INTRODUCTION

The following project is based on ‘Seg2 Tape’. this Tape is a device where

Data can be written or rewound depending upon the need of the user.

PROJECT DESCRIPTION

This a project that develops a portal for executing tapes and drivers using binary files and bash scripts.

The data in the company I am interning is stored in the form of cartridges instead of hard disks.

A cardtrige is a data storing device where data can be stored sequentially.The data is accessed through a pointer of a driver that traverses

the cartridge consisting of various tapes. User authentication has been implemented using LDAP.

By using pipes and subprocess feature in Python, the GUI(developed with tkinter) will show the live status of the command

being executed and the user can even interract with the binary file output in real time using pipes.

**Feasibility Study**:

* Economic feasibility:

As per the requirements, the software to be developed requires

no additional installations of hardware components. Therefore,

The project is economically feasible.

* Technical feasibility:

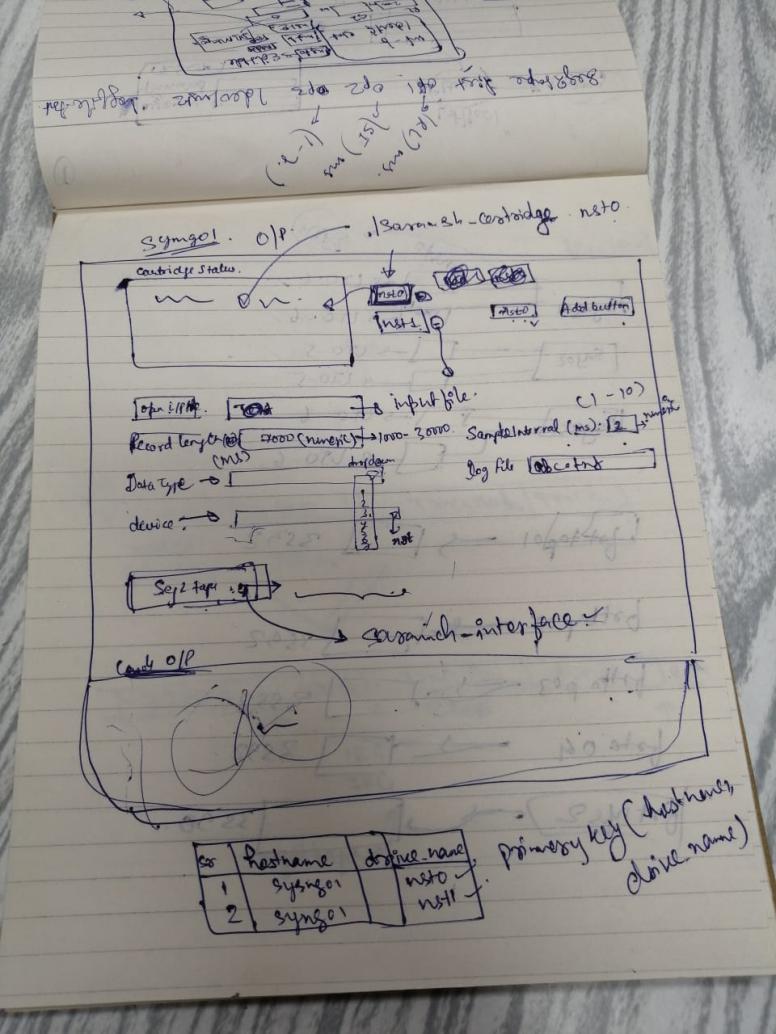
There are high-end workstations in the organization. So, there is no need to make use of optimized codes in order to make the system work. The software requires us to interact with the linux terminal on a regular basis and display the live output via a GUI.

Thus, python language can be used for coding as it provides us with all these functionalities.

REQUIREMENT ANALYSIS AND SPECIFICATION

1. Requirement analysis:

Requirement provided by the customer:



ANALYSIS:

As soon as the application is opened, the user must be authenticated first. After the authentication, the GUI must be visible where the corresponding drives of the concerned server will be available. As the drives are chosen, the corresponding cartridge number must be updated that will be written through the software.

The user must also be able to view the state of the driver used and the contents of the input file chosen.

Requirement Specification:(SRS DOCUMENT)

#### Intended Audience and Intended Use:

The software will be used by the Geo-physicists of the organization to write the data in the in the tape and view the results.

