# OPERATING SYSTEM LAB (COM -312)

## "Online test platform using Linux pattern matching commands and file handling concept "

At

CSE, MODEL INSTITUTE OF ENGINEERING AND TECHNOLOGY

BACHELOR OF TECHNOLOGY (Computer Engineering)

Submitted By

Name: Sashakt Dev Singh Jamwal, Sia, Rakshit Gupta & Ayushmaan Singh Jamwal

Roll No: 2021A1R044, 2021A1R047, 2021A1R050, 2021A1R052

Branch: CSE

Semester: 3rd

Email

2021A1R044@mietjammu.in

2021A1R047@mietjammu.in

2021A1R050@mietjammu.in

2021A1R052@mietjammu.in

# Contents

**ACKNOWLEDGEMENT** .....3

**Abstract** .....4

**1. Introduction**.....5

2. Objective .....5

3. Terminology .....6

4.Algorithm .....9

5.Flowchart .....12

6. Implementation .....13

a. Code .....13

b. Output.....31

7. References .....35

# ACKNOWLEDGEMENT

Under CRIE (Center for Research, Innovation and Entrepreneurship) we both worked under the guidance of mentors of CRIE, who guided us through a way leading to professionalism and practical hands-on work experience. It is indeed with a great pleasure and immense sense of gratitude that we acknowledge the help of these individuals. We are highly indebted to our

Director Ankur Gupta, "MODEL INSTITUTE OF ENGINEERING AND TECHNOLOGY", for the facilities provided to accomplish this main project. We would like to thank our Prof. Ashok Kumar, Head of the Department of Computer Science and Engineering, MIET. For this constructive criticism throughout our project. We feel elated in manifesting our sense of gratitude tour internal project guide. Asst. Saurabh Sharma, Department of Computer Science and Engineering, MIET. He has been a constant source of inspiration for us and we are very deeply thankful to him.

# FACULTY AND MEMBERS

Dr. Ankur Gupta - He is the Director at the Model Institute of Engineering and Technology, Jammu, India, besides being a Professor in the Department of Computer Science and Engineering. Prior to joining academia, he worked as Technical Team Lead at Hewlett Packard, developing software in the network management and e-Commerce domains. He has 2 patents to his name and 20 patents pending. He obtained his B.E, Hons. He is a senior member of the ACM, senior member IEEE and a life-member of the Computer Society of India. He has received competitive grants over Rs. 2 crores from various funding agencies and faculty awards from IBM and EMC.

Prof. Ashok Kumar - He has over 21 years of experience in industry, academics, research and academic administration. He did his Doctorate from IKG Punjab Technical University, Jalandhar, Master of Engineering from Punjab Engineering College, Chandigarh and Bachelor's degree from NIT, Calicut, Kerala. He is Fellow of Institution of Electronics and Telecommunication Engineers (IETE). His research interest areas are Optical Networks, Electronics Product Design, Wireless Sensor Networks and Digital Signal Processing. He is reviewer of many reputed national and international journals. He has also coordinated many national and international conferences.

Mr. Saurabh Sharma - He is working as Assistant Professor in the Department of Computer Science at Model Institute of Engineering and Technology, Jammu and has over 10 years of experience in academics and research. He has done his M. Tech CSE in Web Mining from Maharishi Markandeshwar University, Mullana, Ambala (HR) in the year 2011.His research interest areas are Web Mining, Data Mining, Machine Learning, ANN, Pattern Classification and Object Identification. He has, to his credit, 25 research papers published in journals of national and international repute and he has guided 11 M. Tech Dissertations.

# Abstract

Online test platform is a BASH shell-based tool where tests are conducted online through the internet using a computer system. The main goal of this online examination system is to effectively evaluate the student thoroughly through a totally automated system that not only reduces the required time but also obtains fast and accurate results.

It is a good source of interactivities among the students and between the teacher and students. It is done to improve student's comprehension levels and learning motivation. As one of their tools, online test tools are quite effective. However, in order to use the online test tool, a teacher generally requires a great deal of labour.

For example, a teacher needs to create quizzes and input them in the online test tool.

To solve these problems, we have developed a Bash shell based Online Test Platform which can create quizzes competitively and collaboratively by students for the purpose of reducing the load required for a teacher and promoting interactions among students and between the teacher and students.

Keywords: Bash, Shell Scripting, Index Page, Sign-In, Sign-Up, Pattern Matching, File Handling

# 1. Introduction

Command line test is a BASH shell-based tool that simulates login based online testing scenarios. Initially the user will be provided with a sign-in option where predefined users will be allowed to log in. Upon successful login, this tool will display questions on the screen for the user from the existing database. It will also handle error conditions like time-out. This tool will also store answers provided by users for future verification.

There are a lot of online test platforms which enable students to take tests online. They will typically have a user-interface, backend question bank and evaluation part. They will also support other features like predefined time per question, output reports etc. The idea of this project is to simulate such an online test interface using Linux Shell Scripting and commands.

By implementing this Linux Shell Scripting Project, we will apply Shell programming constructs (ex: loops), Pattern matching commands (ex: grep, sed, etc…) and File handling (ex: permission, directories etc…) aspects during implementation.

We start implementing this Linux Shell Scripting Project by first making an Index Page.

# 2. Objective

Goal of this project is to simulate login based online testing scenario. Initially the user will be provided with a sign-in option where pre-defined users will be allowed to login. Upon successful login this tool will display questions for the user from existing database. It will also handle error conditions like time-out. This tool will also store answers provided by users for future verification.

# 3. Terminology

Index Page

In this, three options will be displayed on the user's screen:

1. Sign-In

2. Sign-Up

3. Exit

Then a pop-up message will be displayed on the user's screen to take the input from the user, and from the input given by the user, the program will proceed accordingly.

For instance, if the user enters the input as '1', then he'll be redirected to the Sign-In Page where the user will be given options to log into the page and further can take the test.

If the user enters the input as '2', then he'll be redirected to the Sign-Up page where the user will be asked some details for Sign-up.

And if the user enters the input as '3', then the program will be terminated.

Sign-In Page

In the sign-in page the user will be asked to enter the email id and password. The email id and password provided by the user are matched with the details of the users present in the user_database.csv. If the details do not match then the message is printed with an incorrect sign in details. If the email id provided by the user is matched with the details present in the user_database.csv. Then the details will be validated. And the user will be redirected to the profile page.

Sign-Up Page

In the sign-up page the user will be asked to enter the Name, E-mail ID, Contact number, Date of Birth and place of birth. All the above details will be stored in different variables. The password should be 8- alphanumeric characters, the email-id should be with "@.", the mobile number should be of 10 digits and the date of birth of the user should be in (dd/mm/yyyy) format. If any of the above conditions are not met then an error message will be displayed on the screen.

Profile Page

In this, five message will be displayed on the user's screen to take the input from the user:

1. Name

2. E-mail

3. Mobile no.

4.DOB

5.Place

All the above details of the users will be printed. Then a pop-up message will be displayed on the user's screen to take the input from the user. For instance:

1.ENTER[T] TO TAKE THE TEST

2.ENTER[E] TO EDIT PROFILE

3.ENTER[L] TO LOGOUT

If the user enters the input "T" then he will be redirected to take test, If the user enters the input "E" then he will be redirected to edit profile page, If the user enters the input "L" then will be redirected to logout, If the user enters any other input, then will be redirected to profile page.

Edit Page

The user will be able to edit the details present in the user_database.csv.

Test Screen

In the test screen, some instructions will be displayed on the user's screen before starting the test. After this, the user will be asked whether he wants to proceed further to take the test or not. If the input enters “Y” as the input, then the user will be redirected to the test screen to take the test. And if the user enters “N” as the input then the user will be redirected to the profile page.

