Software Documentation

on

Enigma Machine Emulator

Apurv Jain

2017KUCP1016

Information Systems and Security Lab (CSP305)

Dr. Ajay Nehra

13 October 2019

Abstract

The ‘Enigma Machine’ is a device which was used by Nazi Germany to encrypt and decrypt secret messages, during the World War II.

Enigma has an electrochemical rotor mechanism that is used to scramble the 26 letters of the alphabet. A person can enter the text on the Enigma’s keyboard, and write down the letter, whose bulb lights up on each key press. This text serves as Ciphertext, which when entered into the machine with the same configurations of the machine before encrypting the text, gives back the Plaintext.

‘Enigma Machine Emulator’ Project aims to provide a GUI Emulator of this original Enigma Machine.

This emulator provides the user to enter the input text in capital case to cipher or decipher the text, either by typing or by using the traditional Enigma style virtual QWERTZ typewriter. The user can also import or export text file for encrypting or decrypting the text.

Table of Contents

1. Objective

1.1 Purpose

Enigma Machine Emulator provides a GUI Interface for the emulation of an original Enigma Machine. The users can encipher and decipher their text for a given setting. The user can select any three motors out of the five given motors (as were provided in an original Enigma Machine), and a reflector out of the given three standard reflectors. The user can also select the Ring settings and Ground (Start) settings for each motor. The user can view the current state of each rotor as well.

The machine provides the user to enter the text (in Upper Case) by typing in the window or by using the traditional Enigma style ‘QWERTZ’ virtual keyboard or by importing a text file. The output can be viewed on the display window, or by exporting the output text into a text file. When the keyboard is used to enter the input, the output emulates the original bulb glowing mechanism, also displaying the output text.

The user can also view the rotors and reflector wiring settings, and can also view how a character is used to give the output through the wires, as one enters it.

1.2 Scope

The emulator can be used by anybody who wants to see and learn the working of an original Enigma Machine.

1.3 Advantages

- Enigma Emulator comes with a user-friendly GUI, which can be used to see and learn the working of the original Enigma Machine.

- The Emulator eradicates the need of an original Hardware machine for anyone who wants to see how the original Enigma works.

- Easy interface and simple options makes it practically possible for anyone with a basic know-how of Computers to run and use the Emulator.

2. Requirement Analysis

2.1Requirement Specification

Software Requirement

1. JRE 1.8 or above

   Note: The detailed requirements for the Java 1.8 are given here at:
   https://www.oracle.com/technetwork/java/javase/certconfig-2095354.html

2.2 Feasibility Study

Feasibility is the determination of whether a project is worth doing. This type of study determines if a project can and should be taken. A feasibility study is carried out to select the best system that meets performance requirements. The document provides the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Economic, Operational and Behavioral feasibilities.

Economic Feasibility

The developing system must be justified by cost and benefit, the criteria to ensure that effort is concentrated on project, which will give best return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require. Here following are estimated costs:

- One-time development cost
- Periodic maintenance cost (if the project is carried on further in future)

Operational Feasibility

This test of feasibility asks if the system will work when it is developed. In this stage, the observed aspects are: Everyone welcomes the new system and there should not be any problem because of ease of access and user-friendly environment. The user is expected to have some basic know-how of a computer system, such as how to run a program, how to type a text etc.

Behavioral Feasibility

Behavioral feasibility deals with the runtime performance of the software, the proposed system must score higher than the present in the behavioral study. The programmer should have end user in mind when the system is designed and while designing the software, programmer should be aware of the conditions related to user's knowledge input, output, calculations etc.

2.3 Product Functions

- **User's perspective**

1. The user can select the rotors and the reflector.
2. The user can set the Ring setting and Start (Ground) setting for each Rotor.
3. The Plugboard settings can be typed in the Plugboard field.
4. The user can save the settings by clicking on the Save Button.
5. The user can restart the application by clicking on the 'Restart' option from the File menu.
6. The user can exit from the application by clicking on the 'Exit' option from the File menu.
7. The user can view the current state of each motor.
8. The user can select any one display option out of three display options from the Display menu.

For Text Display:

1. The user can give the input (in Upper case) by typing or by importing a text file.
2. The output is displayed in the output field.
3. The output can be exported to a text file.

For Keyboard Display:

1. The user can type the input using the traditional Enigma 'QWERTZ' virtual keyboard.
2. The user can see the output on the screen.
3. The output of the each corresponding letter can also be perceived through the Bulb Glowing representation of the output letter (displayed for 0.5 seconds).

