PAYROLL MANAGEMENT SYSTEM

Mini-Project (OPERATING SYSTEM)

Submitted in partial fulfilment of the requirement of University of Mumbai
For the Degree of

(Computer Engineering)

By

1) Mitali Vyankat Mane (C-32) **ID No: TUS3F181934**
2) Bhushan Rajendra Patil (C-29) **ID No: TUS3F181930**
3) Tanuj Palaspagar (C-26) **ID No: TUS3F181926**

Under the Guidance of

Prof. Kishor Sakure

Department of Computer Engineering
TERNA ENGINEERING COLLEGE
Plot no.12, Sector-22, Opp. Nerul Railway station,
Phase-11, Nerul (w), Navi Mumbai 400706
UNIVERSITY OF MUMBAI

Terna Engineering College

NERUL, NAVI MUMBAI

CERTIFICATE

This is to certify that

1) Mitali Vyankat Mane (C-32)	**ID No: TUS3F181934**
2) Bhushan Rajendra Patil (C-29)	**ID No: TUS3F181930**
3) Tanuj Palaspagar (C-26)	**ID No: TUS3F181926**

*Has satisfactorily completed the requirements of the **Mini Project***

Of subject

Operating System

*As prescribed by the **University of Mumbai** Under the guidance of*

Prof. **Kishor Sakure**

Subject Incharge **APC** **HOD**

Index

TABLE OF CONTENTS		
Caption	**Title**	**Page No.**
1	Introduction	4
2	Problem Statement	5
3	Main system details	6
4	Implementation (code)	7
5	Implementation (system)	17
6	Interface	13
7	Conclusion	18
8	Reference	19

CHAPTER 1

Introduction

Objective of OS Project on Payroll System:

This program is based on a simple shell script. It is a simple menu driven program, where the user controls the system by choosing options given by the program.

“Payroll Management System” software developed for a company has been designed to achieve maximum efficiency and reduce the time taken to handle the Payroll activity. It is designed to replace an existing manual record system thereby reducing the time taken for calculations and for storing data.

The following tasks have been performed to meet the specifications of the project
scenario:

1. A script file as **mainmenu.sh** :
   This file contains the code to display the main menu
2. A script file as **file_manager.sh** :
   This file contains the code to change information and delete an employee from employee_master file.
3. A script file as **file_search.sh** :
   This file contains the code to search employees who have joined the company in the current year, their date of retirement in the current year and the department-wise total number of employees needs to be identified.
4. A script file as **print_payslips.sh** :
   This file contains the code to generate pay-slips for each employee in the format.
5. A script file as **print_da.sh** :
   This file contains the code to generate deduction and allowance details report for all grades.

CHAPTER 2

Problem Statement

PAYROLL MANAGEMENT is clearly complex. A major problem is that people are exposed to an unmanageable number of potential contacts.
The project Payroll management is used for calculating and accurately depositing salary in the employees' bank account.
The problem definition for designing is to achieve maximum efficiency and reduce the time taken to handle the Payroll activity

Objective of Project –

The main objective of this project is to create a user friendly and efficient system for professional payroll management, which saves the company's valuable time and man-labour.

About shell script:

A shell script is a computer program designed to be run by the Unix shell, a command-line interpreter. The various dialects of shell scripts are considered to be scripting languages.
Bash is a command processor that typically runs in a text window where the user types commands that cause actions. Bash can also read and execute commands from a file, called a shell script.

System requirements –

Linux/Unix Operating system
Or
Windows Operating System (With Unix environment)
Notepad++ (For EOL conversion to Unix)

Main System

Main system interface consists of the following parts:

Serial No.	File Name	Description
1	mainmenu.sh	Contains the code to display the main menu
2	file_manager.sh	Contains the code to change information and delete an employee from employee_master file.
3	file_search.sh	Contains the code to search employees have joined the company in the current year, their date of retirement in the current year and department-wise total number of employees needs to be identified
4	print_payslips.sh	Contains the code to generate pay-slips for each employee in the format.
5	print_da.sh	Contains the code to generate deduction and allowance details report for all grades.