If the user enters anything else other than yes or no, then a pop-up will appear on the user’s screen with a message “Invalid Input”.

Test Result

1. In this, “RESULT” and “TOTAL SCORE” messages will be displayed on the user's screen.

2. Then a pop-up message will be displayed on the user's screen to take the input from the user,

For instance, if the user enters the input:

1. ENTER[Y] to retake the test

2. ENTER[P] to go to profile page

3. ENTER[V] to view the test

3.If the user enters the input “Y” then he will be redirected to retake test, If the user enters the input “p” then will be redirected to profile page, If the user enters the input “V” then will be redirected to view the test, If the user enters any other input, then “INVALID OPTION” will be displayed.

View Test

In view test, the user will be able to evaluate the test. The question as well as the option selected by the user will be displayed on the screen along with the correct answer. By

viewing the test, the user will be able to evaluate his marks as well as the topics in which the user is lacking.

Test Questions

In this online test platform, the user will be given 10 questions. Each question has a time limit of 30 seconds. If the user failed to attempt the question in the given time, then he will be directly transferred to another question and the user won't be able to attempt that question that the user missed. And the above 10 questions will be randomly picked from a set of 20 questions provided in the question bank.

# 4.Algorithm

The student Will be provided 3 options: -

1. Sign-In

2. Sign-Up

3. Exit

The input given by the user will be stored as argument named "input_user_index"

If "$input_user_index == 1", then the user will be directed to the sign_in_page.sh

The user will be asked to enter the email-id and password.These details will be stored in argument "sign_in_email" and "sign_in_password".

If "${email_id_field[i]} == $sign_in_email ${password_field[i]} == $sign_in_password" then user will be directed to profile_page.sh.

And the username,email-id,mbile number, date of birth and birth place are printed on the screen.And the user will be given 3 options:-

[T] Take test

[E] Edit Profile details

[L] Log out

The input from the user will be saved as argument "user_input".
If "user_input == T" then the user will be directed to test_screen.sh
The user will be provided with 2 choices(Y/N).

If the "user_input == Y" then the user will be directed to test.sh

The user will be provided with 3 options: -

[Y] Retake the test

[N] Profile Page

[V] View test

The above input will be stored as argument "user_input"

If "user_input==Y" then the user will be directed to test.sh

If "user_input==N" then the user will be directed to profile_page.sh

If "user_input==V" then the user will be directed to view_test

The questions,with correct answer and the answer given by the user will be displayed

If "user_input==*" then the user will be directed to test_result

If the "user_input == N" then the user will be directed to profile_page.sh

If the "user_input == *" then the user will be directed to text_screen.sh

If "user_input == E" then the user will be directed to edit_page.sh

If "user_input == L" then the user will be directed to index_page.sh

If "user_input ==*" then "INVALID INPUT" will be printed, and the user will be directed to profile_page.sh

If any of the two detail provided by the user doesn't match the details present in the "user_database.csv".Then the a text message will be printed "Oops!! Incorrect Sign-In Details" and the user will be directed back to index_page.sh.

Else the user didn't give any input then the text will be printed "Enter input to login"

If "$input_user_index == 2", then the user will be directed to sign_up_page.sh

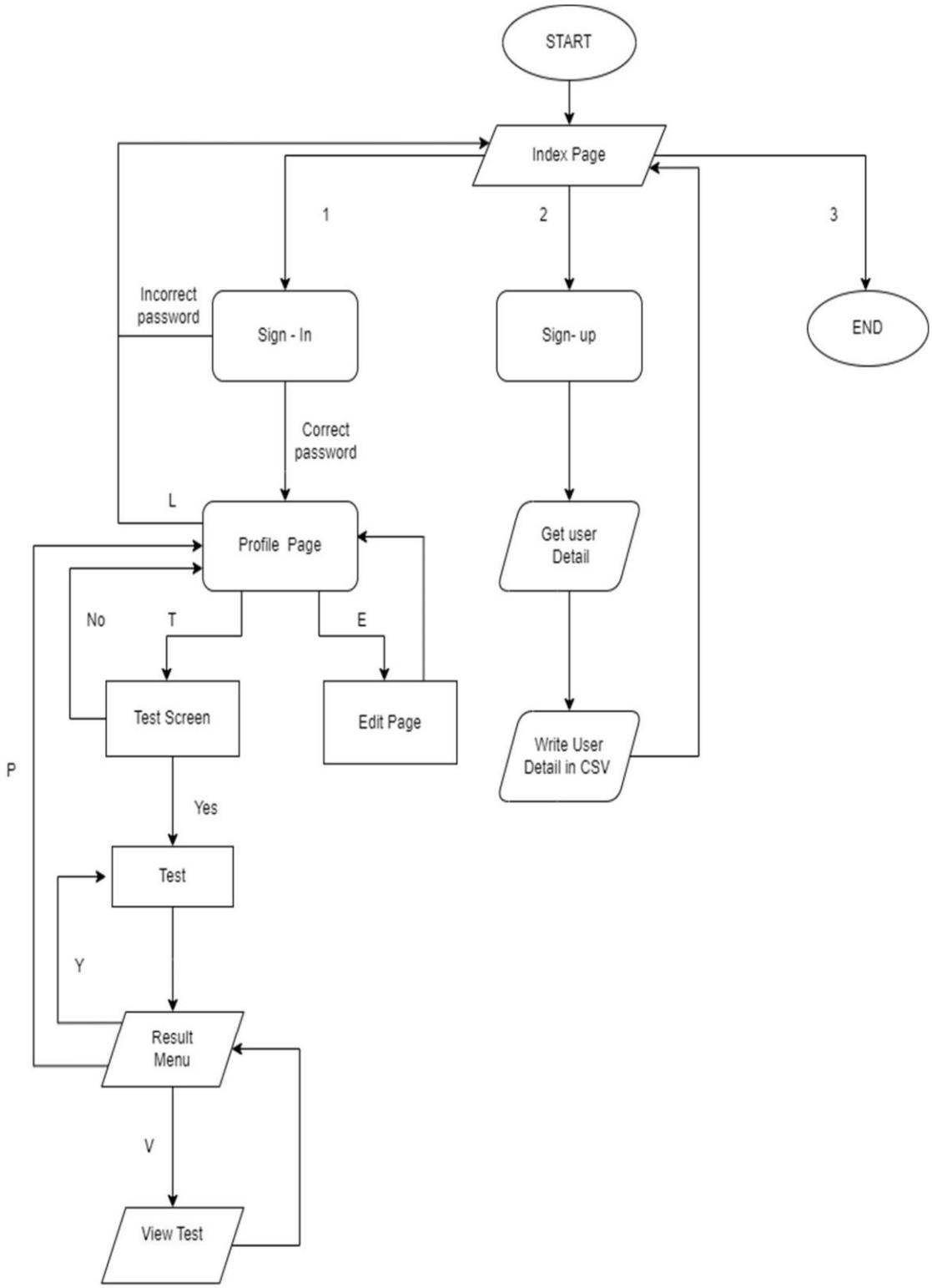
*The user will be asked to enter Name,Password,E-mail id, Contact number, Date of Birth and Birth place. All these will be stored in argument "user_name_from_user","user_password","email_id","mobile_no","DOB" and "place"respectively.

If "$input_user_index == 3", the the program will be closed.

If "$input_user_index == *", then invalid option will be printed, and the user will be directed to the index_page.sh.

Else the user didn't give any input then the text will be printed telling the user to give an input.