For Wires connection Display:

1. The user can see the wires connection at a state for the Rotors and the Reflector.
2. The user can type in letter by letter in the Input Text field, to get the output and to see how the output was generated through the wires.

- **Product Perspective**

  The project is mainly aimed towards providing an emulator for the users to learn and see the working of an original Enigma machine.

  The users are expected to have the basic knowledge of how to run a computer program, type in the input and perceive the output (at least).

2.4 Use Case

Use case no. 1

USE CASE NAME	Settings selection
ACTOR	User
INPUT	Select the Rotors and Reflector, and type in the Plugboard settings
STEPS	1. Actor selects the Rotors. 2. Actor selects the Ring and Start settings for each Rotor. 3. Actor selects the Reflector. 4. Actor types in the Plugboard settings. 5. Then the actor clicks on Save Button to save the settings.
OUTPUT	The settings are saved.

Use case no. 2

USE CASE NAME	Text Display
ACTOR	User
INPUT	Inputs the text
STEPS	1. Actor can input the text in the Input field. 2. Actor can input the text by importing a text file. 3. Actor can export a text file to get the output in the file.
OUTPUT	The output is displayed in the Output text field.

Use case no. 3

USE CASE NAME	Keyboard Display
ACTOR	User
INPUT	Inputs the text
STEPS	The actor inputs the text using the virtual keyboard.
OUTPUT	1. The output for each letter is displayed on the screen. 2. The output of each letter can also be perceived through the Glowing action of a Bulb of the output letter.

Use case No. 4

USE CASE NAME	Wires connection Display
ACTOR	User
INPUT	Inputs the text
STEPS	The actor can see the wires connection at a state for the Rotors and the Reflector, and can input the text in the Input field.
OUTPUT	The output for each letter is displayed, the actor can see how the output was generated through the wires.

Use case No. 5

USE CASE NAME	File Menu (Import and export files)
ACTOR	User
INPUT	Click on the File Menu
STEPS	1. The actor can select to import a text file for the input or to export the file for the output. 2. The actor needs to be in the Text Display mode.
OUTPUT	The corresponding import or export file operation is performed.

Use case No. 6

USE CASE NAME	File Menu (Restart)
ACTOR	User
INPUT	Click on File Menu
STEPS	The actor can restart the application by selecting the 'Restart' option.
OUTPUT	The application is restarted with the initial settings.

Use case no. 7

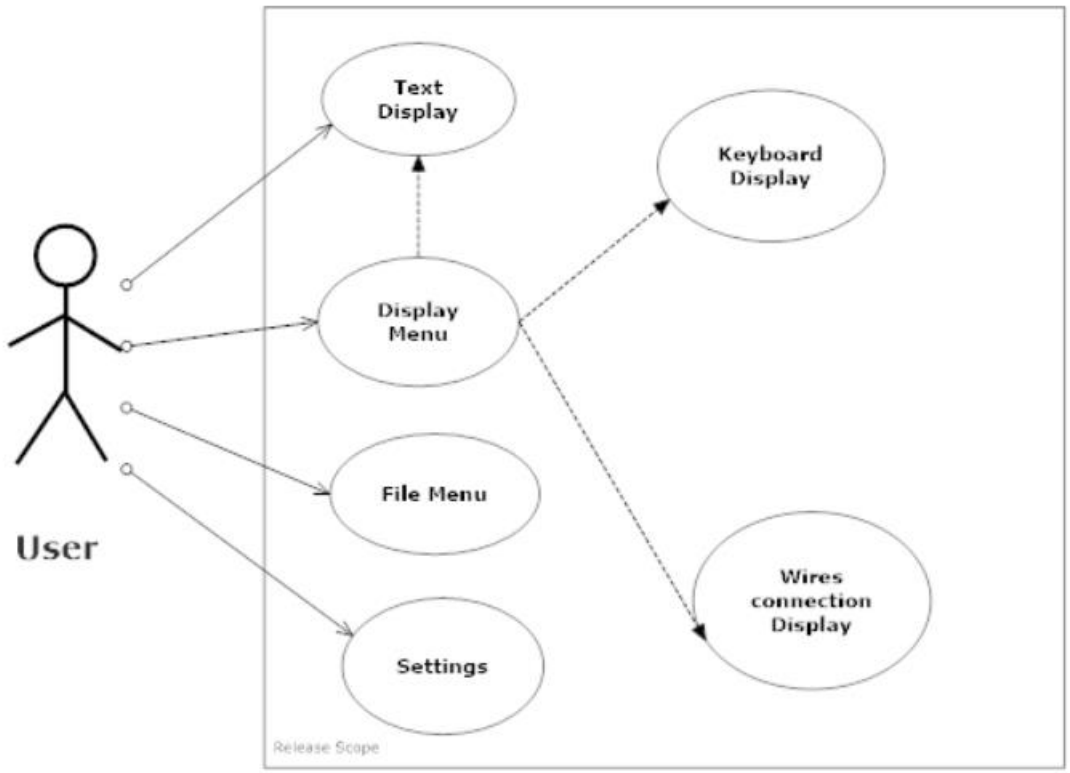
USE CASE NAME	File Menu (Exit)
ACTOR	User
INPUT	Click on File Menu
STEPS	The actor can exit from the application by selecting the 'Exit' option.
OUTPUT	The application is closed.