CHAPTER 3

Implementation

Code (Bash Script code) -

i. MainMenu:

```
#!/bin/bash
get_return()
{
echo -e "Press return \c"
read x
return 0
}
get_confirm()
{
echo -e "Are you sure? \c"
while true
do
read x
case "$x" in
y | yes | Y | Yes | YES )
return 0;;
n | no | N | No | NO )
echo
echo "Cancelled"
return 1;;
*) echo "Please enter yes or no" ;;
esac
done
}
mainmenu()
{
OPT=0
while [ "$OPT" != "5" ]
do
clear
echo
echo
echo
echo -e "\033[30;1m
MAIN MENU \033[0m"
echo
echo "
1. Manager Employee Informations"
echo "
2. Search Employee Informations"
echo "
3. Generate pay-slips for Employees"
echo "
4. Generate Deduction and Allowance
Informations"
echo "
5. Quit"
echo
echo -n " Please enter choice then
press return: "
read OPT
case $OPT in
"1") . file_manager.sh ;;
"2") . file_search.sh ;;
"3") . Print_payslip.sh ;;
"4") . print_da.sh ;;
"5") exit;;
*) echo -e "\n\t\t\t\tInvailid Input"
echo -e "\n\t\t\t\tPress <Enter> key
to continue...\c"
read;;
esac
done
echo "Option chosen: [$OPT]"
return $OPT
}
mainmenu
```

ii. FileManager:

```
file_manager()
{
clear
OPT=0
while [ "$OPT" != "4" ]
do
clear
echo
echo
echo
echo -e "Manager Employee
Informations"
echo
echo "
1. Add New Employees "
echo "
2. Update Employee Informations"
echo "
3. Delete Records Employee
Informations "
echo "
4. Back"
echo
echo -n " Please enter choice: "
read OPT
case $OPT in
"1") add_em;;
"2") update_em;;
"3") delete_em;;
"4") return;;
*) echo -e "\n\t\t\t\tInvailid Input"
echo -e "\n\t\t\t\tPress <Enter> key
to continue...\c"
read;;
esac
done
return $OPT
exit 0
}
add_em()
{
# Check the Employee_Master File file
here
sort_id=`sort Employee_Master`
echo "$sort_id" > E_id.txt
set_id=`tail -1 E_id.txt`
echo "$set_id" > E_id.txt
firstline=`cut -d : -f1 E_id.txt`
if [ -z "$firstline" ]
then
e_id="E001"
else
numid=`expr substr $firstline 3 3
numid=`expr $numid + 1`
if [ $numid -gt 1 -a $numid -le 9 ]
then
e_id="E00${numid}"
elif [ $numid -ge 10 -a $numid -le 99
]
then
e_id="E0${numid}"
elif [ $numid -ge 100 -a $numid -le
999 ]
then
e_id="E${numid}"
else
echo "Data Invailid Input"
exit
fi
fi
#For clearing the screen
echo
echo
read
echo
echo
read
read
echo
read
echo
read
echo
read
echo
read
case
-e "Enter Employee Informations"
-e "Enter First Name: \c"
F_name
-e "Enter Last Name: \c"
-e "Enter Department: \c"
depart
L_name
-e "Enter Date of Birth(D/M/Y): \c"
DoB
-e "Enter Date of Joining(D/M/Y): \c"
DoJ
-e "Enter Grade: \c"
grade
-e "Enter Basic Salary: \c"
basic_salary
"$grade" in
SSK | ssk )
Gr_Sa1=$(($basic_salary*110))
;;
```

```
HSK | hsk )
Gr_Sa1=$(($basic_salary*108))
;;
SK | sk )
Gr_Sa1=$(($basic_salary*107))
;;
SMSK | smsk )
Gr_Sa1=$(($basic_salary*105))
;;
UNSK | unsk )
Gr_Sa1=$(($basic_salary*103))
;;
esac
Gr_Sa=`expr $Gr_Sa1 / 100`
if get_confirm ; then
echo
"$e_id:$F_name:$L_name:$depart:$DoB:$D
oJ:$grade:$basic_salary:$Gr_Sa"
>> Employee_Master
else
remove_em
fi
echo -e -n "\t\tDo you want input more
Employees (y/n)? : \033[0m"
read ans
if [ $ans == 'Y' -o $ans == 'y' ]
then
add_em
else
return 0
fi
}
# Find a Employee to update or Delete
Information
find_em()
{
e_id=""
echo -e "Enter a ID of Employee to
search for in the Employee
Informations"
read SearchStr
if [ "$SearchStr" = "" ]; then
return 0
fi
grep "^$SearchStr" Employee_Master >
temp_file.txt
IFS=":"
read E_id F_Name L_Name DoB DoJ grade
Ba_Sa Gr_Sa < temp_file.txt
IFS=":"
if [ -z "$E_id" ]; then
echo -e "Sorry! Employee has ID is
$SearchStr nothing found"
return 0
fi
echo
echo "ID of Employee: $E_id"
echo "First Name: $F_Name"
echo "Last Name: $L_Name"
echo "Date of Birth: $DoB"
echo "Date of Joining: $DoJ"
echo "Grade: $grade"
echo "Basic Salary: $Ba_Sa"
echo "Gross Salary: $Gr_Sa"
echo
return 1
}
update_em()
{
if [ -z "$E_id" ]; then
echo "You must select a Employee
first"
find_em n
fi
if [ -n "$E_id" ]; then
get_confirm && {
grep -v "^$E_id" Employee_Master >
temp_file.txt
mv temp_file.txt Employee_Master
echo
add_em
}
fi
get_return
return
}
delete_em()
{
if [ -z "$E_id" ]; then
echo "You must select a Employee
first"
find_em n
fi
if [ -n "$E_id" ]; then
echo -e "You are about to delete
$E_id"
get_confirm && {
grep -v "^$E_id" Employee_Master >
temp_file.txt
mv temp_file.txt Employee_Master
echo -e "Employee Informations
deleted"
E_id=""
}
fi
get_return
return
}
file_manager
```

