

*A project report on,*

**“ A BASIC FILE AUTOMATED SYSTEM ”**

*in shell scripting*

**SUBMITTED TO:**

Ms. Deepti

of

***DEPARTMENT of INFORMATION TECHNOLOGY ,  
NATIONAL INSTITUTE OF TECHNOLOGY  
SURATHKAL, KARNATAKA***

**PRESENTED BY :**

Uttam Kumar (16IT250)

Tarun Pratap Singh (16IT143)

Sulabh Tembhurkar (16IT245)

Arjun Kumar (16IT108)

# DECLARATION

We hereby declare that the project work entitled  
    **“A BASIC FILE AUTOMATED SYSTEM”**  
submitted to the **NITK SURATHKAL** , is a record of an  
original work done by us under the guidance of **Ms. Deepti**  
our respected course lecturer of **UNIX**, and this project  
work is submitted in the partial fulfillment of  
the requirements for the completion of **UNIX course** in the  
field of **INFORMATION TECHNOLOGY**. The results  
embodied in this project are all Original.

# ABSTRACT

**Title** : **A Basic Automated System**

## **System Specification:**

The system on which the project developed has the following configuration.

## **Software Specifications:**

Operating System	:	UBUNTU L.T.S 16.04
Reference	:	<a href="http://www.techmint.com">www.techmint.com</a>

## **Hardware Specifications:**

Main Memory	:	8 GB
MicroProcessor	:	Intel
Hard Disk Drive	:	20 GB
Cache Memory	:	512KB.

## **Guidance**

Name	:	Ms Deepti ( UNIX course lecturer)
Branch	:	Information Technology
Institution	:	National Institute Of Technology, Karnataka

## **Project Overview:**

Managing a large amount of data in the present scenario is a huge problem. This is needed for every one. Its better to have a automated system to manage data and keep it protected.

This project deals with general purpose data handling in Linux and other Operating System. The project has a simple basic understanding of shell scripting , and is implemented to protect and manage data.

A basic GUI is implemented to make this project presentable to the user.

## **Advantages**

- Get clear Information
- Time consuming
- Easy to reach destination
- Secures a data file
- Manages data precisely

# TABLE OF CONTENTS

Contents		PageNumber
1.) List Of Figures	--	1-1
2.) Introduction	--	2-2
2.1) Challenges	--	2
2.2) Motivation	--	2
3.) Methodology & Framework	--	3-3
3.1) SystemRequirement	--	3
3.2) Algorithms&Techniques	--	3
4.) Implementation	--	4-6
4.1) ListOfMain Unix Cmnds	--	4
4.2) Syntax of commands	--	5,6
5.) Result and Analysis	--	7-12
6.) Conclusion&References	--	13



# LIST OF FIGURES

Name of Figure	PageNumber
1.) Main-Menu --	7
2.) SystemInformation --	7
3.) ImageManager(input) --	8
4.) ImageManager(output) --	8
5.) SourceFolder(for image manager) --	9
6.) DestinationFolder(for image manager) --	9
7.) Encryption(input) --	10
8.) Encryption(output) --	10
9.) Sample.txt(file input) --	11
10.) Sample.txt.gpg(encrypted file output) --	12
11.) Decryption(input) --	12
12.) Decryption(output) --	13
13.) Sample.txt(decrypted file) --	13



# AN INTRODUCTION TO THE PROJECT

## CHALLENGES :

Data can come from many sources: internally within the organization or externally from data vendors. This creates multi-faceted datasets with each stream of data supplying its own set of standards, including different formats, acceptable ranges, and granularities . Handling multi-faceted data with different standards in an efficient manner has become a major challenge in our fiercely competitive world. However, it gets even more complicated when the required data doesn't get fetched when it is supposed to, creating a time lag. For these reasons, data automation is essential as it allows data to be captured in a timely fashion with a minimum amount of resources.

Automation is a more reliable way of managing data when compared to manual process, since humans are inherently error-prone.

Along with data management comes data protection and saving it from getting corrupted by any external or internal means. So this project includes a way basic program to handle basic regular use data and its encryption and decryption. It saves the user from revising the data from time to time.

This is what lead us to think for this Project and implement it using real life examples.

