**Chapter three : SYSTEM ANALYSIS AND DESIGN**

**3.0 Introduction**

With the many ways, the use the Internet, it’s easy to consider some passwords less important than others. However, all passwords are important because wrongdoers can piece together the information you store online and use it for their benefit. They can even use information you share on social media networks. And commercial websites give customers the ability to store billing and shipping addresses along with credit card information.

**3.1 Description of the Existing System**

Secure password practices result in numerous cryptic passwords which are very difficult to keep track of. It is impossible for most people to consistently remember more than just a few of them. Any time we discuss password managers, the ensuing commentary can sometimes get a little heated. People really love their password managers and we love to hear that too. One of the biggest, if not THE biggest, point of contention, however, is the cloud. Specifically, the cloud as a place to store your password vault, the cache of your credentials that your password manager absolutely needs to keep safe at all costs. This is a clear dividing line for many password manager aficionados. The following are the features found in the existing system.

1. The existing system requires a user must create his/her account and download the system extension on user device.
2. It is a browser based
3. User details are stored both on the local machine and on remote cloud storage server.

**3.2 Analysis of the Proposed System**

The complex methods that attackers can use to gain access to your personal information are becoming more easily accessible to wrongdoers and are increasingly effective. It is important to avoid the common mistakes that give these individuals the opportunity to exploit your personal data. The system to be built has the following functionalities and features.

1. Ability for a user to create a profile.
2. Ability for a user to store files in the system.
3. The user can update his/her files or profile.
4. The system uses a master password to authenticate a user during login process.
5. The system encrypts the user password to avoid being hacked by others.
6. The system stores and secures the user’s profile from others.
7. The system uses a web based platform to function.

**3.2.1 Non-Functional Requirements**

These are the tools that will aid the functional requirements of the proposed system. In addition to the functional requirements are other requirements that don't actually do anything, but that are critical nevertheless. In systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. These nonfunctional requirements, also called "quality attributes," specify system characteristics that are required for acceptance of the system by the end user. It is therefore imperative that you document such requirements in the functional specification. The following are the non-functional requirements of the store management system;

1. **Reliability:** Users of this system will be confident of its functionalities and lose fear of failure of the system.
2. **Availability:** The system will be functional at any point in time the user is in need of it.
3. **Security:** The system will protect the user’s details of transaction and will not disclose the activities performed in the system to any end user. All users of the system must be uniquely identified.
4. **Performance:** The system will be able to satisfy the end user for the reason why it was developed.
5. **Usability:** The system will be developed in a way that the reason for its development will be met, i.e. users will cope and make use of the system appropriately. The system’s user interface intuitive, easy to use and provide an overall positive user experience.
6. **Integrity:** The online store management system will be able to protect and preserve transactions made in the system.

vii. **Maintainability:** The system will be developed in a way that changes can be made easily i.e. for bug fixes or adding new functionality in the future. The system will be made easy for the user to maintain with just reading through the manual or undergoing a few hours training. All source code shall adhere to an agreed upon and well-defined set of coding standards for each development language used.

**3.2.2 Functional Requirements**

[Password manager](https://en.wikipedia.org/wiki/Password_manager) software is used by individuals to organize and encrypt many personal passwords using a single login. This often involves the use of an [encryption key](https://en.wikipedia.org/wiki/Encryption_key) as well. Some of the functional requirements for this system are as follows:

**Logout:** A logout process must be provided. An automatic logout during session inactivity must occur.

**Password security:** Passwords are stored on the database as a secure hash and cannot be decoded. All passwords during logon are transmitted from client to server as a salted hash, an encrypted hash, or an encrypted salted hash.

**Login:** A login process must be provided.

**Password Reset:** The password reset page must utilize the same requirements as the account logon page.

**Account Creation:** The password entry page portion of account creation must be separate from the parts of the account creation page that contain personal data including a secret question.

**3.3.2 Design of the Proposed System**

The system modeling tool used is object oriented analysis and design and it employs the use of unified modeling language (UML). UML is a method of visualizing and documenting information

UML diagram can be of three types:

1. Use case diagram
2. Class Diagram
3. Activity Diagram

**3.3.1 Use Case**

UML Use Case Diagrams can be used to describe the functionality of a system in a horizontal way. That is, rather than merely representing the details of individual features of your system, Use case diagrams can be used to show all of its available functionality. It is important to note, though, that Use case diagrams are fundamentally different from sequence diagrams or flow charts because they do not make any attempt to represent the order or number of times that the systems actions and sub-actions should be executed.

Fig. 3.3.1.1 Use case diagram

**3.3.2 Class Diagram**

The class diagram helps in specifying the structural relationship between parts of the system in an object oriented manner. Each noun or “thing,” describe in the requirement document is qualified to be a class in the system, while each verb or “action,” in the requirement document is a candidate to be a function/method in the system.

Website

USER

1 0…\*

+Id

+UserId

+websiteURL

+webSitePassword

+webSiteUsername

+Id

+email: String

+password: String

+register(email, password):void

+getPasswordFromFile():String

+loginPassword(Password): void

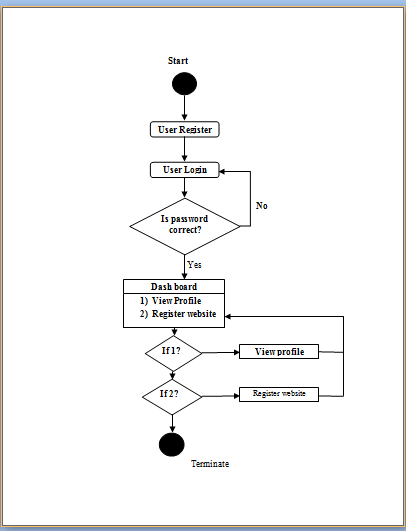
+savePassword(Password): void

+storeUserDetail

Fig 3.3 class of Diagram of the system

**3.3.3 Activity Diagram**

The activity diagram defines or explains the work flow (sequence of action) of the system during program execution. It models the actions the objects performs and specify the order in which it performs them, it is model in flow chart and the activity display in the flow chart can be branched, sequential or concurrent.



**Fig 3.4 Activity Diagram of the system**

**3.3.5 Input Design**

The input design is concerned with data capture and data entry. The user registers and login into the system to interact with the system with the help of the HTML form field. It allows the user to fill in the fields and click on the submit button for processing by PHP script on the server.

**3.3.6 Output Design**

The output is the result form that display after a user has submitted an input query to the system.

**3.3.7 Database Design**

This design shows us how the database is been structured and how data are stored in there.

1. **User Table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| FIELD | DATA TYPE | SIZE | NULL | KEY | DEFAULT | EXTRA |
| Id | Int | 11 |  | Primary Key |  |  |
| Username | Varchar | 30 | Null |  |  |  |
| Password | Varchar | 30 | Null |  |  |  |
| Phone number | Varchar | 30 | Null |  |  |  |

1. **website Table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Id | Int | 25 | Null | Primary Key |  |  |
| Username | Varchar | 25 | Null |  |  |  |
| Password | Varchar | 30 | Null |  |  |  |
| Email | Varchar | 30 | Null |  |  |  |
| Phone No | Varchar | 11 | Null |  |  |  |

* + 1. **System architecture**

The architecture of the system design is a 3-tier application. The tiers are the presentation tier, middle tier and data tier. The presentation tier is the user interface and it is designed using HTML, CSS and BOOSTRAP framework for the design of the graphical user interface. The middle tier which is also known as the business logic connects the presentation tier and data tier together. The middle tier of the system is design using PHP programming language, while the data tier which together with the middle tier is known as the server side or backend of a system is the part of the system that is responsible for storing the data in a database. The database management system use for the design of this system is MYSQL server. The system architecture is shown below.

Server

Middle Tier Data Tier

SQL Database

PHP

HTML5, CSS3

Presentation Tier

Figure 3.3.4 shows the system architecture