iii. FileSearch:

```
file_search()
{
clear
OPT=0
while [ "$OPT" != "4" ]
do
clear
echo
echo
echo
echo -e "Search Employee
Informations"
echo
echo "
1. Search Employees joined the company
in the current
year "
echo "
2. Search Employees have their date of
retirement in
the current year"
echo "
3. Search Employees in same
Department"
echo "
4. Back"
echo
echo -n " Please enter choice then you
need search: "
read OPT
case $OPT in
"1") search_join;;
"2") retirement_em;;
"3") department_em;;
"4") return;;
*) echo -e "\n\t\t\t\tInvailid Input"
echo -e "\n\t\t\t\tPress <Enter> key
to continue...\c"
read;;
esac
done
echo "Option chosen: [$OPT]"
return $OPT
}
search_join()
{
clear
year_cur=$(date "+%Y")
grep "$year_cur" Employee_Master >
temp_file.txt
read bien < temp_file.txt
if [ -z "$bien" ]; then
echo -e "There are no employees in
this company"
else
num_join=$(wc -l temp_file.txt)
echo -e "The number of employees
joining the company this year:
$num_join"
echo -e ""
echo -e "Staff List:"
cat temp_file.txt
fi
echo -e ""
get_return
}
department_em()
{
clear
echo
echo -e " Enter Department"
read depart
grep "$depart" Employee_Master >
temp_file.txt
read bien1 < temp_file.txt
if [ -z "$bien1" ]; then
echo -e "Phong ban go khong co trong
cong ty"
get_return
else
num_join=$(wc -l temp_file.txt)
echo -e "The room style is not in the
company $depart call: $num_join"
echo ""
echo -e " The staff list in the room
$depart collect:"
cat temp_file.txt
bien1=""
get_return
fi
}
file_search
```

iv. PrintPayslip:

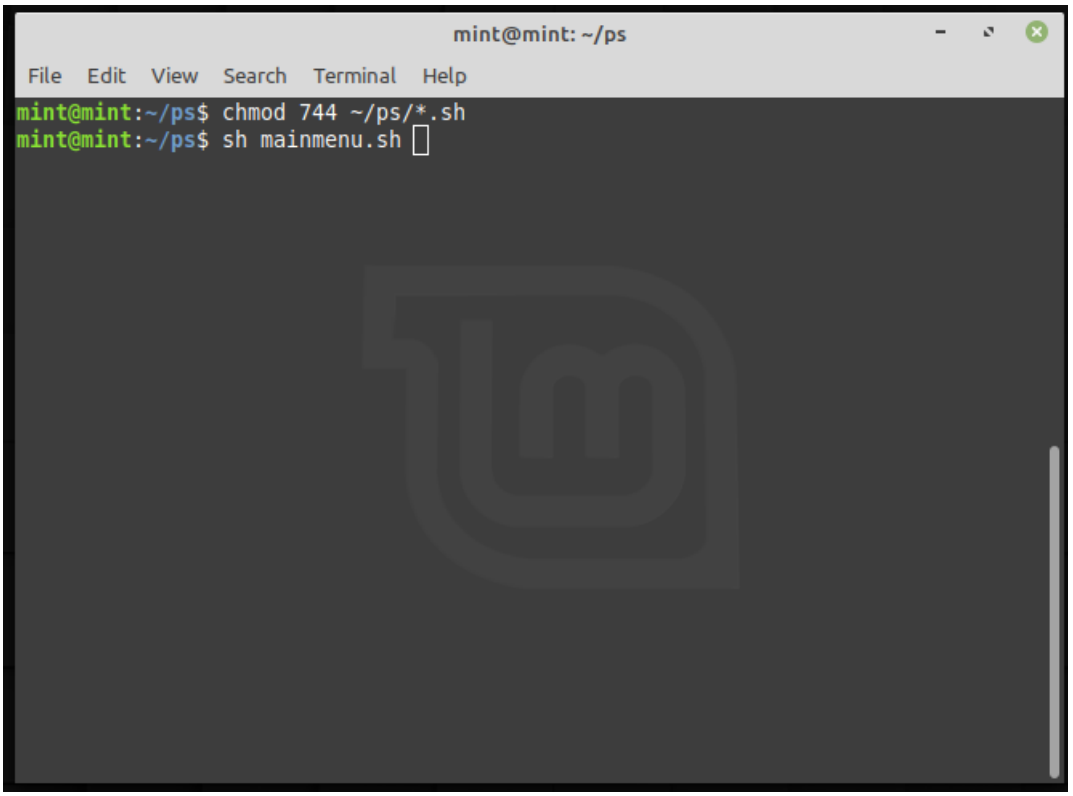
```
Print_payslip()
{
echo -e "Enter a ID of Employee to
generate Pay-slip"
read Str
if [ "$Str" = "" ]; then
return 0
fi
grep "^$Str" Employee_Master >
temp_file.txt
IFS=":"
read E_id F_Name L_Name depart DoB DoJ
grade Ba_Sa Gr_Sa <
temp_file.txt
IFS=":"
if [ -z "$E_id" ]; then
echo -e "Sorry! Employee has ID is
$Str nothing found"
return 0
fi
echo ""
echo "
Deezy Corp, Pay-Slip"
echo ""
echo "Employee ID: $E_id
F_Name: $F_Name
L_Name: $L_Name"
echo "Department: $depart
Grade: $grade"
echo "Basic Salary: $Ba_Sa
Gross Salary: $Gr_Sa"
echo ""
echo "
Manager"
echo "
Accounts Department"
echo""
get_return
}
Print_payslip
```

v. PrintDA:

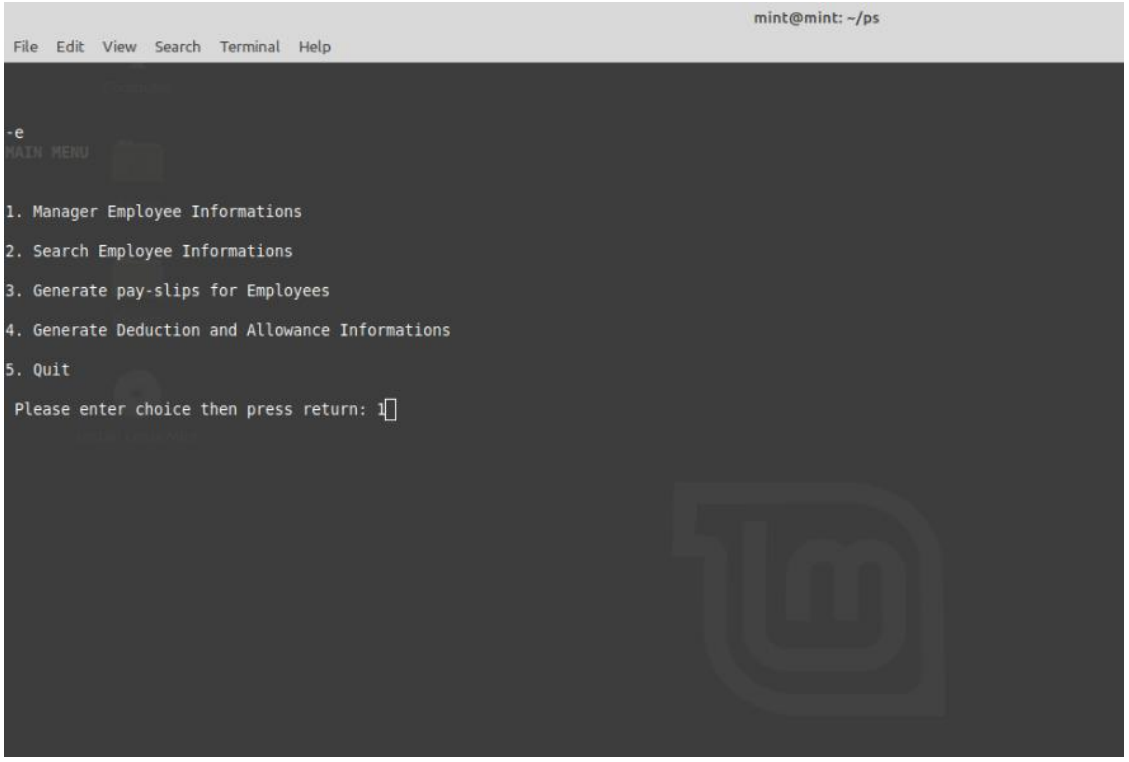
```
print_da()
{
echo -e "Enter a Grade to Generate
details report for Deduction and
Allowance"
read Str
if [ "$Str" = "" ]; then
return 0
fi
grep "^$Str" Grade.txt > temp_file.txt
IFS=":"
read grade deduc allow < temp_file.txt
IFS=":"
if [ -z "$grade" ]; then
echo -e "Sorry! Grade is $Str nothing
found"
return 0
fi
echo ""
echo "
Deduction and Allowance of $grade"
echo ""
echo "
Grade
Deduction
Allowance"
echo "
$grade
$deduc
$allow"
echo ""
get_return
}
print_da
```

