# CHAPTER 1

# INTRODUCTION

## 1.1 Background Study

We study from internet about this topic “**File security system**” that what is going in this software and why it is necessary for a computer user and also study how can we modify it. And also the working of this type of software by study the different types of algorithm which are being used in this software .Basically the file security software which are present right now tries to alter the originality of the text into some encrypted form. The major task of File Security System is to provide the user the flexibility of passing the information implementing the encryption standards as per the specification and algorithms proposed and store the information in a form that is unreadable. The Application should have a reversal process as of which should be in a position to decrypt the data to its original format upon the proper request by the user.

Our main motive for making this software is that currently the file security system which are present right now in the market is for computer only but we make this for both pc and mobiles.

We are using **data encryption standard (DES)** algorithm for encryption and decryption of data. We also provide a user friendly GUI and we also encrypt the password provided by the user on its file so that no one can access it.

## 1.2 Project Scope

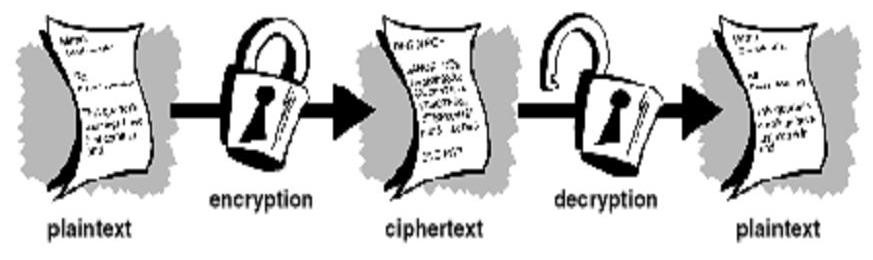
This file security system is different from other because we are try to decrease time consumed by the file for encryption and decryption. This system has a very wide scope in future because currently file security system which are present in market are not use for both pc and mobiles but our system is useful for both.It is low cost and effective so this can also be used by many user just like school and in companies

# CHAPTER 2

# OVERALL DESCRIPTION

## 2.1 Product Perspective

As we all know that this is a technical world and everything got computerized in every sector whether its private or government sector so there is large amount of confidential data are store in the computers so it is very important to secure these data from some kind of unwanted material here we realize the need of file security system this is the reason for origin. The file security system which we are using currently doesn’t provide the feature which a user want just like they are slow not working on all kind of files they just depend on a particular type. They store the password of any user in database without



Figure

any security so any one can access the database will take the user password and decrypt file also when the software is not installed on the pc there is no security on that files so these are the few reason where improvement is needed. Our product is a new product and also a replacement for the existing software of this type because it contained solution for all the problems which are present in the earlier file security system and also some more feature.

## 2.2 Product Features

The major task of File Security System is to provide the user the flexibility of passing the information implementing the encryption standards as per the specification and algorithms proposed and store the information in a form that is unreadable. The Application should have a reversal process as of which should be in a position to decrypt the data to its original format upon the proper request by the user.

* Our main motive for making this software is that currently the file security system which are present right now in the market is for computer only but we make this for both pc and mobiles.

* We are using **data encryption standard (DES)** algorithm for encryption and decryption of data .We also provide a user friendly GUI.

* We also encrypt the password provided by the user on its file so that no one can access it.

* There is no special memory or extra space or drive is needed for storing encrypted files. Files are remain same at the position where they are present before the encryption.

* If the software get uninstalled from the computer after that no one can access the file which are encrypted and the file will remain same in the encrypted form.

* Low cost and processing speed is faster than other file security system.

## 2.3 User Classes and Characteristics

* **Admin**: - There will be class admin which is the authorized person for these functions login, create new user, view user, edit, delete, password reset, show all files.

* **User**: - There will be class user which is authorized person for these function login, Secure a file, View list, Reset a file.

## 2.4 Operating Environment

**Hardware Requirements: -**

|  |  |
| --- | --- |
| **Hardware** | **Reason** |
| PROCESSOR: PENTINUM-2,INTEL-i7 | FOR HIGH PROCESSING SPEED |
| HARD DISK: 4GB TO 1TB | BELOW 40GB IT CAN’T STORE MORE DATA |
| RAM: 256 MB TO 4GB | BECAUSE THIS SOFTWEAR IS SMALL IN SIZE SO IT CAN WORK IN ANY SPEED |

Table 1: hardware requirement.

**Software Requirement: -**

|  |  |
| --- | --- |
| **Software** | **Reason** |
| OPERATING SYSTEM: XP,WINDOW 7,8 | IT CAN WORK IN ANY OF THEM |
| JAVA | WITHOUT JAVA IT CANNOT WORK |

Table 2: software requirement

**Server Side:**

* Web Server : APPACHETOMCAT 6.0 and above
* Database server : MY SQL5.0 and above  Operating System : Window XP or above  Processor: INTEL CORE i3.
* 1GB RAM

**Client Side:**

* A reliable internet connection. ADSL / Broadband connections are recommended.
* Operating System : Window XP or higher
* Processor: INTEL i3
* 256 RAM or higher
* Microsoft Office 2003 or higher

## 2.5 Design and Implementation Constraints

This system is designed for computer and mobiles.

* System should be java compatible because we are making this software in java and without java installation software cannot work.

* The second constraints for this system is currently we are making this for .exe files only.

## 2.6 Assumptions and Dependencies

In the development of this system assumption which have been made primarily is that which we are encrypted is present in the hard drive means in the computer. The second assumption is that the files should be of **.exe** format.

The most important assumption is that the system should be java compatible otherwise this security system should not work because it made in java.

**CHAPTER 3**

**EXTERNAL INTERFACE REQUIREMENTS**

## 3.1 User Interfaces

In this file security system both user and admin can interact with the system .Admin has a power of adding and deleting any new user or the existing one. But user can only browse the file which he or she want to encrypt or decrypt and rest of the work is down by the admin. Once this software is get installed and user can apply password on any file then the file get encrypt and cannot open until and unless we apply the right password to them.