# 5.Flowchart


START
Index Page
1
2
3
Incorrect password
Sign - In
Sign- up
END
Correct password
L
Profile Page
Get user Detail
No
T
E
Test Screen
Edit Page
Write User Detail in CSV
P
Yes
Test
Y
Result Menu
V
View Test

# 6. Implementation

## a. Code

### i.Color.sh

```
#! /bin/bash

#background black="\e[40m"
red="\e[41m" green="\e[42m"
bg_yellow="\e[43m"
yellow="\e[33;7m" blue="\e[44m"
purple="\e[45m" cyan="\e[46m"
white="\e[47m"



normal="\e[0m"
 invisible="\e[8m"
reverse="\e[7m"
blink="\e[5m" bold="\e[1m"
lowintensity="\e[2m"
underline="\e[4m"




#regular text color rg_black="\e[30m"
rg_red="\e[31m" rg_green="\e[32m"
rg_yellow="\e[33m" rg_blue="\e[34m"
rg_purple="\e[35m" rg_cyan="\e[36m"
rg_white="\e[37m"
```

### i. start.sh

```
clear bash index_page.sh
```

### ii. index_page.sh

```
#!/bin/bash source
color.sh
        echo -e "$bold$blue                Test Your Knowledge                $normal"
```

```
        echo -e "$green
$normal"
        echo -e "$green                                                $normal"
              echo -e  "$bold\n 1.Sign-In"
        echo    " 2.Sign-Up"           echo -e "
3.Exit\n$normal"
                    echo -e "$green
$normal"
        echo -e "$green                                                $normal"
#prompts the user for the user-input
                          echo -en "$bold[Enter your Choice]:$normal"
             read input_user_index if [
$input_user_index ] then

#validates the user_index_page input
                    case $input_user_index in
1)          clear
                              bash sign_in_page.sh  ;;
2)          clear
                              bash sign_up_page.sh ;;
3)          clear
                    echo -e "\n\n\n\t\t\t**********\t$yellow This program has stopped executing as per
your input $normal\t**********\t\t\t"                    echo -e "\n\t\t\t\t**********\t $yellow Thank
you for
using Software $normal \t**********\t\t\t\n"
                    sleep 2
clear               exit ;;
             *)            clear
                                 echo -en "\n\n$red[ERROR]$normal Invalid Option
$red[ERROR]$normal\n\n"
                              bash index_page.sh ;;
             esac

else
              clear
                        echo -en "\n$red[ERROR]$normal Please give an input
$red[ERROR]$normal\n"
```

```
        bash index_page.sh fi
```

### iii. profile_page.sh

```
#!/bin/bash source
color.sh

#interpreters only if 6 arguments are passes which carries the userinfo if [ $# -eq 6 ] then

#page-layout
        echo -e "$bold$blue                        Profile
$normal"
        echo -e "$green
$normal"
              echo -en "[Name]:$1"       echo -en "\n[E-
mail-ID]:$2"    echo -en "\n[Mobile-No]:$3"
                 echo -en "\n[DOB]:$4"    echo -en
              "\n[Place]:$5"
                      echo -e "\n$green
$normal"
        echo -e "$green
$normal"
        echo -e "$green$bold   *Enter [T] to take the Test
$normal"
        echo -e "$green$bold   *Enter [E] to Edit Profile Details
$normal"
                              echo -e "$green$bold   *Enter [L] to Log-Out
$normal"
        echo -e "$green
$normal"
                            echo -en "$bold[Enter your Choice]$normal:"
                    read user_input

#validates the user_input
                    case $user_input in

                    "T") clear
                                    bash test_screen.sh $1 $2 $3 $4 $5 $6                    ;;
```

```
                "E")    clear
                                bash edit_page.sh $1  $2 $3 $4 $5 $6;;
                "L")    clear
                                bash index_page.sh ;;
            *)              clear
                                echo  -e "\n$red[ERROR]$normal INVALID INPUT
$red[ERROR]$normal"
                                bash profile_page.sh $1 $2 $3 $4 $5 $6;;
            esac

else
            clear
                    echo -e "\n$red[ERROR]$normal Sign-in or Log-in
$red[ERROR]$normal\n"
        bash index_page.sh fi
```

### iv. sign_in_page.sh

```
#import the files  source color.sh

#welcome heading flag=0
                                echo -e "\n$green$bold                ☞  Welcome to SIGN-IN
☜                       $normal"        echo -e "$blue$bold                  Test Your Knowledge
$normal"
      echo -e "$green
$normal"
      echo -e "$green                                                $normal"

#prompt the user for the email-id if existing user echo -en "[E-mail]:"
read sign_in_email
#prompt the user for password for the existing account echo -en "[Password]:"
read -s sign_in_password

      echo -e "\n$green
$normal"
      echo -e "$green                                                $normal"
#if the input is blank then Thrown an error asking the user to pass an valid entry
```

```
if [[ $sign_in_email =~  ^[[:punct:][:alnum:]]+$  ]] then

#accessing the user_database file for validating the user            user_database=('cat user_database.csv')
user_database_line_count=(`cat user_database.csv | wc -l`)      password_field=(`cat user_database.csv |
cut -d ';' -f 2`)      email_id_field=(`cat user_database.csv | cut -d ';' -f 3`)
mobile_number_field=(`cat user_database.csv | cut -d ';' -f 4`)              dob_field=(`cat
user_database.csv | cut -d ';' -f 5`)    place_field=(`cat user_database.csv | cut -d ';' -f 6`)
user_name_database=(`cat user_database.csv | cut -d ';' -f 1`)   echo ${user_name_from_database[@]}

#validating the email-id and password from the "sign_in" user with the existing "user_database.csv"
                        for((i=0; i<$user_database_line_count; i++))
                do
                                        if [ ${email_id_field[i]} == $sign_in_email -a
${password_field[i]}  == $sign_in_password   ]
                                then
  flag=0; index=$i    echo  "index:$index"
                                        break
else                                    flag=1
fi
                done



#does goes to the test_screen if user is logged in sccessfully else prints the error
                        if [ $flag -eq 0 ]
                then
                                clear
                                        bash profile_page.sh  ${user_name_database[$index]}
${email_id_field[$index]} ${mobile_number_field[$index]}
${dob_field[$index]} ${place_field[$index]} $index

                 else
                                clear
                        echo -e "\n$red$bold[ERROR]$normal Oops!! Incorrect SignIn Details
$red$bold[ERROR]$normal "                       echo -e "Existing User$bold[sign-in]$normal\tNew
User$bold[sign-up]$normal"
                                bash index_page.sh
```

```
            fi
else
            clear
                        echo -e "\n$red[ERROR]$normal Enter input to login
$red[ERROR]$normal" fi
```

v. sign_up_page.sh

```
#!/bin/bash
#imports the required files  source color.sh

#screen head  echo -e "\n$green$bold                    Welcome to SIGN-UP
$normal"
 echo -e "$blue$bold                    Test Your Knowledge
$normal"
 echo -e "$green
$normal"
 echo -e "$green                                                $normal"

#prompt  the  user  for  user_name  echo  -ne
"[Name]:"  read  user_name_from_user  if  [
$user_name_from_user ] then
#gets the existing user from the user_database flag=1
user_name_from_database=(`cat user_database.csv | cut -d ';' -f 1`)
line_count_from_database=(`cat user_database.csv | wc -l `)
#check if the user-name is present in the database for((i=0;
i<$line_count_from_database; i++)) do
if [ ${user_name_from_database[i]} == $user_name_from_user  ] then
            flag=0
```

```
        break fi
done

#validates the user_name  if [[  $flag -
eq 1 ]] then
                user_name=$user_name_from_user
                echo -n "[Password]:"
            read -s  user_password
#validates the password for 8 - alpha numeric character or thrown an error
                if [ ${#user_password} -gt 8 ]
             then
                                if [[ $user_password =~  [a-zA-Z]*..*[0-9]+[a-zA-Z]*   ]]
                        then
                                        echo -en "\n[Confirm Password]:"
                                    read -s  re_user_password

#validates the password and the confirm password or thrown an error
                                        if [  $user_password == $re_user_password  ]
                                then
                                    echo -en "\n[Email-Id]:"
read email_id

#validates the email-id with "@ ." or throw an error
                                                if [[ $email_id =~ .@.*\.[a-z]+   ]]
                                            then

                                                        echo -n "[Mobile-No]:"
                                                    echo -n "+91-"; read mobile_no

#validates the modile-number for 10-digits or throw an error
                                                            if [[ $mobile_no =~ ^[0-9]{10}$  ]]
                                                     then
                                                    echo -n "[DOB(dd/mm/yyyy)]:"
                                read DOB
#validates the Date of Birth of the User else throw an error
                                                                if [[ $DOB =~ ^[0-3][0-
9]/[0-1][0-9]/[0-9]{4}$  ]]
```

```
                                                                    then
                                                                    echo -n "[Place]:"
                                        read place
else                                                                clear
                    echo -e
"$red[ERROR]$normal Invalid Date try again with \"dd/mm/yyyy\"
$red[ERROR]$noraml"
                                                                    bash sign_up_page.sh
                            fi                                              else
                    clear                                                   echo -e
"\n$red[ERROR]$normal Enter the valid Phone-Number
$red[ERROR]$normal\n"
                                                                        bash sign_up_page.sh

fi


                                            else
                                                        clear
    echo -e "\n$red[ERROR]$normal Enter the valid email-id $red[ERROR]$normal\n"
                                                            bash sign_up_page.sh
                                        fi
                    else
                                            clear
                                                echo -e "\n$red[ERROR]$normal Mismatching
Password $red[ERROR]$normal\n\n"
                                                bash sign_up_page.sh
                                    fi

                        else
                                    clear
                                        echo -e "\n$red[ERROR]$normal Password Should
have atleast 8-alphaNumberic Charcters $red[ERROR]$normal\n\n"
                                        bash sign_up_page.sh
                        fi

                else
```

```
                clear
                        echo -e "\n$red[ERROR]$normal Password Should be atleast
8 alpha-numeric characters $red[ERROR]$normal\n"                bash
sign_up_page.sh
        fi


else
        clear
                echo -e "\n$red[ERROR]$normal User-Name  is Already Present
$red[ERROR]$normal\n"
          bash sign_up_page.sh
fi
else
        clear
                echo -e "\n$red[ERROR]$normal Please Enter the user_name
$red[ERROR]$normal\n"
          bash sign_up_page.sh
fi
#passes all the fields to user_database.csv file
echo "$user_name;$user_password;$email_id;$mobile_no;$DOB;$place;" >> user_database.csv
clear bash index_page.sh
```

### vi. edit_page.sh

```
#!/bin/bash
source color.sh
 if [ $# -eq 6 ] then

#This function prints the success message after the successful editing of the information  function
success_msg()
{
            clear
    echo -e "$bold$blue              EditProfile              $normal"
    echo -e "$green
$normal"
    echo -e "$green
$normal"
```

```
                                echo -e "\n\t\t\t$blue$bold Successfully Edited $normal"

        echo -e "\n$green
$normal"
        echo -e "$green
$normal"
        echo -e "$green
$normal"
                read n
                clear
                        bash profile_page.sh  $1 $2 $3 $4 $5 $6


}


#index variable to access the values  index=$((${6}+1))


#page layout
        echo -e "$bold$blue                      EditProfile                     $normal"
        echo -e "$green
$normal"
        echo -e  "$bold\t\t Enter The Option to edit $normal"
                   echo -en "1)Name:$1"
        echo -en "\n2)E-mail-ID:$2"            echo -en
"\n3)Mobile-No:$3"            echo -en "\n4)DOB:$4"
        echo -en "\n5)Place:$5"
        echo -e "\n$green
$normal"
        echo -e "$green
$normal"
        echo -e "$green
$normal"
        echo -en "$bold[Enter your Choice]$normal:"               read user_input
```

```
#validates the user_input for appropriate value case $user_input in

1)              echo -en "$bold[Enter the New-Value]:$normal"  read  new_entry
       user_name_database=(`cat user_database.csv | cut -d ';' -f 1`)
                              sed  "${index}s/;/ /g" user_database.csv | sed
"${index}s/${user_name_database[$index-1]}/$new_entry/g"  |sed
"${index}s/ /;/g" > new1.txt && mv new1.txt user_database.csv
              success_msg $1 $2 $3 $4 $5 $6       ;;            bash index_page.sh
2)              echo -en  "$bold[Enter the New-Value]:$normal"  read  new_entry
              email_id_field=(`cat user_database.csv | cut -d ';' -f 3`)            sed  "${index}s/;/ /g"
user_database.csv | sed
"${index}s/${email_id_field[$index-1]}/$new_entry/g" |sed "${index}s/
/;/g">new1.txt && mv new1.txt user_database.csv
                      success_msg ;;
                    bash index_page.sh

3)              echo -en "$bold[Enter the New-Value]:$normal"  read  new_entry
              mobile_number_field=(`cat user_database.csv | cut -d ';' -f 4`)               sed  "${index}s/;/ /g"
user_database.csv | sed
"${index}s/${mobile_number_field[$index-1]}/$new_entry/g"  |sed
"${index}s/ /;/g">new1.txt && mv new1.txt user_database.csv
              success_msg $1 $2 $3 $4 $5 $6       ;;            bash index_page.sh

4)              echo -en "$bold[Enter the New-Value]:$normal"  read  new_entry
              dob_field=(`cat user_database.csv | cut -d ';' -f 5`)       sed  "${index}s/;/ /g"
user_database.csv | sed
"${index}s/${dob_field[$index-1]}/$new_entry/g"  |sed "${index}s/
/;/g">new1.txt && mv new1.txt user_database.csv
              success_msg $1 $2 $3 $4 $5 $6       ;;            bash index_page.sh

5)              echo -en  "$bold[Enter the New-Value]:$normal"
                    read  new_entry
              place_field=(`cat user_database.csv | cut -d ';' -f 6`)    sed  "${index}s/;/ /g"
user_database.csv | sed
"${index}s/${place_field[$index-1]}/$new_entry/g"  |sed "${index}s/
/;/g">new1.txt && mv new1.txt user_database.csv
              success_msg $1 $2 $3 $4 $5 $6       ;;            bash index_page.sh
```

```
*)              clear
                    echo -en "\n$red[ERROR]$normal INVALID CHOICE
$red[ERROR]$normal\n"
          bash edit_page.sh $1 $2 $3 $4 $5 $6  esac  else
                              echo -en "Sign-in or Sign-up to edit the information"
          bash index_page.sh fi
```

### vii. test_screen.sh

```
#!/bin/bash
#imports the required files source color.sh

#validates the sign_in from the user if [ $# -eq 6 ]
then
#prompts the user for Instruction before Test and the user-input for start the Test
          echo -e "\n$green$bold                [SuccessFull Logged-IN]                $normal"
                              echo -e "$bold$blue                    INSTRUCTION
$normal"
                    echo -e "$green
$normal"
                    echo -e "$green
$normal"
                    echo -e "$green
$normal"
                         echo -e "$green$bold--> Each Question Carries 1 mark
$normal"
                       echo -e "$green$bold--> No Negative Marks
$normal"
                            echo -e "$green$bold--> Each Question Has 30 Seconds To answer
$normal"          echo -e "$green$bold--> Un-Answered Question will Automatically marked as 0
marks          $normal"
                              echo -e "$green$bold--> Time will be Started as you Enter [Y]
$normal"
                            echo -e "$green$bold--> To Exit from [TEST] Please Enter [N]
$normal"
      echo -e "$green
```

```
$normal"            echo -e "$green                    $normal$blue$bold ALL THE BEST
!!!$normal$green                    $normal"
          echo -e "$green                                          $normal"

#prompt the user for the Choice
                    echo -ne "$bold[Enter Your Choice(Y/N)]:$normal"
                read  test_input

#validates the choice
                 case $test_input in

                "Y")  clear
                              bash test.sh $1 $2 $3 $4 $5 $6;;
                "N")  clear
                              bash profile_page.sh $1 $2 $3 $4 $5 $6
                           exit ;;
             *)             clear
                              echo -e "\n\n$red[ERROR]$normal Invalid Input
$red[ERROR]$normal\n\n"
                               bash test_screen.sh  $1 $2 $3 $4 $5 $6;;

             esac
else
              clear
                    echo -e "\n\n$red[ERROR]$normal Please Log-In
$red[ERROR]$normal\n\n"
          bash index_page.sh fi
```

### viii. test.sh

```
#!/bin/bash
#import the required files source color.sh

function time1()
{
for((i=30; i>=1; i--)) do      echo -ne " :$blue$bold[Enter your Option]$normal"
echo -ne "
$rg_green[$normal$bold$i$normal$rg_green]Seconds-
```

```
Remaining$normal\033[0K\r"
                sleep 1
 done

}



#This Fuction prints the test_result taken by the user function test_result()
{




#imports the required files  source color.sh



 #validates for user-info and user-score after the test is taken  if [ $#  ]  then

 #page-layout
        echo -e "\n$green$bold                  [Test-SuccessFully Completed]                    $normal"
        echo -e "$bold$blue                        Result
$normal"
        echo -e "$green
$normal"
        echo -en "$green
$rg_red Welcome  $normal"          echo -en "$bold
$1$normal\n"
        echo -e "$green
$normal"
        echo -e "$green$bold   --> Your Total Score is: $7
$normal"
        echo -e "$green
$normal"
        echo -e "$green                                              $normal"
        echo -e "$green
```

```
$normal"
        echo -e "$green$bold   *Enter [Y] to retake the Test
$normal"
        echo -e "$green$bold   *Enter [P] to go to Profile Page
$normal"
        echo -e "$green$bold   *Enter [V] to View The Test
$normal"
        echo -e "$green                    $normal$blue$bold Congratulations !!!$normal$green
$normal"
        echo -e "$green                                              $normal"

 #prompts the user for the choice         echo -ne "[Enter Your
Option]:"          read  user_input

#validate the user for the input          case $user_input
in

       "Y")    clear
               bash test.sh $1 $2 $3 $4 $5 $6;;
       "P")    clear
               bash profile_page.sh $1 $2 $3 $4 $5 $6;;
       "V")    clear
               view_test $1 $2 $3 $4 $5 $6;;

       *)      clear
               echo -e "\n$red[ERROR]$normal INVALID OPTION
$red[ERROR]$normal\n\n"
               test_result $1 $2 $3 $4 $5 $6 $7 ;;
        esac else
clear
       echo -e  "\n\n$bold Please Sign-in or Log-in to Take a Test
$normal"
       bash index_page.sh  fi
}


#this function prints the question and user answer with the correct answers
```

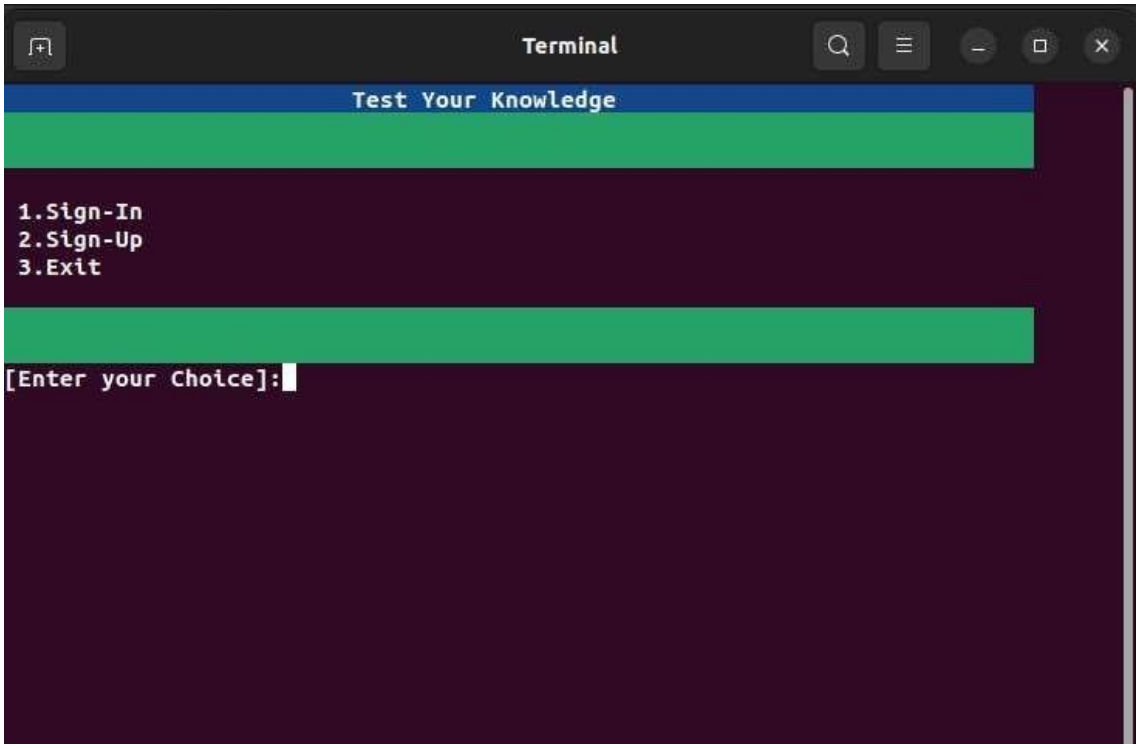
```
function view_test()
{

for((i=0; i<10; i++)) do
                clear
        echo -e "\n$blue$bold                        [View-Test]
$normal"
        echo -e "$green
$normal"
        echo -e "$green                                                $normal"
        echo -e "$green
$normal"            echo -e "\n$(($i+1)))${questions[i]}"          echo -e
"\n$blue$bold    your Answer$normal:
${user_answer_for_rand[i]}"          echo -e "$green$bold correct
Answer$normal:
${evaluation_sheet_for_rand[i]}\n"
                echo -e "$green
$normal"
        echo -e "$green
$normal"
              read n
done
clear
bash profile_page.sh $1 $2 $3 $4 $5 $6


}


#validates if the user-info is present if [ $1 ] then

#access the  total-count-of-questions
          count_of_question=(`ls ./QUESTION_BANK | grep -i -regexp=.*\.txt | wc -l`)
              count=1
                total_marks=0
```

```
#random numbers are generated from the count of the questions (only ten questions are generated) and
the correct answers are also accessed
        rand=(`shuf -i 1-$count_of_question -n  10`)             evaluation_file=(`cat
./QUESTION_BANK/answer.csv`)
                        evaluation_sheet=(` sed 's/;/ /g'   ./QUESTION_BANK/answer.csv
`)

                for(( i=0; i<10; i++))
            do
#page-layout
                        clear

                        rearranged_evaluation_sheet=${evaluation_sheet[${rand[i]}-1]}

                                        echo -e "\n$blue$bold                    [ Test
]                       $normal"
                      echo -e "$green
$normal"
                      echo -e "$green
$normal"
                      echo -e ""
                      echo -e "$count)`cat
./QUESTION_BANK/question${rand[i]}.txt`"
                        echo -e ""
        echo -e "$green                                  $normal"
                        echo -e "$green
$normal"
                        count=$(($count+1))

                      time1 &
                          read -t 30   user_answer
#validates the user-input with the answer
                          if [ $user_answer ]
                    then
kill $!                   clear
                                  if [ $user_answer ==
${evaluation_sheet[${rand[i]}-1]} ]
```