#### Product Scope:

The software is built in such a way that it can be understood by both people with technical and non technical background. The product can further be enhanced by making the code more optimized so that not much strain is put on the workstations.

* Functional requirements:

1. User can view the state of the cartridge
2. User should be able to add or remove drivers
3. User should be able to rewind the tape
4. User should see the live output of operations
5. User must be authenticated before he or she can

use the software.

1. A database should be maintained to keep track of

all the operations performed by a user.

* Non-Functional requirements:

1. The user should be able to understand the GUI easily
2. The software must be portable and reliable
3. The software should be efficient both performance as well

as space wise

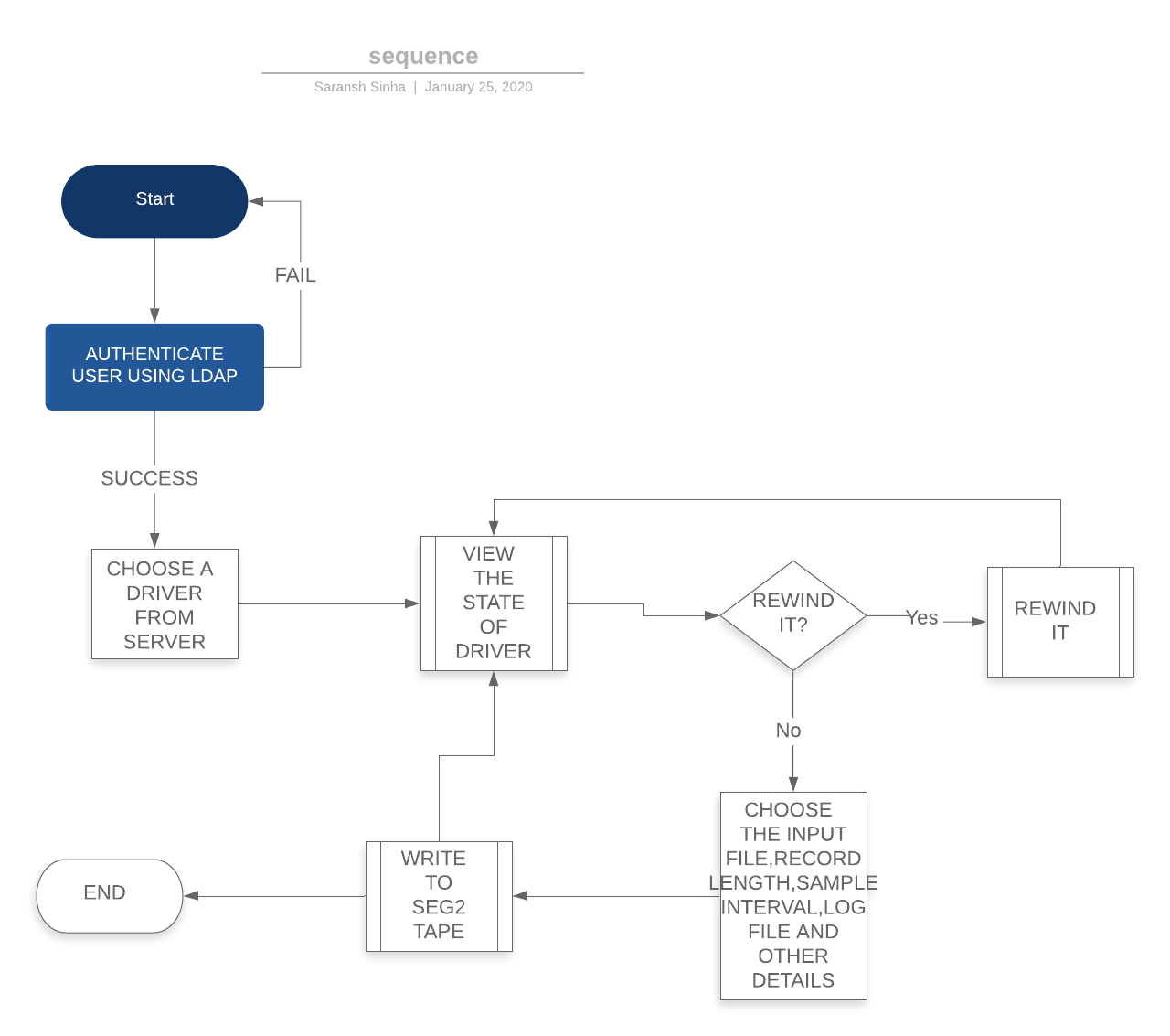
1. Organization’s ethical and legislative standards must not be violated
2. The software should exhibit interoperabilty and efficient delivery of results.
3. The software must meet the required Company’s standards.

