Proyecto 2

Anggelo Santiago Son Mux
201709502

MANUAL TÉCNICO

El proyecto presentado realiza el monitoreo del disco duro, CPY y memoria RAM que se encuentra en uso o libre del sistema. El programa fue escrito en Golang y utilizando wails para la visualización de las gráficas.
La solución para el monitoreo de los dispositivos USB se realizó con el lenguaje de programación Golang, esos programas habilitan o deshabilitan el uso de los dispositivos USB, también conservan un log de las acciones realizadas como lo es copiar archivos desde la USB al sistema o del sistema al USB.

Especificaciones técnicas utilizadas para la realización del proyecto

- SO: (Virtualización VirtualBox) Ubuntu 20.04.5 LTS
- Memory: 4.6 GB
- Disk Capacity: 23.3 GB

Aplicaciones y Librerías utilizadas

- Visual Code: editor de código fuente.
- Golang: lenguaje de programación concurrente y compilado con tipado estático inspirado en la sintaxis de C, pero con seguridad de memoria y recolección de basura.
- Wails: permite escribir aplicaciones de escritorio utilizando Go y tecnologías web.
- Node: es un entorno de ejecución para JavaScript construido con V8, motor de JavaScritp de Chrome.
- ApexCharts: es una biblioteca de gráficos moderna que ayuda a los desarrolladores a crear páginas web interactivas.

Inicialización del proyecto utilizando Wails

Para que el comando de Wails sea reconocido en la terminal se deben realizar los siguientes export:

```
export GO111MODULE=on
export PATH=$PATH:~/go/bin
```

La instalación de Wails se realizó con el siguiente comando:

```
go install github.com/wailsapp/wails/v2/cmd/wails@latest
```

El Proyecto se realizó utilizando una plantilla de Vue, pueden utilizarse diferentes platillas las cuales se pueden encontrar en la página oficial de Wails Para poder crear un proyecto utilizando wails se debe ejecutar el siguiente comando;

```
wails init -n myproject -t vue
```

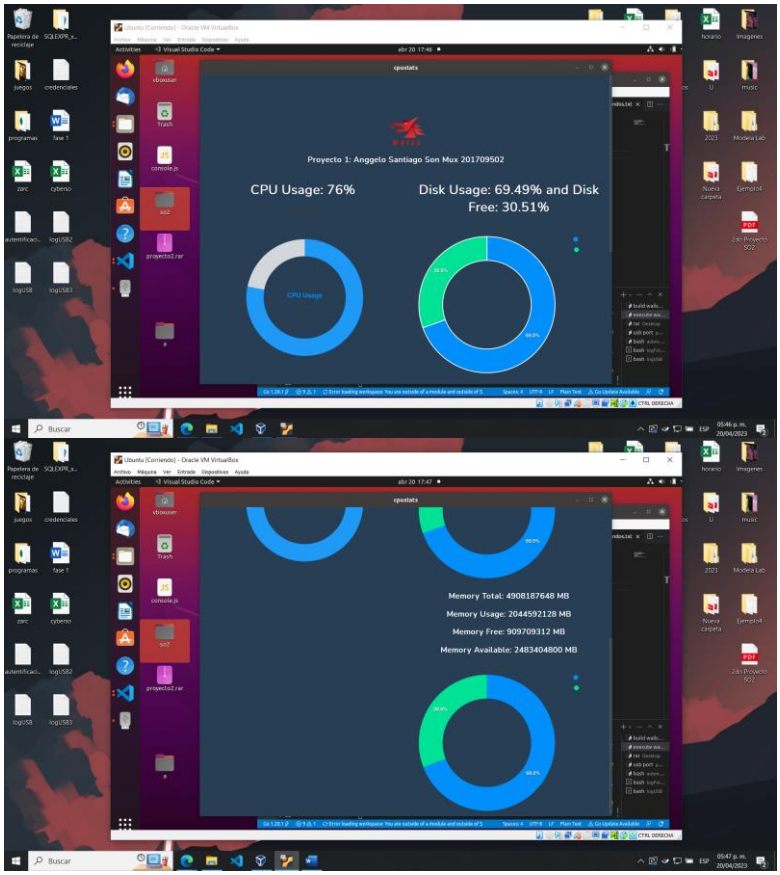
La construcción de un proyecto utilizando Wails se realizó con el siguiente comando:

wails build -debug

La ejecución del proyecto se realiza con el comando **./[ProjectName]** dentro de la ubicación del ejecutable creado con el comando anterior **./201709502**

Aplicación usando Wails

La aplicación fue escrita utilizando el lenguaje de programación Golang, muestra un entorno gráfico similar al administrador de tareas de Windows donde el usuario puede ver la cantidad de CPU que se está utilizando, la cantidad de disco duro, utilizado y libre, del sistema y la memoria RAM que se está utilizando y libre.