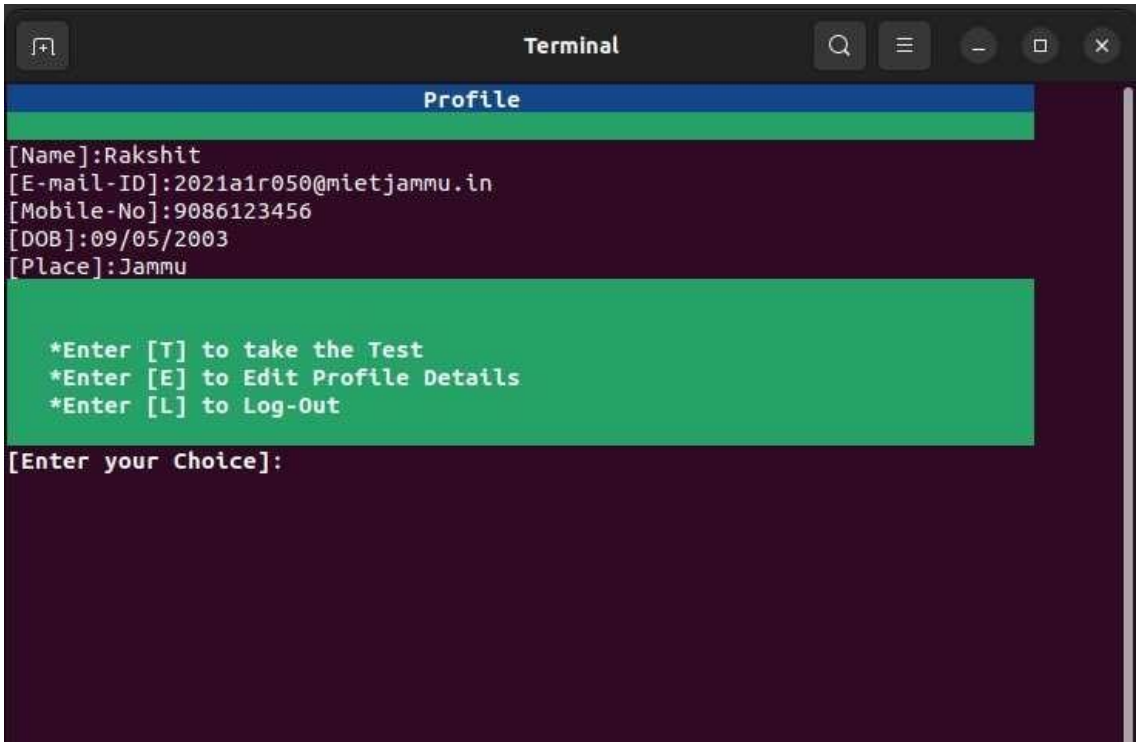
```
                                then
                                        total_marks=$(($total_marks+1))

                                fi
                        fi

                questions[i]=`cat ./QUESTION_BANK/question${rand[i]}.txt`
user_answer_for_rand[i]=$user_answer

evaluation_sheet_for_rand[i]=${evaluation_sheet[${rand[i]}-1]}
        done

        clear
                test_result $1 $2 $3 $4 $5 $6 $total_marks
#                       bash  test_result.sh  $1 $2 $3 $4 $5 $6 $total_marks $questions
$user_answer_for_randi $evaluation_sheet_for_randi

else
        clear
                echo -e "\n$red[ERROR]$normal Please Sign-in [Test-Software]
$red[ERROR]$normal\n"
          bash index_page.sh
fi
```

# b. Output

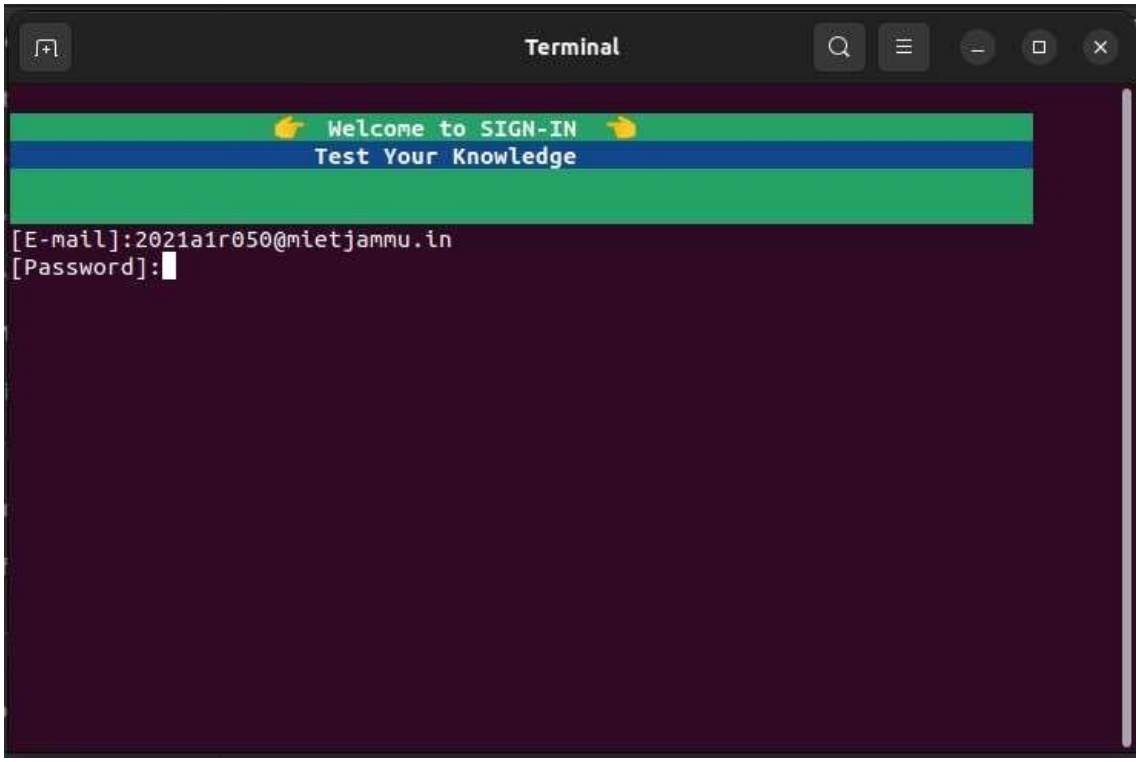
i. Index Page



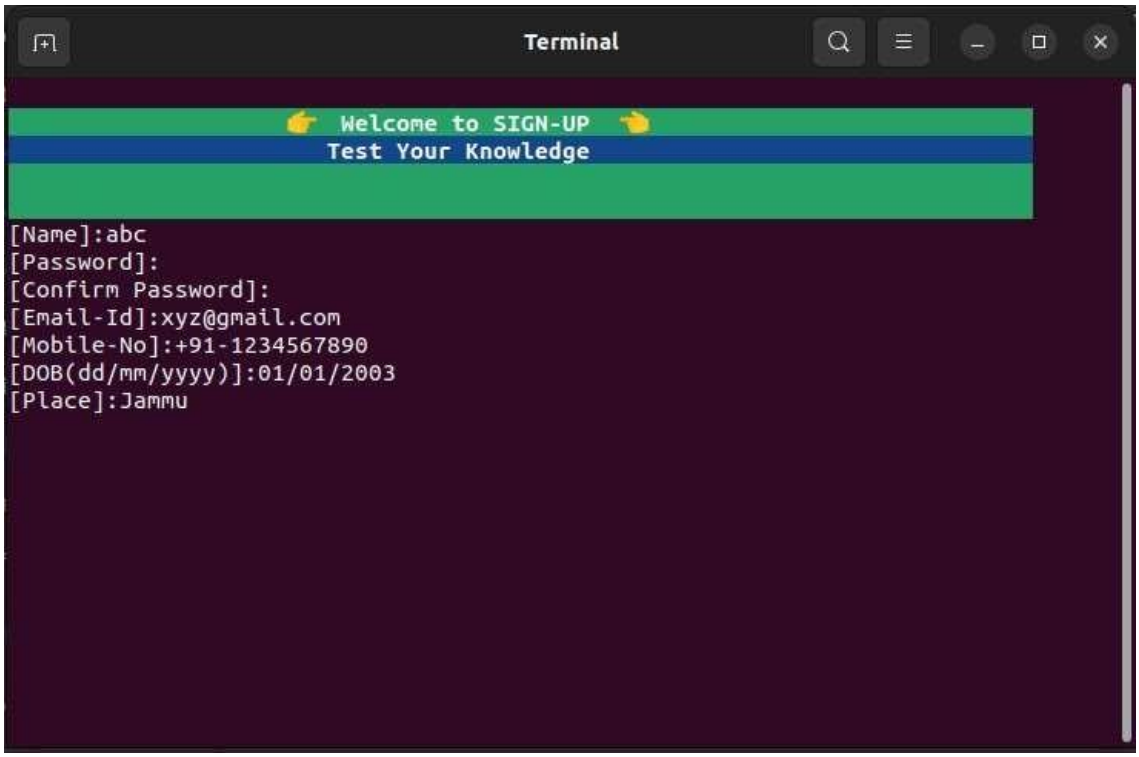

ii. Profile Page



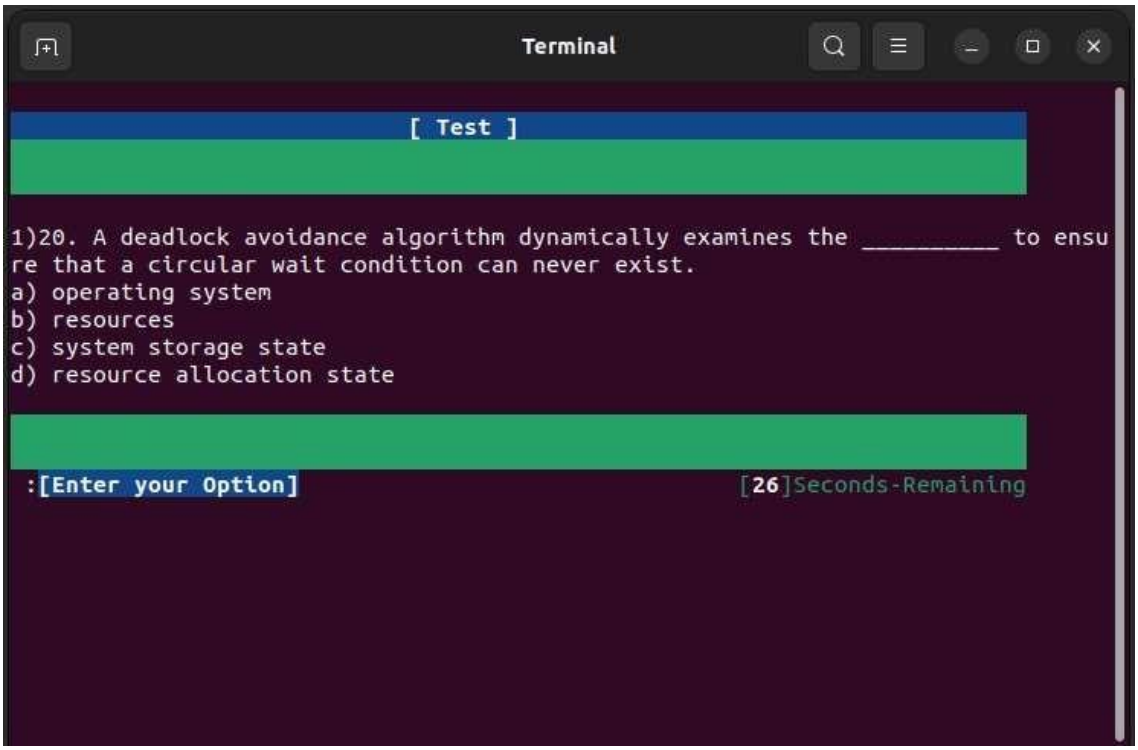
iii. Sign In Page



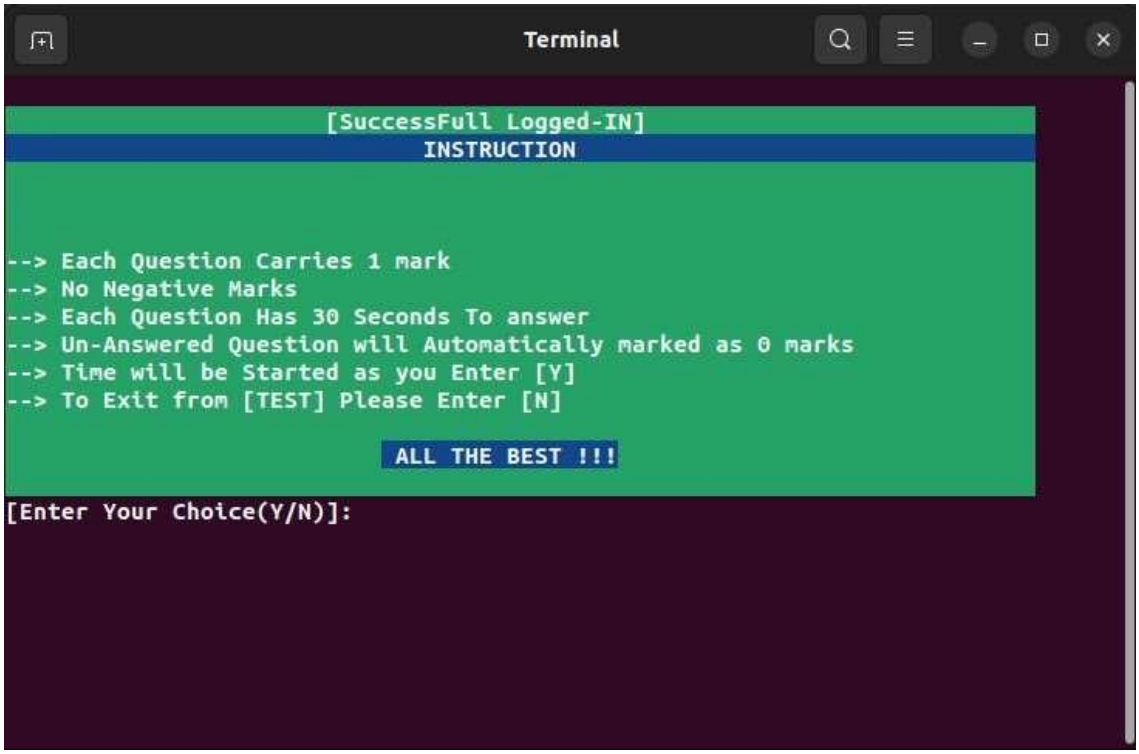

iv. Sign Up Page



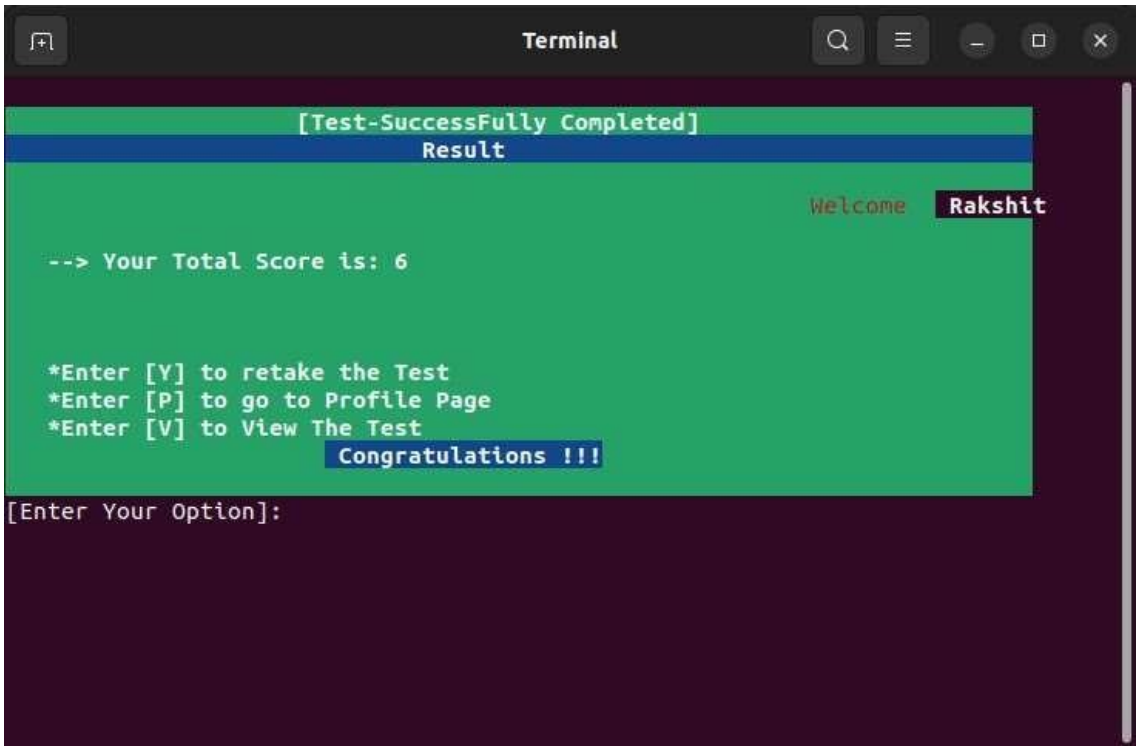