Use case no. 8

USE CASE NAME	Display Menu
ACTOR	User
INPUT	Click on Display Menu
STEPS	The actor selects which display mode to use, by clicking on the corresponding option (1. Text mode, 2. Keyboard mode, 3. Wires connection mode).
OUTPUT	The corresponding display is generated.

Use Case Diagrams

1. User



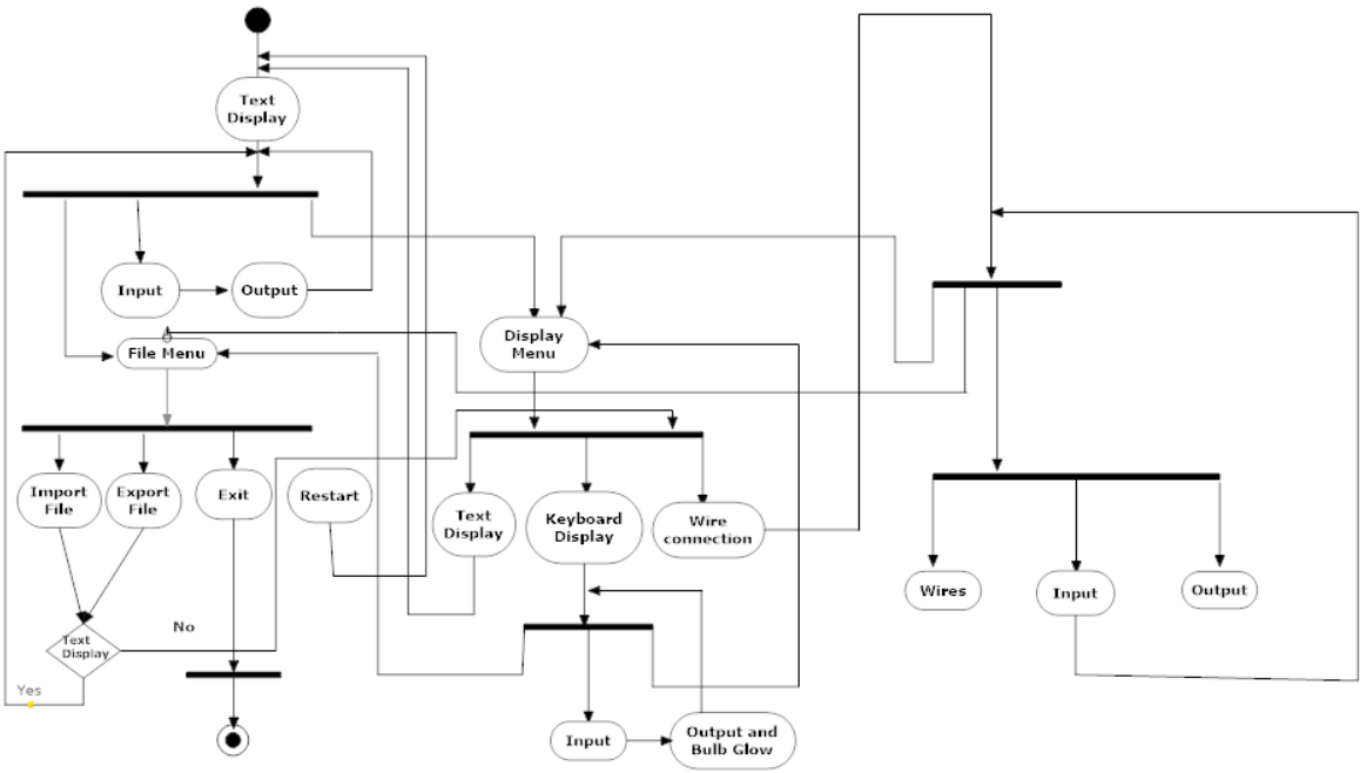
3. System Design (SDS)

3.1 Class Diagrams

Note: Please refer to Java Documentation (given along with project files).