* Interface Requirements:

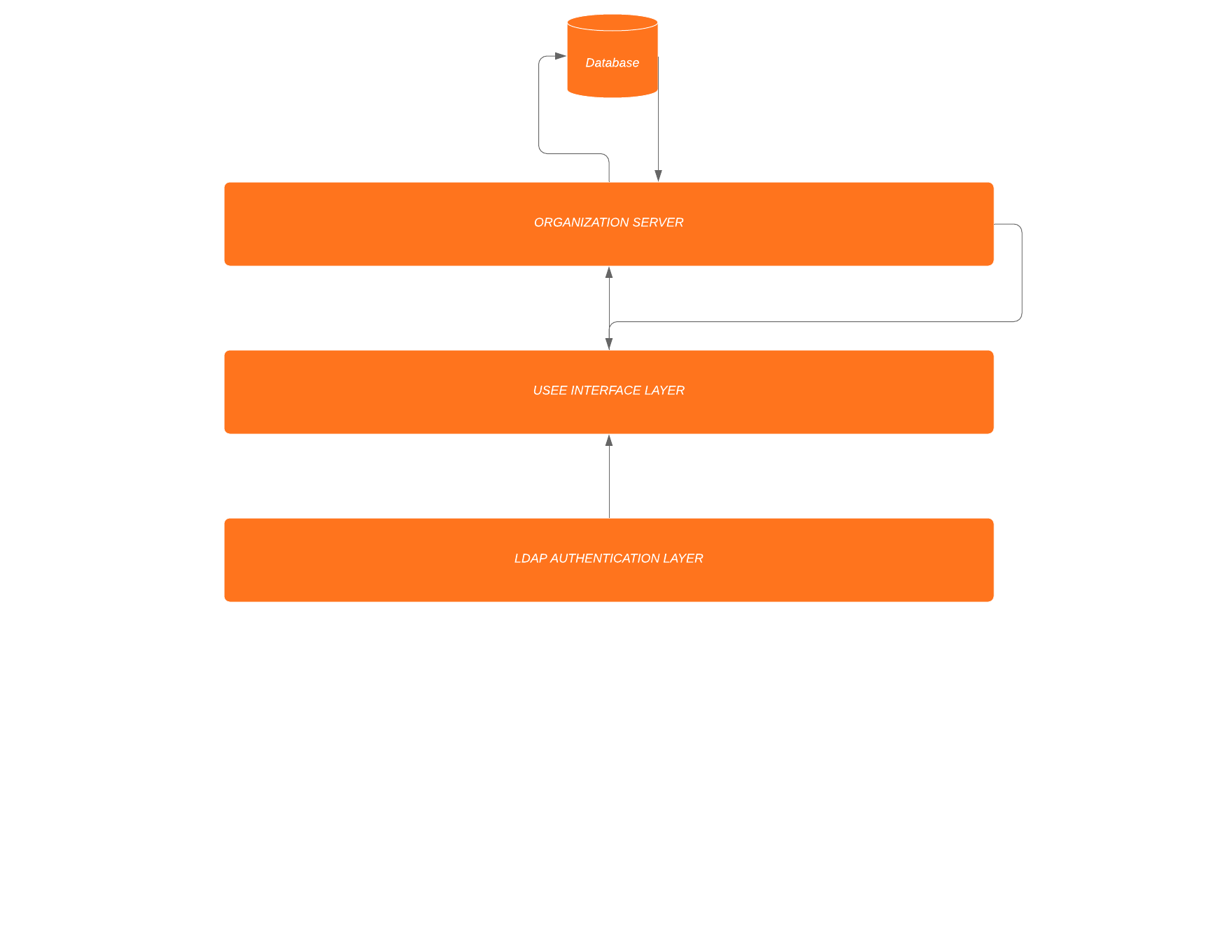
1. Event handling should be done efficiently
2. Appropriate alert system must be implemented so that the user can be notified about the issue in a clear manner.
3. Proper connectivity with the LDAP server and ONGC database.

