**Highly Confidential Security System**

Software Design Document

Name (s): Sahil Narula (12BCE0416), Poorva Arora (12BCE0277)

Submitted to: Prof. Geetha Mary

Date: 15-April-2015

**TABLE OF CONTENTS**

[1. INTRODUCTION 3](#_Toc416801554)

[1.1 Purpose 3](#_Toc416801555)

[1.2 Scope 3](#_Toc416801556)

[1.3 Overview 3](#_Toc416801557)

[1.4 Reference Material 3](#_Toc416801558)

[2. SYSTEM OVERVIEW 4](#_Toc416801559)

[2.1 ER Diagram 5](#_Toc416801560)

[2.2 Class Diagram 6](#_Toc416801561)

[3. Specific Requirements 6](#_Toc416801562)

[3.1 Activity Diagram 6](#_Toc416801563)

[3.2 Sequence Diagram 10](#_Toc416801564)

[3.3 Collaboration Diagram 14](#_Toc416801565)

[4. APPENDICES 15](#_Toc416801566)

# INTRODUCTION

## Purpose

The purpose of the Detailed Design Document is to define the detailed design for all components of the **Highly Confidential Security System** which are specified in the [SRS]. The low-level components are designed, coded and tested

## Scope

The application implements server based optimal storage for users data. The application back-end is able to execute multiple evented IO using a single thread. By hiding the complexity of the technology and providing users fast and efficient platform to store and access their confidential items or passwords, the system will be easy to use.

## Overview

This DDD is organized as follows. Chapter 2 of this document is a short introduction to the general context of the application to be made and to the background of this project. Chapter 3 describes the overall architecture and all the components in a consistent way. Chapter 4 consists of a dictionary for related keywords. Chapter 5 consists of Various components description for the project. Chapter 6 includes the User Interface for the project

## Reference Material

*This section is optional.*

**PHP**: <http://php.net/>

**NODEJS**: <http://nodejs.org/>

**ANGULARJS**: <https://angularjs.org/>

**MySQL:** <https://www.mysql.com/>

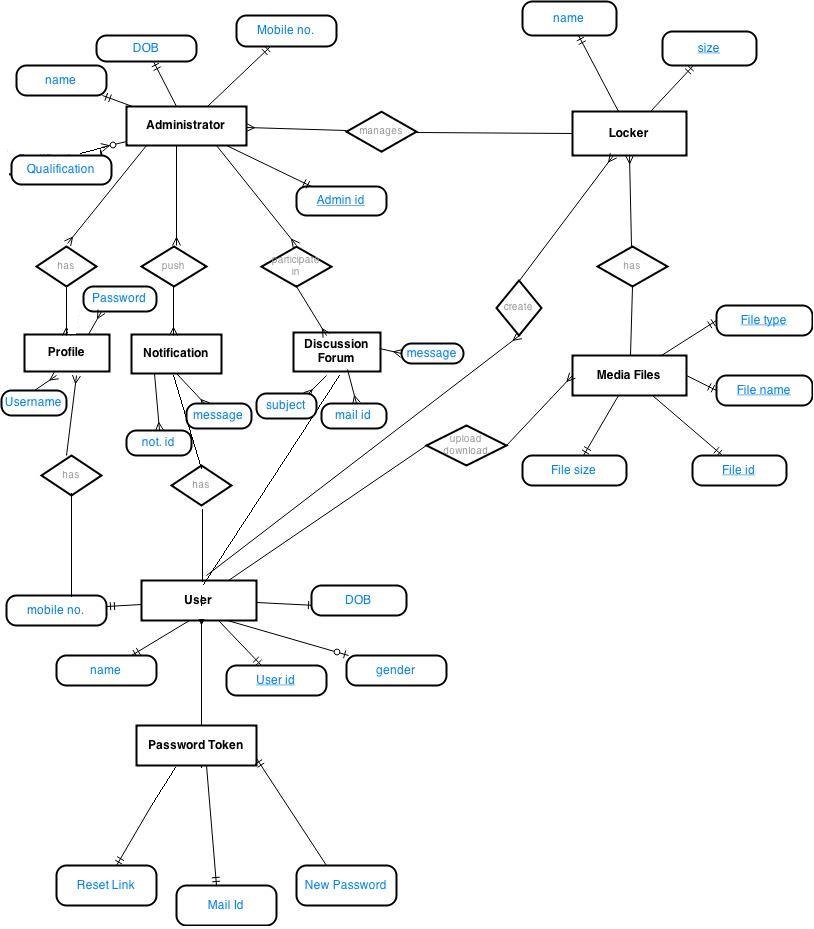
**HTML:** <http://en.wikipedia.org/wiki/HTML>

**CSS**: <http://en.wikipedia.org/wiki/Cascading_Style_Sheets>

# SYSTEM OVERVIEW

The project is divided into two modules frontend and backend. Frontend consists of all the assets to improve user experience. Back-end consists of restful apis to provide optimal routes to send data back and forth. Both frontend and backend are further divided into sub modules. Frontend application consists of various assets and client side controllers which includes resources to connect to remote rest apis. Backend consists of sub modules of various controllers and models for database.

## ER Diagram



**Entity–relationship model** (**ER model**) is a [data model](http://en.wikipedia.org/wiki/Data_modeling) 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](http://en.wikipedia.org/wiki/Database) such as a [relational database](http://en.wikipedia.org/wiki/Relational_database). The main components of ER models are [entities](http://en.wikipedia.org/wiki/Entities) (things) and the relationships that can exist among them.

