VISVESVARAYA TECHNOLOGICAL UNIVERSITY,
BELGAUM- 590014

A Mini Project Report

On

SOCIAL HAVALDAR

submitted in partial fulfillment of the requirement for the degree of

Bachelor of Engineering in

Computer Science and Engineering

Submitted By

B CHETAN RAO – 2BV13CS132

Under the guidance of

Prof. MAHESH S PATIL

.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.V.B. COLLEGE OF ENGINEERING & TECHNOLOGY,

ABSTRACT

Social Havaldar is a program that acts as a bank vault, or safe, where you can keep your private information or files hidden and secure. Everything in the vault is protected with an advanced encryption, and requires a password (your password) to open the vault to access the information.

Files and file structures can be added to the vault by a simple drag and drop operation. It is also possible to create picture passwords which is the part of GUA and which is the uniqueness of our system.

To open a file stored in vault the user can decrypt it from the vault and this sets the file permissions of the selected file to default. The software enables the user to store information about credit/debit cards which are highly confidential and can be accessed only by the authorized user.

The software also keeps track of the intruders into the system by taking an image of user on three wrong password entries.

ACKNOWLEDGEMENTS

We have been bestowed the privilege of expressing our gratitude to everyone who helped us in completing our project.

First, we would like to thank our project guide Prof. Mahesh Patil, Department of Computer Science and Engineering, BVBCEET, Hubli, whose guidance and knowledge was great use to do us during this project work. His continuous guidance throughout our project helped us complete in time.

We would like to thank our project coordinator Prof. Padmashree Desai, Department of Computer Science and Engineering, BVBCEET, Hubli, for her continuous support and reviews during the progress of or project.

We would like to express our sincere thanks to Prof. K.R. Biradar, HOD, , Department of Computer Science and Engineering, BVBCEET, Hubli, for providing us an opportunity for carrying out our mini project work successfully.

We would also like to thank Prof. Ashok Shettar, Vice-chancellor, KLETech and to Dr. Prakash Tewari, Principal, BVBCET, Hubli.

The software tools used in various phases of our project are:

- NetBeans for coding the functional requirements.
- CodePro for validating the code
- Junit testing tool for unit testing

Team members
Raghavi Bhat
B Chetan Rao
Shivendra Dubeer
Saurabh Kumar

Chapter No. **Table of Contents** **Page No.**

1. PREAMBLE

1.1 Introduction

Privacy is an important aspect when any domain of PC users are concerned. Any computer user will have personal files, be it of any type like images, text files, audio notes or videos.

Social Havaldar is an application developed to provide a separate private folder to store all the private files of users.

1.2 Problem Definition

A Desktop Utility meant to provide security by creating folders with to user information like videos, folders, photos and private wallet information.

1.3 Scope and objectives

The primary purpose of this project is to protect the confidentiality of digital data stored on computer systems.

The application of this desktop utility is widespread. As security is a major concern in today's world, this utility can be put to use in almost every field to secure any kind of information.

Some fields are as follows:

- Banking
- Military
- Hospitals
- Scientific experimental data
- Personal use

It can be efficiently extended to any platform.

2. LITERATURE SURVEY, PROBLEM DEFINITION, PROPOSED SOLUTION

2.1 Existing Systems

There are a few existing systems in market that provide similar functions as Social Havaldar. Some of them are

- Crisplock
- Advanced file folder
- Password protect windows

The existing systems do not have well defined user interfaces when compared with our proposed system. Most of the functionalities of the existing systems are constrained to specific domains of user groups, whereas the functionalities of our proposed system are quite general to all domains of user groups.

The proposed system includes a few features from existing systems with the addition of unique features like Graphical User Authentication(GUA) and facility to store private wallet information.

2.2 Description of target users and justification of choice of problem

The users for our proposed system can belong to any domain of user group who use a PC. Any user will have personal information to be stored in their computers which are not likely to be accessible by everyone. Also sometimes the users wish to store information about their credit/debit cards in a safe place which is highly confidential and is accessed only by them whenever needed. Our selection of the problem definition is more likely to provide the users with the type of security they want for their personal information and hence is a relevant topic to be developed as a system.

2.3 Advantages of proposed system

2.3.1 Improved Graphical User Interface

The system provides an easy way to store private information into the system providing improved GUI.

2.3.2 Graphical User Authentication

The system provides a unique feature that allows the user to set up a unique picture password that unlocks only on clicking the image at particular spots for particular number of times. This feature provides additional security to the folders containing personal files.

2.3.3 Mail Recovery of passwords

The system also provides the user with the option of email recovery of passwords in case the password is forgotten.

2.4 Constraints of proposed system

The system is unable to recover any data that gets deleted in the folder. It also deletes the original data file from the computer once the file is put in folder.

3. SOFTWARE REQUIREMENT SPECIFICATIONS

3.1 Overview of SRS

It describes the purpose of the system and its intended audience. Introduces the concepts of user requirements, the functional requirements, use case descriptions, non-functional requirements, test planning. It describes the overall behavior of the system using diagrammatic representation like use case diagrams for the security folder.

3.2 Intended Audience

The SRS can be referred by stake holders, developers of system and also users who wish to learn about the system.

3.3 Requirement Specifications

3.3.1 Functional Requirements

Req -ID	Functional Requirement	Priority
1	The user shall be able to sign up to the system. The user can select either one or both of the security options. 1.1 The user shall be able to set a character password. 1.2 The user shall be able to set a picture password.	High
2	The user shall be able to encrypt any file in the computer through the system either by drag and drop or by browse method	High
3	The user shall be able to set file permissions to the selected file. The file shall be 3.1 Read only 3.2 Hidden	Medium
4	The user shall be able to login to the system any time by entering the previously set character password and/or picture password.	High
5	The system shall take a picture of the user on entering the password wrongly for continuous three times.	Medium
6	The user shall be able to reset password.	High
7	The user shall be able to email recovery of password in case password is forgotten.	High

Table 1: Functional Requirements

3.3.2 Use case diagrams

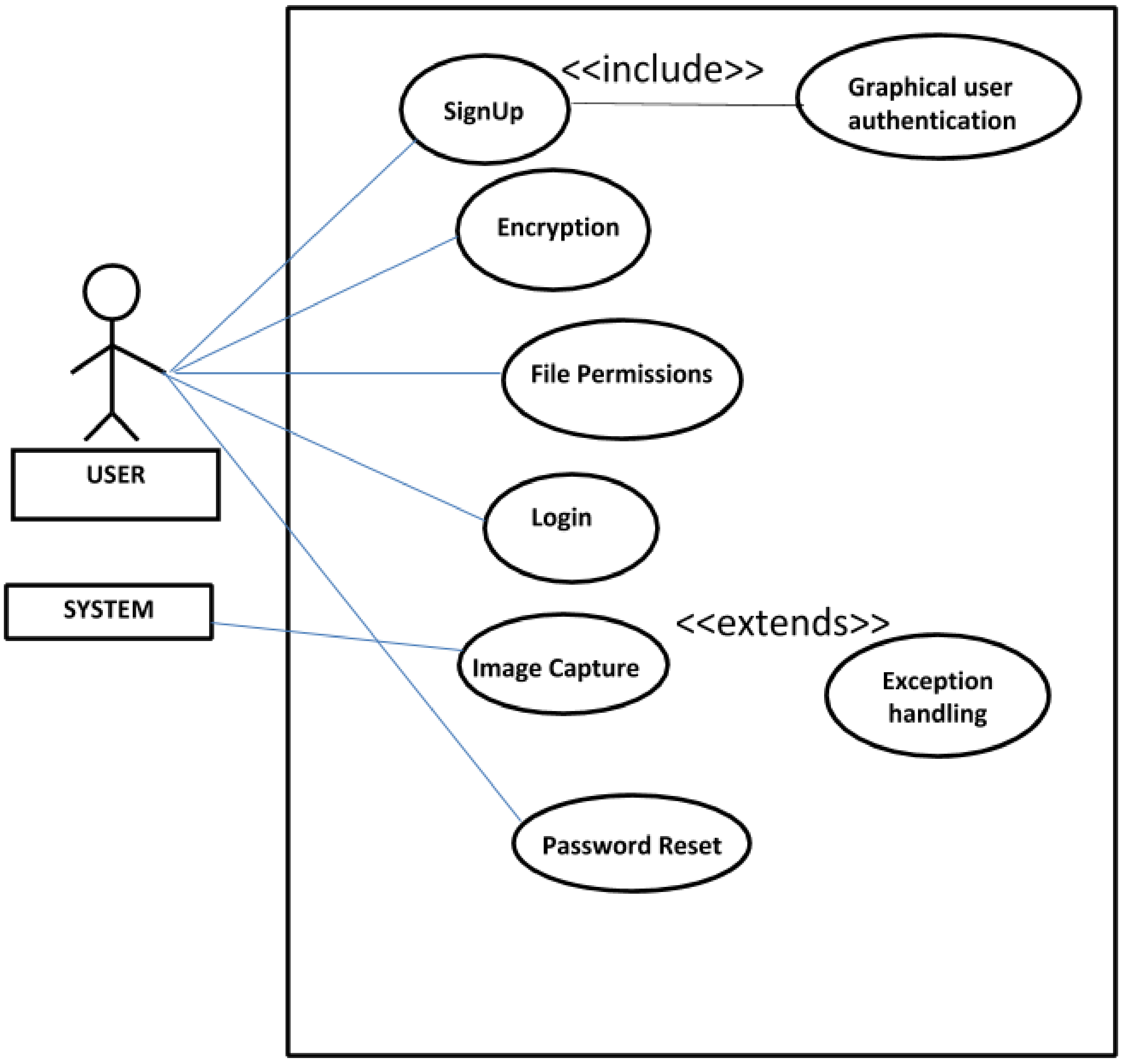


Fig.1: Use case diagram

Use case descriptions

3.3.2.1 Use case: Sign Up

- Actors: Users.
- Pre condition:

The application must be launched .

The application must be in a state to save new password.

In case of GUA, there must be valid images available that can be set for password.

- Post condition:

The user password is saved.

- Success scenario:

1. The user installs the application.
2. For GUA, the background image is chosen.
3. The application allows the user to prioritize the objects shown in the image.
4. Thus a pattern is created to unlock.
5. The other option is to set character password.
6. The user selects a set of characters in a particular sequence that is saved as password.
7. The user enters a valid email ID for the purpose of recovery of password if forgotten.
8. The user clicks on ‘Save’ button to save the created password.

- Exception scenario:

6: Character password does not satisfy minimum length requirement.

2: The background image resolution isn't appropriate for processing.

7: The user does not happen to save the newly created password.

3.3.2.2 Use case: Encryption

- Actors: Users.
- Pre condition:

The application is launched and password is entered.

The application must display option "Browse" to search the files that the user wants to encrypt.

The application must also accept files that user drags and drops into the application for encryption.

- Post condition:

The user chooses browse option to select file, or

The user chooses drag and drop to select file

- Success scenario:

1. The user chooses browse option and selects a file or drag-drop option and drops a file into the application.
2. The application determine the format of the file.
3. The applications encrypts the selected file using format specific encryption algorithm.
4. Encryption successful message is displayed.

- Exception scenario:

2: The application fails to determine format of the file and cannot encrypt it.

3: The application does not have a specific encryption algorithm for the determined format.

3.3.2.3 Use case: File Permissions.

- Actors: Users.
- Pre condition:

The application is launched and user successfully logs in.

User sets the required filed permissions as per his needs to a particular file.

- Post condition:

The file permissions are successfully applied to the file chosen by the user.

- Success scenario:

1. The application is opened and user is asked for password.
2. User inputs the password.
3. The submit button is pressed.
4. User is successfully logged in.
5. User selects the desired file from computer.
6. User sets the necessary file permissions.

- Exception scenario:

5: (i) After selecting the File Permissions if the user fails to Press the OK button then default permissions would still exist on the file.

5: (ii) Admin authentication required to change file permissions.

3.3.2.4 Use case: Login

- Actors: Users.
- Pre condition:

The application is launched.

The application must be in state to accept the password given to it.

The application must be in state to recognize the wrong password.

- Post condition:

The entered password is accepted and the application is opened.

- Success scenario:

1. The application is opened and user is asked for password.
2. User inputs the password.
3. The screen is displayed wherein the set pattern (password) is drawn.
4. Then the submit button is pressed.
5. The appropriate message is displayed.

- Exception scenario:

2: Password is incorrect.

3: If Password is not recognized.

3.3.2.5 Use case: Image Capture

- Actors: Users.
- Pre condition:

The application is launched.

The application must be in state to accept the password given to it.

The application must be in state to recognize the wrong password.

The application must have permission to access WEB cam on the respective system.

- Post condition:

A Image is captured of the user, no three wrong password tries.

- Success scenario:

1. The application is opened and user is asked for password.
2. User inputs the password.

3. Then the submit button is pressed.

4. On three continues wrong tries, image is captured of the user.

5. The image is displayed on next successful login.

- Exception scenario:

4: System permission to access WEBCAM is denied.

5: If there's no WEBCAM on the system, this feature doesn't work.

3.3.2.6 Use case: Password Reset

- Actors: Users.
- Pre condition:

The application is launched.

The application must prompt the user to enter the password.

The application must display options for the user to reset password.

- Post condition:

The user chooses to reset password using master reset.

The password is set to new password.

- Success scenario:

1. The user chooses Reset password.
2. User changes old password to new password according to his preference.
3. The user enters new password again.
4. The password is set to new password.

- Exception scenario:

2: The new password set has less than minimum number of characters.

3: The new password re entered does not match with new password entered first.

3.3.2.7 Use case: Email Recovery

- Actors : Users

- Pre conditions:

  The application is launched.

  The user has opened log in window.

- Post conditions:

  The password is mailed to the email ID of user stored during sign up.

- Success scenario:

1. The user tries to login into the software.

2. The user forgot the password.

3. The user clicks on ‘Forgot password?’ button.

4. The previously set password is mailed to the email ID of user thatwas entered during the sign up.

- Exception Scenario:

4: The password is not mailed due to network connection problems.

3.3.3 Non Functional Requirements

3.3.3.1 Performance: Login to the software requires 2sec on average and 4sec maximum for completion. At a time only one user can login and whenever the system is degraded by some unexpected errors the user cannot have access to certain files until the user completes re-login.

3.3.3.2 Safety : If the system fails catastrophically the data stored will be lost permanently.

3.3.3.3 Security: The system is secure from brute force hacking.

3.3.3.4 Reliability: Performs encryption and decryption of files with consistent efficiency.

3.3.3.5 Availability: The software is available in working condition for 98% of times, can be used continuously without any functional issues for 98% of login.

3.4 Software and Hardware Requirement Specifications

3.4.1 Minimum Hardware Requirements

3.4.1.1 System should have an inbuilt camera.

3.4.1.2 Core 2DUO processor

3.4.1.3 1GB of free space

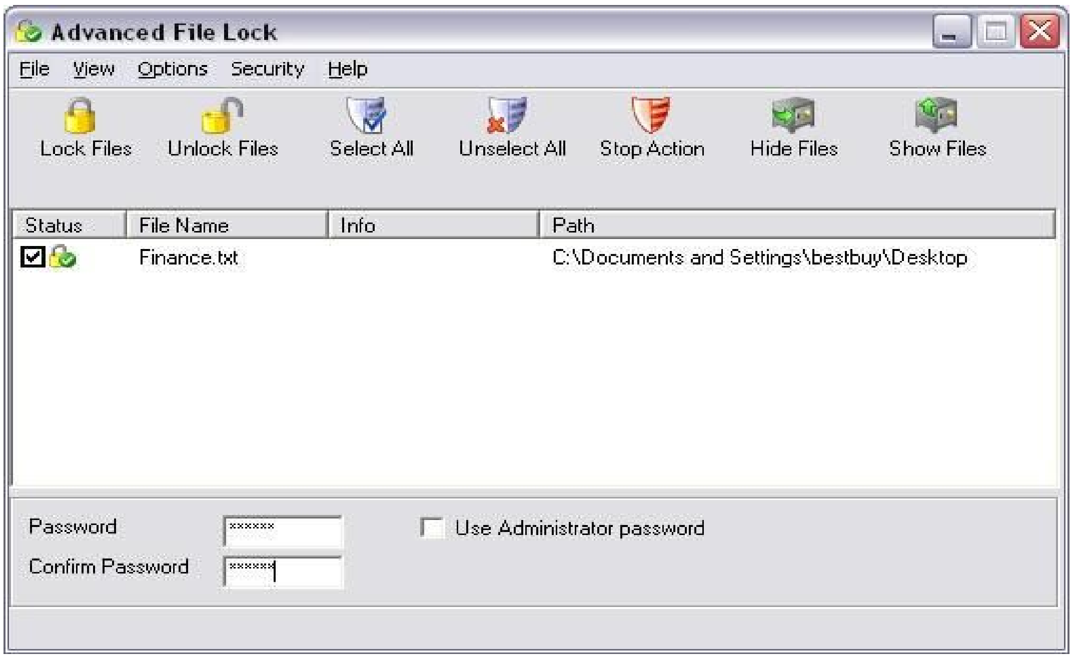
3.4.1.4 2GB RAM

3.4.2 Software Requirements

3.4.2.1 JDK version 1.4 onwards.

3.5 Comparison of GUI of existing systems

Advanced File Locker



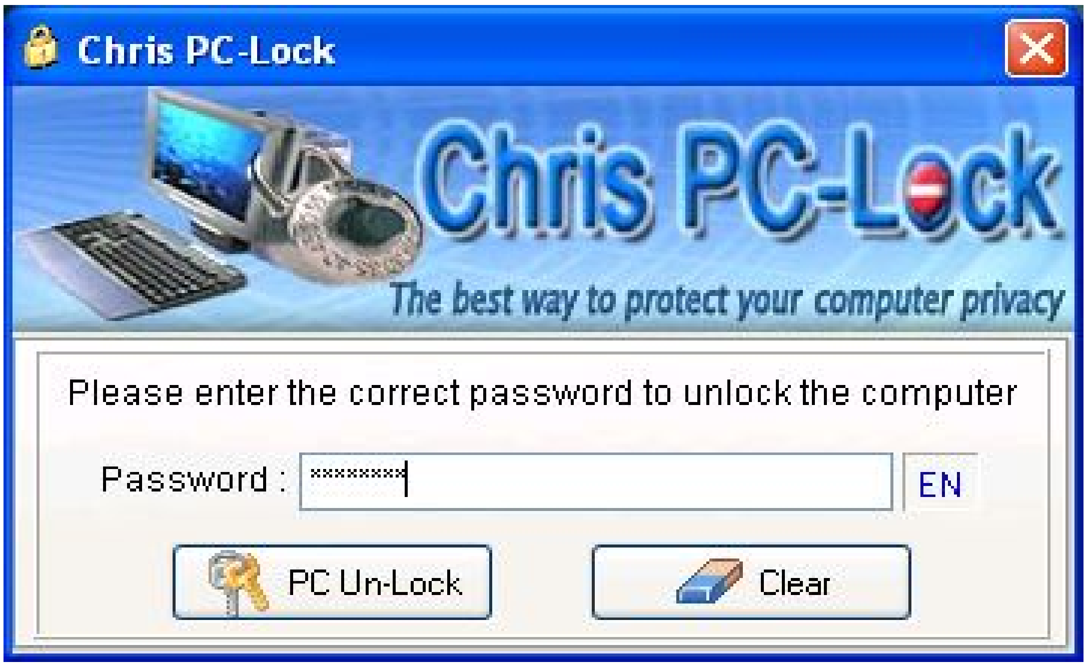
Status	File Name	Info	Path
☑	Finance.txt		C:\Documents and Settings\bestbuy\Desktop