3.2 Activity Diagram

1. User



4.TESTING

The importance of software testing and its implications with respect to software quality cannot be over emphasized. Software testing is a crucial element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as system element and the attendant “costs” testing. It is not unusual for a software development organization to expend 40 percent of total project effort on testing. Testing is a process of executing a program with the intent of finding an error. If testing is conducted successfully (according to the objective stated), it will uncover errors in the software. As a secondary benefit, testing demonstrates that software functions that appear to be working according to specification, that performance requirement appear to have been met.

Unit Testing

Unit testing focuses on verification effort of the smallest unit of software design –the module. Using the detail design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The module interface is tested to ensure the information properly follows into and out of the program under test. Boundary conditions are tested to ensure that all statements in the module have been executed at least once. And finally all error-handling paths are tested.

Module Testing

Module testing deals with testing of each module separately. Each module is tested to check whether it works according to the requirement and perform and desired results. Module testing tells about errors in each module and if each module works properly then the whole system will also work properly. Module testing saves a lot of time in detecting errors at a later stage. In integration testing we combine different modules, which may be getting called, form one another and then they are tested together so as to check whether they are working properly or not.

Integration Testing

Integration testing is a systematic technique for constructing the program structure while at the same time to uncover the errors associated with the interface. The objective is to take unit tested module and build a program structure that has been detected by designing.

System Testing

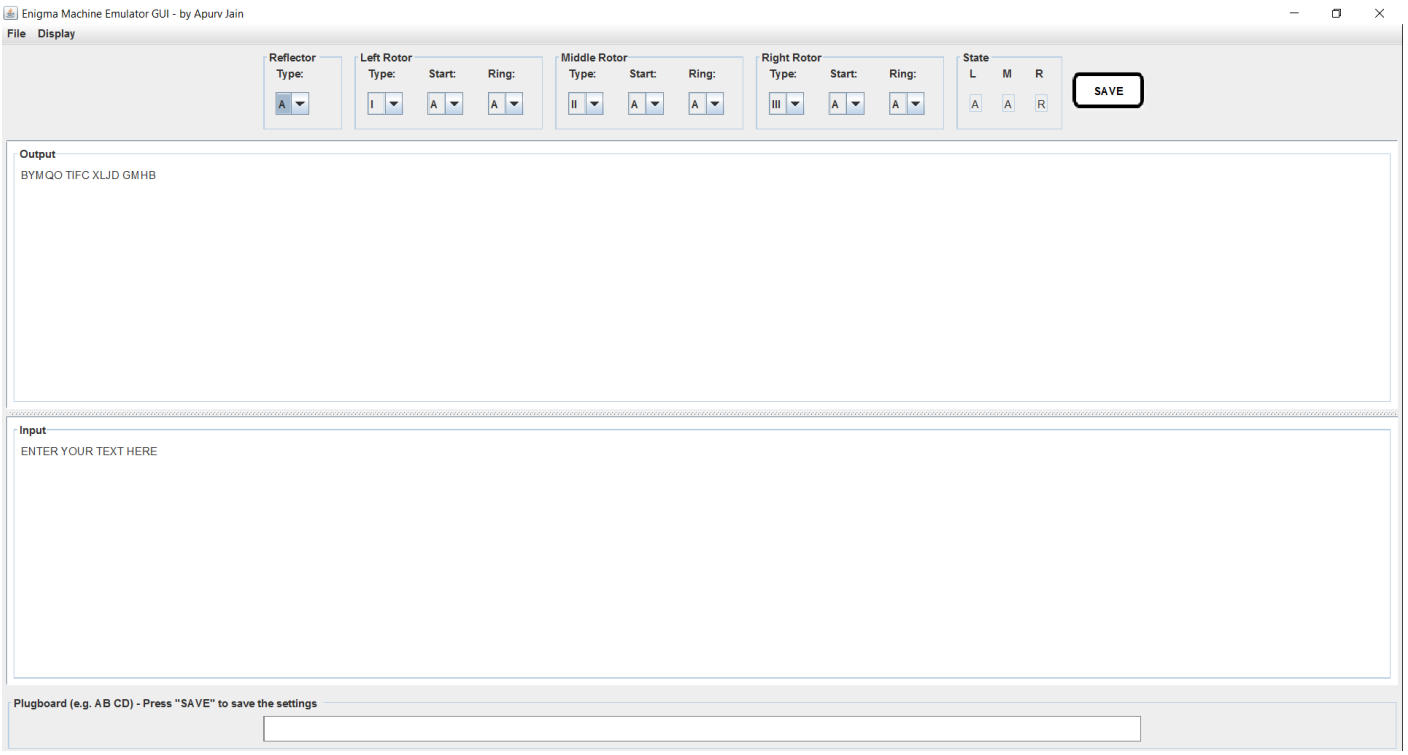
System testing is actually a series of different tests whose primary purpose is to fully exercise the computer base systems. Although each test has a different purpose, all works should verify that system elements have been properly integrated and performs allocated functions.