## Class Diagram

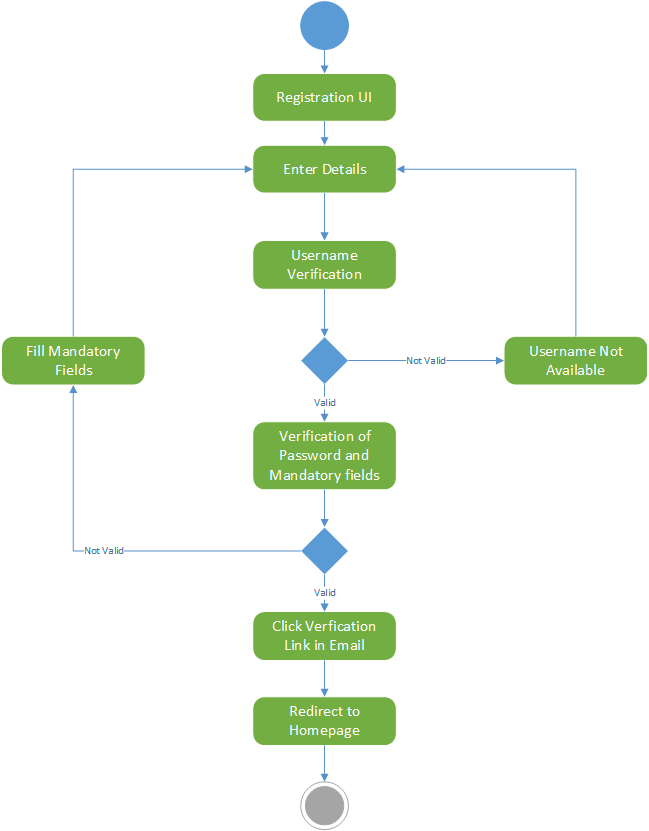
**Class diagram** in the [Unified Modeling Language](http://en.wikipedia.org/wiki/Unified_Modeling_Language) (UML) is a type of static structure diagram that describes the structure of a system by showing the system's [classes](http://en.wikipedia.org/wiki/Class_(computer_science)), their attributes, operations (or methods), and the relationships among objects.

# Specific Requirements

## Activity Diagram

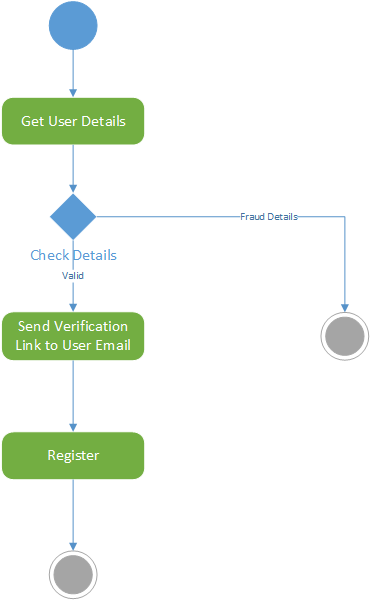
### 3.1.1 User Registration

The user is made to fill all the mandatory fields, each user must choose a unique user name. If the user has filled a name that is already present in the database, then the user will be prompted to fill in a different username. After this the user must enter the new password twice. When the user clicks the submit button the database verifies all the mandatory fields are filled or not. When all the fields are filled the verification mail is automatically send to user’s mail, after click the verification link the user is registered, otherwise the user is made to fill the fields again.



### 3.1.2 Validate Registration

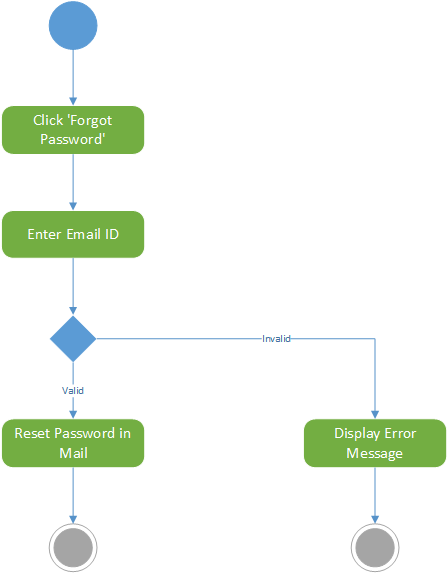
The user enters the registration details. The system validates the details and if the details are found invalid the system rejects it else the system sends the verification mail to the user. The user needs to follow the verification link to complete his registration.



### 3.1.2 Password Reset

If the user forgets his password then he should click on the Forget password link. Then the system

prompts him to enter his Mail ID. The system validates the Mail ID. If the Mail ID is valid then the system sends the password reset mail to that Mail ID. If the Mail ID is invalid then the system displays the error message.



## Sequence Diagram

A **Sequence diagram** is an [interaction diagram](http://en.wikipedia.org/wiki/Interaction_diagram) that shows how processes operate with one another and what is their order. It is a construct of a [Message Sequence Chart](http://en.wikipedia.org/wiki/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**.



Figure 1: Login Sequence Diagram



Figure 2: Registration Sequence Diagram



Figure 3: Upload Sequence Diagram



Figure 4: Download Sequence Diagram

## Collaboration Diagram

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among [software](http://searchsoa.techtarget.com/definition/software) [object](http://searchsoa.techtarget.com/definition/object)s in the Unified Modeling Language (UML). The concept is more than a decade old although it has been refined as modeling paradigms have evolved.



Figure 5: Login Collaboration Diagram



Figure 6: Registration Collaboration Diagram



Figure 7: Upload Collaboration Diagram



Figure 8: Download Collaboration Diagram

# APPENDICES

*PHP:* <http://php.net/>

*Angularjs:* <http://angularjs.org>

*Html:* <http://en.wikipedia.org/wiki/HTML>

*CSS:* <http://en.wikipedia.org/wiki/Cascading_Style_Sheets>

*MySQL:* <https://www.mysql.com/>