DESIGN:

* Sequence diagram:



* Architectural Design:



IMPLEMENTATION:

Tools and modules used for the implementation:

1. Language:

Python version 3.6 has been used for the development of the software. This version of python has modules like:

* Tkinter(for developing the GUI)
* Subprocess(for communicating with the terminal)

1. Python pipes:

The [subprocess](https://docs.python.org/3/library/subprocess.html" \l "module-subprocess" \o "subprocess: Subprocess management.) module allows you to spawn new processes, connect to their input/output/error pipes, and obtain their return codes.

When used, the internal [Popen](https://docs.python.org/3/library/subprocess.html" \l "subprocess.Popen" \o "subprocess.Popen) object is automatically created with stdout=PIPE and stderr=PIPE. The stdout and stderr arguments may not be supplied at the same time as capture\_output. If you wish to capture and combine both streams into one, use stdout=PIPE and stderr=STDOUT instead of capture\_output.

The timeout argument is passed to [Popen.communicate()](https://docs.python.org/3/library/subprocess.html" \l "subprocess.Popen.communicate" \o "subprocess.Popen.communicate). If the timeout expires, the child process will be killed and waited for. The [TimeoutExpired](https://docs.python.org/3/library/subprocess.html" \l "subprocess.TimeoutExpired" \o "subprocess.TimeoutExpired) exception will be re-raised after the child process has terminated.

The input argument is passed to [Popen.communicate()](https://docs.python.org/3/library/subprocess.html" \l "subprocess.Popen.communicate" \o "subprocess.Popen.communicate) and thus to the subprocess’s stdin. If used it must be a byte sequence, or a string if encoding or errors is specified or text is true. When used, the internal [Popen](https://docs.python.org/3/library/subprocess.html" \l "subprocess.Popen" \o "subprocess.Popen) object is automatically created with stdin=PIPE, and the stdin argument may not be used as well.

1. Mysqlb:

MySQLdb is an thread-compatible interface to the popular MySQL database server that provides the Python database API.

MySQLdb is a thin Python wrapper around \_mysql which makes it compatible with the Python DB API interface (version 2). In reality, a fair amount of the code which implements the API is in \_mysql for the sake of efficiency.

The DB API specification [PEP-249](http://www.python.org/peps/pep-0249.html) should be your primary guide for using this module. Only deviations from the spec and other database-dependent things will be documented here.

Therefore, it has been used as a connector to connect the software with the Company’s database.

1. LDAP:

LDAP, or lightweight directory access protocol, is a communications protocol that defines the methods in which a directory service can be accessed. More broadly speaking, LDAP shapes the way that the data within a directory service should be represented to users, defines requirements for the components used to create data entries within a directory service, and outlines the way that different primitive elements are used to compose entries.

Since LDAP is an open protocol, there are many different implementations available. The OpenLDAP project is one of the most well supported open source variants.

Entries are all added to an LDAP system as branches on trees called **Data Information Trees**, or **DITs**.

For instance, an entry for a person named John Smith might be placed beneath a “People” entry for an organization under example.com. Since there might be multiple John Smiths in the organization, a user ID might be a better choice for the entry’s RDN. The entry might be specified like this:

dn: uid=jsmith1,ou=People,dc=example,dc=com

objectClass: inetOrgPerson

cn: John Smith

sn: Smith

uid: jsmith1

LDAP has been used for user authentication before he or she can use the software to write data to the tape.

1. Shell Scripting:

A shell is special user program which provide an interface to user to use operating system services. Shell accept human readable commands from user and convert them into something which kernel can understand. It is a command language interpreter that execute commands read from input devices such as keyboards or from files

As shell can also take commands as input from file we can write these commands in a file and can execute them in shell to avoid this repetitive work. These files are called **Shell Scripts** or **Shell Programs**. Shell scripts are similar to the [batch file](https://en.wikipedia.org/wiki/Batch_file" \t "/home/saransh/Documents\\x/_blank) in MS-DOS. Each shell script is saved with **.sh** file extension

A shell script comprises following elements –

* Shell Keywords – if, else, break etc.
* Shell commands – cd, ls, echo, pwd, touch etc.
* Functions
* Control flow – if..then..else, case and shell loops etc.