## MOTIVATION FOR THE WORK :

Well Being a student we too have many issues related to managing data. such as image handling and other documents handling. Many a times we need a sorted format of data for our use. Once if a data is sorted it saves much of our time in doing works related to that data in our daily life.

The final goal is to feed the collected data into trade and risk applications. Hence it made us to work on this project so that we could solve our own issues along with learning something new. However, it gets even more complicated when the required data doesn't get fetched when it is supposed to, creating a time lag. For these reasons, data automation is essential as it allows data to be captured in a timely fashion with a minimum amount of resources, an experienced data management platform along with basic data automation is the only viable solution.





# METHODOLOGY AND FRAMEWORK

## SYSTEM REQUIREMENT :

The system on which the project developed has the following configuration..

### Software Specifications:

Operating System	:	UBUNTU L.T.S 16.04
Reference	:	<a href="http://www.techmint.com">www.techmint.com</a>

### Hardware Specifications:

Main Memory	:	8 GB
MicroProcessor	:	Intel
Hard Disk Drive	:	20 GB
Cache Memory	:	512KB.

## ALGORITHMS AND TECHNIQUES USED :

Algorithms are used for calculation, data processing, and automated reasoning.” Whether you are aware of it or not, algorithms are becoming a ubiquitous part of our lives. Here in this project we have used a basic shell scripting concept in order to implement automation in data management and it works in an interactive mode with the user. Exceptions while using the source code have been managed in order to avoid any adverse condition to the user. Automation is a more reliable way of managing data when compared to manual process, since humans are inherently error-prone. File fetching and handling is one of the most powerful features of UNIX , hence this project implements and shows a smart use of file manipulation, in management of data. We have used some extra techniques of getting source information of the file and using it in sorting them and placing them in the suitable folder. The final goal is to feed the collected data into trade and risk applications. Hence it made us to work on this project so that we could solve our own issues along with learning something new. However, it gets even more complicated when the required data doesn't get fetched when it is supposed to, creating a time lag.

# IMPLEMENTATION

## MAIN UNIX COMMANDS USED :

**tr (translate) command ----** is a command in UNIX operating systems. It is an abbreviation of *translate* or *transliterate*, indicating its operation of replacing or removing specific characters in its input data set.

**sed (stream editor) command ----** is a UNIX utility that parses and transforms text, using a simple, compact programming language.

**echo (for output) command ----** is in DOS , OS/2, Microsoft Windows, Unix and Unix-like operating systems that outputs the strings it is being passed as arguments. It is a command typically used in shell-script and batch files to output status text to the screen or a file.

**which (for checking) command ----** is a Unix command used to identify the location of executables. Returns pathname of the file

**stat (system call) command ----** is a Unix system call that returns file attributes about an inode.

**identify (for image) command ----** program is a member of the ImageMagick(1) suite of tools. It describes the format and characteristics of one or more image files. It also reports if an image is incomplete or corrupt.

**find (for search) command ----** is a command-line-utility that searches one or more directory trees of a file system, locates files .

**cut (to extract) command ----** is a Unix command line utility which is used to extract sections from each line of input, usually from a file.

**basename (for extracting name) command ----** is a standard UNIX computer program. When basename is given a pathname, it will delete any prefix up to the last slash (' / ') character and return the result.

**hostnamectl (for host name) command ----** may be used to query and change the system hostname and related settings.

**gpg (encryption) command ----** perform a reasonable action depending on the type of file it is given as input (an encrypted message is decrypted, a signature is verified, a file containing keys is listed).

## SYNTAX OF THE COMMANDS :

**tr (translate) command ----** \$ tr [OPTION] SET1 [SET2]

**sed (stream editor) command ----**

\$ sed OPTIONS... [SCRIPT] [INPUTFILE...]

**echo (for output) command ----**

\$ echo [SHORT-OPTION]... [STRING...]

**which (for checking) command ----** \$ which -a *[filename]* ...

**stat (system call) command ----** \$ stat [OPTION]... *FILE*...

**identify (for image) command ----**

\$ identify [options].... *Input\_file*....