v. Test Page



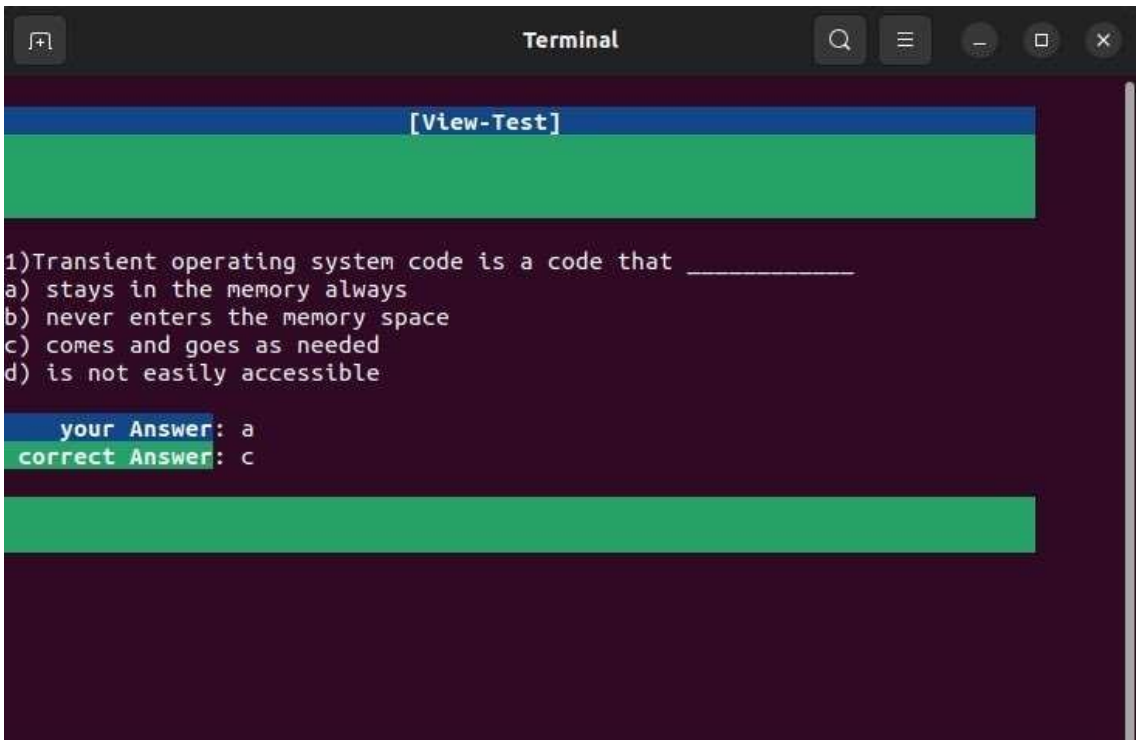

vi. Test Screen



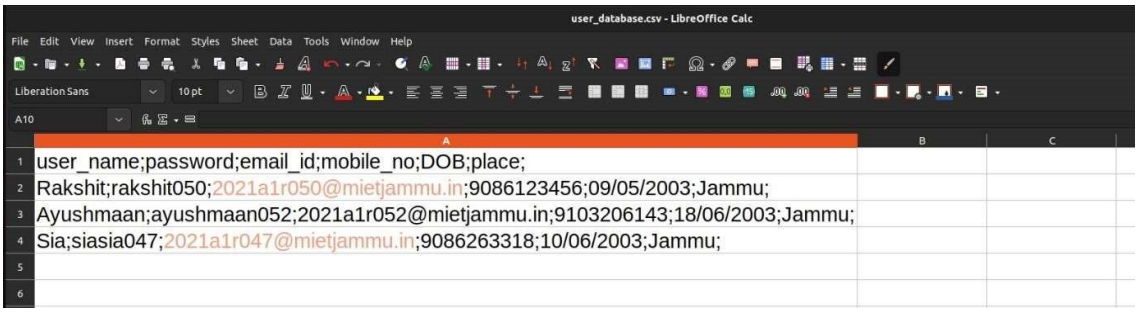

vii. Result Page




viii. View Test




ix. User Database




# 7. References

1. Geeksforgeeks

2. allproject4u

3. Mr. Saurabh Sharma (Assistant Professor, MIET)