i.e. RAM), that is, communication between this combination and the computer's users (e.g., via the keyboard or mouse), its storage devices (e.g., disk or tape drives) or other computers. Process creation is the creation of a new process.

2. PROCESS MANAGEMENT:

Process management is an integral part of any modern-day operating system (OS). The OS must allocate resources to processes, enable processes to share and exchange information, protect the resources of each process from other processes and enable synchronization among processes. To meet these requirements, the OS must maintain a data structure for each process, which describes the state and resource ownership of that process, and which enables the OS to exert control over each process.

a. **Process Creation:**

Operating systems need some ways to create processes. In a very simple system designed for running only a single application (e.g., the controller in a microwave oven), it may be possible to have all the processes that will ever be needed be present when the system comes up. In general-purpose systems, however, some way is needed to create and terminate processes as needed during operation.

There are four principal events that cause a process to be created:

- System initialization.
- Execution of process creation system call by a running process.
- A user request to create a new process.
- Initiation of a batch job.

3. FILE MANAGEMENT:

File management systems are useful in organising and managing the files present in a particular directory. It can be defined as “The system that an operating system or program uses to organize and keep track of files”. For example, a *hierarchical file system* is one that uses directories to organize files into a tree structure.

Following are some of the attributes of a file:

- **Name** It is the only information which is in human-readable form.
- **Identifier**. The file is identified by a unique tag (number) within file system.
- **Type.** It is needed for systems that support different types of files.
- **Location.** Pointer to file location on device.
- **Size.** The current size of the file.

4. GRAPHIC USER INTERFACE:

Graphical User Interface, a GUI (pronounced as either G-U-I or gooey) allows the use of icons or other visual indicators to interact with electronic devices, rather than using only text via the command line. For example, all versions of Microsoft Windows utilize a GUI, whereas MS-DOS does not. The GUI was first developed at Xerox PARC by Alan Kay, Douglas Engelbart, and a group of other researchers in 1981. Later, Apple introduced the Lisa computer, the first commercially available computer, on January 19, 1983. Below is a picture of the Windows 7 Desktopand an example of a GUI.

How does a GUI work?

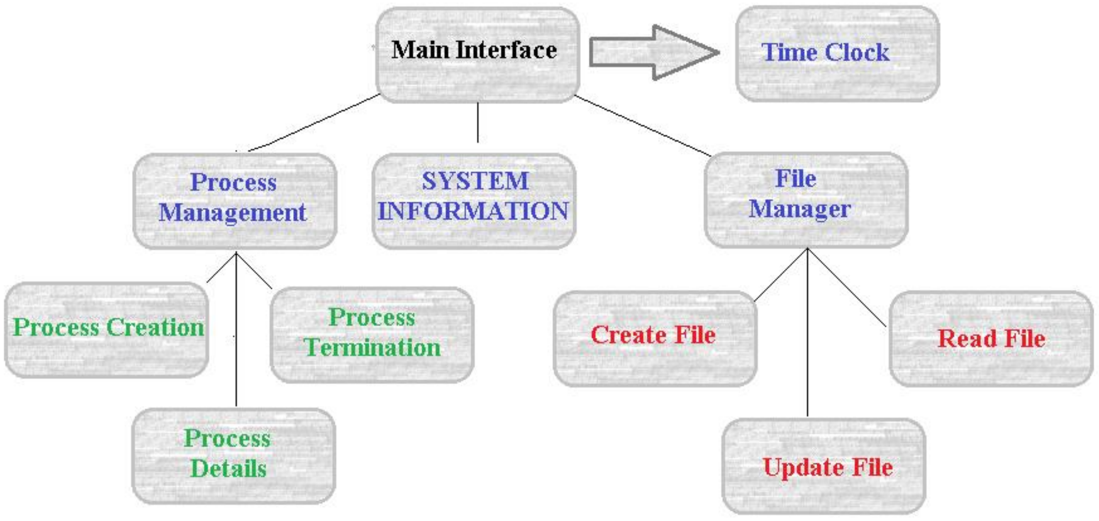
A GUI uses windows, icons, and menus to carry out commands, such as opening, deleting, and moving files. Although many GUI operating systems are navigated through the use of a mouse, the keyboard can also be utilized by using keyboard shortcuts or arrow keys.

What are the benefits of GUI?

Unlike a command line operating system or CUI, like Unix or MS-DOS, GUI operating systems are much easier to learn and use because commands do not need to be memorized. Additionally, users do not need to know any programming languages. Because of their ease of use, GUI operating systems have become the dominant operating system used by today's end-users.