**find (for search) command ----**

\$ find path... [expression]

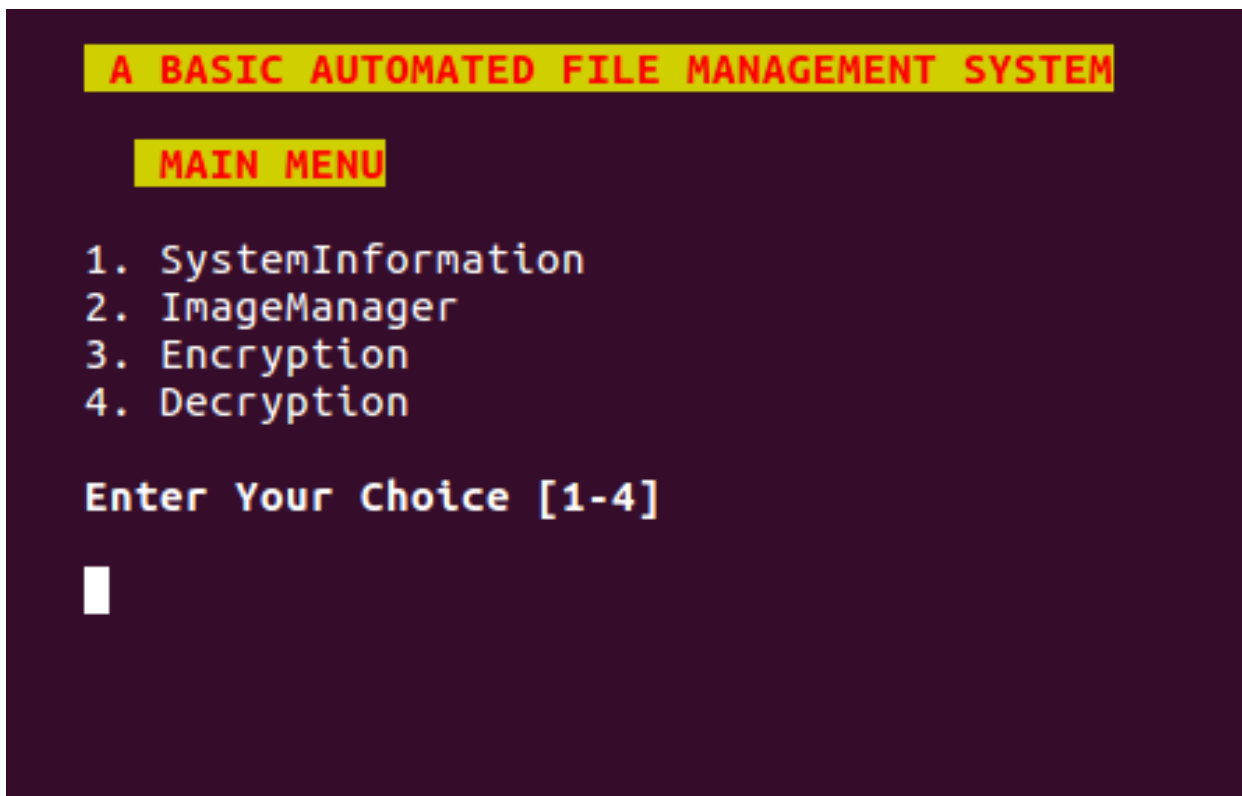
**gpg (encryption) command ----**

\$ gpg [--options file] [options] command

**cut (to extract) command ----** \$ cut [OPTION]... [FILE]...

# RESULTS AND ANALYSIS

1.) The execution code start within an user-interactive interface asking for input from the user for what to proceed with.

A screenshot of a terminal window with a dark purple background. At the top, a yellow banner contains the text "A BASIC AUTOMATED FILE MANAGEMENT SYSTEM" in red, uppercase letters. Below this, another yellow banner contains the text "MAIN MENU" in red, uppercase letters. Under the banner, a list of four options is displayed in white text: "1. SystemInformation", "2. ImageManager", "3. Encryption", and "4. Decryption". Below the list, the prompt "Enter Your Choice [1-4]" is shown in white text. At the bottom left, there is a small white rectangular cursor indicating where the user should enter their choice.

```
A BASIC AUTOMATED FILE MANAGEMENT SYSTEM

MAIN MENU

1. SystemInformation
2. ImageManager
3. Encryption
4. Decryption

Enter Your Choice [1-4]

█
```

2.) Prompts the user enter keys for specified actions. Suppose if user enters 1 then he enters in a non interactive interface displaying available information from the system.

```

HOSTNAME INFORMATION

Static hostname: airgaz-Inspiron-5555
Icon name: computer-laptop
Chassis: laptop
Machine ID: 4e48fe8c52494c8fa9265a3ae1883d77
Boot ID: 32d1dfd1d1c742c5853cbb5808729762
Operating System: Ubuntu 16.04.3 LTS
Kernel: Linux 4.10.0-40-generic
Architecture: x86-64

FILE SYSTEM DISK SPACE USAGE

Filesystem      Size  Used Avail Use% Mounted on
udev            3.4G     0   3.4G   0% /dev
tmpfs           699M    27M   673M   4% /run
/dev/sda5       15G   8.7G   5.1G  64% /
tmpfs           3.5G    36M   3.4G   2% /dev/shm
tmpfs           5.0M    4.0K   5.0M   1% /run/lock
tmpfs           3.5G     0   3.5G   0% /sys/fs/cgroup
tmpfs           699M   100K   699M   1% /run/user/1000

FREE AND USED MEMORY

Mem:           total      used      free      shared  buff/cache  available
Swap:              0          0          0          0           0           0

SYSTEM UPTIME AND LOAD

00:10:16 up 1 day, 11:34,  1 user,  load average: 0.79, 0.64, 0.57

CURRENTLY LOGGED-IN USERS

airgaz  tty7          2017-11-23 18:46 (:0)

TOP 5 MEMORY-CONSUMING processes

%MEM %CPU COMMAND
 5.3  3.8 soffice.bin
 5.2  4.9 firefox
 5.1  2.2 Web Content
 3.6  0.8 Web Content
 3.6  5.4 Web Content

CHECKING FILE SYSTEM USAGE

The remaining available space in /dev/sda5 is critically low. Used: 64%
The remaining available space in total is critically low. Used: 35%

Done.

Do You Wish To Continue ? (1.(Yes) 2.(No))

```

This interface provides an complete information about system.

Well when one of the inputs is entered for “yes” or “no” then the execution returns to the main menu ,Providing the user for further continuation.

3.) when user enters the other option specified by there key.  
Option 2 specifes for image manager,this takes the user to an interactive interface.

```

Welcome To ImageManager

Enter The Name Of The Source Folder
imagesource

Enter The Name Of The Destination Folder
picture

Initiating Process

```

It asks the user to enter the source from where images are to be taken and sorted on the basis of their data such as date .Then it asks for the destination folder where the data are to be stored.

It initiates the process of image managing and takes the user to non-interactive mode where it updates the user with the processes happening in the background.

```

Welcome To ImageManager

Enter The Name Of The Source Folder
imagesource

Enter The Name Of The Destination Folder
Pictures

Initiating Process

Welcome To ImageManager
Image[/home/airgaz/imagesource/Screenshot from 2017-11-15 18-18-09.png] processed
Image[/home/airgaz/imagesource/Screenshot from 2017-11-15 18-18-07.png] processed
Image[/home/airgaz/imagesource/Screenshot from 2017-11-15 18-18-05.png] processed
Image[/home/airgaz/imagesource/874960.jpg] processed
Mission Succesfully finished :->

Do You Want To Remove The Left Empty Folder ? [ 1.(Yes) or 2.(No) ]

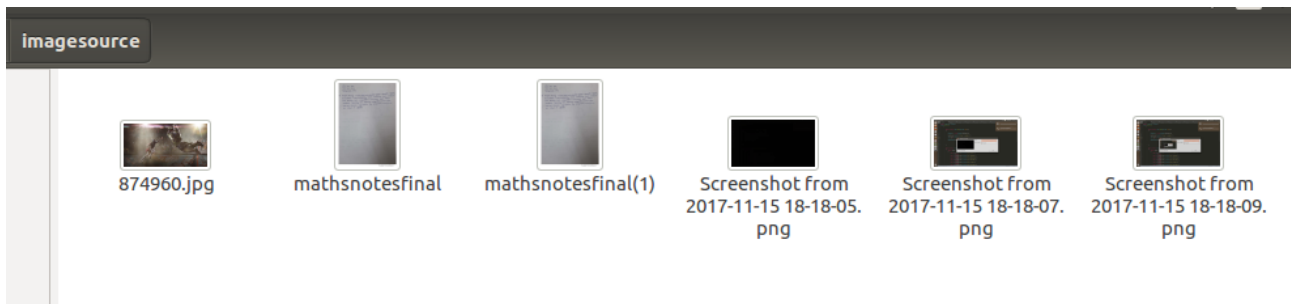
```

Asks the user wheather to remove the left empty folder or not.