Acceptance Testing

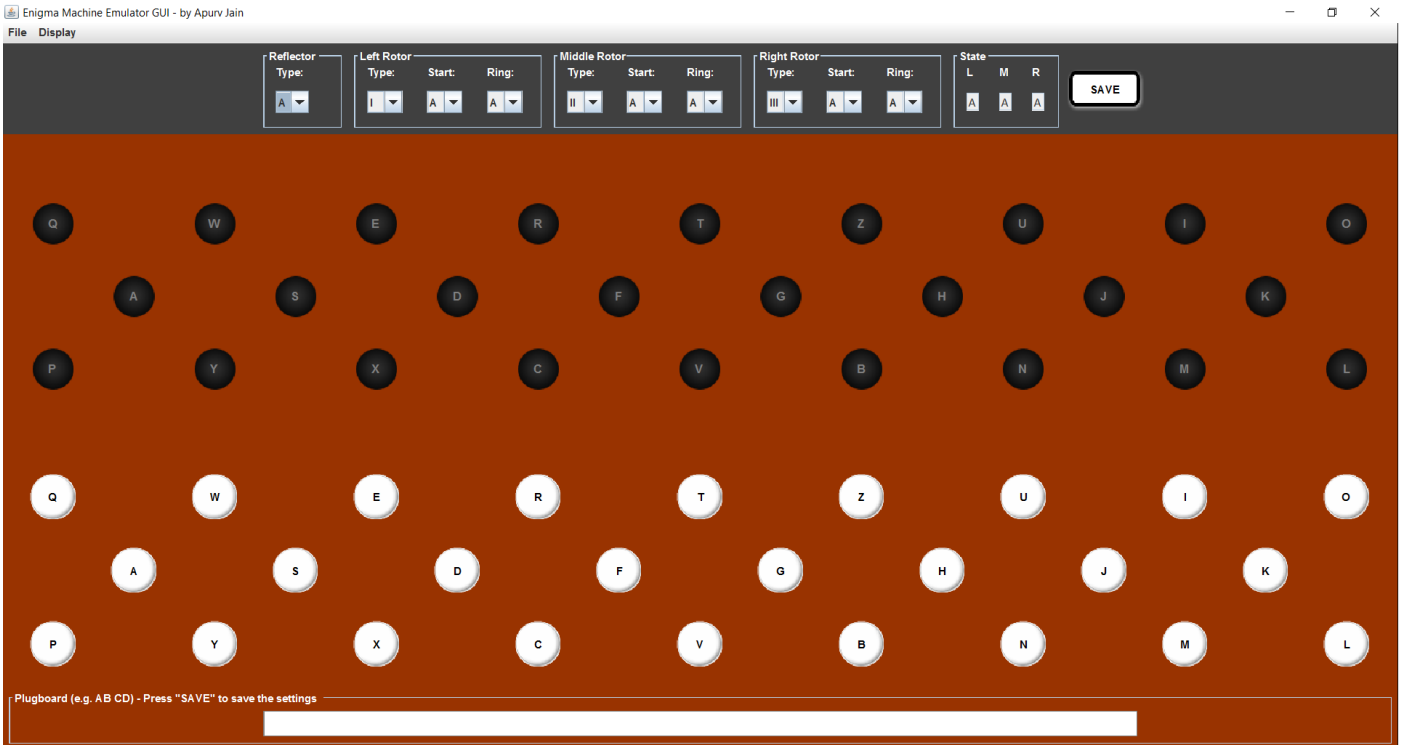
All the requirements that were specified are verified.

