Software Design Description

Prepared by CODEFELLAS2

for the project LINUX PASSWORD VAULT

METU - Department of Computer Engineering

CENG 492 Senior Design Project II

Spring 2015-2016

Ali Can Oğul - 1746304

Table of Contents

1. Overview

1.1 Scope

This Software Design Description (SDD) document provides necessary information about the project Linux Password Vault. This document includes design principles of the software with its requirements, functionalities and necessary definitions. These information is aimed to guide any programmer to understand our design and be an assistant in the development phase.

1.2 Purpose

The purpose of this document is to outline the design views of the project which will satisfy functional and nonfunctional requirements stated in the SRS Document of Linux Password Vault. Purpose of this document is serving as a guideline throughout development phase of the project for developers.

1.3 Intended Audience

The main audience of this document includes the developers of this project and Middle East Technical University CENG492 course management. Developers are supposed to use this document in the development phase to the structure and design of each component.

2. Definitions

LPV	Linux Password Vault
SSL	Secure Sockets Layer
TLS	Transport Layer Security
PGP	Pretty Good Privacy
TOTP	Time Based One Time Password
GPG	GNU Privacy Guard
SRS	Software Requirement Specification
OTP	One Time Password
AES	Advanced Encryption Standard
RSA	Rivest-Shamir-Adleman Encryption
DBMS	Database Management System
ER Diagram	Entity Relationship Diagram
SDD	Software Design Description
API	Application Programming Interface
SQL	Structured Query Language
MySQL	My Structured Query Language

3. Conceptual Model For Software Design Descriptions

In this section, we will present a conceptual model for the SDD. This conceptual model mainly explains the context in which SDD is prepared and how it will be used by the stakeholders and developers. Basic terms, concepts and context of SDD will be given in this part.

3.1 Software Design In Context

The project aims to create a service where all passwords can be stored securely, accessed by the authorized personnel only on a need-to-know basis, reset when needed. All activities will be logged. Ruby and MySQL are the main components that are going to be used in the development stage of this project.

This project will be designed with modular approach. By obeying this approach it will be easy to respond quickly and efficiently to the changing demands' of the customers.

3.2 Software Design Descriptions Within The Life Cycle

3.2.1 Software Design Descriptions Within The Life Cycle

This document is prepared by considering the Software Requirement Specification (SRS) document. The requirements in the SRS document (interface, functional, non-functional and logical database requirements) and also the demands of the candidate users specify the design of the project.

3.2.2 Influences On Software Life Cycle Products

During the progress, some features of our project have been changed, according to new requirements and constraints that were learned after SRS report. In addition, in this progress, when the group decides to change a property for a better usage and appearance, we have replaced this property with the new one. Our test procedure will be done through the guide of SDD.

3.2.3 Design Verification And Design Role In Validation

Software design description is the primary reference for the verification and validation of whether the software product designed fulfills the specified requirements. The requirements for each specific intended use of the software product are modeled in the design view parts of the document. The verification and validation of the design view models are carried out based on this document. SDD serves a compass role during the test-driven development process. It is used to drive the test cases.

4. Design Description Information Content

4.1 Introduction

In this part of the Software Design Description document, the design and implementation of the Linux Password Vault will be identified. This section will include information about SDD identification, identified design stakeholders and design concerns, selected design viewpoints with their type definitions of its allowed design elements and design languages, design views, design overlays and design rationale.

4.2 SDD Identification

This is the initial version of SDD document of the project. The date of issue is 3rd of June, 2016. The scope of the document is stated in section 1. The copyright of the document belongs to Codefellas2. UML is used to represent each design viewpoint.

4.3 Design Stakeholders And Their Concerns

Design stakeholders Linux Password Vault project is Codefellas2, its advisor Dr. Onur Tolga Şehitoğlu and its assistant Çağlar Seylan. Stakeholders' main concern is completing the project until the deadline which is the end of the 2nd semester of 2015-2016 educational year. The team and the assistant Çağlar Seylan and advisor Dr. Onur Tolga Şehitoğlu meet once in a week and the team gets feedback from them to realize their situation and improve it.