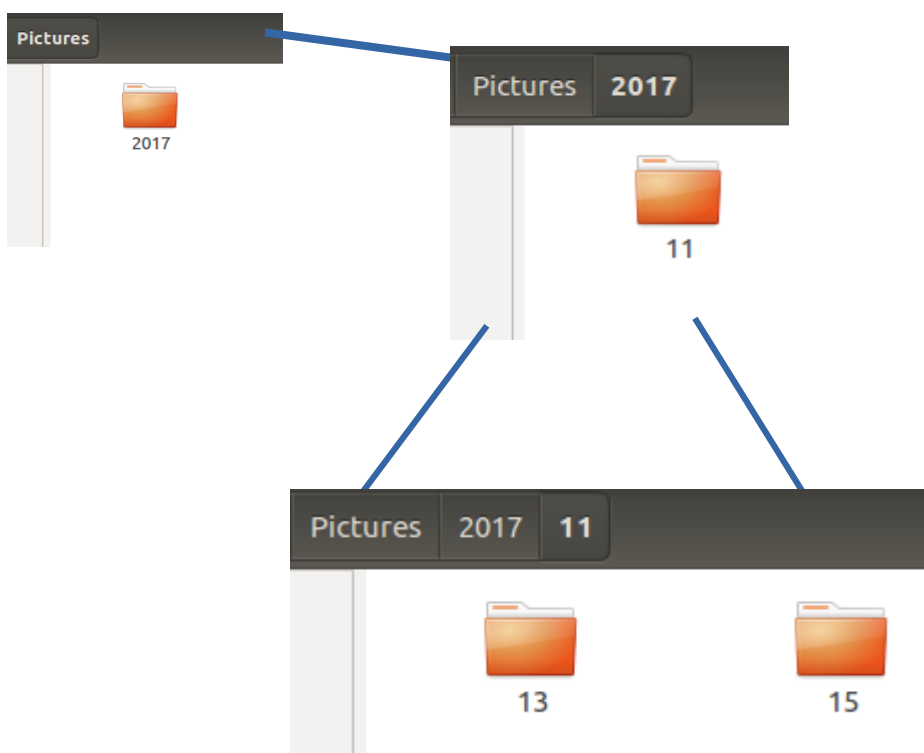
Then after completion of the process we will have a glance at the **source folder** and the **destination folder** .

**SOURCE FOLDER :** \$HOME/airgaz/imagesource is the path of the source folder,before manipulation it had the input images to be given,and looked like somewhat this.



after the completion of the execution the source folder files are moved to the destination folder.

**DESTINATION FOLDER :** \$HOME/airgaz/Pictures is the path to the destination folder, hence we get all the files sorted in different folders according to there data in the image in the suitable folder.



Once the execution gets completed the code returns to the main menu and asks the user for further inputs.