Interface –

For executing the project run the following commands:



Main menu :



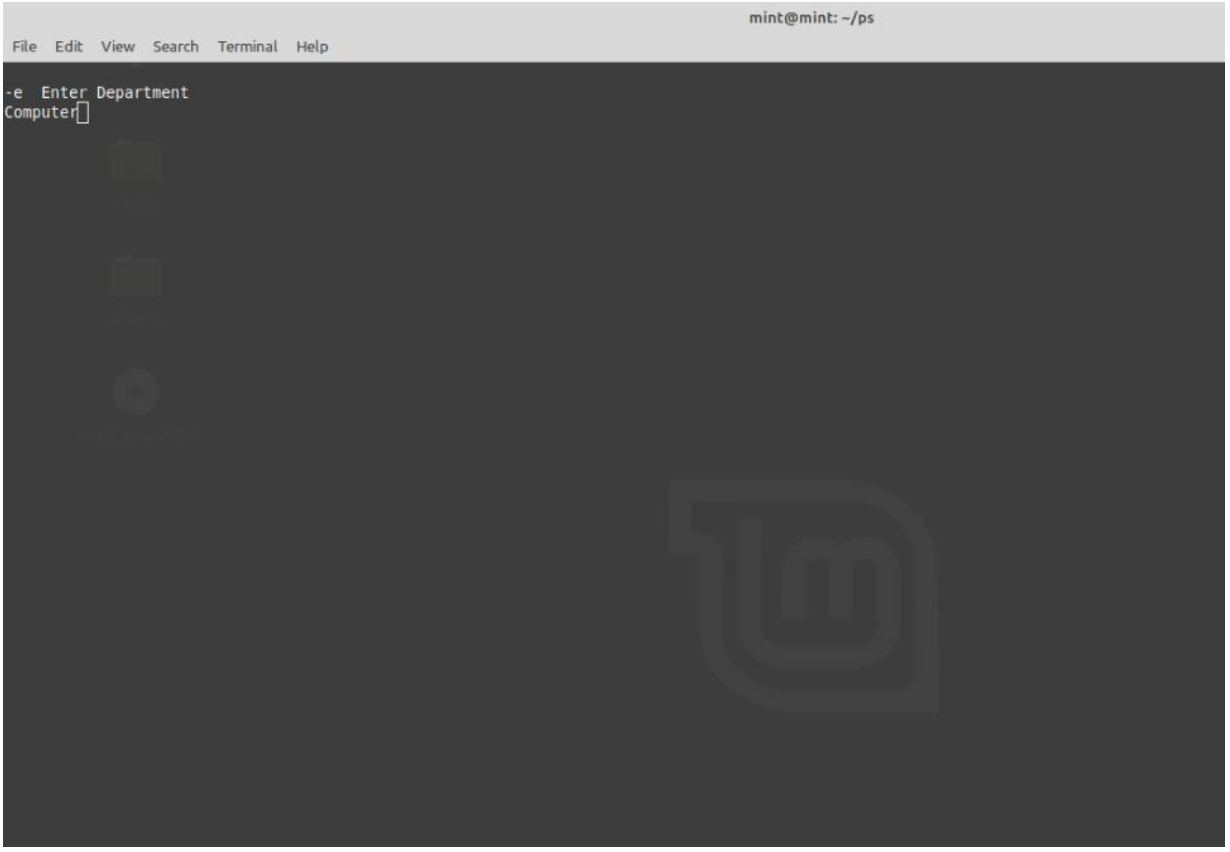
Another example:

```
mint@mint: ~/ps
File  Edit  View  Search  Terminal  Help

-e
MAIN MENU

1. Manager Employee Informations

2. Search Employee Informations

3. Generate pay-slips for Employees

4. Generate Deduction and Allowance Informations

5. Quit

 Please enter choice then press return: 2
```