5. User Interface

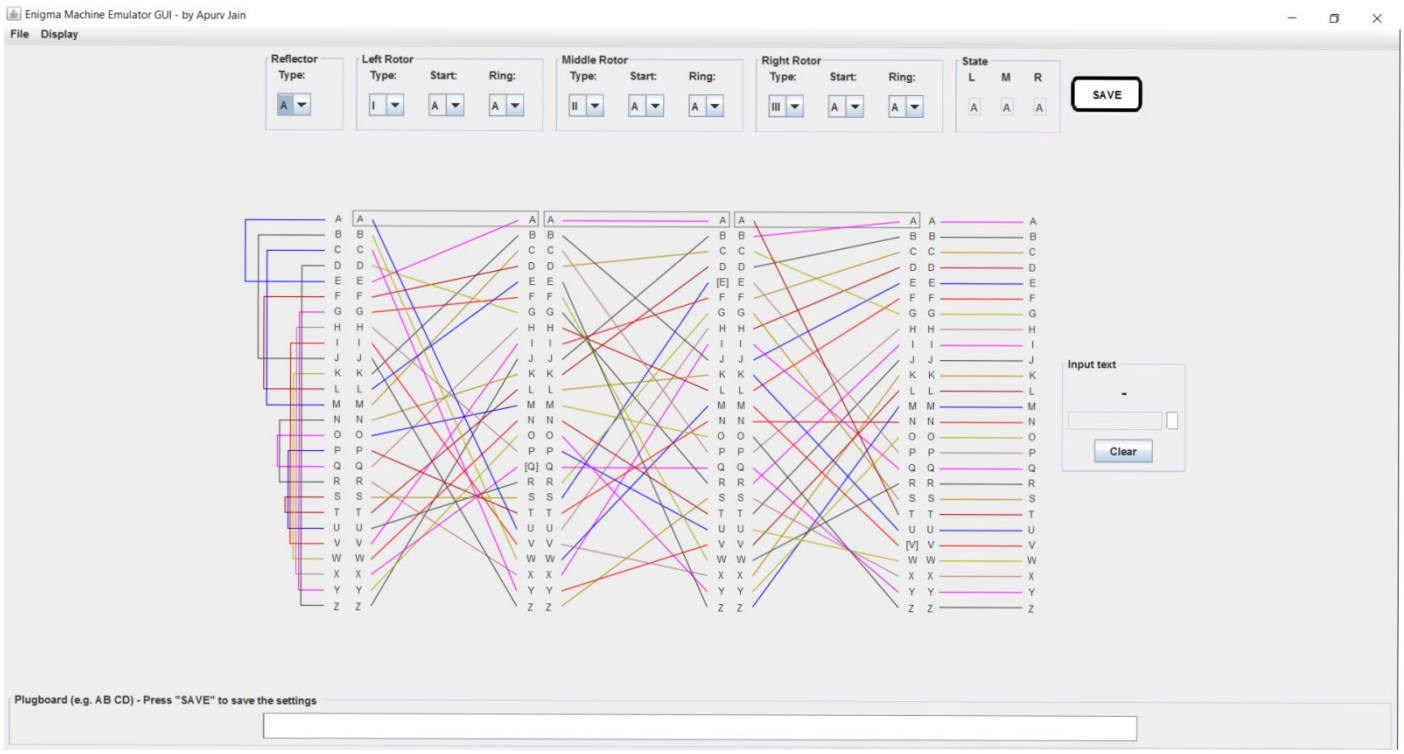
Text Display



Keyboard Display



Wires connection Display



6. Conclusion

This Emulator provides a GUI alternative for the original ‘Enigma Machine’.
The Emulator can be used by anyone who wants to learn and see the working of an Enigma machine. The emulator eradicates the need for having a Hardware model of Enigma. Its user-friendly GUI makes it easy for a user with some basic know-how of using a Computer, to use it.

7. References

Books:

- Schildt, Herber. *Java: The Complete Reference.* New York: Mc-Graw Hill, 2014.

Online Articles:

- "Enigma". Wikipedia, https://en.wikipedia.org/wiki/Enigma_machine .
- "Working principle of the Enigma". Crypto museum, https://www.cryptomuseum.com/crypto/enigma/working.htm .

Special Mention:

- Bawab, Amir. *Enigma machine simulator.* https://github.com/amirbawab/Enigma-machine-simulator .