Password ******

Confirm Password ******

Fig.2: Advanced file locker GUI

Chrisplock

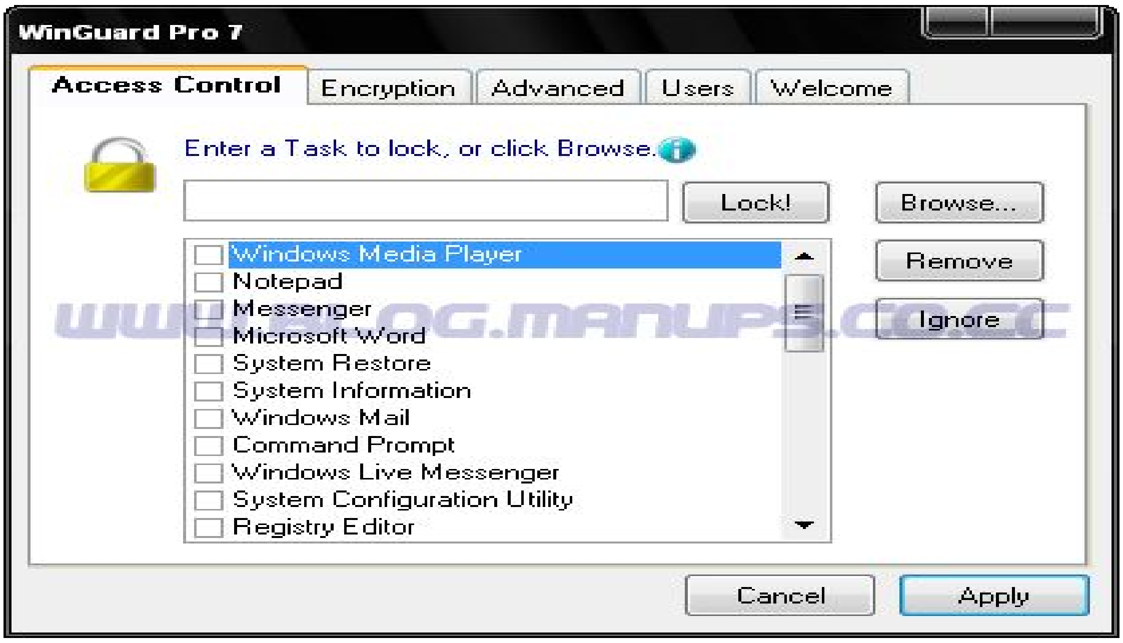


Please enter the correct password to unlock the computer

Password : ********

Fig.3: Chrisplock GUI

Password protect Windows

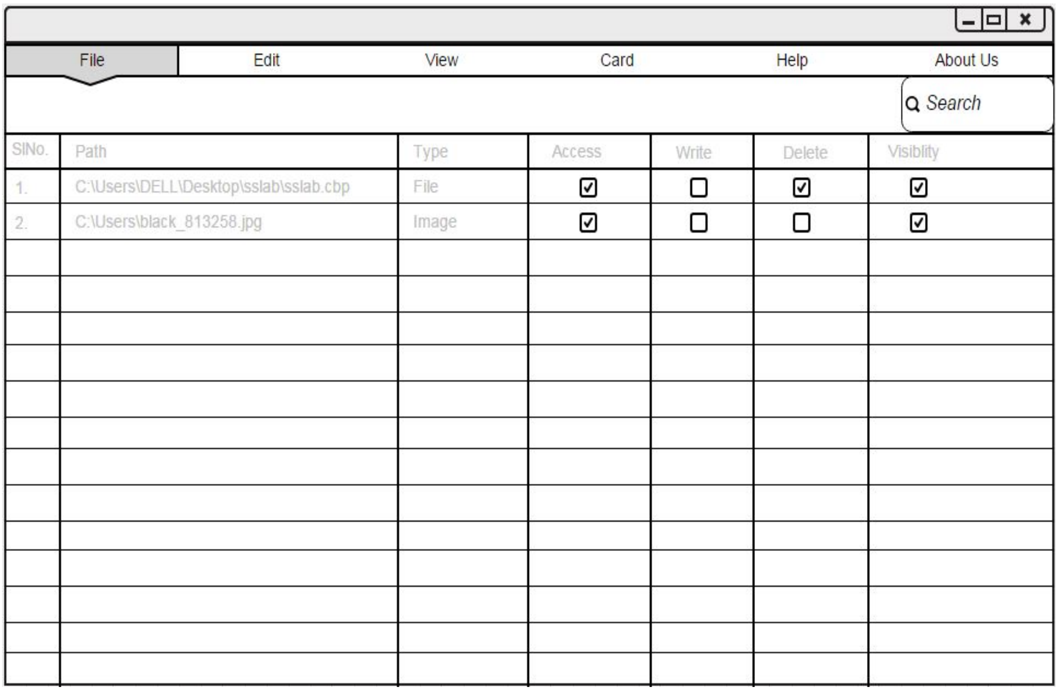


Enter a Task to lock, or click Browse.

- [ ] Windows Media Player
- [ ] Notepad
- [ ] Messenger
- [ ] Microsoft Word
- [ ] System Restore
- [ ] System Information
- [ ] Windows Mail
- [ ] Command Prompt
- [ ] Windows Live Messenger
- [ ] System Configuration Utility
- [ ] Registry Editor

Fig.3: Password protect Windows GUI

3.4 Proposed GUI for the system



SlNo.	Path	Type	Access	Write	Delete	Visiblity
1.	C:\Users\DELL\Desktop\sslab\sslab.cbp	File	☑	☐	☑	☑
2.	C:\Users\black_813258.jpg	Image	☑	☐	☐	☑

Fig.4: GUI of Social Havaldar

3.7 Acceptance Test Plan

Test ID	Requirements ID	Input Description	Expected Output	Actual Output
3.7.1	1	User Sets valid password	Password saved message displayed.	Password saved message is displayed.
3.7.2	1	User Sets password and Password length insufficient.	Password length insufficient message displayed.	Password not set message is displayed.
3.7.3	2	File chosen using browse option/ drag-drop	File Encrypted	File encrypted message is displayed.
3.7.4	2	Invalid File chosen using browse option/ drag-drop	File format not detected.	File doesnot exist message displayed
3.7.5	3	File Permission chosen by user for the file	File permission changed	No message displayed but file permissions changed.

3.7.6	3	Unauthorized File Permission chosen by user for the file	Admin authentication required to change file permissions.	No message displayed and file permissions donot change.
3.7.7	4	User enters genuine password	Successfully logged in	Login complete message displayed.
3.7.8	4	User enters wrong password	Wrong password. Please re-enter password	Invalid username or password message displayed.
3.7.9	5	User enters 3 wrong password	Image captured	Invalid username or password message displayed and image is captured.
3.7.10	5	User enters 3 wrong password in system that doesn't have a WEBCAM	Image not captured	Invalid username or password message displayed and image is not captured.
3.7.11	6	User chooses password reset option	The password is set to new password.	Password successfully changed message is displayed.
3.7.12	6	User chooses password reset option and enters invalid password.	Password reset failed, new password is not activated.	Password reset failed message is displayed.
3.7.13	7	User chooses email recovery of password.	Username and password is mailed to the stored email ID of user.	Password successfully mailed message is displayed and username and password are mailed.
3.7.14	7	User chooses email recovery of password when there is no internet connection.	Username and password is not mailed to the user.	Password could not be mailed message is displayed and username and password is not mailed.

Table 2: Acceptance test plan

4. SYSTEM DESIGN

4.1 Level 0 DFD

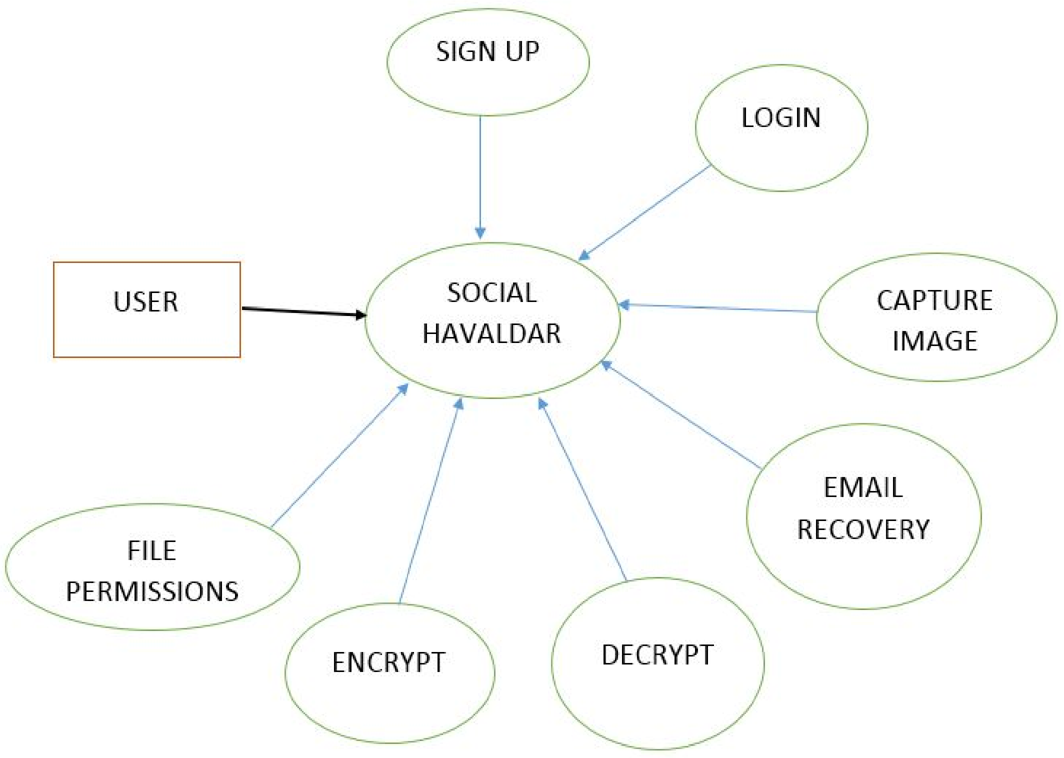


Fig.5: Level0 DFD

The level 0 DFD described above shows the interaction of different modules of our application. User is the source represented by square shape and the different interacting modules are represented in oval shape.