* **Admin login:** - This is the login interface in which admin has to login himself first of all so that he can access the further options. Every authentic user has a username and password to login in the system.

User apply password on any file then the file get encrypt and cannot open until and unless we apply the right password to them.

* **Choice:** - Choice is the second interface of this system in which some choices are provided for admin. After logging in admin has to choose one of them like add a new user or delete the existing user.

* **Remove password:** - This is the another interface for the admin in this admin can remove password from that files whose password are forgot by the user by applying the master password.

These all are the user interface to system.

## 3.2 Hardware Interfaces

The user will interact with the software through: monitor, keyboard, & mouse; Input will be received via mouse and keyboard, and feedback provided through the monitor GUI.

#### Software Interfaces

1. PC : - This system is has been currently developed for computers because this is computerized world and there is a need to provide security to the data. So for this project we have take a pc with minimum hardware and software requirement and also java compatible.

1. Eclipse-IDE-Galileo : - In this project we have used eclipse-IDE-Galileo(integrated development environment) to code the project.

1. ArgoUML: - To make UML diagrams we have used ArgoUML tool.

1. EDraw: - This software has been used in our project to make E-R diagram and Data Flow Diagrams.

1. We have used MySQL as database for our project to store all records related to file informations , password, user name , user id etc.

1. Visual Paradigm11.2:- This software is used for making CRC and use case diagram.

## 3.3 Communications Interfaces

We can use GUI (graphical user interface) for communications interface.

# CHAPTER 4

# OTHER NONFUNCTIONAL REQUIREMENTS

## 4.1 Performance Requirements

For best performance the system speed should be fast and also java compatible.

## 4.2 Safety Requirements

The safety requirement for file security system is that the software do not get uninstalled otherwise the files which are encrypted will be lost or not in the position to access them. Second thing is that the user cannot forget his username or password because without it user is not possible to access the file. The file which is encrypted should not be deleted by someone.

## 4.3 Security Requirements

## 1:-Authentication

Using a cryptographic system, we can establish the identity of a remote user (or system). A typical example is the SSL certificate of a web server providing proof to the user that he or she is connected to the correct server.

The identity is not of the user, but of the cryptographic key of the user. Having a less secure key lowers the trust we can place on the identity.

### 2:- Non-Repudiation

The concept of non-repudiation is particularly important for financial or e-commerce applications. Often, cryptographic tools are required to prove that a unique user has made a transaction request. It must not be possible for the user to refute his or her actions.

For example, a customer may request a transfer of money from her account to be paid to another account. Later, she claims never to have made the request and demands the money be refunded to the account. If we have non-repudiation through cryptography, we can prove – usually through digitally signing the transaction request, that the user authorized the transaction.

### 3:-Confidentiality

More commonly, the biggest concern will be to keep information private. Cryptographic systems were originally developed to function in this capacity. Whether it be passwords sent during a log on process, or storing confidential medical records in a database, encryption can assure that only users who have access to the appropriate key will get access to the data.

### 4:-Integrity

We can use cryptography to provide a means to ensure data is not viewed or altered during storage or transmission. Cryptographic hashes for example, can safeguard data by providing a secure checksum.

#### 5:-Brute force attack

For any cipher, the most basic method of attack is brute force — trying every possible key in turn. The [length of the key](http://en.wikipedia.org/wiki/Key_length) determines the number of possible keys, and hence the feasibility of this approach. For DES, questions were raised about the adequacy of its key size early on, even before it was adopted as a standard, and it was the small key size, rather than theoretical cryptanalysis, which dictated a need for a replacement algorithm. As a result of discussions involving external consultants including the NSA, the key size was reduced from 128 bits to 56 bits to fit on a single chip.

# CHAPTER 5

# DESIGN SPECIFICATIONS

## 5.1 Assumptions

In the development of this system assumption which have been made primarily is that which we are encrypted is present in the hard drive means in the computer. The second assumption is that the files should be of **.exe** format.

The most important assumption is that the system should be java compatible otherwise this security system should not work because it made in java.

## 5.2 Constraints

This system is designed for computer and mobiles.

* System should be java compatible because we are making this software in java and without java installation software cannot work.

* The second constraints for this system is currently we are making this for .exe files only.

## 5.3 Design Methodology

We are using DES (data encryption standard) algorithm for this tool to encrypt or decrypt the text

## 5.4 Key schedule:-

The algorithm which generates the sub-keys. Initially, 56 bits of the key are selected from the initial 64 by *Permuted Choice 1* (*PC-1*) — the remaining eight bits are either discarded or used as parity check bits. The 56 bits are then divided into two 28-bit halves; each half is thereafter treated separately. In successive rounds, both halves are rotated left by one and two bits (specified for each round), and then 48 sub-key bits are selected by *Permuted Choice 2* (*PC-2*) — 24 bits from the left half, and 24 from the right. The rotations (denoted by "<<<" in the diagram) mean that a different set of bits is used in each sub-key; each bit is used in approximately 14 out of the 16 sub-keys.

The key schedule for decryption is similar — the sub-keys are in reverse order compared to encryption. Apart from that change, the process is the same as for encryption. The same 28 bits are passed to all rotation boxes.

## 5.5 Risk and Volatile areas

The key purpose of classifying risk is to get collective view point on a group of factors. These are the types of factor which will help project managers to identify the group that contributes the maximum risk. A best and most scientific way of approaching them is based on attributes. Risk classification is considered as an economical way of analyzing risk and their cause by grouping similar risk together into classes. Software risks could classified as internal or external. Those risks

That come from risk factors within the organization call internal risk whereas the external risk come from out of the organization and are difficult to control. Internal risk are project risks, process risks and product risks. External risk are generally business with the vendor, technical risks, customer’s satisfaction and so on. In general there are many risk in software engineering which is very difficult to identify all of them. Some of most important risk in software engineering project are categorized as software requirement risks, software cost risks, software scheduling, software quality risks and software business risks. These risks are detail below.

* SOFTWARE REQUIREMENT RISKS

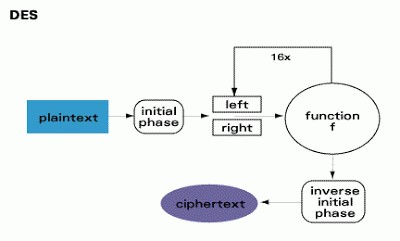
* 1. Lack of analysis for change of requirements.
  2. Change extension of requirements.
  3. Lack of report for requirements.
  4. Poor definition of requirements.
  5. Ambiguity of requirements.
  6. Change of requirements.
  7. Inadequate of requirements.
  8. Impossible requirements.
  9. Invalid requirements.

* SOFTWARE COST RISKS

* 1. Lack of good estimation in projects.
  2. Unrealistic schedule.
  3. The hardware does not work well.
  4. Human errors.
  5. Lack of testing.
  6. Lack of monitoring.
  7. Complexity of architecture.
  8. Large size of architecture.
  9. Extension of requirements change.
  10. The tool does not work well.
  11. Personnel change, Management change, Technology change, Environment change.
  12. Lack of reassessment of management cycle.

**CHAPTER 6**

# ARCHITECTURE



Figure

# CHAPTER 7

# OVERVIEW

This system is a software that is use to secure a file from un-authorized user. It can be used in public and private sector. It mostly used in military services and intelligence bureau. Where every document is a secret from even those who work with them. Anyone can use who want to make privacy of his/her file.

## 7.1 Modules

The modules of this system are as following:-

* **Login:** - This is the login interface in which admin or user has to login himself/herself first of all so that he can access the further options. Every authentic user has a username and password to login in the system.

* **Choice:** - choice is the second interface of this system where admin/user has to choose which action they want to perform after login. Admin can add, delete, edit user, and he can reset password and view all file. User can add file, view file, reset file.

* **Add/Delete file:** - this interface is only for user of system. He/she can secure a file by simply adding file to system or he/she can view file by simply resetting it.

* **Add/Delete/Edit user:** - this interface is only for admin, where he/she can add user of system or delete user from system.

* **View files:** - admin or user can view file list by this interface. User can view only file which he/she secured in past. But admin can view all the files which all the user secured in past.

* **Chance password:** - this interface is for only admin where he/she can change admin password or master password.

# CHAPTER 8

# UML DIAGRAMS

The Unified Modeling Language (UML) is a general-purpose modeling language in the field of software engineering, which is designed to provide a standard way to visualize the design of a system. The UML diagrams of this project are as following:-

## 8.1 Requirement Paragraph: -

Make a software which can be used to secure a file from unauthorized user. Files are of only data file (not mp3 or mp4 or any media file).Any cryptographic technique can be used. Only two, admin and user can access software. Admin can create user, view user, edit user, and view all files. User can protect a file, view file and reset a file. To access the software it is very necessary that admin and user has to login first.

## 8.2 Noun Phrase Analysis: -

After analyzing the requirement paragraph we can see that there are following nouns:-

1. Software
2. File
3. User
4. Admin
5. Cryptographic technique

From the upper list of nouns we can easily say that users of this system are:- 1. Admin

2. User

List of verbs which can be find out from the paragraph are:- 1. Login

1. Secure
2. Create user
3. Edit user
4. View user
5. View all user
6. Secure a file
7. View files
8. Reset a file

## 8.3 Use Case Diagram: -

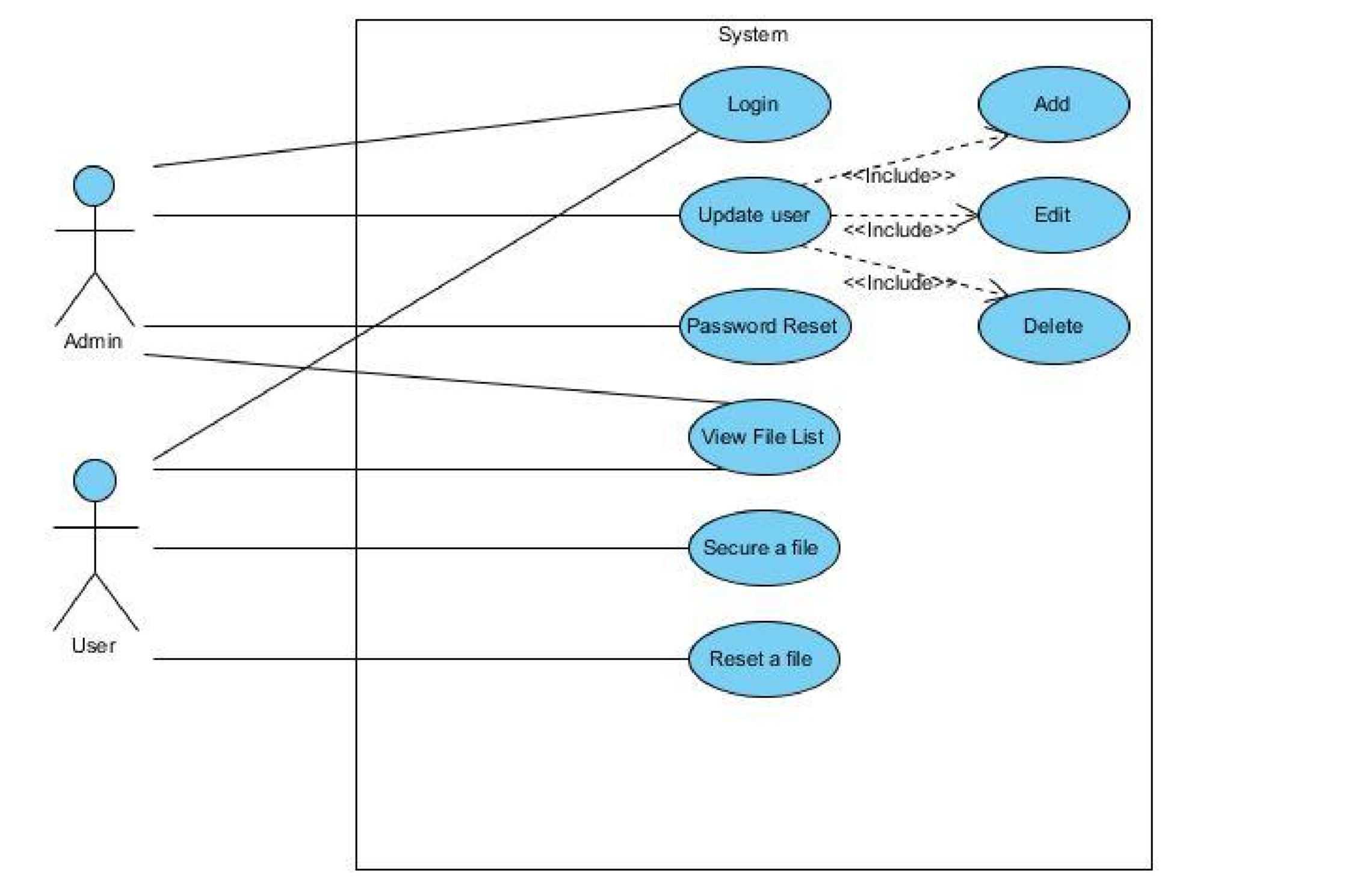
A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well. The use case diagram of this File Security system is as following:-

Figure : USE CASE DIAGRAM OF SYSTEM

**Description of above diagram:**

In the above use case diagram there are two **actors** admin and user. They are only two who can access the software according to their level of access. Admin can do login, update user, password reset and view file list. While User can login, view file list, secure a file and reset a file.

## 8.4 Sequence Diagram: -

A **Sequence diagram** is an interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams** or **event scenarios**. Sequence diagram for this project is as following:-

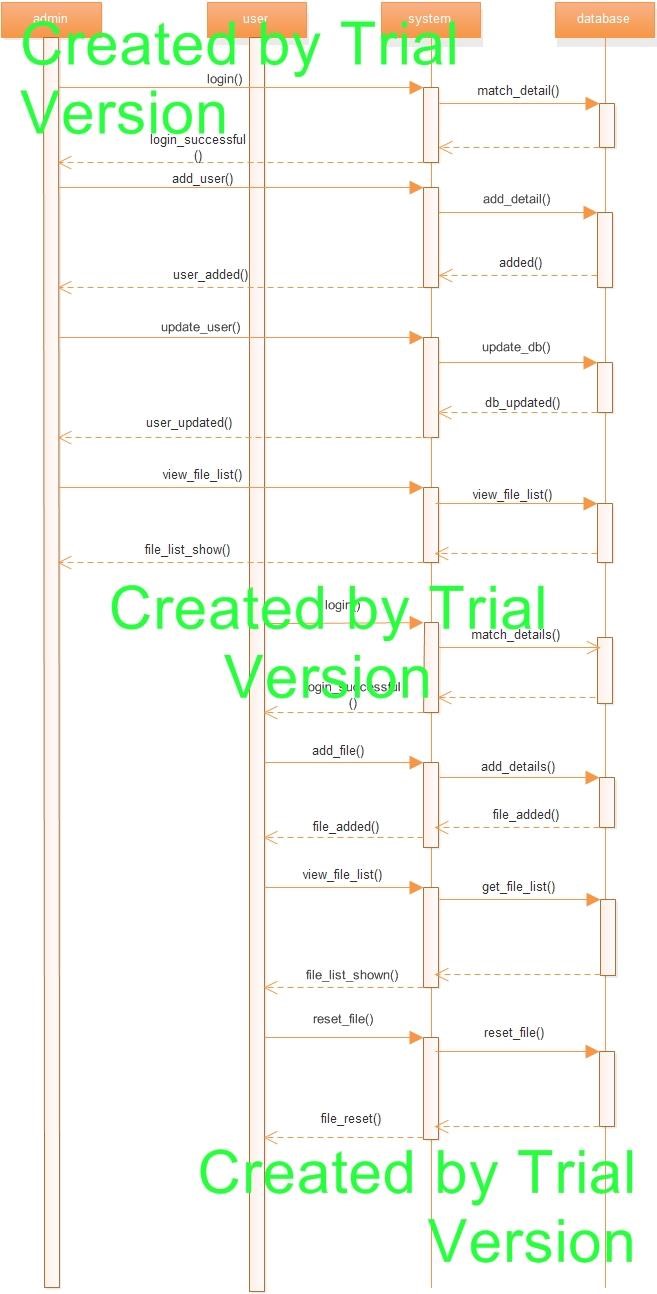


Figure : SEQUENCE DIAGRAM

**Description of above diagram:**

In above diagram there are four life-line which are as follows admin, user, system and database. Explaining above diagram let’s take login function of admin, here firstly login function is called by the admin to the system, from system it check the username and password from the database. If the provided information by the system is correctly matched with the database then the authentication is provided by the system to the admin. This is how function are work through the life-line i.e. from one life-line to another. Whenever any function is called by any life-line it checked to its corresponding life-line. Here life-line shows the different levels of software.

## 8.5 Data Flow Diagram: -

A **data flow diagram** (**DFD**) is a graphical representation of the "flow" of data through an information system, modeling its *process* aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated.DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart). Data Flow Diagram for this system is as following:-

1. **Level DFD:**

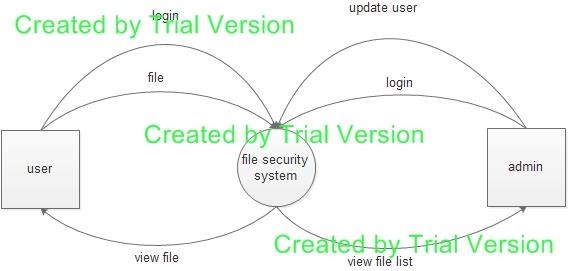


Figure : 0 LEVEL DFD

1. **Level DFD:**

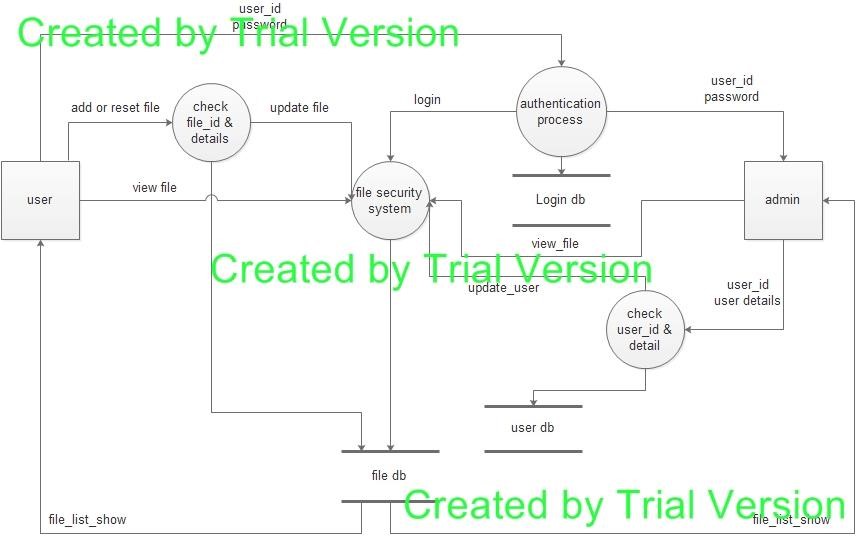


Figure : LEVEL 1 DFD

## 8.6 Activity Diagram: -

**Activity diagrams** are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e. workflows).Activity diagrams show the overall flow of control. Various activity diagrams for this system are as following:-

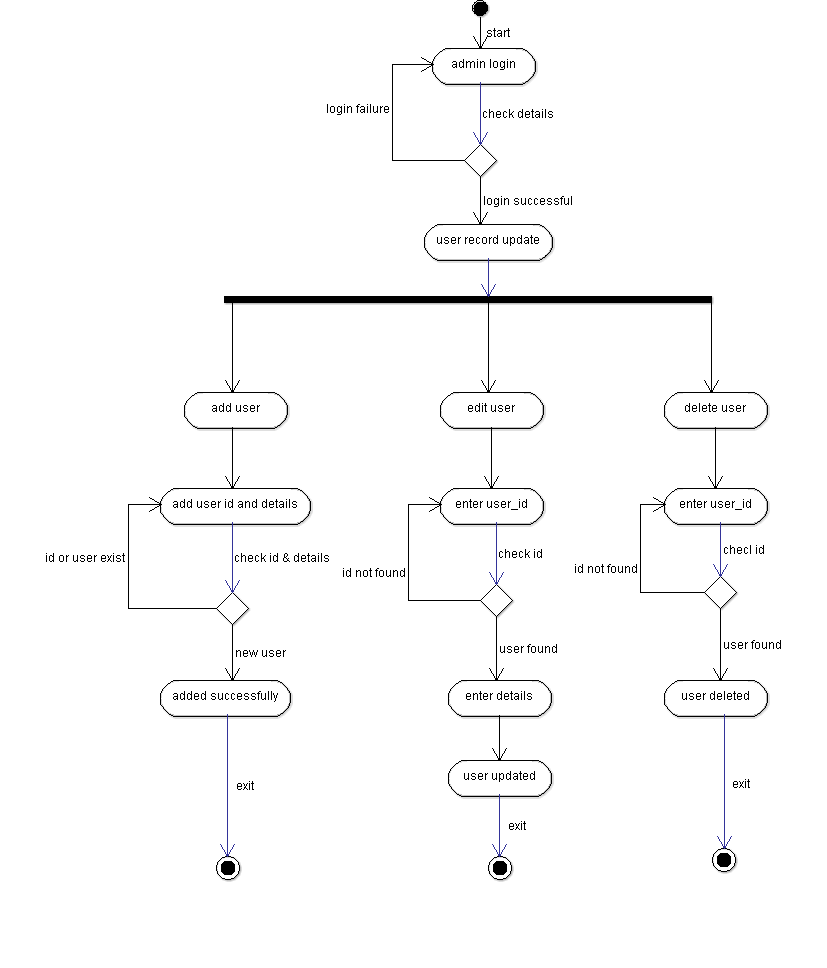


Figure : ACTIVITY DIAGRAM OF UPDATE USER.

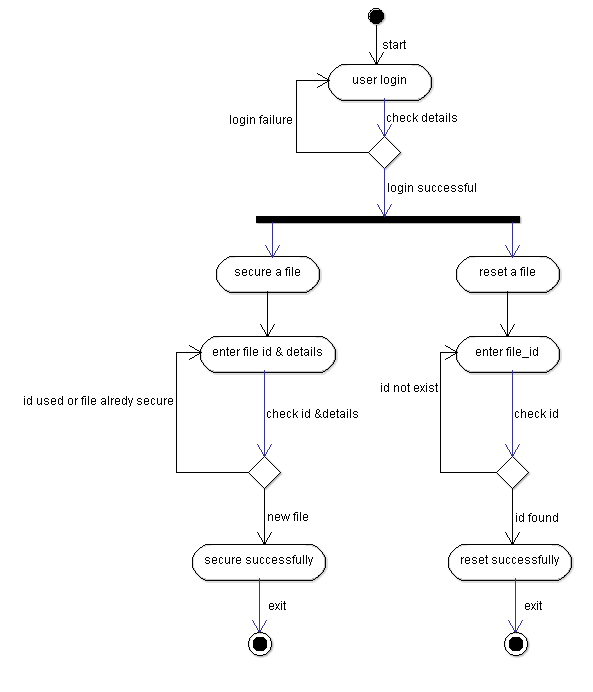


Figure : ACTIVITY DIAGRAM OF UPDATE FILE LIST.

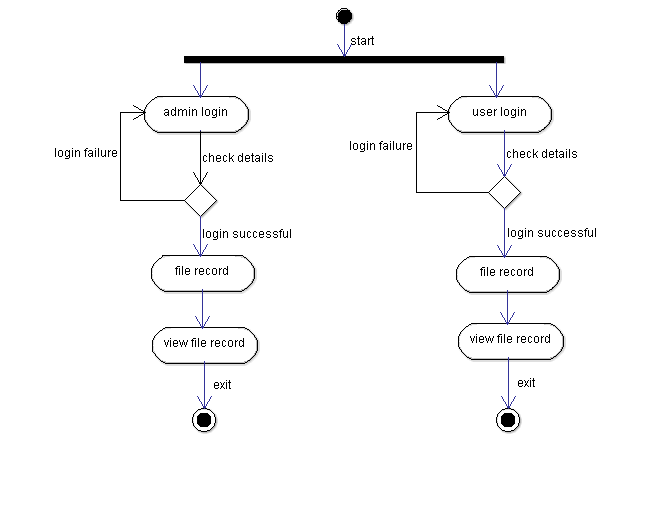


Figure : ACTIVITY DIAGRAM OF VIEW FILE LIST.

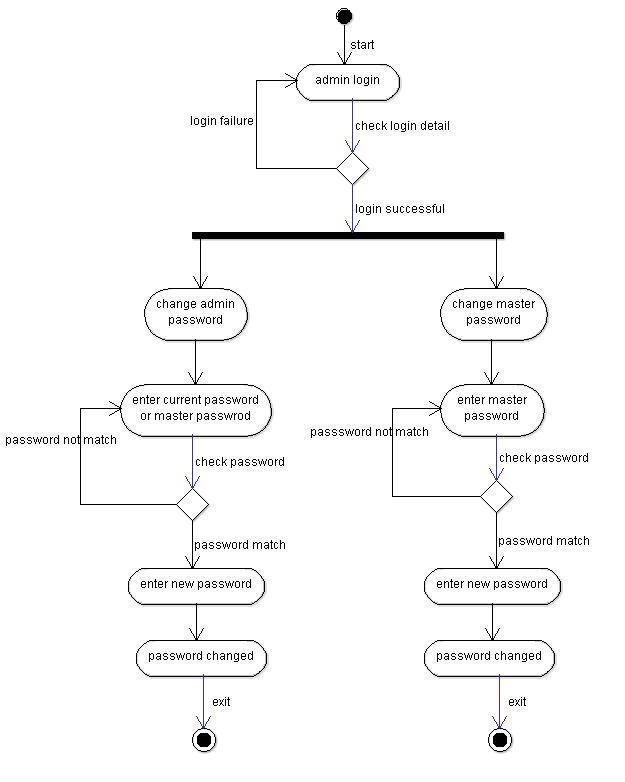


Figure : ACTIVITY DIAGRAM OF PASSWORD RESET

## 8.7 Class-Responsibility-Collaboration Diagram:-

CRC Cards are used to represent objects. The class of the object can be written at the top of the card, responsibilities listed down the left side, collaborating classes are listed to the right of each responsibility. We say "can be written" because once a CRC session is in full swing participants usually only need a few cards with the class name and virtually no cards written out in full. The CRC cards diagram for this system is as following:-



Figure : CRC CARDS

## 8.8 E-R Diagram: -

In software engineering, an **entity–relationship model** (**ER model**) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them, and databases. The Entity – Relationship diagram for this system is as below:-

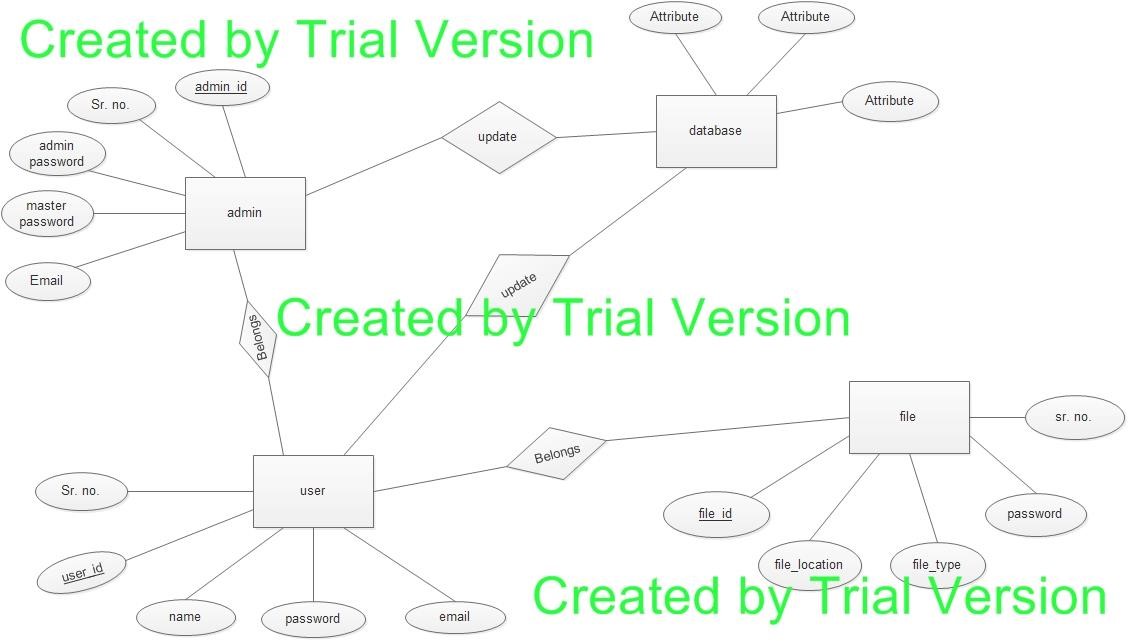


Figure : ER DAIGRAM

# CHAPTER 9

# DATABASE SCHEMA

## 9.1 Tables, Fields and Relationships

In this project to store data we have made following tables:-

#### 1. Admin table: -

This table is used to store the data of the admin of the system. This table has three fields username, password and name. Whenever an admin login to the system then his details are matched from this table. If the details are matched then he get access. The detailed schema of the table is as following:-

|  |  |  |  |
| --- | --- | --- | --- |
| Field-Name | DataType | Allows Null | Default Value |
| S.No | int(10) | No | none |
| Admin Id | int(10) | No | none |
| Admin Passward | varchar(50) | No | None |
| Email Id | varchar(50) | No | none |

Table 3: Admin table

#### 2. File Information: -

This table is used to store all the information about the encrypted file and these information are s.no, file id, file location, file type, password.

|  |  |  |  |
| --- | --- | --- | --- |
| Field-Name | DataType | Allows Null | Default Value |
| S.No | int(10) | No | None |
| File Id | int(10) | No | None |
| File location | varchar(50) | No | None |
| File type | varchar(50) | No | None |
| password | varchar(50) | No | None |

Table 4: File table

#### 3. User information: -

This table store the information of the user for example s.no, user id, user password, email.

|  |  |  |  |
| --- | --- | --- | --- |
| Field-Name | DataType | Allows Null | Default Value |
| S.No | int(10) | No | None |
| User Id | int(10) | No | None |
| User Password | varchar(50) | No | None |
| Email | varchar(50) | No | None |

Table 5: User table

### 9.1.1 Databases

In our project we are using MySQL database for storing our data. The data will be on server which is Apache 2.

# CHAPTER 10

# COST ESTIMATION

After reading all the requirements, layouts and functions of this proposed system the cost can be estimated on the basis of following points:-

1. As the system is for colleges and institutions so the database will be large so a good database management is required.
2. 3-4 skilled people (software developer) are required.
3. Approximately about 6 months’ time is required.

The basic COCOMO equations take the form:-

**Effort Applied (E)** = ab(SLOC)bb **[ person-months ]**

**Development Time (D)** = cb(Effort Applied)db **[months]**

**People required (P)** = Effort Applied / Development Time **[count]**

Where, **SLOC** is the estimated number of delivered lines (expressed in thousands ) of code for project. The coefficients *ab*, *bb*, *cb* and *db* are given in the following table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Software Project** | **ab** | **bb** | **cb** | **db** |
| organic | 2.4 | 1.05 | 2.5 | 0.38 |
| Semi-detached | 3.0 | 1.12 | 2.5 | 0.35 |
| embedded | 3.6 | 1.20 | 2.5 | 0.32 |

Table 6: Value of variables in COCOMO model.

Now in our project the no of effective source line of code (SLOC) is **1740** (includes only data declaration statements and executable statements) and the amount payble to the person working on the project per month is $100 (₹6149). So according to the COCOMO Model the estimated cost is:-

**Software Development (Elaboration and Construction)**

Effort = 6.3 Person-months

Schedule = 6.7 Months

Cost = $630

Total Equivalent Size = 2000 SLOC

**Acquisition Phase Distribution:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase** | **Effort (Personmonths)** | **Schedule (Months)** | **Average Staff** | **Cost (Dollars)** |
| **Inception** | 0.4 | 0.8 | 0.4 | $38 |
| **Elaboration** | 1.5 | 2.5 | 0.6 | $151 |
| **Construction** | 4.8 | 4.2 | 1.1 | $479 |
| **Transition** | 0.8 | 0.8 | 0.9 | $76 |

Table 7: Acquisition Phase Distribution

**Software Effort Distribution for RUP/MBASE (Person-Months):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase/Activity** | **Inception** | **Elaboration** | **Construction** | **Transition** |
| **Management** | **0.1** | **0.2** | **0.5** | **0.1** |
| Environment/CM | **0.0** | **0.1** | **0.2** | **0.0** |
| **Requirements** | **0.1** | **0.3** | **0.4** | **0.0** |
| **Design** | **0.1** | **0.5** | **0.8** | **0.0** |
| **Implementation** | **0.0** | **0.2** | **1.6** | **0.1** |
| **Assessment** | **0.0** | **0.2** | **1.1** | **0.2** |
| **Deployment** | **0.0** | **0.0** | **0.1** | **0.2** |

Table 8: Software Effort Distribution for RUP/MBASE (Person-Months)

# CHAPTER 11

# SOFTWARE TESTING

## 11.1 What is Software Testing

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of a software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

#### Verification

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

#### Validation

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

## 11.2 Basics of Software Testing

There are two basics of software testing: blackbox testing and whitebox testing.

##### Blackbox Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

##### ****White****box ****Testing****

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing.

Black box testing is often used for validation and white box testing is often used for verification.

## 11.3 Types of Testing

#### Types of testing

There are many types of testing like

* Unit Testing
* Integration Testing
* Functional Testing
* System Testing
* Stress Testing
* Performance Testing
* Usability Testing
* Acceptance Testing
* Regression Testing
* Beta Testing

##### Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

##### Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

##### Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

##### System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

##### Stress Testing

Stress testing is the testing to evaluate how system behaves under unfavorable conditions. Testing is conducted at beyond limits of the specifications. It falls under the class of black box testing.

##### Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

##### Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

##### Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

##### Regression Testing

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

##### Beta Testing

Beta testing is the testing which is done by end users, a team outside development, or publicly releasing full pre-version of the product which is known as beta version. The aim of beta testing is to cover unexpected errors. It falls under the class of black box testing.

## 11.4 Test Cases

A **test case**, in [software engineering](http://en.wikipedia.org/wiki/Software_engineering), is a set of conditions under which a tester will determine whether an [application](http://en.wikipedia.org/wiki/Software_application), [software system](http://en.wikipedia.org/wiki/Software_system) or one of its features is working as it was originally established for it to do. The mechanism for determining whether a software program or system has passed or failed such a test is known as a [test oracle](http://en.wikipedia.org/wiki/Oracle_(software_testing)). In some settings, an oracle could be a [requirement](http://en.wikipedia.org/wiki/Requirement) or [use case](http://en.wikipedia.org/wiki/Use_case), while in others it could be a [heuristic](http://en.wikipedia.org/wiki/Heuristic). It may take many test cases to determine that a software program or system is considered sufficiently scrutinized to be released. Test cases are often referred to as [test scripts](http://en.wikipedia.org/wiki/Test_script), particularly when written - when they are usually collected into [test suites](http://en.wikipedia.org/wiki/Test_suite).

## 11.5 Our Project Test Cases

### 11.5.1 Test case No. 1

**Test Case no. 01**

|  |  |
| --- | --- |
| **Test Case ID: T\_01** | **Test Case Name: Run Application** |
| **System: File Security System** | **Subsystem: Open/close Application** |
| **Designed by: Sandeep Yadav** | **Design Date: 08/03/2015** |
| **Executed by: Khemanshu Rao** | **Execution Date: 08/03/2015** |
| **Short Description: Make test cases to open and close the application** |  |

**Pre-conditions: We have a computer system having IDE for java.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| 1 | Double-click on run.bat. | | The system should open the application | PASS | Actual result is same as expected |
| 2 | Click on the cancel button on interface. | The system displays the desktop. | | PASS | Actual result is same as expected |

**Post-conditions: The system displays the desktop after closing the application.**

### 11.5.2 Test Case No. 2

**Test Case no. 02**

|  |  |
| --- | --- |
| **Test Case ID: T\_02** | **Test Case Name: Authorization** |
| **System: File Security System** | **Subsystem: Username & Password** |
| **Designed by: Sandeep Yadav** | **Design Date: 08/03/2015** |
| **Executed by: Khemanshu Rao** | **Execution Date: 08/03/2015** |
| **Short Description: Make test cases for authorization of application** |  |

**Pre-conditions: Our application is running on the system.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| 1 | Enter valid username in text box and leave the blank password text box. Click submit button | | The system should display proper error message "unauthorized User" | PASS | Actual result is same as expected |
| 2 | Enter invalid username in text box and leave the blank password text box. Click submit button. | The system should display proper error message "unauthorized User" | | PASS | Actual result is same as expected |
| 3 | Enter the invalid username and valid password and click on submit button | The system should display proper error message "unauthorized User" | | PASS | Actual result is same as expected |
| 4 | Enter the valid username and invalid password and click on submit button | The system should display proper error message "unauthorized User" | | PASS | Actual result is same as expected |
| 5 | Enter valid username and password in textbox and click on sign-in | The system should redirect to file security system window. | | PASS | Actual result is same as expected |

**Post-conditions: The system should give access to authorized user.**

### 11.5.3 Test Case No 3

**Test Case no. 03**

|  |  |
| --- | --- |
| **Test Case ID: T\_03** | **Test Case Name: Menu Testing** |
| **System: File Security System** | **Subsystem: Menu** |
| **Designed by: Sandeep Yadav** | **Design Date: 08/03/2015** |
| **Executed by: Khemanshu Rao** | **Execution Date: 08/03/2015** |
| **Short Description: Make test cases for testing of menu of application** |  |

**Pre-conditions: Our application is running on the system with authorization.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| 1 | Click on "Sender" button. | | The system should open sender window | PASS | Actual result is same as expected |
| 2 | Click on "Receiver" button. | The system should open receiver window | | PASS | Actual result is same as expected |
| 3 | Click on "Exit" button. | The system should close the window | | PASS | Actual result is same as expected |

**Post-conditions: The system should open proper window.**

### 11.5.4 Test Case No 4

**Test Case no. 04**

|  |  |
| --- | --- |
| **Test Case ID: T\_04** | **Test Case Name: Sender window testing** |
| **System: File Security System** | **Subsystem: Sender window** |
| **Designed by: Sandeep Yadav** | **Design Date: 08/03/2015** |
| **Executed by: Khemanshu Rao** | **Execution Date: 08/03/2015** |
| **Short Description: Make test cases for testing of sender window** |  |

**Pre-conditions: Sender window of our application is running on the system.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| 1 | Typing the string in text box. | | String should be taken as input by system. | PASS | Actual result is same as expected |
| 2 | Selecting a text file. | Content of file are should be copied in text box | | PASS | Actual result is same as expected |
| 3 | Encrypt the text. | Text should be changed into cypher text. | | PASS | Actual result is same as expected |
| 4 | Save the cypher text. | Cypher text should be saved in the selected text file. | | PASS | Actual result is same as expected |
| 5 | Click on "back button". | Sender window should be closed. | | PASS | Actual result is same as expected |

**Post-conditions: The sender window should close after clicking on “back” button.**

### 11.5.5 Test Case No 5

**Test Case no. 05**

|  |  |
| --- | --- |
| **Test Case ID: T\_05** | **Test Case Name: Receiver window testing** |
| **System: File Security System** | **Subsystem: Receiver window** |
| **Designed by: Sandeep Yadav** | **Design Date: 08/03/2015** |
| **Executed by: Khemanshu Rao** | **Execution Date: 08/03/2015** |
| **Short Description: Make test cases for testing of receiver window** |  |

**Pre-conditions: Receiver window of our application is running on the system.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | | **Pass/ Fail** | **Comment** |
| 1 | Typing the cypher text in text box. | | Cypher text should be taken as input by system. | PASS | Actual result is same as expected |
| 2 | Selecting a cypher text file. | Content of file are should be copied in text box | | PASS | Actual result is same as expected |
| 3 | Decrypt the cypher text. | Cypher text should be changed into plain text. | | PASS | Actual result is same as expected |
| 4 | Save the plain text. | Plain text should be saved in the selected text file. | | PASS | Actual result is same as expected |
| 5 | Click on "back button". | Receiver window should be closed. | | PASS | Actual result is same as expected |

**Post-conditions: The receiver window should close after clicking on “back” button.**

# CHAPTER 12

# REFERENCES

1. Silberschatz,Korth,Sudarshan “Database System Concepts”:5thEd. New York:McGrawHill(May 2006)
2. Grady Booch,James Rambaugh,Ivar Jacobson “Unified Modeling Language User Guide” :1stEd. Addition Wesley(April 1998)
3. SAMS“Teach yourself UML in 24 hours”:4thEd.USA (March 2004).