Design of Project:

Flow of Project Controls:



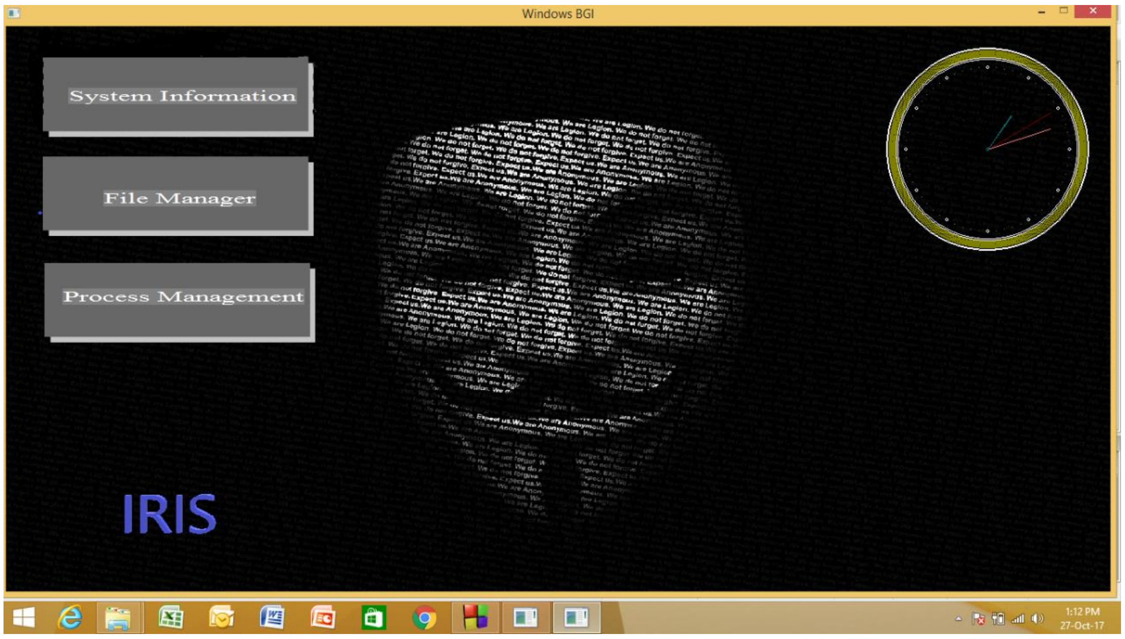
Modules in our Project:

1. **Main UI:**
   This Particular Module acts as the initial page and contains all the linking facilities to the other modules. It basically behaves in a similar way to how the desktop screen works. It contains Buttons to open up the new windows for System Information, File Manager and process Management. On the other Side it has a continuously running clock that Takes Care of the Time Management Calls.
2. **Process Management:**
   This module will contain the system calls for process creation and termination and a window to display the Process Information and attributes. In windows Process Management can be carried out by functions in the header file Windows.h and w32api.h. The module will have options to run executables from windows under the main process and as a result pause the main process. In addition a special feature is added that displays all the currently running process in the system.
3. **System information:**
   System Information Includes the Hardware Information about the no of processors Type of Processors, the page sizes, allocation limits for the current user and other relevant details. All these details are stored under the structure SYSTEM_INFO and can be updated using the function getsysteminfo();
4. **Analog Clock (Time Management):**
   This Module uses functions from the header file time.h to open and get the current system time and has a GUI for displaying it in the form of a continuous analog clock. The time function is a system call that gets the current time in terms of seconds from the first second of 1970.
5. **File Manager:**
   The System calls in c for File management include the following:

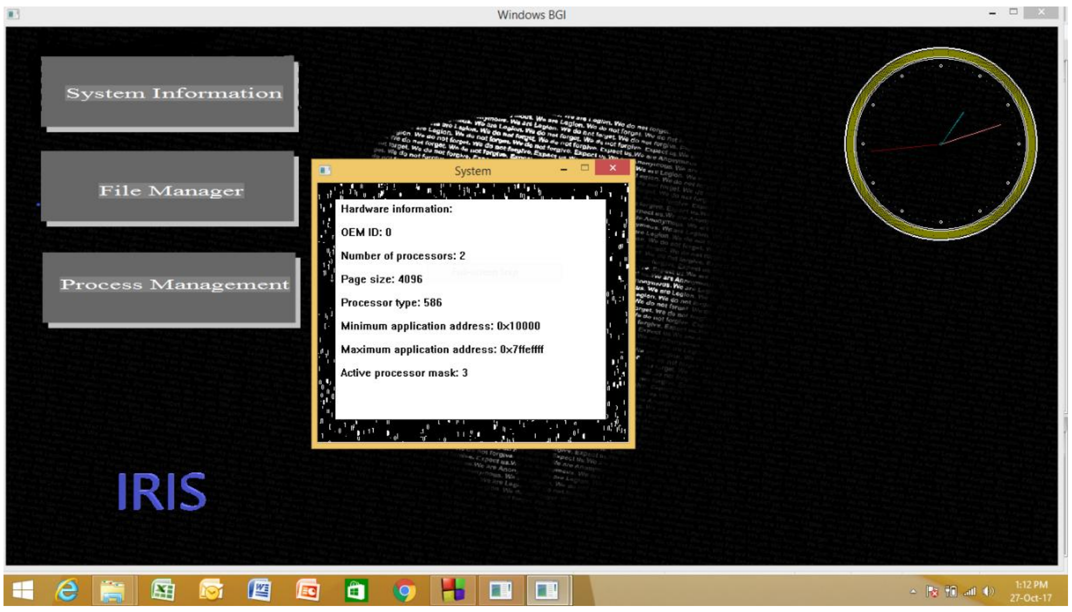
open: system call to open a fileopen returns a file descriptor, an integer specifiying the position of this open file in the table of open files.
creat: (no 'e') system call to create a file
close: system call to close a file
read: read data from a file opened for reading
write: write data to a file opened for writing