4.) when the user enters the **3<sup>rd</sup>** option the code returns a menu asking for **directories** where the file that is to be **encrypted** is present. Here if the user enters the wrong **Directory code** it **doesn't terminate** and runs until the user enters the correct directory, hence this way an exception is handled in the code. Here when the user enters the correct filename along with extensions it asks for passphrase from the user for securing filename **sample.txt** after that, it asks for the final destination where the encrypted data is to be stored and removes the original file.

```
Enter a Valid Directory Name. . . . .
***** Welcome To The File Encrypter *****

Enter The Directory Where The File Is Present :
unixproject/decrypted

Enter the Exact File Name with extension
sample.txt

encrypting file. . . . .
Enter passphrase: █

Enter a Valid Directory Name. . . . .
***** Welcome To The File Encrypter *****

Enter The Directory Where The File Is Present :
unixproject/decrypted

Enter the Exact File Name with extension
sample.txt

encrypting file. . . . .

File Encrypted Succesfully

Now removing the original file. . . . .

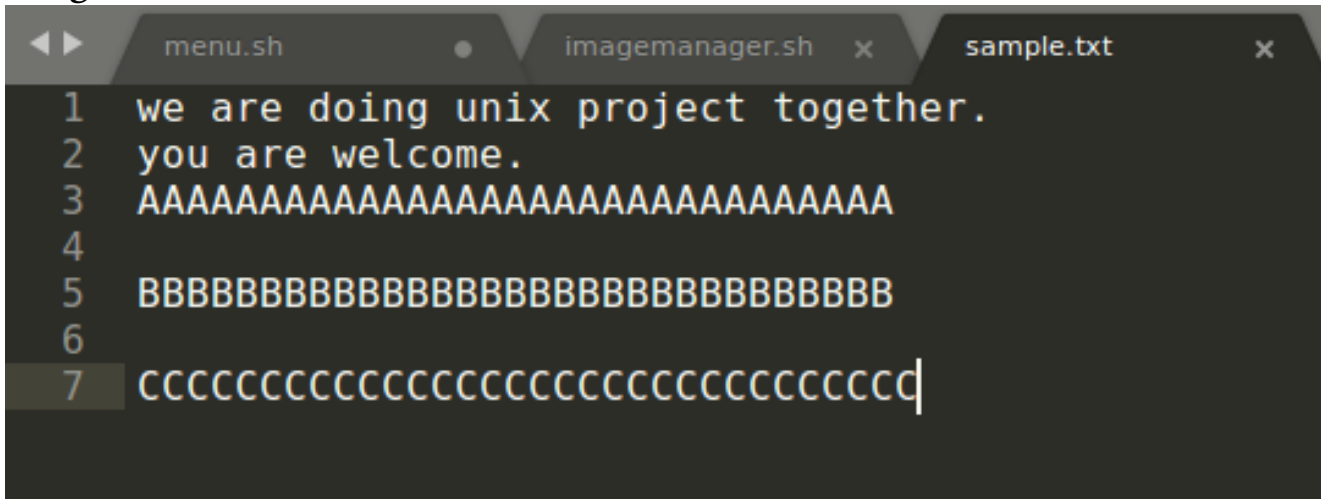
Enter the Location where File Is To be Stored
decrypted

Moving The Encrypted File To decrypted Folder . . . . .
***** Task Completed *****

Do You Wish To Continue ? (1.(Yes) 2.(No)
█
```

Once after the complete execution of the program we will check the result wheather the file has been encrypted or not.

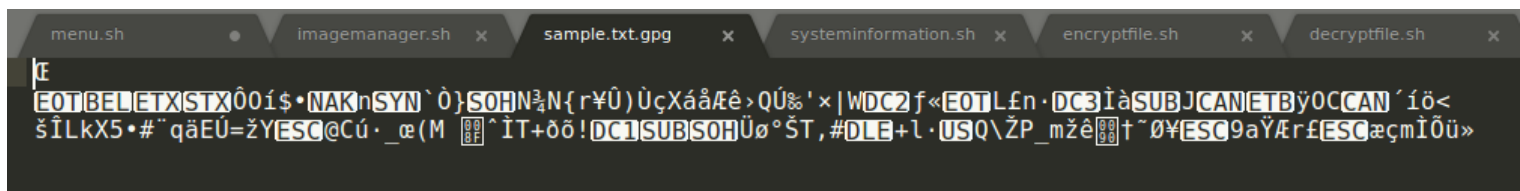
**Sample.txt** : this was the sample file which we encrypted,hence the original data file is.....



```
1 we are doing unix project together.
2 you are welcome.
3 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
4
5 BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
6
7 CCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

we have the encrypted file named as

**Sample.txt.gpg** : this is the encrypted file which cant be accesed without the passphrase.