4.4 Design Views

The project will be implemented with modular approach. The team can respond changing needs by updating only necessary modules and fix the bugs in small modules. Object oriented principles will be applied so that new features can be integrated without much effort. Users will use this program by running it in their Linux console and they will login to the system with specified user name and password. Then they can store domains' passwords safely to the database system. After that they can retrieve the password back and may or may not authorize other users to access it. Users also can be seperated into groups and those groups may or may not have authorization to access a domain password.

4.5 Design Viewpoints

In this section, each design viewpoint are explained briefly.

- Context viewpoint describes the relationships and interactions between the users and the system. Use case diagrams is used to represent the each function.
- Composition viewpoint describes the main structure of the application. It shows the interactions between the components of the system such as the database, server and user interface. Overall system architecture is shown by using component diagram.
- Interaction viewpoint describes the interactions and relations between database, server and user interface for each user operation. Sequence diagram is used to represent those relations.

4.6 Design Elements

Design elements will be explained thoroughly in section 5.

4.7 Design Overlays

In the Design Viewpoints section, all the existing information are described and there is no additional information to present in this section.

4.8 Design Rationale

The software is designed by object-oriented approach in order to make the product easily maintainable. The object oriented design helps to classify the objects of the software so that a new object can easily be added to design or an existing component can be easily deleted.

While designing the database of the system, tables are created according to both ER Diagram and class diagram, shown in Section 5, in order to synchronize the database and models.

Test driven development is chosen for implementation of the system. The main benefit of writing the test cases first is that it provides a better understanding of the actual code and thinking more about extreme cases.

Also, while writing the code, function and variable names are intentionally chosen to specify what they do. Thus, their functionalities can be understood easily in future.

4.9 Design Languages

Unified Modeling Language (UML) is selected as a part of design viewpoint and it will be used for clarifying design viewpoints.

5. Design Viewpoint

5.1 Introduction

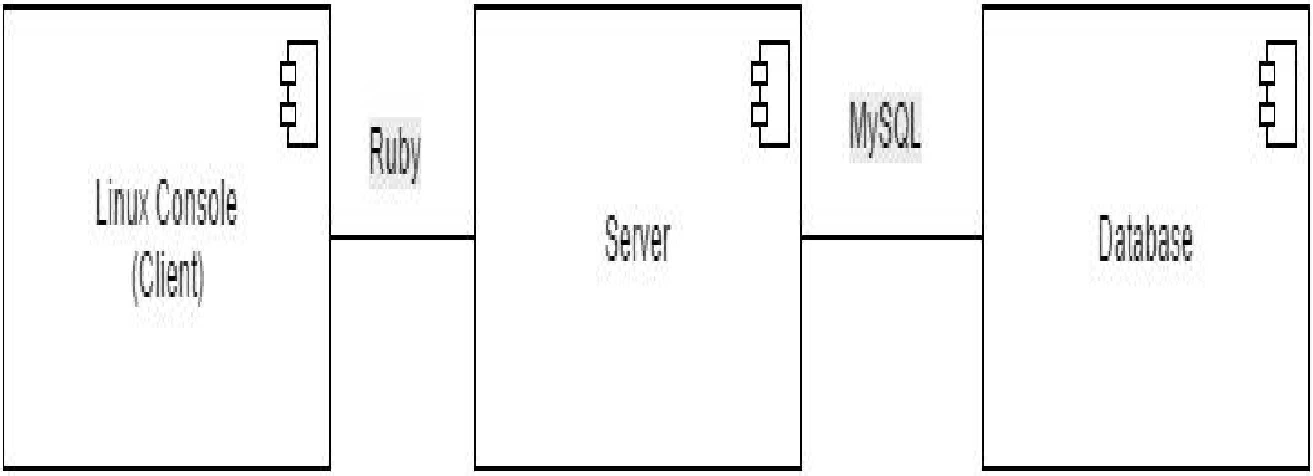
In this section following viewpoints are illustrated:

- Composition viewpoint
- Context viewpoint
- Interaction viewpoint

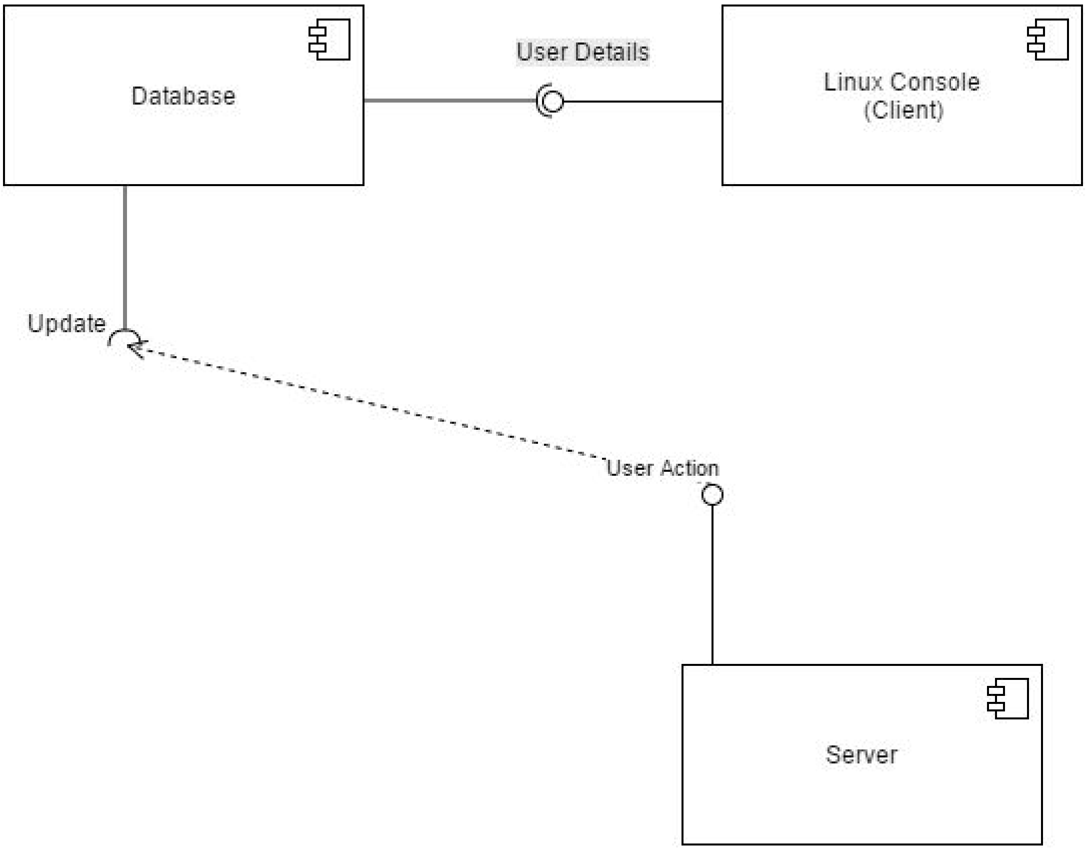
5.1 Composition Viewpoint

We are aiming to explain physical and logical components of our system by introducing composition viewpoint. UML is used under this title.

Logical relation between components are showed the component diagram below.



There is also a deployment diagram below.

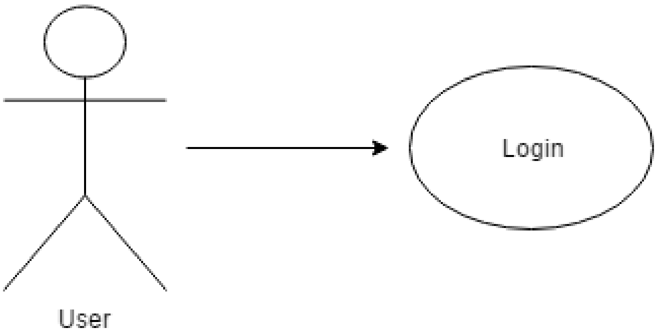


5.2 Context Viewpoint

System has two main users: users and admins. Admins are capable of doing whatever users are capable of doing.

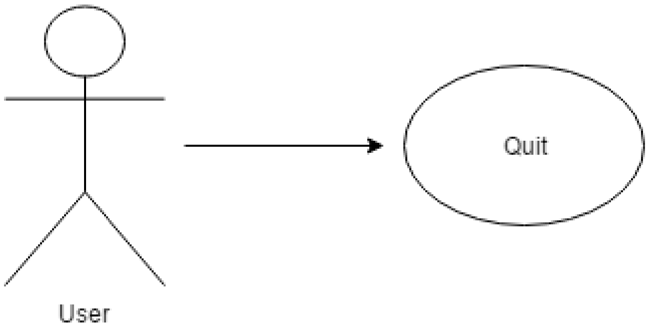
5.2.1 Login Use Case

User logins to the system.



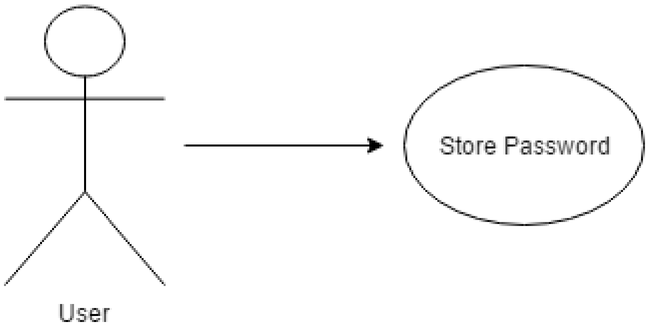
5.2.2 Quit Use Case

User logs out from the system.



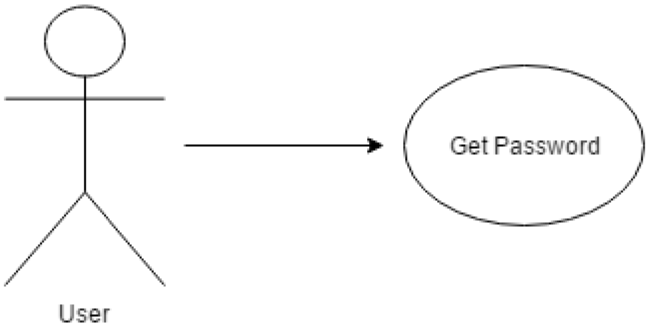
5.2.3 Store Password Use Case

User may store the password securely in the database.



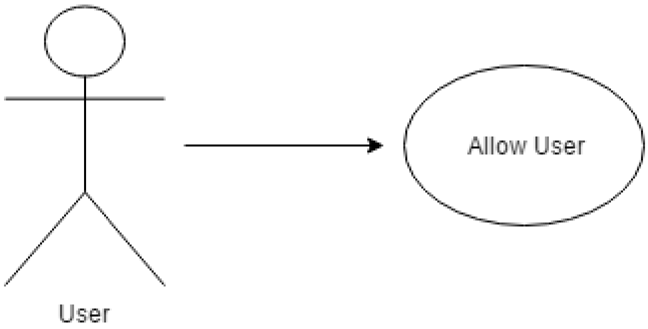
5.2.4 Get Password Use Case

User may get a domain’s password from the database.



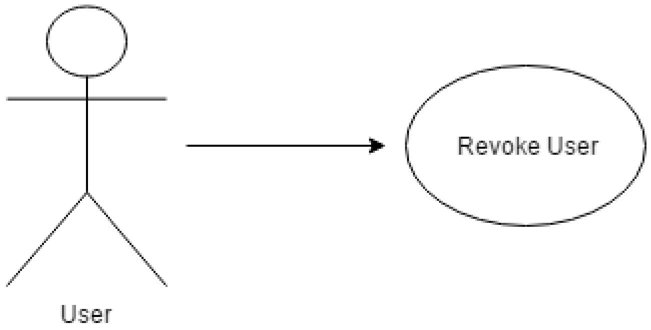
5.2.5 Allow User Use Case

User may allow another user to access to the domain password.



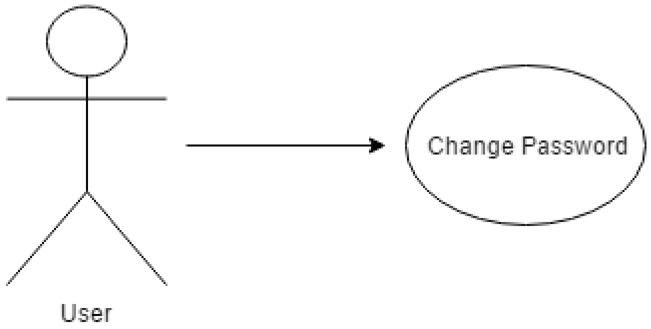
5.2.6 Revoke User Use Case

User may revoke another user’s acces to the domain password.



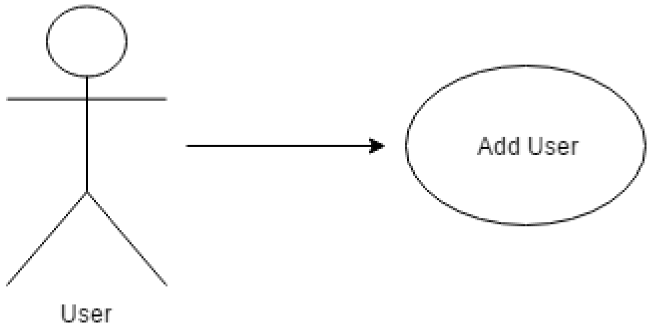
5.2.7 Change Password Use Case

User may change password of the domain.



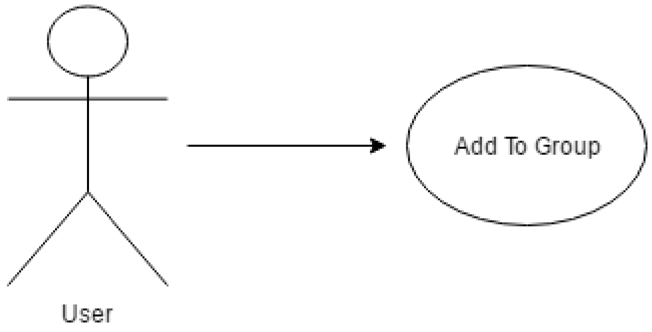
5.2.8 Add User Use Case

User may add another user to the system.



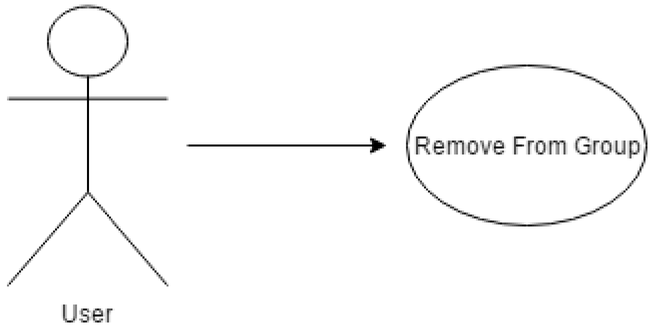
5.2.9 Add To Group Use Case

User may add another user to the group.