Screenshots of Project:

MAIN WINDOW:

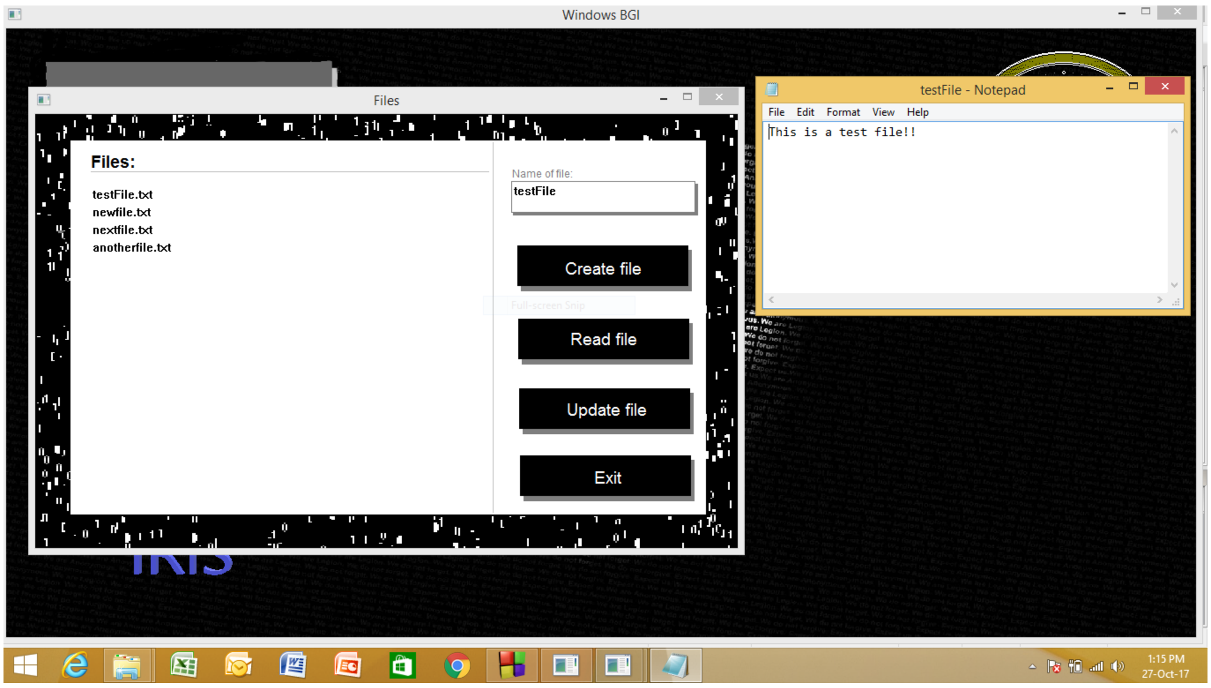


SYSTEM INFORMATION:



```
Hardware information:
OEM ID: 0
Number of processors: 2
Page size: 4096
Processor type: 586
Minimum application address: 0x10000
Maximum application address: 0x7ffeffff
Active processor mask: 3
```

Read Files:

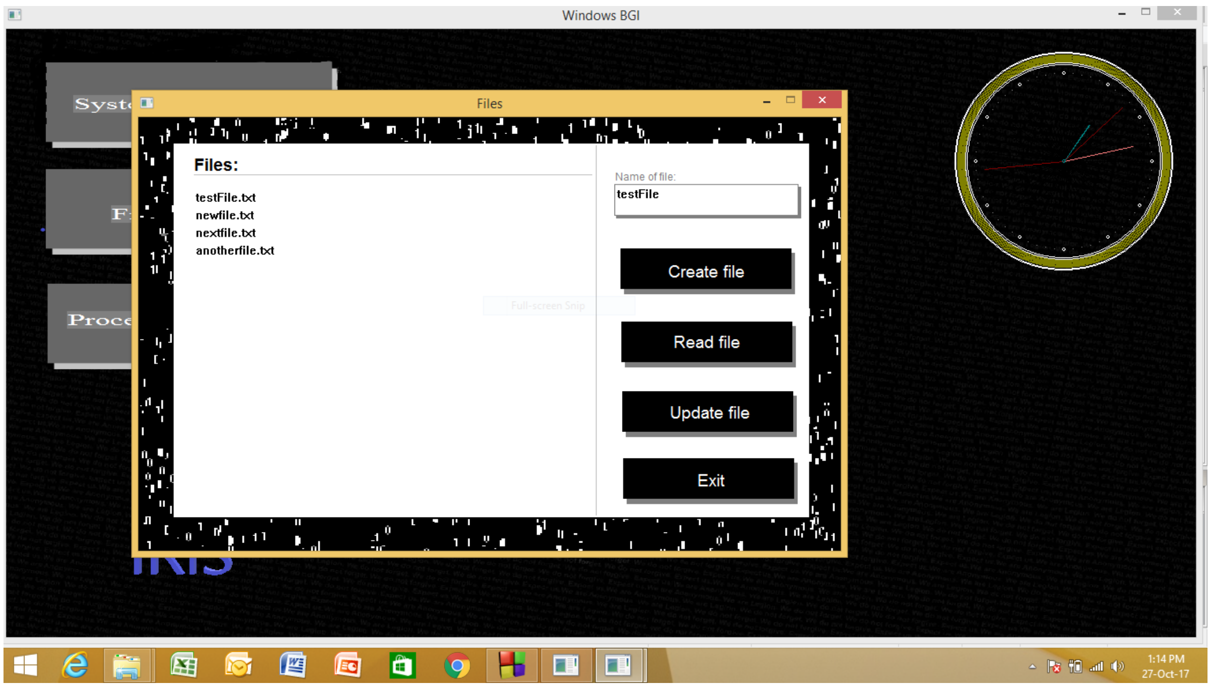


testFile.txt
newfile.txt
nextfile.txt
anotherfile.txt

testFile

```
This is a test file!!
```

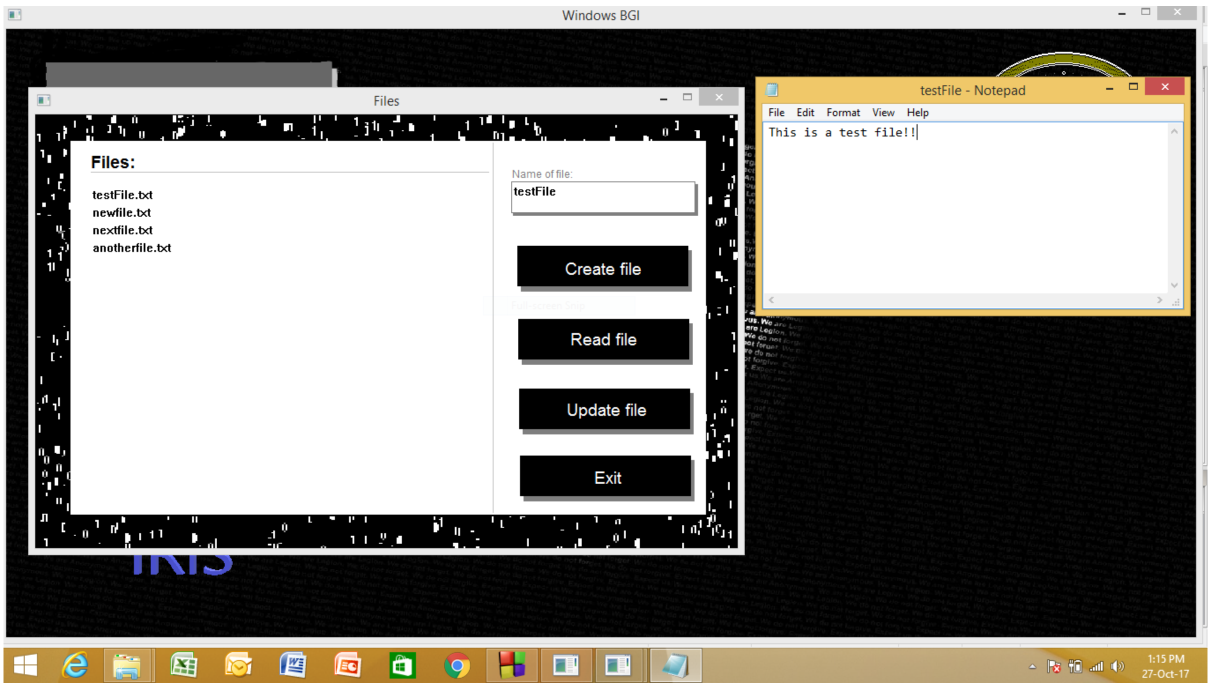
Create Files:



testFile.txt
newfile.txt
nextfile.txt
anotherfile.txt

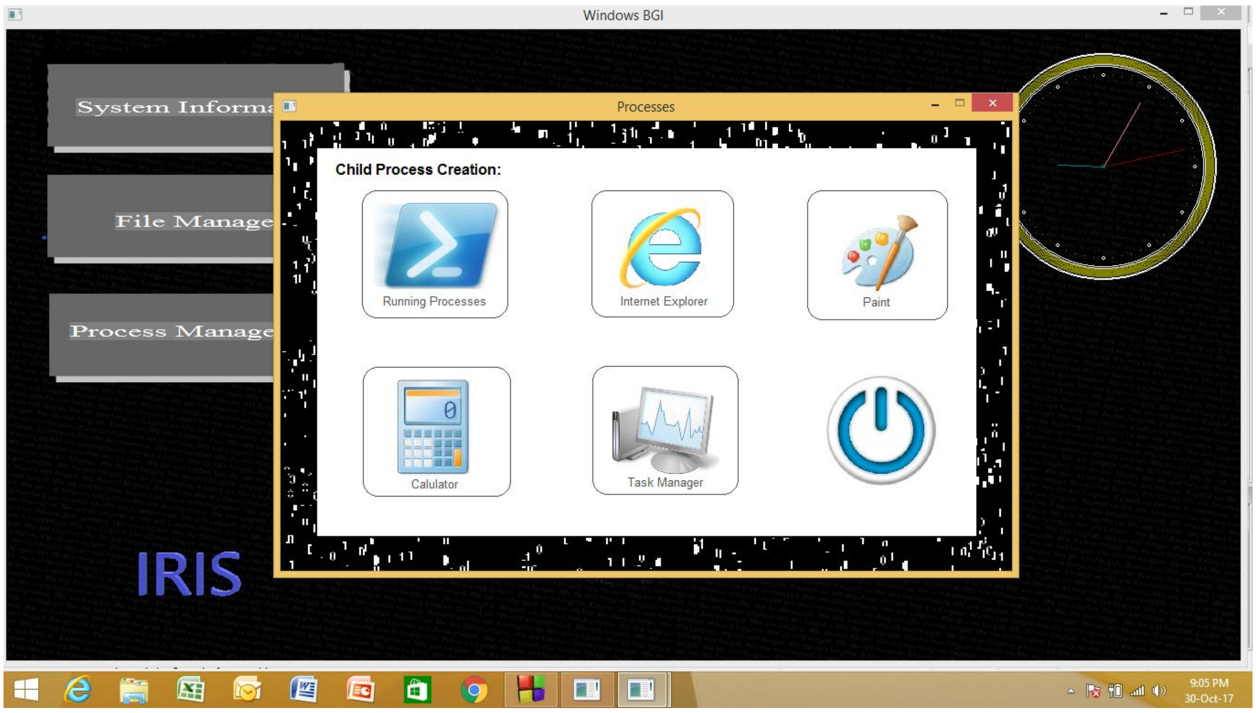
testFile

Update Files:

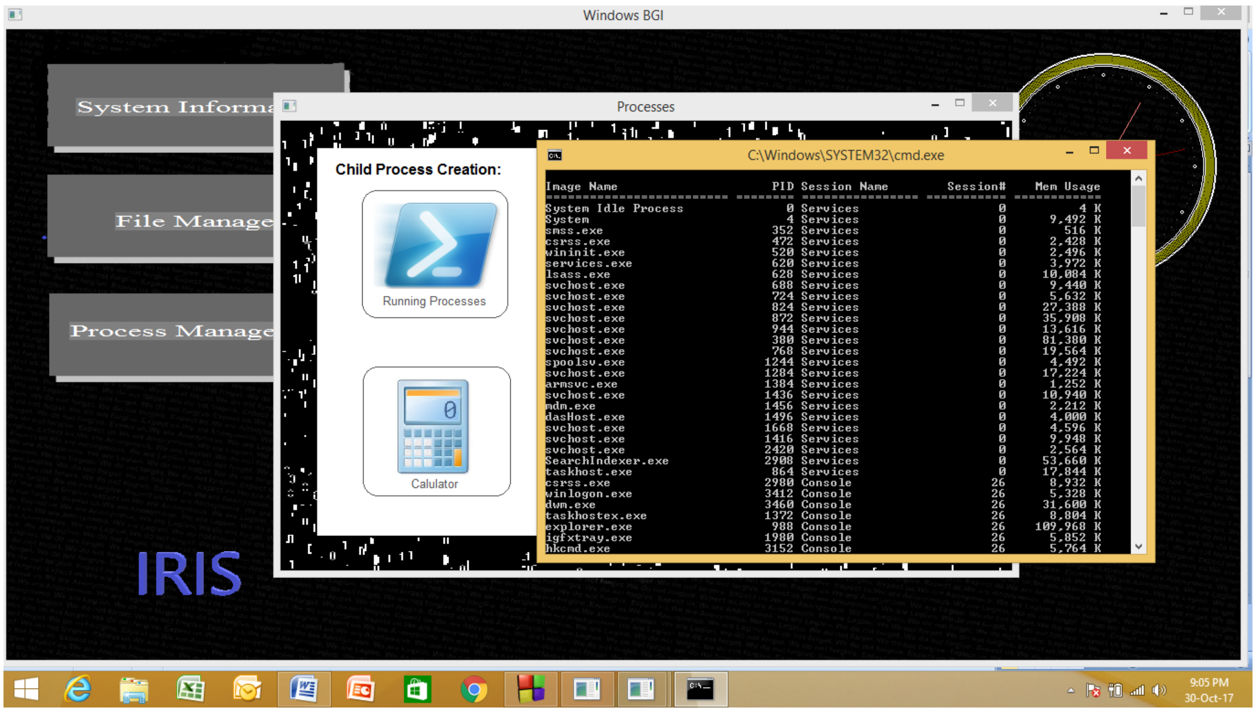


```
This is a test file!!
```

Process Management

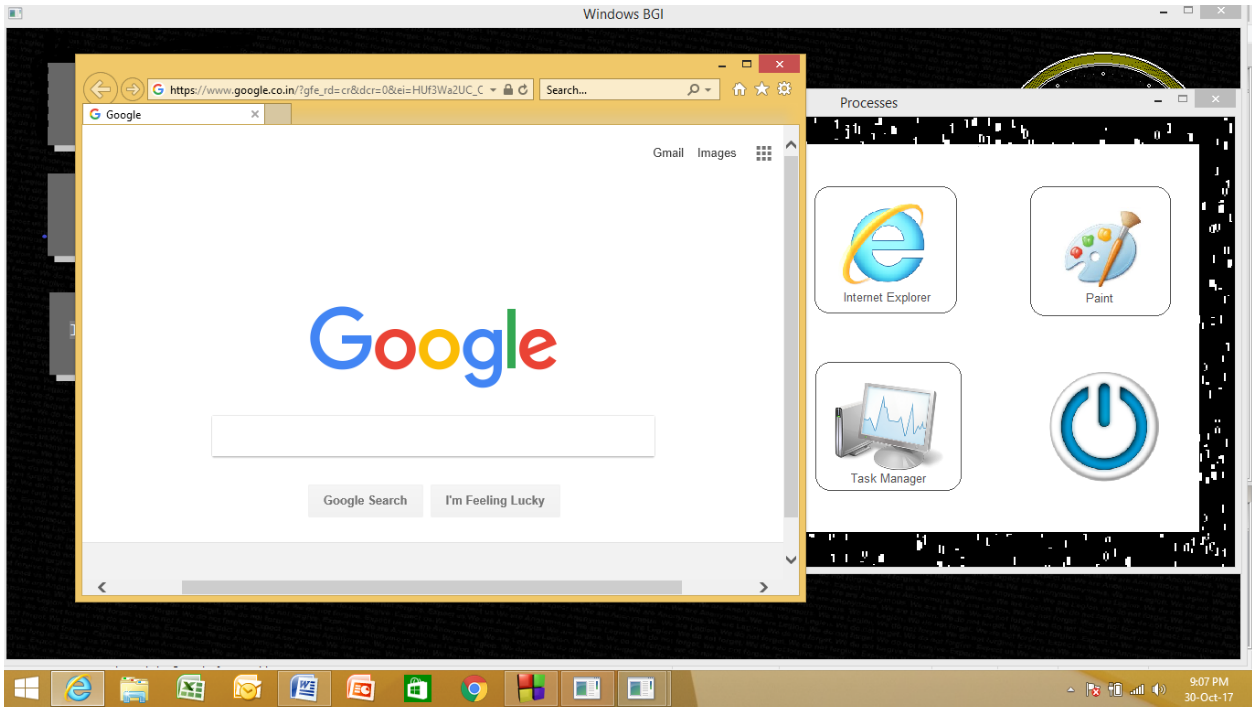


Running processes:

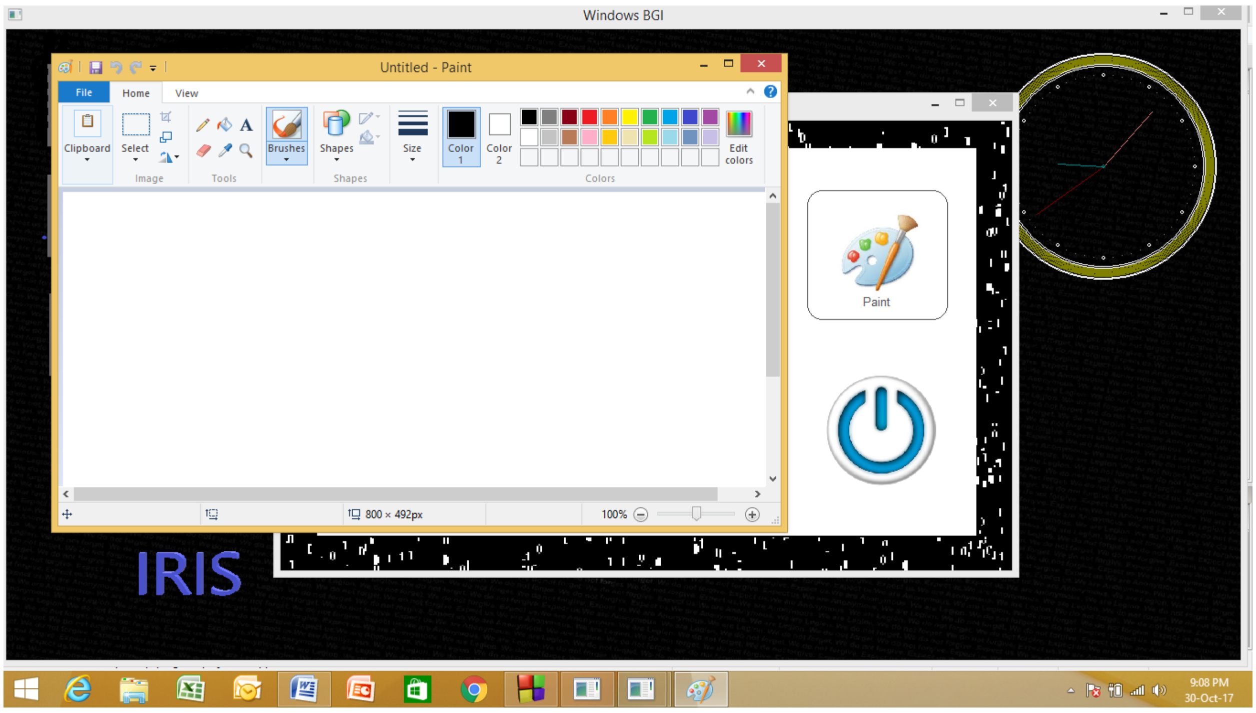


```
Image Name                     PID Session Name        Session#    Mem Usage
========================= ======== ================ =========== ============
System Idle Process              0 Services                   0          4 K
System                           4 Services                   0      9,492 K
smss.exe                       352 Services                   0        516 K
csrss.exe                      472 Services                   0      2,428 K
wininit.exe                    520 Services                   0      2,496 K
services.exe                   620 Services                   0      3,972 K
lsass.exe                      628 Services                   0     10,084 K
svchost.exe                    688 Services                   0      9,440 K
svchost.exe                    724 Services                   0      5,632 K
svchost.exe                    824 Services                   0     27,388 K
svchost.exe                    872 Services                   0     35,908 K
svchost.exe                    944 Services                   0     13,616 K
svchost.exe                    380 Services                   0     81,380 K
svchost.exe                    768 Services                   0     19,564 K
spoolsv.exe                   1244 Services                   0      4,492 K
svchost.exe                   1284 Services                   0     17,224 K
armsvc.exe                    1384 Services                   0      1,252 K
svchost.exe                   1436 Services                   0     10,940 K
mdm.exe                       1456 Services                   0      2,212 K
dasHost.exe                   1496 Services                   0      4,000 K
svchost.exe                   1668 Services                   0      4,596 K
svchost.exe                   1416 Services                   0      9,948 K
svchost.exe                   2420 Services                   0      2,564 K
SearchIndexer.exe             2908 Services                   0     53,660 K
taskhost.exe                   864 Services                   0     17,844 K
csrss.exe                     2980 Console                   26      8,932 K
winlogon.exe                  3412 Console                   26      5,328 K
dwm.exe                       3460 Console                   26     31,600 K
taskhostex.exe                1372 Console                   26      8,804 K
explorer.exe                   988 Console                   26    109,968 K
igfxtray.exe                  1980 Console                   26      5,852 K
hkcmd.exe                     3152 Console                   26      5,764 K
```

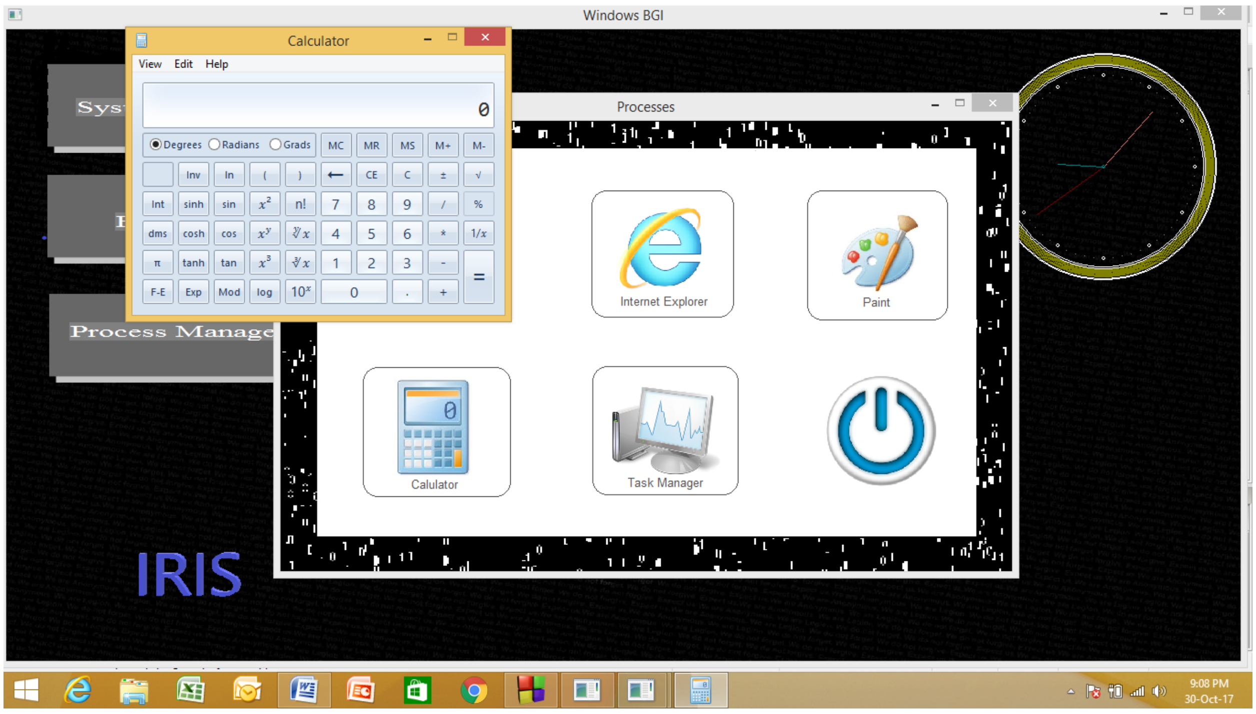
Internet Ex:



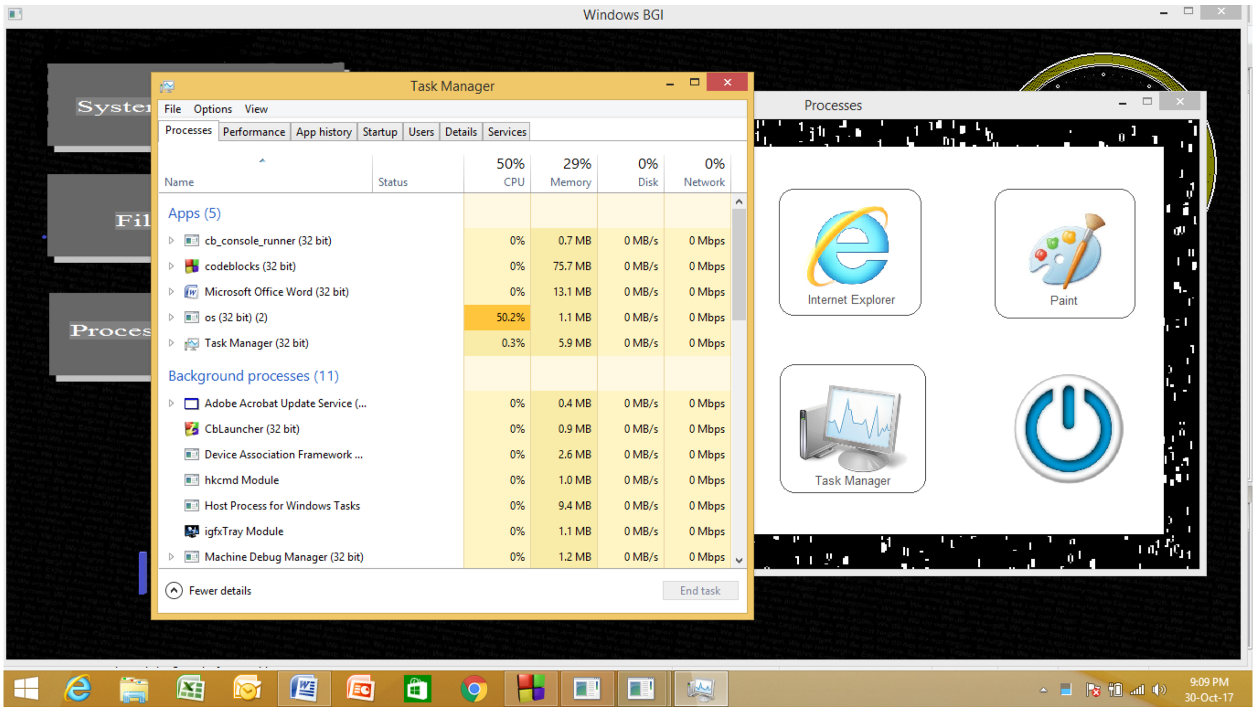
Paint:



Calculator:



Task Manager:



Task Manager

File Options View

Processes | Performance | App history | Startup | Users | Details | Services

Name	Status	50% CPU	29% Memory	0% Disk	0% Network
Apps (5)					
cb_console_runner (32 bit)		0%	0.7 MB	0 MB/s	0 Mbps
codeblocks (32 bit)		0%	75.7 MB	0 MB/s	0 Mbps
Microsoft Office Word (32 bit)		0%	13.1 MB	0 MB/s	0 Mbps
os (32 bit) (2)		50.2%	1.1 MB	0 MB/s	0 Mbps
Task Manager (32 bit)		0.3%	5.9 MB	0 MB/s	0 Mbps
Background processes (11)					
Adobe Acrobat Update Service (...		0%	0.4 MB	0 MB/s	0 Mbps
CbLauncher (32 bit)		0%	0.9 MB	0 MB/s	0 Mbps
Device Association Framework ...		0%	2.6 MB	0 MB/s	0 Mbps
hkcmd Module		0%	1.0 MB	0 MB/s	0 Mbps
Host Process for Windows Tasks		0%	9.4 MB	0 MB/s	0 Mbps
igfxTray Module		0%	1.1 MB	0 MB/s	0 Mbps
Machine Debug Manager (32 bit)		0%	1.2 MB	0 MB/s	0 Mbps

Fewer details   End task

Conclusion:

Thus a graphic user interface for system calls was implemented in windows by using the C++ language. This GUI touched upon most of the basic system calls to manage process, files and also retrieve system information such as time and hardware details.

Such a graphic system is convenient to use and helps in visualisation of the method of processes are generated and managed. Such a system is a basic prototype and further work to develop this and add additional features can also be undertaken as a future goal for this project.

References:

1. https://www.webopedia.com/TERM/F/file_management_system.html
2. http://www.geeksforgeeks.org/input-output-system-calls-c-create-open-close-read-write/
3. http://www.linfo.org/system_call.html
4. https://www.cs.colorado.edu/~main/bgi/doc/
5. https://en.wikipedia.org/wiki/Process_management_(computing)#Process_creation
6. https://www.computerhope.com/jargon/g/gui.htm