Aplicación puertos USB

La aplicación fue escrita utilizando el lenguaje de programación Golang, al ejecutara la aplicación muestra un menú en el cual se pueden escoger las opciones;

- Bloquear los puertos USB
- Habilitar los puertos USB

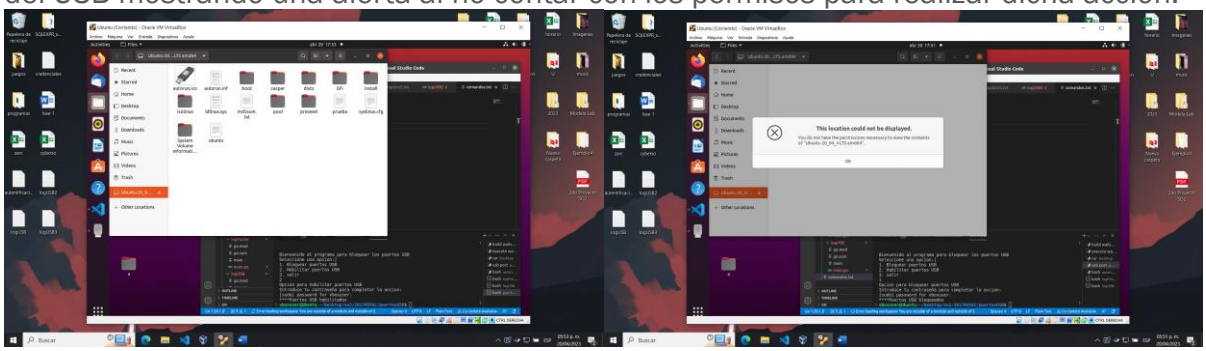
Al momento de seleccionar la opción deseada el usuario deberá ingresar su contraseña del sistema para poder completar la acción.

```
Bienvenido al programa para bloquear los puertos USB
Seleccione una opcion:]
1. Bloquear puertos USB
2. Habilitar puertos USB
3. salir
1
Opcion para bloquear puertos USB
Introduce tu contraseña para completar la accion:
[sudo] password for vboxuser:
****Puertos USB bloqueados
```

```
Bienvenido al programa para bloquear los puertos USB
Seleccione una opcion:]
1. Bloquear puertos USB
2. Habilitar puertos USB
3. salir
2
Opcion para habilitar puertos USB
Introduce tu contraseña para completar la accion:
[sudo] password for vboxuser:
****Puertos USB habilitados
```

La funcionalidad para el bloqueo de los dispositivos USB se realizo modificando los permisos del directorio media con **chmod 000 /media/**, el cual es el encargado de la administración de los dispositivos USB, y para la habilitación de los puertos USB se utilizó **chmod 777 /media/**

Cuando los puertos USB se encuentran habilitados se puede navegar en los archivos que contiene, si los puertos USB se encuentran deshabilitados impide el acceso al contenido del USB mostrando una alerta al no contar con los permisos para realizar dicha acción.



Aplicación log USB

La aplicación fue escrita utilizando el lenguaje de programación Golang, se cuenta con una aplicación la cual monitorea los cambios realizados en el contenido de la USB y otra aplicación la cual monitoria los cambios realizados en el contenido de un directorio especificado, al momento de que se produzca un cambio en alguno de los dos directorios los programas escribirán el cambio realizado en un archivo para llevar un registro.

```
vboxuser@Ubuntu:~/Desktop/so2/logUSB/logFolder$ go build main.go
vboxuser@Ubuntu:~/Desktop/so2/logUSB/logFolder$ ./main
archivo ".goutputstream-LTVN31" copiado en la ruta /home/vboxuser/Desktop/a
archivo "prueba" copiado en la ruta /home/vboxuser/Desktop/a
^C
vboxuser@Ubuntu:~/Desktop/so2/logUSB/logFolder$ ./main
archivo ".goutputstream-FU6Y31" copiado en la ruta /home/vboxuser/Desktop/a
archivo "prueba" copiado en la ruta /home/vboxuser/Desktop/a
^C
vboxuser@Ubuntu:~/Desktop/so2/logUSB/logFolder$

vboxuser@Ubuntu:~/Desktop/so2/logUSB/logUSB$ go build main.go
vboxuser@Ubuntu:~/Desktop/so2/logUSB/logUSB$ ./main
archivo ".goutputstream-NQGP31" copiado en la ruta /media/vboxuser/Ubuntu 20_04
_4 LTS amd64
archivo "ubuntu" copiado en la ruta /media/vboxuser/Ubuntu 20_04_4 LTS amd64
^C
vboxuser@Ubuntu:~/Desktop/so2/logUSB/logUSB$
```