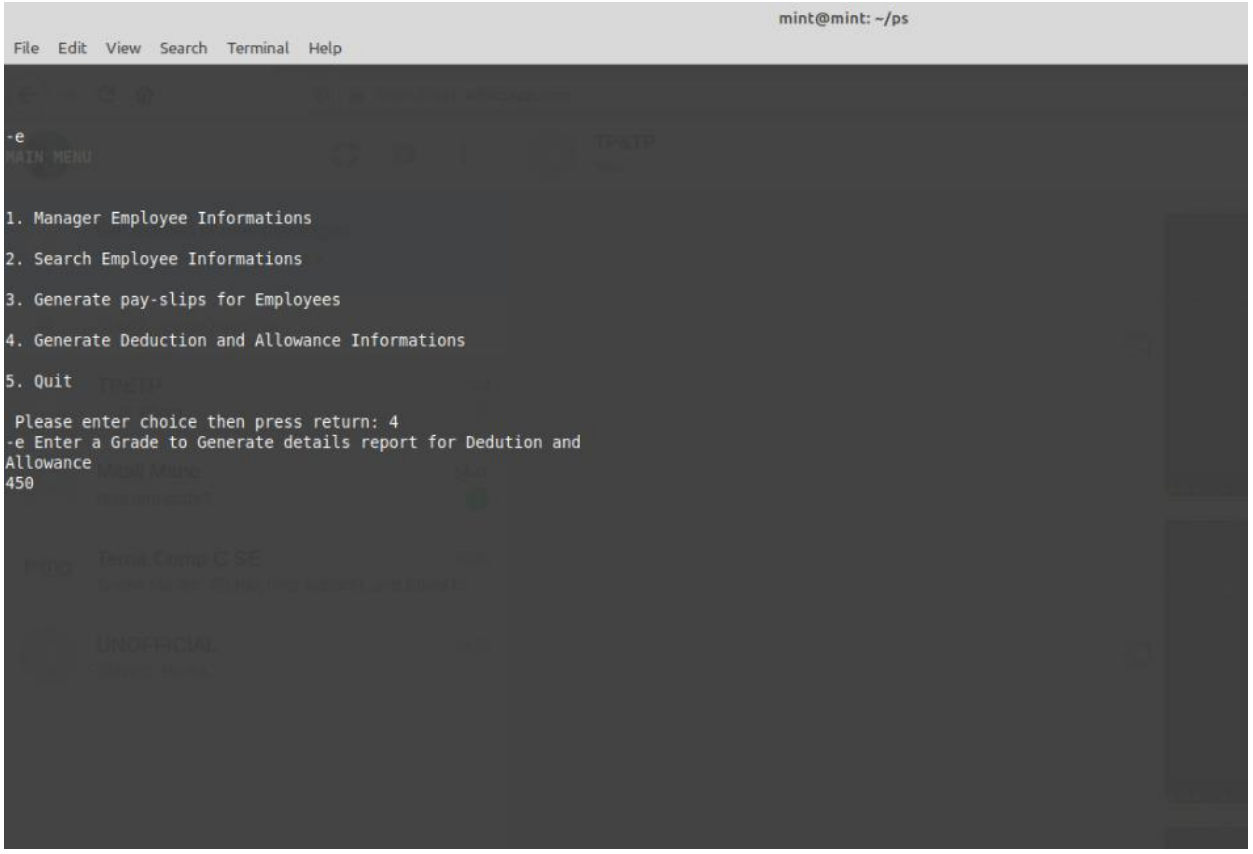
After selecting 2nd option, a sub menu will be shown:

```
mint@mint: ~/ps
File  Edit  View  Search  Terminal  Help

-e
Search Employee
Informations

1. Search Employees joined the company in the current
year

2. Search Employees have their date of retirement in
the current year

3. Search Employees in same Depatment

4. Back

 Please enter choice then you need search: 3
```