4.2 Detailed DFD for the proposed system

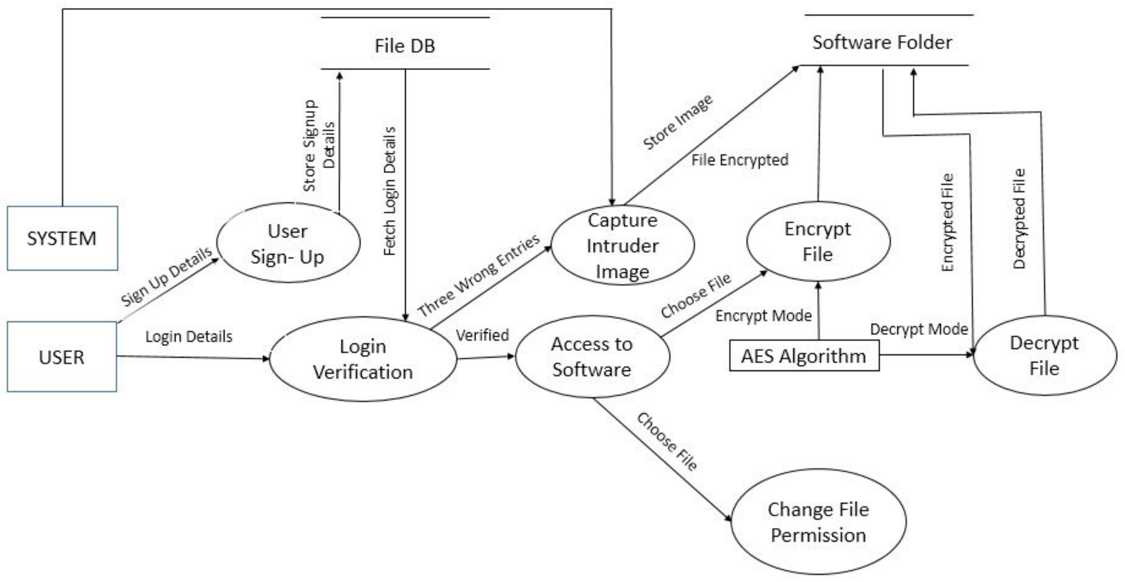
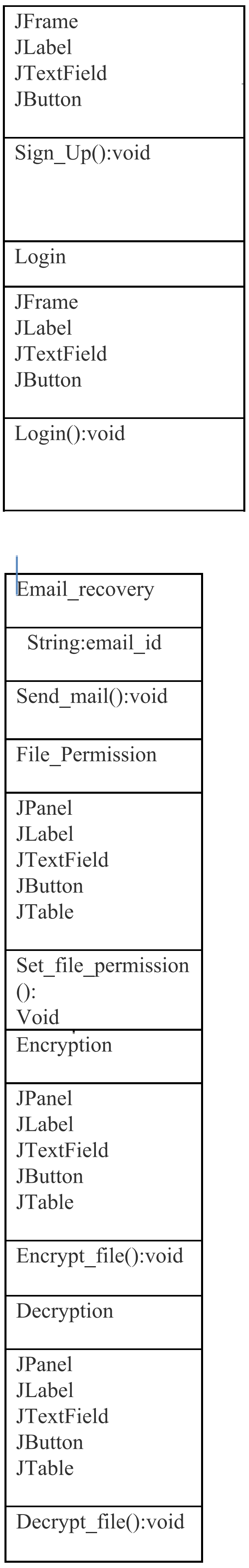


Fig.6 : Detailed DFD

4.3 Class diagram


JFrame
JLabel
JTextField
JButton
Sign_Up():void
Login
JFrame
JLabel
JTextField
JButton
Login():void
Email_recovery
String:email_id
Send_mail():void
File_Permission
JPanel
JLabel
JTextField
JButton
JTable
Set_file_permission():
Void
Encryption
JPanel
JLabel
JTextField
JButton
JTable
Encrypt_file():void
Decryption
JPanel
JLabel
JTextField
JButton
JTable
Decrypt_file():void

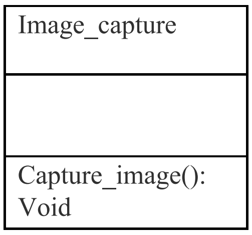


Fig.7: Class diagram

We have used inheritance by extending Jframes. We have tried our best to eliminate the irrelevant and amplify the essential while designing the GUI. We have encapsulated all the methods and variables related to a module into one class. All processing data are hidden from end users. The end users need not worry about the way we are implementing the system rather use it to store private information. Thus we have achieved data abstraction.

4.4 Class coupling matrix

	Sign_up	Login	File_Permissions	Encryption	Decrypyion	Image_capture	Mail_recovery
Sign_up	-	Data Coupling	-	-	-	-	Data Coupling
Login	-	-	Interaction	Interaction	Interaction	Interaction	Interaction
File_permissions	-	-	-	-	-	-	-
Encryption	-	-	-	-	-	-	-
Decryption	-	-	-	Data Coupling	-	-	-
Image_capture	-	Interaction	-	-	-	-	-
Email_recovery	-	-	-	-	-	-	-

Table 3: Coupling Matrix

5. IMPLEMENTATION

5.1 Introduction

We have implemented the modules in pair of two. Pair programming is an agile software development technique in which two programmers work together at one workstation. One of us was the driver who wrote code and the other, the observer, pointer or navigator, reviewed each line of code as it is typed in. the two of one working on one module switch roles frequently. While reviewing the observer came up with ideas for improvements and likely future problems to address. The module for login, sign up, GUA, mail recovery and other GUI was pair programmed by Shivendra and Chetan. The modules for encryption, decryption, image capture and others was pair programmed by Raghavi and Saurabh. We also followed the test first concepts of extreme programming. Test driven programming was helpful in writing bug-free code.

5.2 Module Description

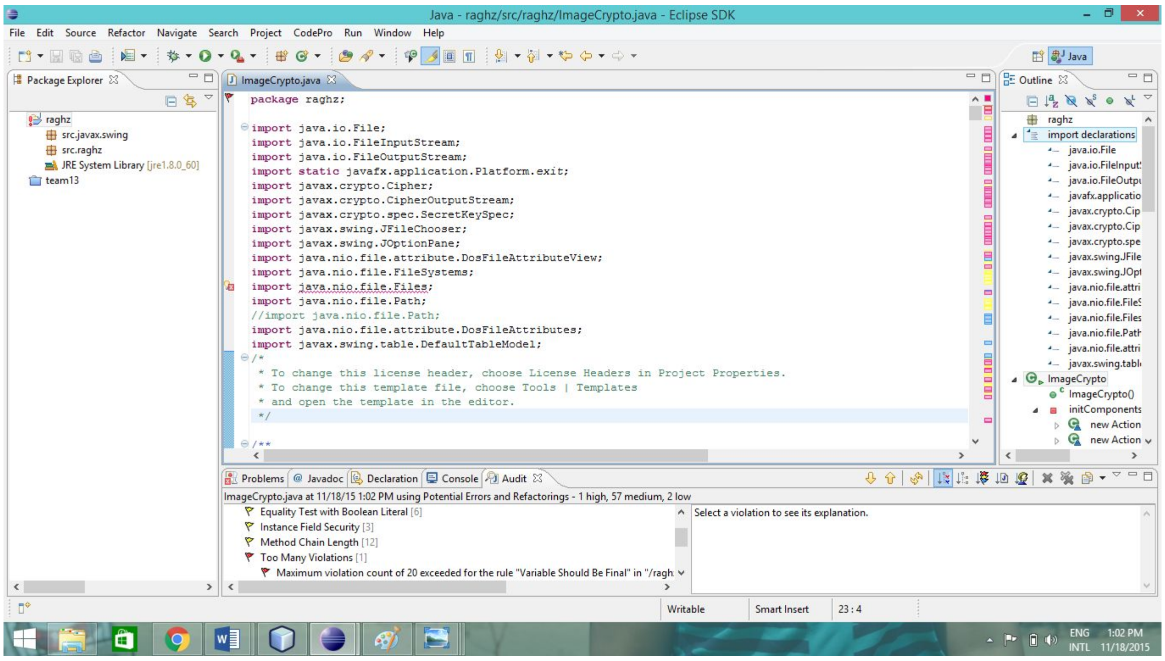
5.2.1 Sign up

Input – The user details are taken for the first time and stored.

Output – The user is acknowledged of successful sign up and a new account is created.

Pseudocode – The user information is stored in a file for further use and encrypted to avoid unauthorized access.

CodePro Review

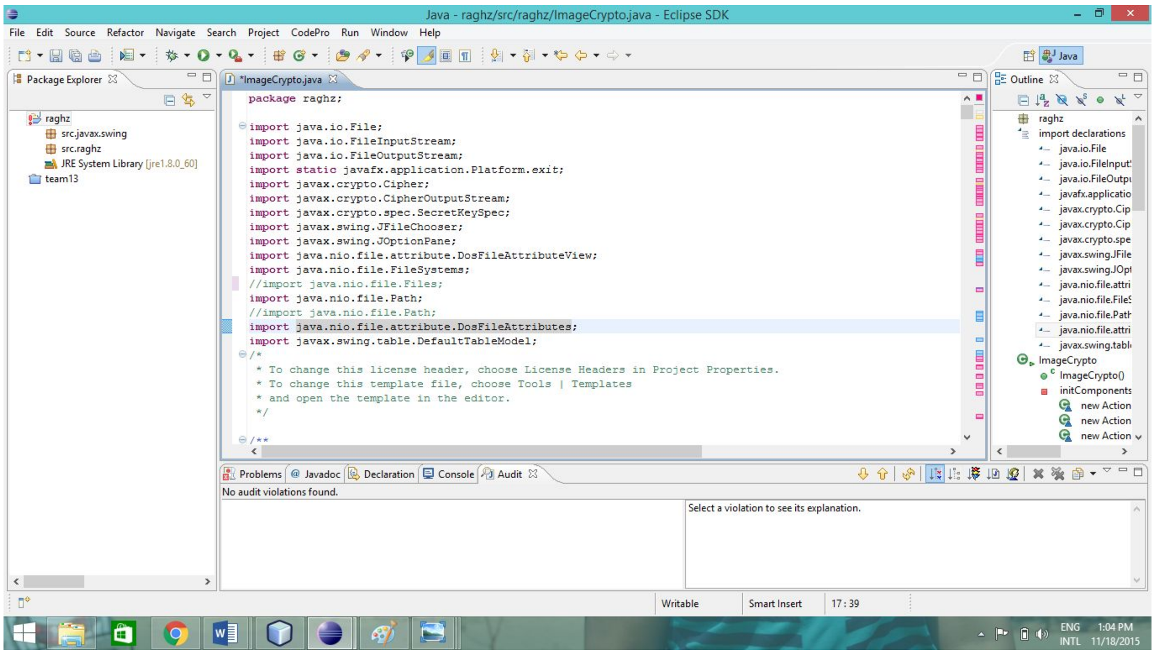


Java - raghz/src/raghz/ImageCrypto.java - Eclipse SDK

```java
package raghz;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import static javafx.application.Platform.exit;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import java.nio.file.attribute.DosFileAttributeView;
import java.nio.file.FileSystems;
import java.nio.file.Files;
import java.nio.file.Path;
//import java.nio.file.Path;
import java.nio.file.attribute.DosFileAttributes;
import javax.swing.table.DefaultTableModel;
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
```

```
ImageCrypto.java at 11/18/15 1:02 PM using Potential Errors and Refactorings - 1 high, 57 medium, 2 low
    Equality Test with Boolean Literal [6]
    Instance Field Security [3]
    Method Chain Length [12]
    Too Many Violations [1]
        Maximum violation count of 20 exceeded for the rule "Variable Should Be Final" in "/ragh

Select a violation to see its explanation.

Writable    Smart Insert    23 : 4
```

Fig.8: Snapshot of codepro review for sign up module



```
package raghz;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import static javafx.application.Platform.exit;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import java.nio.file.attribute.DosFileAttributeView;
import java.nio.file.FileSystems;
//import java.nio.file.Files;
import java.nio.file.Path;
//import java.nio.file.Path;
import java.nio.file.attribute.DosFileAttributes;
import javax.swing.table.DefaultTableModel;
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
```

```
No audit violations found.
Select a violation to see its explanation.
```

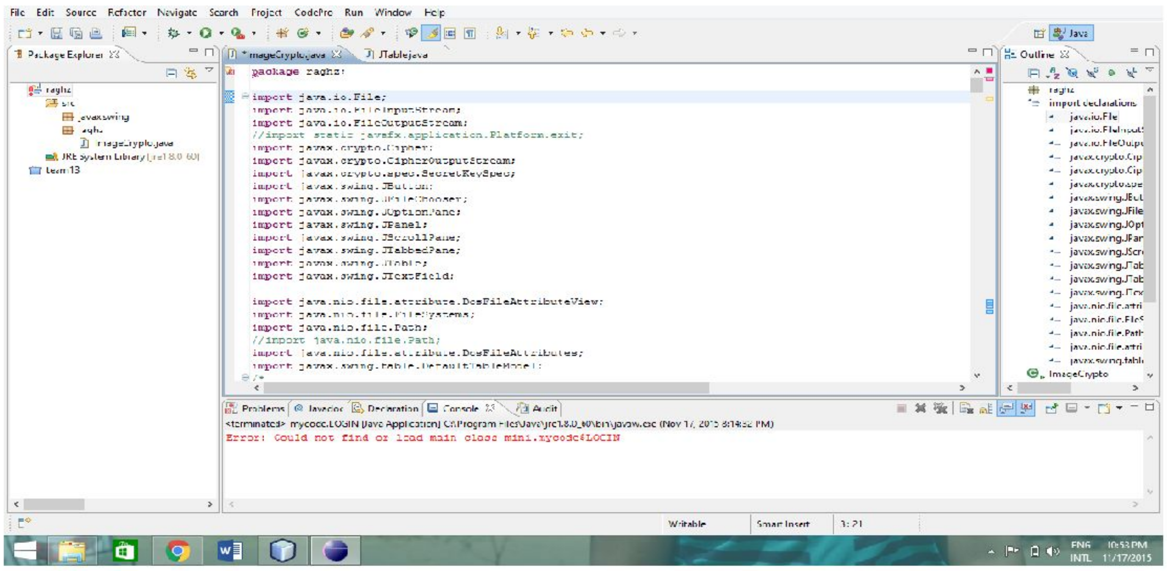
Fig.9: Snapshot of codepro review

5.2.2 Login

Input – User set username and password.

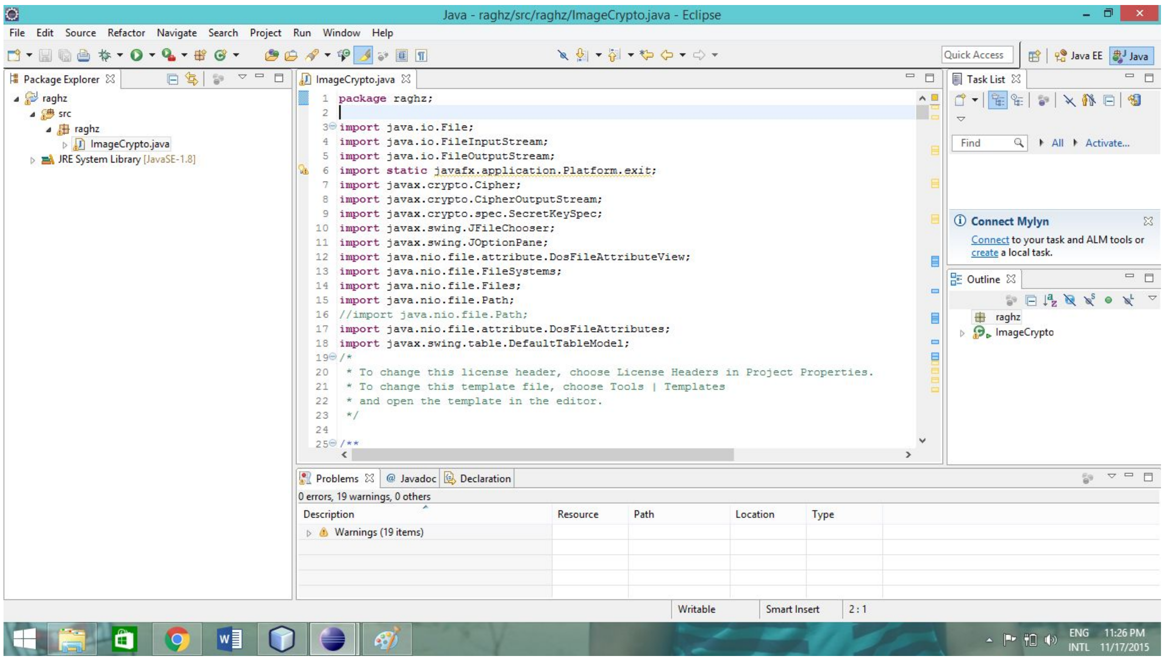
Output – The user is given access to the folder containing personal files.

Pseudocode – The entered username and password is checked with the stored username and password. If the username and password entered matches with the stored ones the folder opens, else the user is notified of wrong username and password.



```
package raghz;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
//import static javafx.application.Platform.exit;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JButton;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTabbedPane;
import javax.swing.JTable;
import javax.swing.JTextField;

import java.nio.file.attribute.DosFileAttributeView;
import java.nio.file.FileSystems;
import java.nio.file.Path;
//import java.nio.file.Path;
import java.nio.file.attribute.DosFileAttributes;
import javax.swing.table.DefaultTableModel;
/*
```

```
<terminated> mycode.LOGIN [Java Application] C:\Program Files\Java\jre1.8.0_60\bin\javaw.exe (Nov 17, 2015 8:14:52 PM)
Error: Could not find or load main class mini.mycode$LOGIN
```



Java - raghz/src/raghz/ImageCrypto.java - Eclipse

```
package raghz;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import static javafx.application.Platform.exit;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import java.nio.file.attribute.DosFileAttributeView;
import java.nio.file.FileSystems;
import java.nio.file.Files;
import java.nio.file.Path;
//import java.nio.file.Path;
import java.nio.file.attribute.DosFileAttributes;
import javax.swing.table.DefaultTableModel;
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
```

```
0 errors, 19 warnings, 0 others
Description                    Resource    Path    Location    Type
  Warnings (19 items)
```

Fig.10: Snapshot of codepro review for login module

5.2.3 Image Capture

Input – Three entries of wrong password.

Output – The image of intruding user is taken and stored.

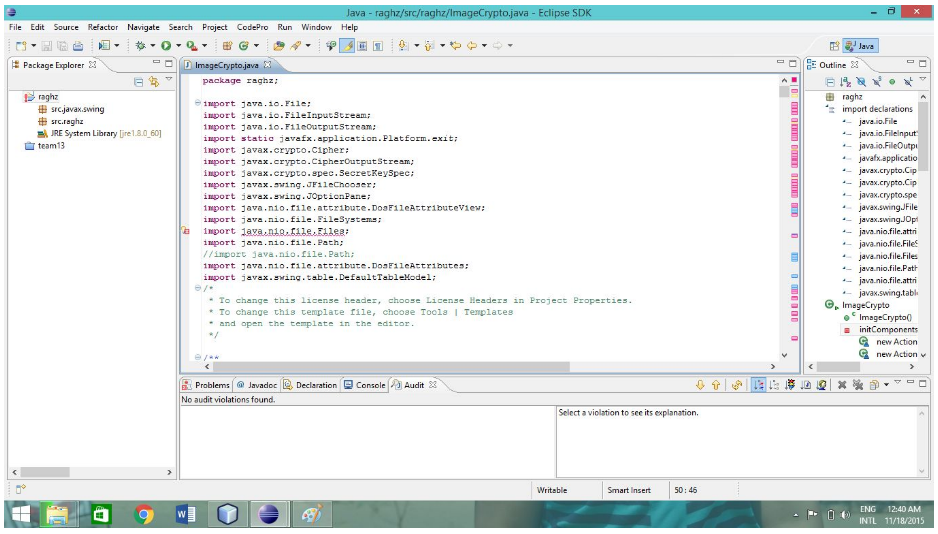
Pseudocode – On three wrong password entries the webcam takes a picture of the intruding user and saves the image in separate folder named Thief.

5.2.4 Encryption

Input – The user files that needs to be stored in the folder.

Output – The files encrypted and stored in folder.

Pseudocode – The personal files of user those are intended to be stored in folder is converted to byte array and by applying Bouncy castle algorithm the byte array file is encrypted and stored.



```
package raghz;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import static javafx.application.Platform.exit;
import javax.crypto.Cipher;
import javax.crypto.CipherOutputStream;
import javax.crypto.spec.SecretKeySpec;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import java.nio.file.attribute.DosFileAttributeView;
import java.nio.file.FileSystems;
import java.nio.file.Files;
import java.nio.file.Path;
//import java.nio.file.Path;
import java.nio.file.attribute.DosFileAttributes;
import javax.swing.table.DefaultTableModel;
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */

/**
```

```
No audit violations found.
Select a violation to see its explanation.
```

Java - raghz/src/raghz/ImageCrypto.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project CodePro Run Window Help
Package Explorer
raghz
src.javax.swing
src.raghz
JRE System Library [jre1.8.0_60]
team13
ImageCrypto.java
Outline
Problems
Javadoc
Declaration
Console
Audit
Writable
Smart Insert
50 : 46

Fig 11: Snapshot of codepro review for encryption module

5.2.5 Decryption

Input – The encrypted files from folder that needs to be taken out of folder.

Output – The original files in the same form as existed before putting in folder.

Pseudocode – The encrypted files from the folder is decrypted using the same AES algorithm, i.e, the byte array of file is converted back to its original format.

5.2.6 Mail Recovery

Input – User request for email recovery of password.

Output – The username and password set by the user at the time of signup is mailed to the users email ID.

Pseudocode – The user request for mail recovery is processed and username and password is mailed to the user’s email ID.

6. Testing

6.1 Introduction

Type of testing practice used: Unit testing

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual function or procedure. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing.

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

Unit testing finds problems early in the development cycle. This includes both bugs in the programmer's implementation and flaws or missing parts of the specification for the unit. The process of writing a thorough set of tests forces the author to think through inputs, outputs, and error conditions, and thus more crisply define the unit's desired behaviour. The cost of finding a bug before coding begins or when the code is first written is considerably lower than the cost of detecting, identifying, and correcting the bug later; bugs may also cause problems for the end-users of the software.

In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is considered to be a bug either in the changed code or the tests themselves. The unit tests then allow the location of the fault or failure to be easily traced.

6.2 Module Test Plan and Test Cases

6.2.1 Sign up

Test ID	Input Description	Expected Output
1	User Sets valid password	Password saved message displayed.
2	User Sets password and Password length insufficient.	Password length insufficient message displayed.
3	User doesn't enter email ID during signup.	Sign up incomplete message displayed.

Table 1: Test cases for sign up module

6.2.2 Login

Test ID	Input Description	Expected Output
1	User enters genuine password	Successfully logged in message displayed.
2	User enters wrong password	Invalid username and password message displayed.
3	User enters wrong username.	Invalid username and password message displayed.

Table 2: Test cases for login module

6.2.3 Image capture

Test ID	Input Description	Expected Output
1	User enters 3 wrong password	Image captured , no message displayed.
2	User enters 3 wrong password in system that doesn’t have a WEBCAM	Image not captured no message displayed.
3	User enters wrong username or password twice.	No image captured. Only message displayed.

Table 3: Test cases for Image capture module

6.2.4 Encryption

Test ID	Input Description	Expected Output
1	File chosen using browse option/ drag-drop	File encrypted message shown and encrypted file is stored.
2	Invalid File chosen using browse option/ drag-drop	File doesn’t exist message displayed.

Table 4: Test cases for encryption module

6.2.5 Mail Recovery

Test ID	Input Description	Expected Output
1	User chooses password reset option	Unique ID mailed
2	User chooses password reset option when there is no proper connectivity.	Mail not sent, password recovery failed message displayed.

Table 5: Test cases for mail recovery module

7. RESULTS

The Social Havaldar system is activated for the first time when the user performs Sign up operation successfully. The following screen appears when the user signs up for the first time.

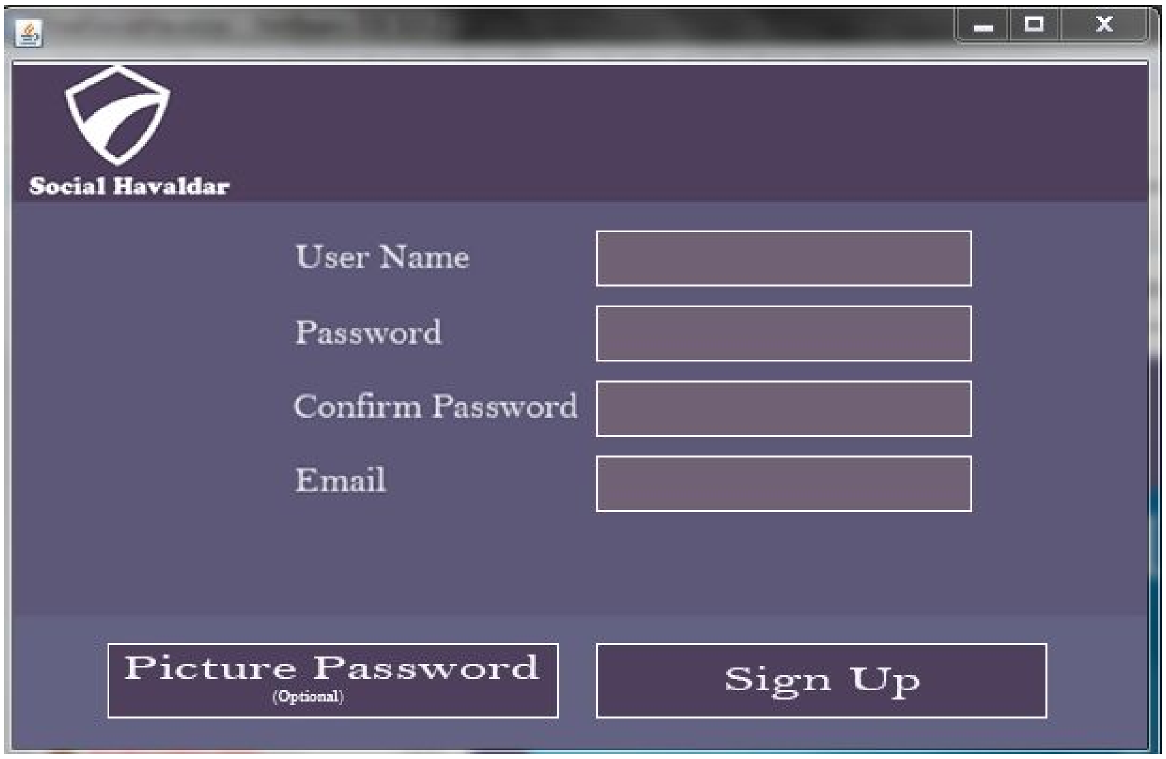


Fig 1: Snapshot of sign up page

Once the user signs up into Social Havaldar by setting valid username and password, and/or picture password the use can login into the system. The following screen appears when the user wants to login into Social Havaldar.

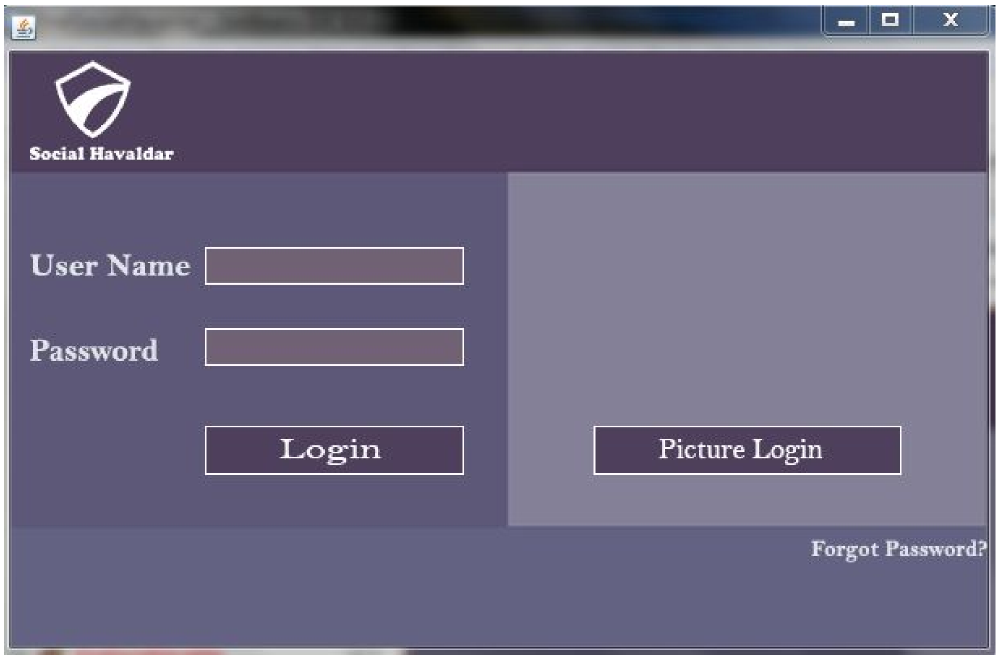


Fig 2: Snapshot of login window

The picture password which is the part of GUA is also an option for security of the folder. The following window appears when the user selects picture password.

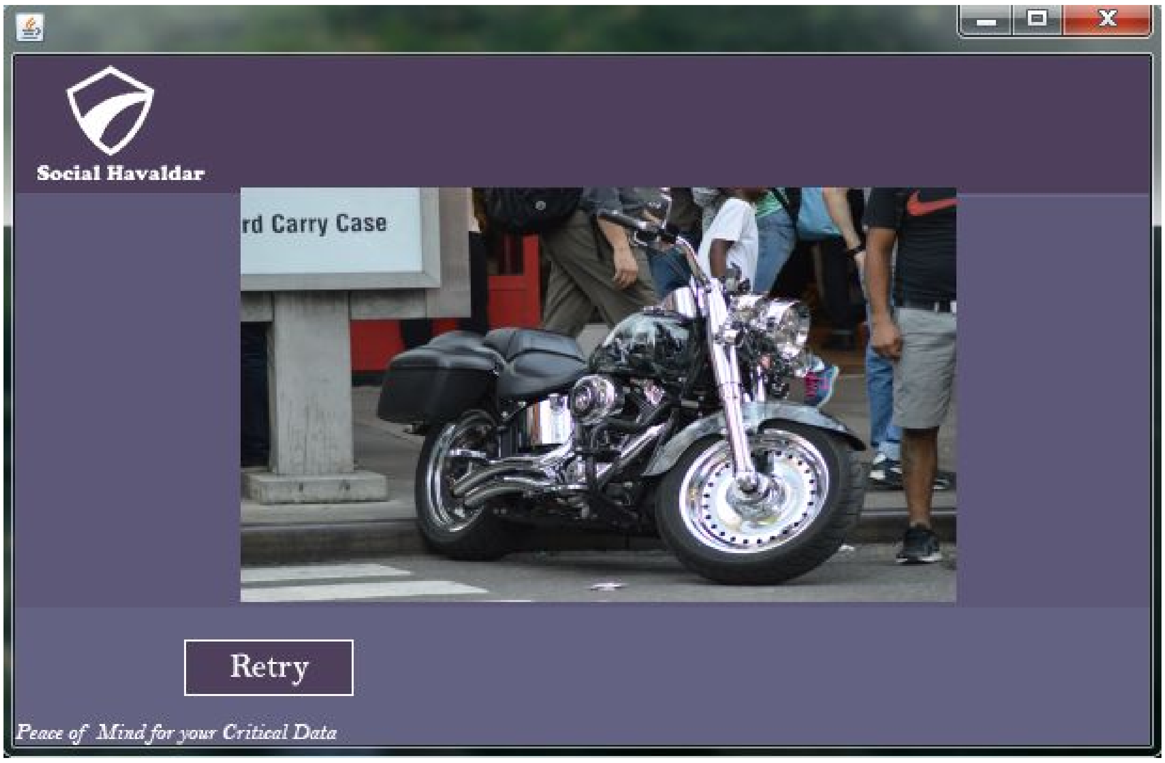


Fig 3: Snapshot of picture password window

Once the user performs the login operation successfully by entering correct username and password and/or picture password, user can choose any file they desire either by browse option or by drag and drop option and encrypt it. The user can also set file permissions to the selected file. The following screen appears for encryption.

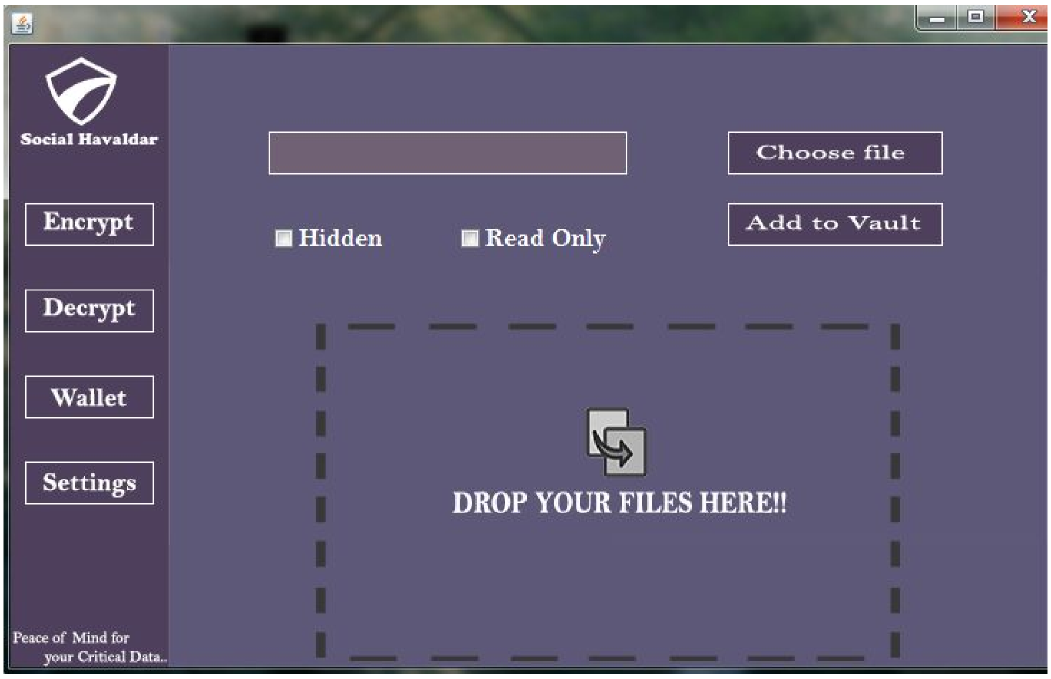


Fig 4: Snapshot of Encryption and file permissions window

The user can also decrypt the encrypted files anytime from the encrypted files folder. The following screen appears for decryption of encrypted files.

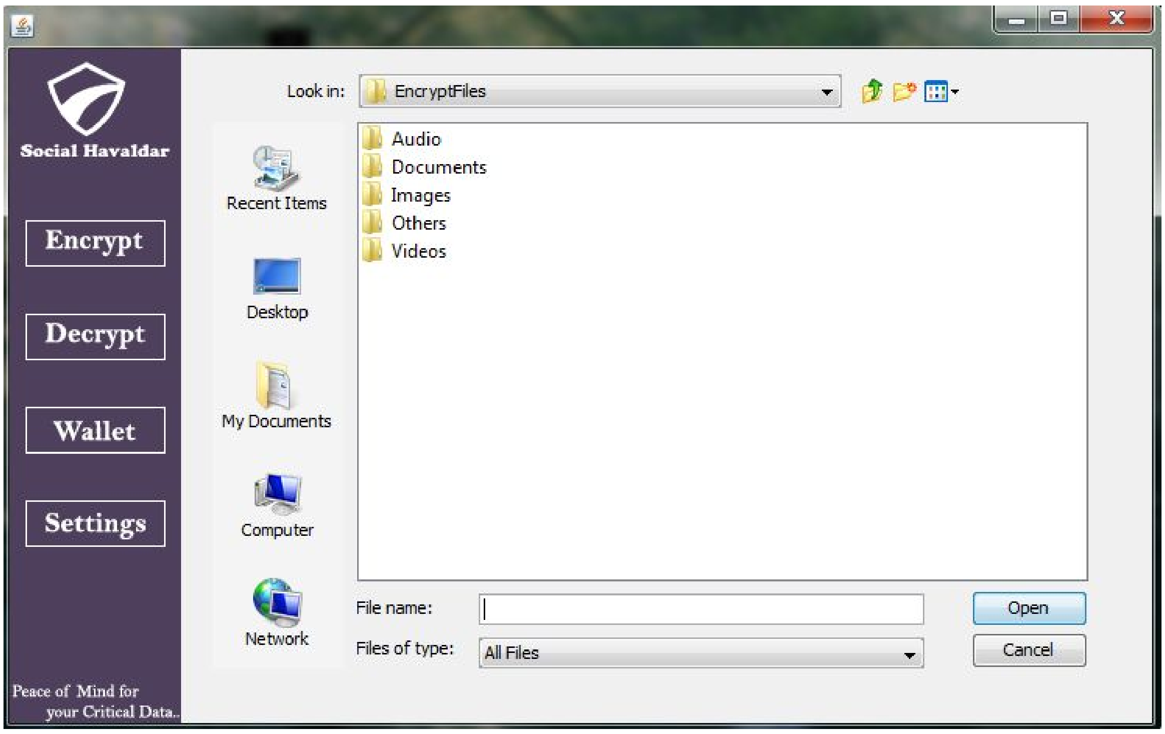


Fig 5: Snapshot of decryption window

In addition to all the functional requirements mentioned above in the SRS section, Social Havaldar also provides the feature to store information regarding wallet or credit/debit cards so that only user can access these private wallet information whenever required in any transactions. The following are the screens that appear for wallet storage.

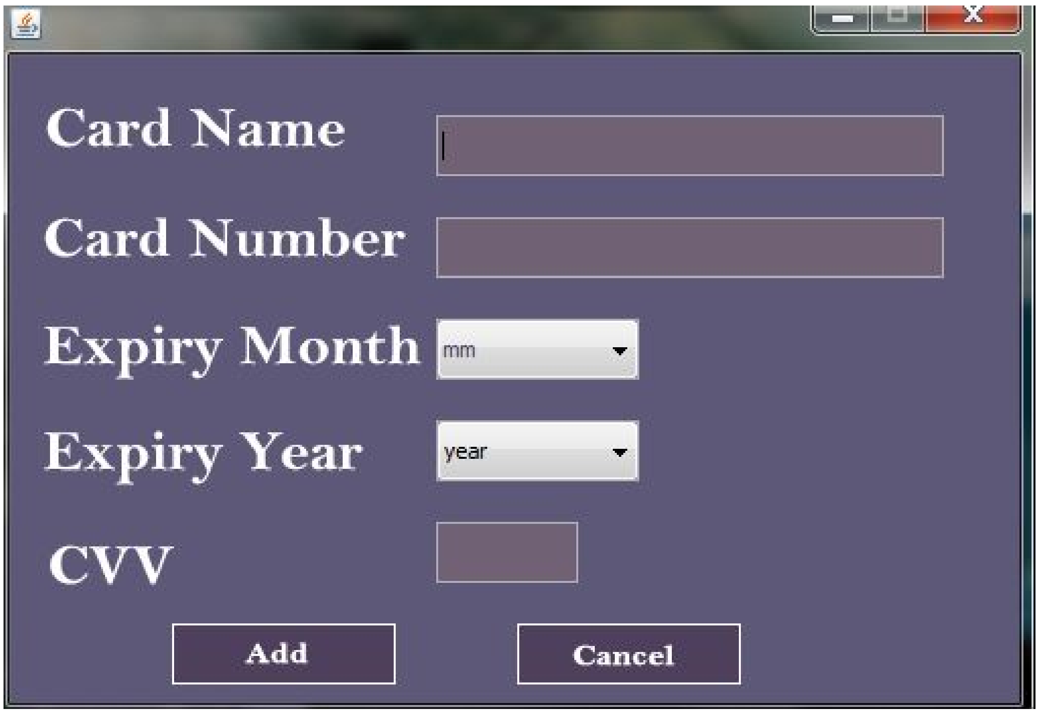
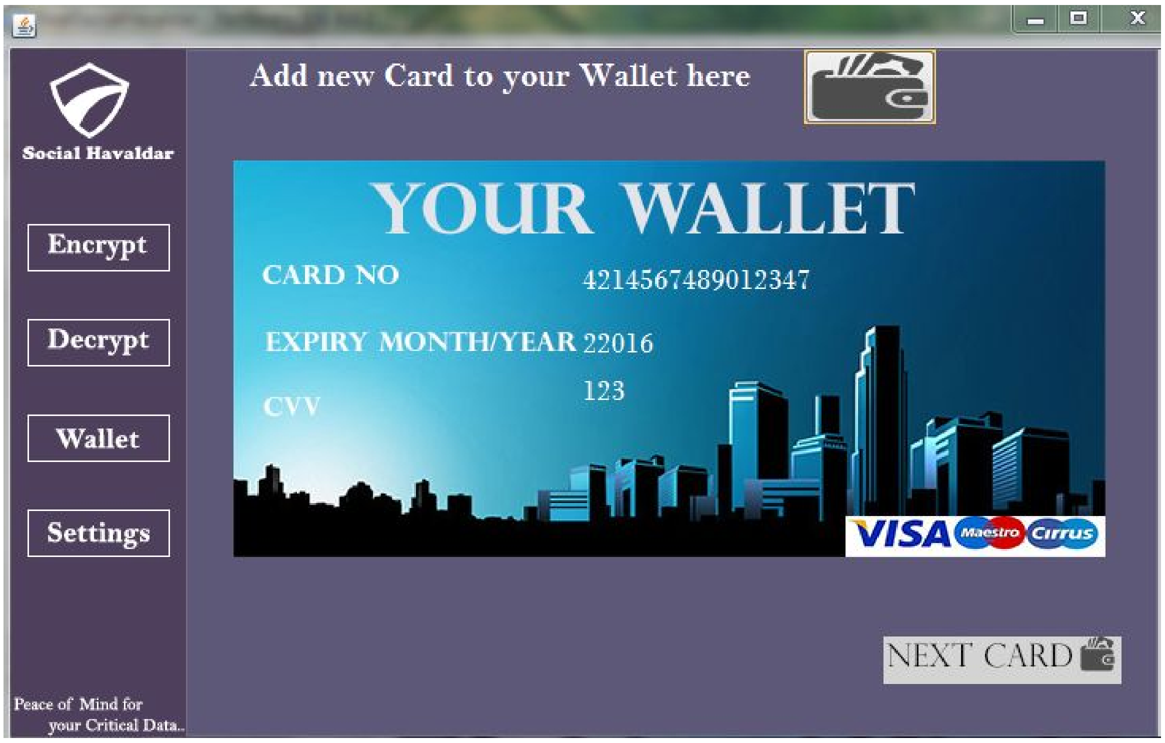


Fig 6: Snapshot of wallet storage



CARD NO 4214567489012347

EXPIRY MONTH/YEAR 22016

CVV 123

Fig 8: Snapshot of viewing the stored cards in wallet

8. SUMMARY OF OUR LEARNINGS

The project has helped us analyse the various stages of software development life cycle. We have understood each phase of the software development life cycle- problem analysis, design, coding, testing and implementation- effectively. The project has also helped us understand the Java coding language better. We have learnt the importance of all work products of different stages like SRS document, testing report, DFDs etc. In addition to all these aspects the project has helped us learn personal skills like team work, project management, compatibility and so on.

9. CONCLUSION AND FUTURE SCOPE

9.1 Conclusion

9.1.1 The utility can be used to store all personal files in a folder whose contents are accessed by only authorized users.

9.1.2 The utility provides the user with different file permissions for the selected file like hidden and read only.

9.1.3 The utility keeps track of intruders who try to hack into our system by taking the image of intruding users.

9.1.4 The utility also provides ability to recover passwords when password is forgotten by mail recovery.

9.1.5 The utility provides increased security to the files in the folder by Graphical User Authentication which enables the user to set a picture as password and can be unlocked only on specific number of clicks on specific spots on the picture.

Limitations:

The utility provides only one account and so only one user can maintain his files in the folder.

9.2 Future Scope

There is high scope for improvements in different aspects of our utility. The system can be extended to provide multiple accounts so that more than one user can maintain different folders in the same computer.

10. REFERENCES/BIBLIOGRAPHY

10.1 http://www.google.com

10.2 http://wikipedia.com

10.3 http://www.netbeans.org

10.4 http://www.opensuse.org

10.5
http://www.itcsolutions.eu/2011/08/24/how-to-encrypt-decrypt-files-in-java-with-aes-in-cbc-mode-using-bouncy-castle-api-and-netbeans-or-eclipse/

10.6 https://github.com/sarxos/webcam-capture

11. APPENDIX

11.1 Glossary

GUA – Graphical User Authentication

GUI – Graphical User Interface

Encryption – The conversion of data into a form(ciphertext) that cannot be easily read and understood by unauthorized people.

11.2 Explanation on JUnit tool

JUnit is a unit testing framework for the Java programming language. JUnit has been important in the development of test driven development, and is one of a family of unit testing frameworks which is collectively known as xUnit that originated with SUnit.

Junit promotes the idea of "first testing then coding", which emphasis on setting up the test data for a piece of code which can be tested first and then can be implemented. This approach is like "test a little, code a little, test a little, code a little…." which increases programmer productivity and stability of a program code that reduces programmer's stress and the time spent on debugging.

Features:

11.2.1 JUnit is an open source framework which is used for writing and running tests.

11.2.2 Provides Annotation to identify the test methods.

11.2.3 Provides Assertions for testing expected results.

11.2.4 Provides Test runners for running tests.

11.2.5 JUnit test allow you to write code faster which increases quality.

11.2.6 JUnit is elegant simple. It is less complex and takes less time.

11.2.7 JUnit tests can be run automatically and they check their own results and provide immediate feedback. There's no need to manually comb through a report of test results

11.2.8 JUnit tests can be organized into test suites containing test cases and even other test suites.

11.3 Explanation on CodePro

CodePro Analytix is the premier Java software testing tool for Eclipse developers who are concerned about improving software quality and reducing developmens costs and schedules. The Java software audit features of the tool make it an indispensable assistant to the developer in reducing errors as the code is being developed and keeping coding practices in line with organizational guidelines.

11.4 Explanation on Bouncy castle Algorithm

Bouncy castle is a collection of APIs used in cryptography. It includes APIs for bothjava and C# programming algorithms. The APIs are supported by a registered Australian charitable organization : Legion of the Bouncy Castle Inc.

The Bouncy castle architecture consists of two main components that support the base cryptographic capabilities. These are known as the 'light-weight' API, and the Java Cryptography Extension (JCE) provider.

The low level API is the set of API that implement all the underlying cryptographic algorithms. As the low level API is just java code, the Java Virtual Machine (JVM) does not impose any restrictions on the operation of the code.

The JCE compatible providers are built upon the low-level APIs. As such, the source code for the JCE provider is an example of how to implement many of the common crypto projects using low-level API.