so2 > logUSB > log > registro2.txt

```
1  archivo ".goutputstream-FU6Y31" copiado en la ruta /home/vboxuser/Desktop/a
2  archivo "prueba" copiado en la ruta /home/vboxuser/Desktop/a
3  archivo ".goutputstream-NQGP31" copiado en la ruta /media/vboxuser/Ubuntu 20_04_4 LTS amd64
4  archivo "ubuntu" copiado en la ruta /media/vboxuser/Ubuntu 20_04_4 LTS amd64
5
```

CODIGO

Para la obtención de las estadísticas de la memoria RAM se utilizó la librería mem que proporciona “gopsutil”, esta función proporciona el total, usado, libre, disponible de la memoria RAM en el sistema.

```
func (s *Stats) GetMemoryUsage() *MEMORYUsage {
    UsageStats, err := mem.VirtualMemory()
    if err != nil {
        s.log.Errorf("unable to get memory stats: %s", err.Error())
        return nil
    }

    return &MEMORYUsage{
        Total:     int(UsageStats.Total),
        Used:      int(UsageStats.Used),
        Free:      int(UsageStats.Free),
        Available: int(UsageStats.Available),
    }
}
```

En la visualización de la grafica se utilizó apexchart el cual proporciona la función para implementar gráficas, en la grafica de la memoria ram se muestra la memoria libre y usada del sistema

```
<div class="MEMORYUsage">
  <h3>Memory Total: {{MEMORYTotalMessage}} MB</h3>
  <h3>Memory Usage: {{MEMORYUsageMessage}} MB</h3>
  <h3>Memory Free: {{MEMORYFreeMessage}} MB</h3>
  <h3>Memory Available: {{MEMORYAvailableMessage}} MB</h3>
  <apexchart type="donut" width="480" :options="memoryoptions" :series="
</div>
/template>
```

Para el bloqueo de los puertos USB se creó un menú en consola utilizando el lenguaje de programación golang.

```
fmt.Println("\n\n\nBienvenido al programa para bloquear los puertos US
fmt.Println("Seleccione una opcion:]")
fmt.Println("1. Bloquear puertos USB")
fmt.Println("2. Habilitar puertos USB")
fmt.Println("3. salir")
```

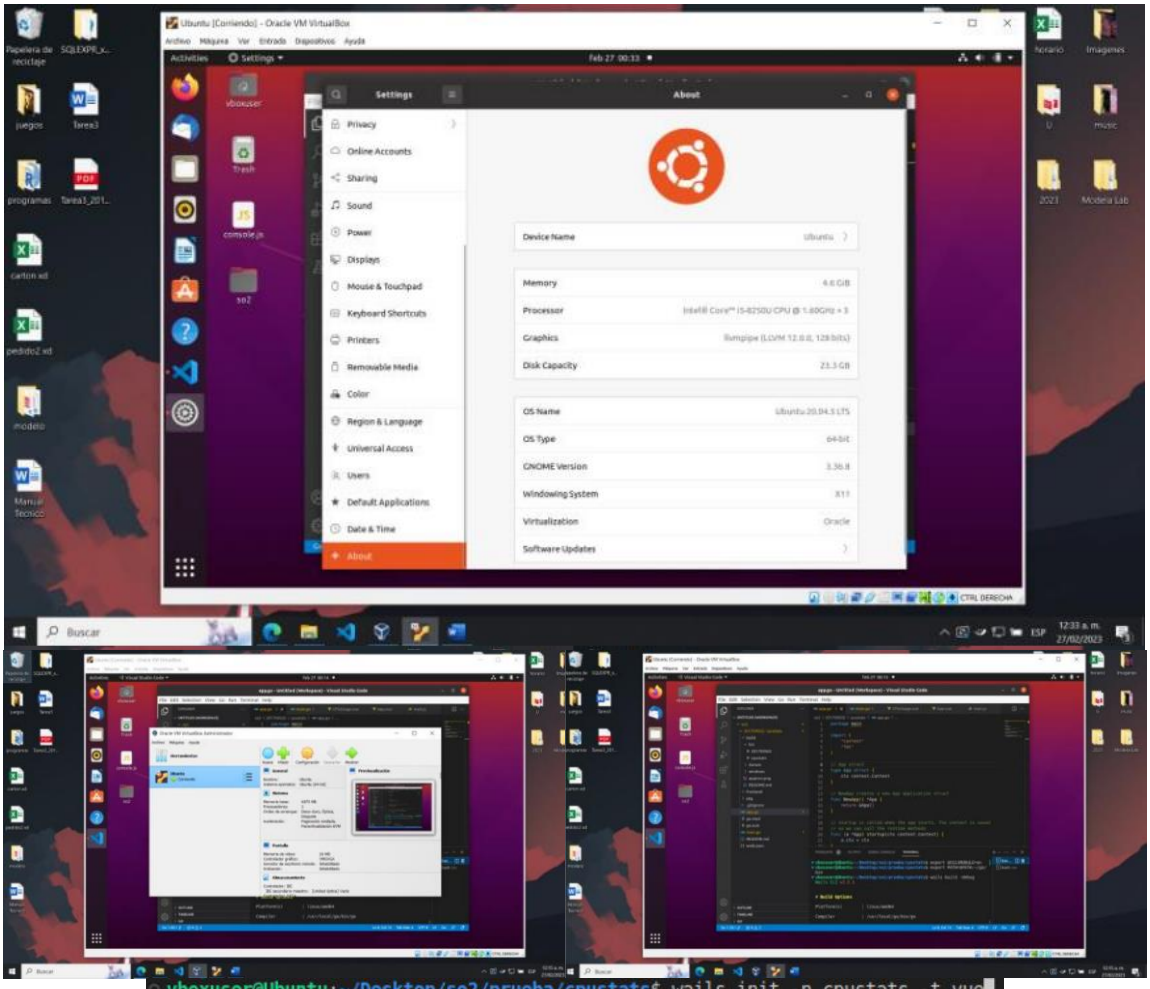
Se implementaron lianas de comando para la autenticación del usuario, con la finalidad de poder permitir la acción que se quiere realizar en el sistema.