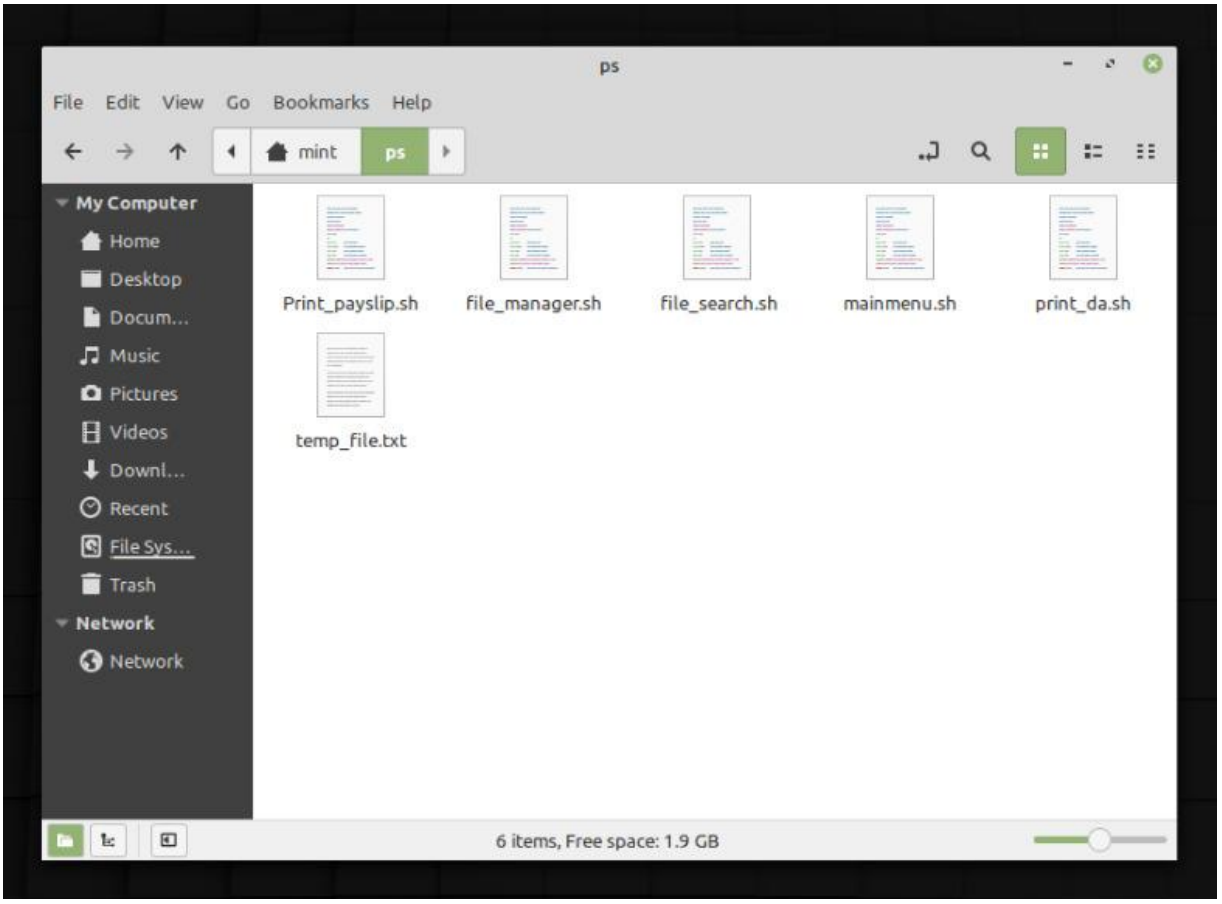
Searching by department:



Generation of Deduction and Allowance Information (DA):



The final generated + required files for the running of this system:



After our initial steps to using the system, a new file is created in the parent directory, where mainmenu.sh file resides. It is named temp_file.txt, and it contains the information that is fed through the server system. This file is used to display the required information. It essentially works as a naïve Database and all information will be lost (system will be reset) if this file gets corrupted or deleted.

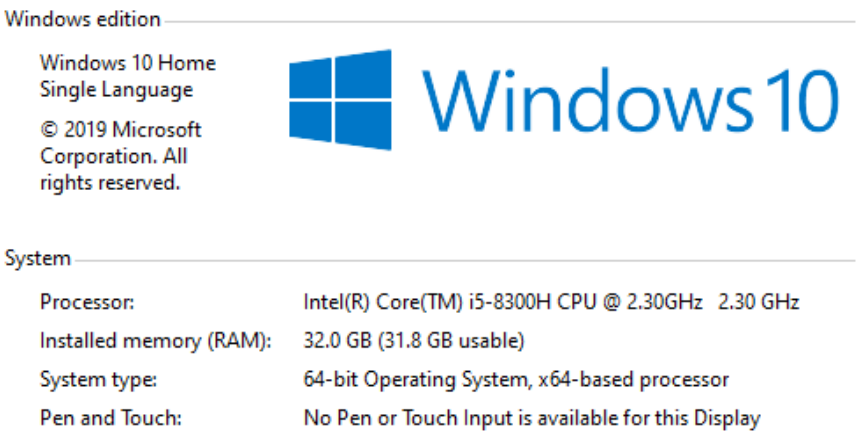
Host System Environment-

This system was written and executed on a Linux Mint Virtual system hosted by VirtualBox open-source utility in windows, host being Windows 10 OS.

Guest Info:

```
[code]
System:     Host: mint Kernel:
5.0.0-32-generic x86_64 bits: 64
compiler: gcc v: 7.4.0
            Desktop: Cinnamon 4.4.5
wm: muffin dm: LightDM Distro: Linux
Mint 19.3 Tricia
            base: Ubuntu 18.04 bionic
Machine:   Type: Virtualbox System:
innotek product: VirtualBox v: 1.2
serial: <filter>
            Chassis: Oracle
Corporation type: 1 serial: <filter>
            Mobo: Oracle model:
VirtualBox v: 1.2 serial: <filter>
BIOS: innotek v: VirtualBox
            date: 12/01/2006
CPU:        Topology: Single Core
model: Intel Core i5-8300H bits: 64
type: MCP arch: Kaby Lake
            rev: A L2 cache: 8192 KiB
            flags: lm nx pae sse sse2
sse3 sse4_1 sse4_2 ssse3 bogomips:
4608
            Speed: 2304 MHz min/max:
N/A Core speed (MHz): 1: 2304
Graphics:  Device-1: VMware SVGA II
Adapter driver: vmwgfx v: 2.15.0.0 bus
ID: 00:02.0
            chip ID: 15ad:0405
            Display: x11 server: X.Org
1.20.4 driver: vmware unloaded:
fbdev,modesetting,vesa
            resolution: 1400x1050~60Hz
            OpenGL: renderer: llvmpipe
(LLVM 8.0 256 bits) v: 3.3 Mesa 19.0.8
compat-v: 3.1
            direct render: Yes
Network:   Device-1: Intel 82540EM
Gigabit Ethernet driver: e1000 v:
7.3.21-k8-NAPI port: d020
            bus ID: 00:03.0 chip ID:
8086:100e
            IF: enp0s3 state: up
speed: 1000 Mbps duplex: full mac:
<filter>
            Device-2: Intel
82371AB/EB/MB PIIX4 ACPI type: network
bridge driver: piix4_smbus
            v: N/A port: d200 bus ID:
00:07.0 chip ID: 8086:7113
Drives:     Local Storage: total: N/A
used: 127.6 MiB
Partition: ID-1: / size: 1.93 GiB
used: 127.6 MiB (6.5%) fs: overlay
source: ERR-102
USB:        Hub: 1-0:1 info: Full
speed (or root) Hub ports: 12 rev: 1.1
chip ID: 1d6b:0001
            Device-1: 1-1:2 info:
VirtualBox USB Tablet type: HID
driver: hid-generic,usbhid
            rev: 1.1 chip ID:
80ee:0021
```

Host Info:



CHAPTER 4

Conclusion –

Hereafter the completion of the project we got familiar with the shell script and its features. Due to its lack of time and we are beginners in the programming aspect that we expected can't be developed by us.

As a whole, the project has been a good learning experience for us. We have gained knowledge about the various aspects of shell-script. At the same time, we have developed a deep understanding about menu-driven systems using shell script and Unix environments.

We still want to emphasize that the program is not complete by itself. There is still a lot of room for improvement. Graphics may be added to the program to make it more attractive.

Reference –

1. Scripting:

- *Bash scripting cheat sheet* *(devhints.io/bash)*
- *Learning the bash Shell (Book by Cameron Newham)*

2. Designing System:

- *Finding included in BE project by students of Department of Computer Science and Engineering The People's University of Bangladesh(PUB):*

https://www.academia.edu/8828018/Design_and_Development_Of_Payroll_Management_System/

- *General Payroll Info:*
  https://www.hrpayrollsystems.net/payroll-systems/

- *General Guide and other references from:*
  https://smallbusiness.chron.com/design-payroll-system-75850.html

3. Project Guides:

- *https://projectabstracts.com/list-of-payroll-management-system-projects*
- *http://www.sourcecodeonline.com/list?q=mini_project_in_operating_system_concepts*