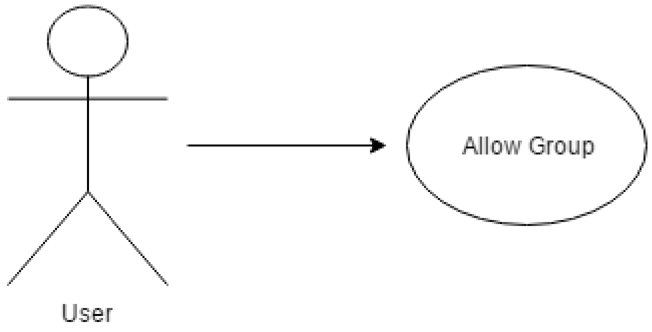
5.2.10 Remove From Group Use Case

User may remove another user from group.



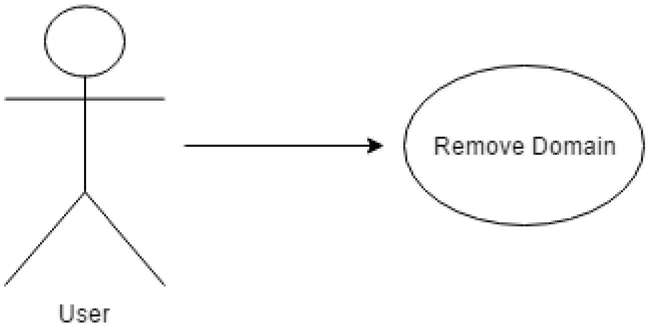
5.2.11 Allow Group Use Case

User may allow all group members to access to the domain password.



5.2.12 Remove Domain Use Case

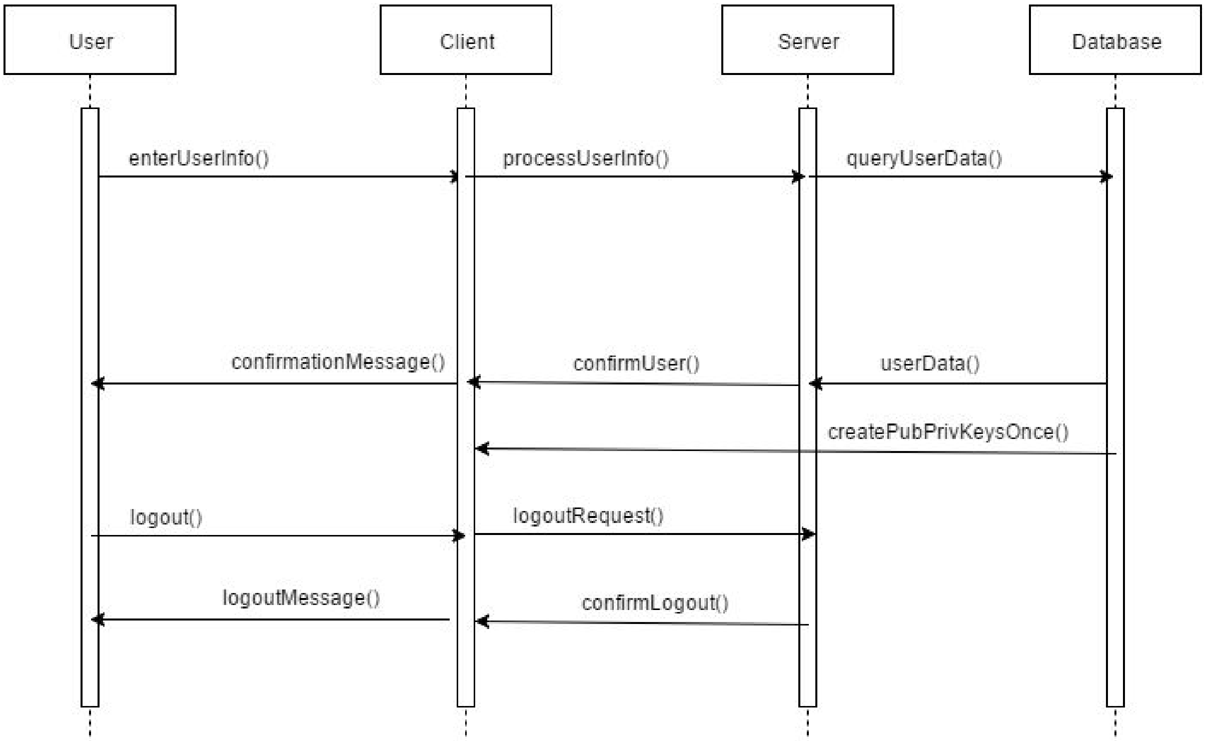
User may remove domain from the database.



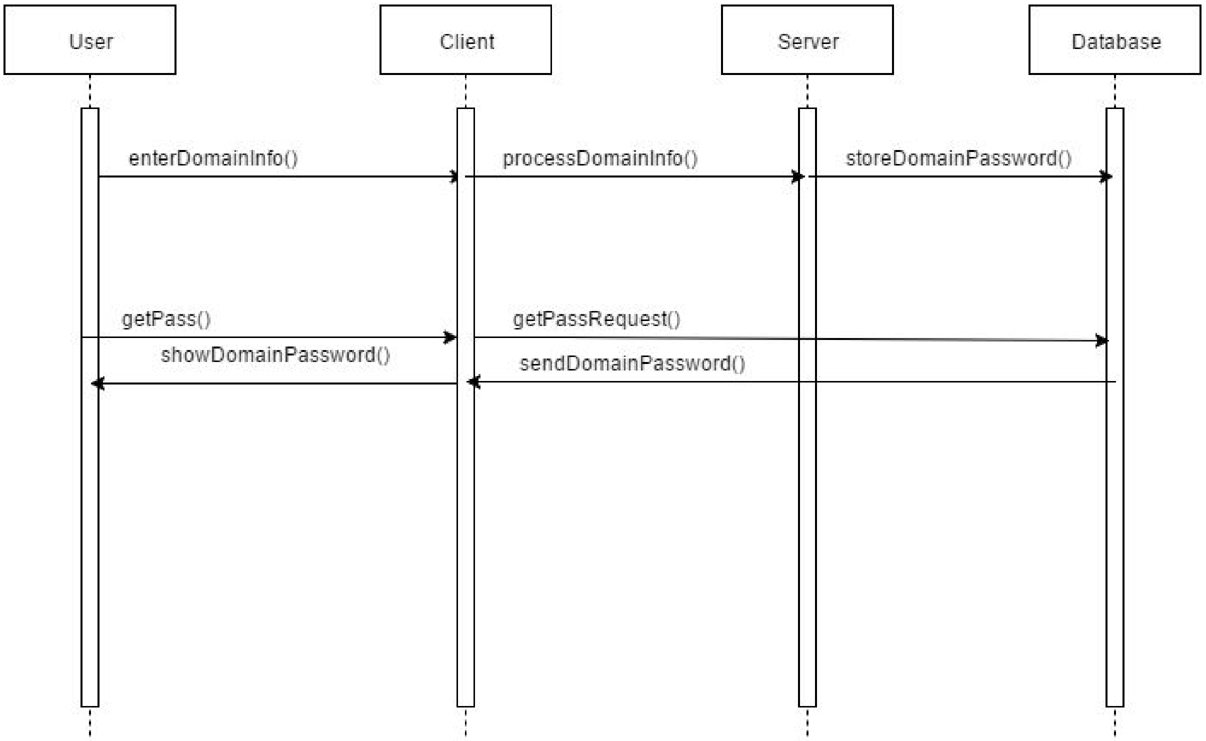
5.3 Interaction Viewpoint

In this section, system users’ interaction with the system and system objects in terms of messages among objects in fulfilling required actions is described. UML sequence diagrams are used to visualize data and work flow

5.3.1 Login Logout Interaction



5.3.2 Store And Get Password Interaction


User
Client
Server
Database
enterDomainInfo()
processDomainInfo()
storeDomainPassword()
getPass()
getPassRequest()
showDomainPassword()
sendDomainPassword()