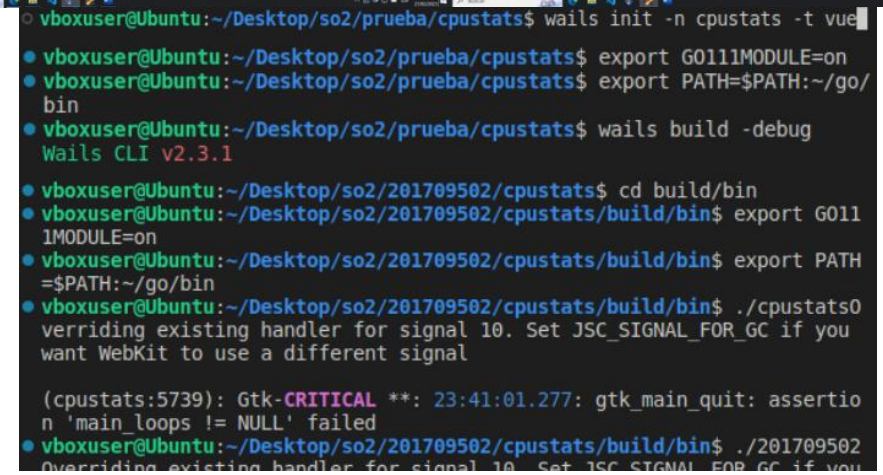
```
// Solicitar la contraseña del usuario y verificar si tiene permis
fmt.Println("Introduce tu contraseña para completar la accion: ")
cmd := exec.Command("sudo", "-v")
cmd.Stdin = os.Stdin
cmd.Stdout = os.Stdout
cmd.Stderr = os.Stderr
if err := cmd.Run(); err != nil {
    fmt.Println("Contraseña incorrecta")
    return
}
```

Para la habilitación de puertos USB se modificaron los permisos del directorio /media utilizando 777 para habilitar los permisos y 000 para deshabilitar los permisos e impedir la utilización de dispositivos usb

```
unblockCommand := "sudo chmod 777 /media/"
_, err2 := exec.Command("bash", "-c", unblockCommand).Output()
if err2 != nil {
    fmt.Println("Error")
    log.Fatal(err2)
}
fmt.Println("****Puertos USB habilitados")
```

Anexos





- Es posible realizar aplicaciones sencillas amigables al usuario para el monitoreo del sistema.
- Con la modificación de los permisos en directorios clave es posible el bloqueo o habilitación de los dispositivos USB.
- La utilización de contraseñas en los programas permite una mejor seguridad en la información o acciones que se quieran ejecutar en el sistema