```
EOTBELUETXSTX00i$•NAKhnSYN`0}SOHN2N{r¥Ü)ÛçXáâÆê>QÚ%`x|WDC2f«EOTLfn•DC3IàSUBJCANETBÿOCCAN`íö<
šÎLkX5•#"qäEU=žYESC@Cú•_æ(M [^IT+ôô!DCTSUBSOHÜø°ŠT,#DLB+L•USQ\ŽP_mžê[†~Ø¥ESC9aŸ&rſESCæçmİÖü»
```

hence, the encrypted file is stored in the folder specified by the user, hence the essence of encryption is achieved.

After the execution of this program segment , the program returns to the main menu,hence the user selects wheather to go further r to stop there.

Once it goes to the main menu ,user enters the choice from the main menu.

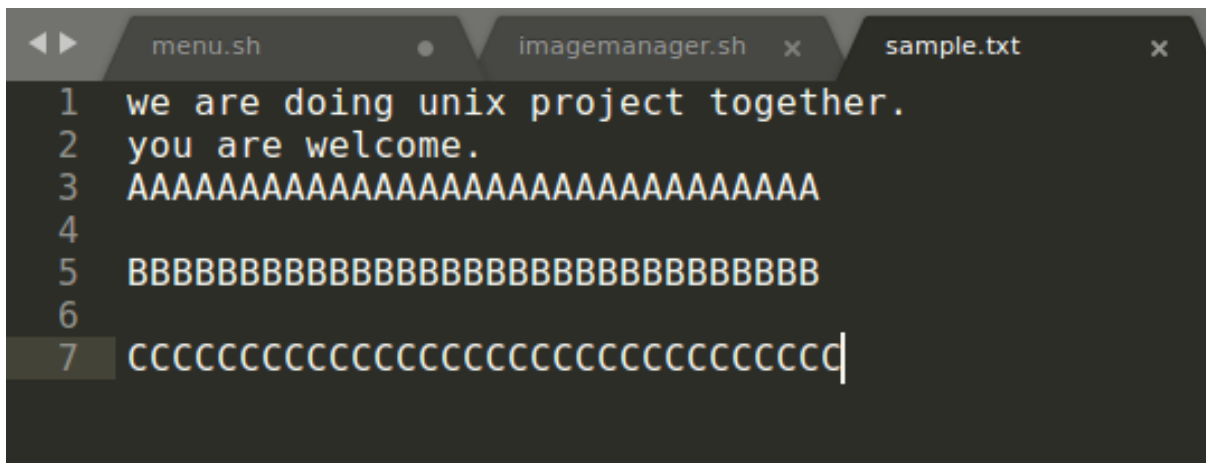
5.) This choice takes the user to the code segment where the encrypted file can be decrypted back once the user has acces to the password .here an interface appears in fron of the user and asks for the source file path and if a valid path is entered it moves further asking for the file to be decrypted.....

```
***** Welcome To The File Decrypter *****
Enter The Directory Where The File Is Present :
encrypted
Enter the Exact File Name with extension( example; filename.extension.gpg )
sample.txt.gpg
Decrypting file. . . . .
Enter the Name For The New File
█
```

it asks the user to enter the name of the new file to be created for the decryption data to be stored.

Once execution finalizes it asks for the **passphrase** which is used when encrypting the file.

Hence the decrypted file looks like the normal file accesible by the user.

A screenshot of a terminal window with three tabs: 'menu.sh', 'imagemanager.sh', and 'sample.txt'. The 'menu.sh' tab is active. The terminal displays a script with seven lines of code. Line 1: 'we are doing unix project together.' Line 2: 'you are welcome.' Line 3: 'AAAAAAAAAAAAAAAAAAAAAAAAAAAAA' Line 4: (empty) Line 5: 'BBBBBBBBBBBBBBBBBBBBBBBBBBBBBB' Line 6: (empty) Line 7: 'CCCCCCCCCCCCCCCCCCCCCCCCCCCCC' followed by a cursor. The lines are numbered 1 through 7 on the left side of the terminal.

```
1 we are doing unix project together.  
2 you are welcome.  
3 AAAAAAAAAAAAAAAAAAAAAAAAAAAAA  
4  
5 BBBBBBBBBBBBBBBBBBBBBBBBBBBB  
6  
7 CCCCCCCCCCCCCCCCCCCCCCCCCCCC|
```

Once the execution finishes the code segment again returns to the main menu and asks for further inputs.

# CONCLUSION

We Thought of managing data and storing it safely by any means of automated data management. And we were succesfully able to manage our data with the source code written by our group. Hence our project at a basic level can help managing data in the course of daily life and would make data(here images) easily accesible to the user at not at the cost of security of the data. Hence unix provides us with enough of the commands to store,manipulate and encrypt our personal data easily.The project developed a better understanding of the concepts in UNIX and surely made us curious to go further in this project.

## REFERENCES :

- 1.) YOUR UNIX by SUMITABHA DAS
- 2.) Study materials from [www.techmint.com](http://www.techmint.com)
- 3.) *YouTube* video tutorials on unix