Shell scripting has been used to kill the subprocesses in the system. A recursive algorithm has been developed which traverses through all the children of the process and kills them while backtracking except the process itself.

INTEGRATION AND TESTING:

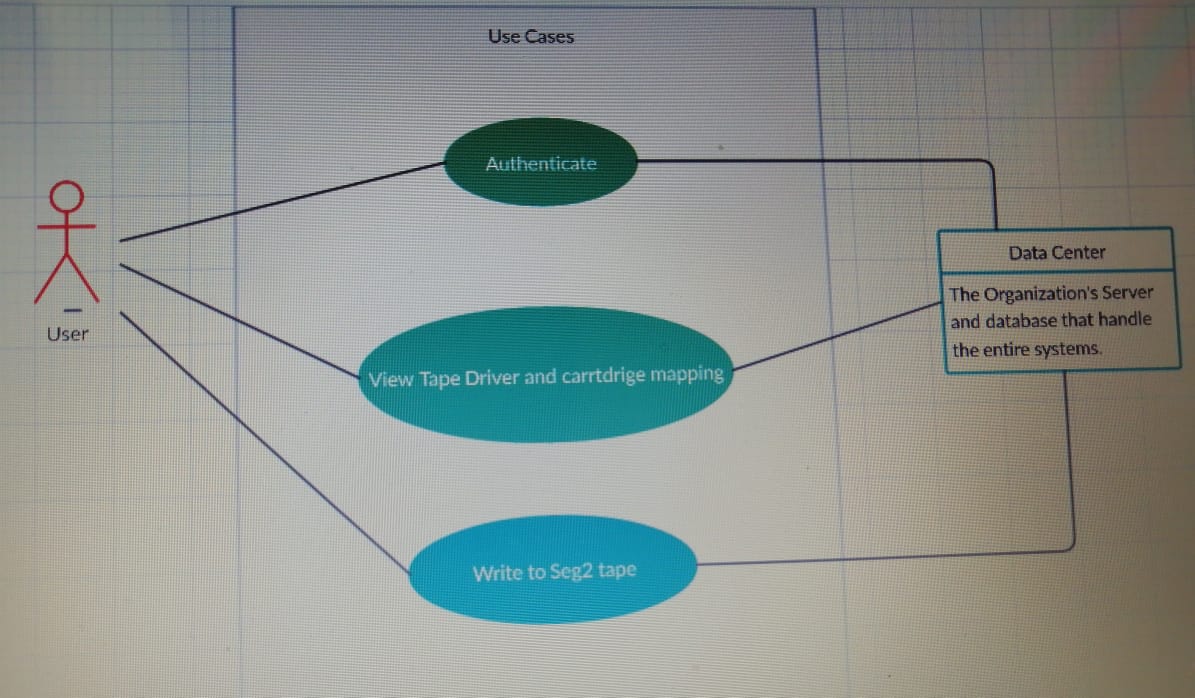
After all the modules were developed successfully, each of them were tested separately. After the unit testing, they were integrated and then the integrated system was verified against

various uses cases.

INTEGRATION TESTING:

|  |  |  |  |
| --- | --- | --- | --- |
| S No. | Description | Expected output | Actual output |
| 1. | Checking LDAP  Authentication by providing incorrect details | An alert window signifying ‘invalid credentials’ should pop up | An alert window signifying ‘invalid credentials’ pops up. |
| 2. | Checking LDAP  Authentication by providing correct details | Access should be granted to the user for Seg2 tape. | Access granted |
| 3. | Checking all the driver status and cartridge mapping | As different drivers are selected, the corresponding cartridge should be visible | As different drivers are selected, the corresponding cartridge is visible |
| 4. | Entering invalid data in the parameters for writing the data | An alert window signifying ‘Please enter valid details’ should pop up | An alert window signifying ‘Please enter valid details’ pops up |
| 5. | Stopping a process that is writing the data | Process should stop executing any further. | Process stops and the user can start it again. |

USE CASE DIAGRAM :



After this phase, the software was ready to be used by the employees of the organization and they can now easily write and manipulate data to the tape without going into the details of operating system operations and the back end information. The GUI has been designed in such a way that even people with not much technical knowledge of the system can understand the working easily.

DEPLOYMENT

After the requirement phase, design phase and the testing phase, the system is now finally ready to be deployed for the use by employees of the Organization.

The following are the screenshots of the final developed software: