**TASK 1: SIMPLE CRUD APPLICATION IN PHP AND MSQL**

**Table of Contents**

[**Chapter 1: Introduction** 2](#_Toc82977562)

[**1.1 What is CRUD** 2](#_Toc82977563)

[**1.2 What is Front end (user interface)** 2](#_Toc82977564)

[**1.2 What is Backend (user interface)** 3](#_Toc82977565)

[**Chapter 2: System Analysis and Design** 3](#_Toc82977566)

[**2.1 System Narrative** 3](#_Toc82977567)

[**2.2 Functional Requirements** 3](#_Toc82977568)

[**2.3 Non-functional Requirements** 3](#_Toc82977569)

[**2.4 System design diagrams** 4](#_Toc82977570)

[**2.4.1 Use case diagram** 4](#_Toc82977571)

[**2.4.2 Data Schema** 4](#_Toc82977572)

[**2.4.3 Languages used.** 5](#_Toc82977573)

# **Chapter 1: Introduction**

## **1.1 What is CRUD**

CRUD is an acronym meaning:

C- Create

R- Read

U- Update

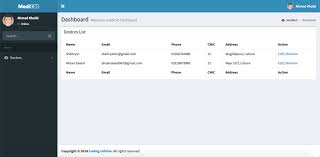
D- Delete

These are the four most important functionality of any dynamic system. A dynamic system is system that allows user interact with the system by generating the pages/data in real time, as per the request (done by client) to the server, a respective response will trigger from the server end and will reach the client end (your end). The response depends upon the client-side code.

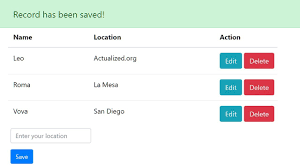
Example, in my system, the client can create a post, they can view the created post, the can either update or delete the specific post. All this is done in the front-end (user interface) programming but facilitated by the back-end programming

## **1.2 What is Front end (user interface)**

When you open a new browser tab, type in a URL, and press enter. The site loads instantly. It nearly takes your breath away with its ultra-clean layout, well-constructed pages, and impressive visuals. So basically, this is what you see when you load your URL on the browser.



This is where a user can get the delete, edit, view and add buttons.



## **1.2 What is Backend (user interface)**

So, what makes the front-end of a website possible? Where is all that data stored? This is where the back-end comes in. The back- end of a website consists of a server, an application, and a database.

A back-end developer builds and maintains the technology that powers those components which, together, enable the user-facing side of the website to even exist in the first place.

This is the root of any development; it powers the system.

# **Chapter 2: System Analysis and Design**

## **2.1 System Narrative**

It is a simple web application where a user can register, then proceed to login. A user cannot access the dashboard after signing up unless he or she logs in. This is good especially for authentication. It restricts only logged in users to enjoy the system services such as edit, delete, create posts. Once a user is logged in, he or she can add a post/ blog, the user can then edit the post if he/she wants to.

When you visit the site, the first page is the index.php, a page where anyone can see posts of different users but the user cannot enjoy the CRUD services unless logged in.

**NOTE:** A user can only delete and edit a post that he or she has created and no other. For now, a user cannot delete his or her profile but I will add that functionality.

## **2.2 Functional Requirements**

The system allows a user to register, then login in order to enjoy the system services. The system authenticates any user on login. The user can then create a post, they can decide on either to delete or edit the post.

## **2.3 Non-functional Requirements**

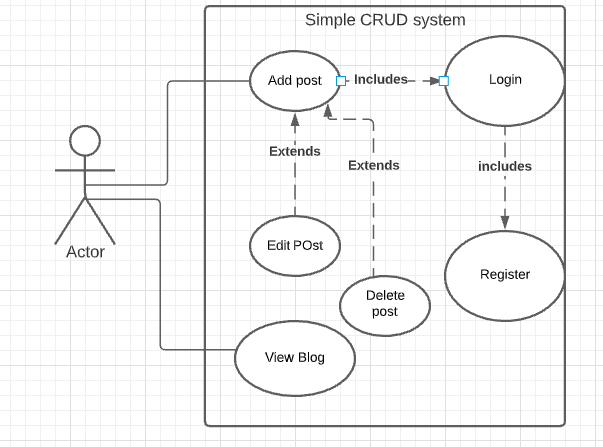
The system has portability in that it is be able to be supported on different devices and operating systems. The system also has usability in that it has user-friendly interface and functionality which should not prove any difficulty of use. The system has supportability in that it has a simple user manual that allows different users to use the application. The system also has scalability in that it can be expanded to a larger user base.

## **2.4 System design diagrams**

System design helps us specify specific needs and requirements of a project through the engineering of a coherent and well running system. For this project different diagrams such as a use case diagram, the Data Flow Diagram (DFD), the entity Relationship Diagram and a D2 schema were be used to design the project by clearly describing the elements of the project.

### **2.4.1 Use case diagram**

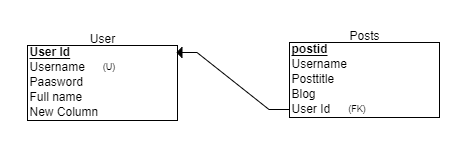
Here, the diagram describes the system functional requirements.



**Figure 2.4.1 Use case diagram describing all the modules in the system and what they can do.**

### **2.4.2 Data Schema**

It shows the actual database structure for the system. For my system, it had only two tables because it was a simple blog system. If a user is new, he or she registers and the data is pushed into the user table. Where a user id is autogenerated with an auto increment method acting as a primary key. A primary key is a unique identifier in a table so that if a user adds a post, it will refer back to the primary key. When a person adds a post, it goes to the posts table with the user id as the foreign key in the posts table. Foreign key is an inherited key from the user table. This will show you who posted what post.



### **2.4.3 Languages used.**

Languages used in my system are like:

* php for back-end
* simple html
* css
* Javascript

The site was deployed on 000webiste and below is the link

<https://finsensetrial.000webhostapp.com/index.php>

You can only edit once you have uploaded a blog. As a user, since you will be on session, you can edit or delete the post.