

**MicroLink Information Technology College**

**Department of Computer Science**

**Web Based E-Health System**

**System Design Document (SDD)**

**For Addis Hiwot General Hospital**

**Prepared by:**

1. Biruk Endalew
2. Dawit Tadesse
3. Eyosiyas Nigussie
4. Kirubel Maregn
5. Yohannes Melese

**Advisor:**

Tensaye Desalegn

19 June 2022

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 7/5/2022 | 1.0 | This is the first RAD document | All group members |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 4

1.1 Purpose 4

1.2 Scope 4

1.3 Definitions, acronyms, and abbreviations 4

2. References 4

3. Goals and Trade-offs 4

4. System Decomposition 4

4.1 Layers and Partitions 5

4.2 System Topology 5

5. Concurrency Identification 5

6. Hardware/Software Allocation 5

6.1 System Performance 5

6.1.1 General system performance 5

6.1.2 Input/Output Performance 5

6.1.3 Processor allocation 5

6.1.4 Memory allocation 5

6.2 Connectivity 6

6.3 Network architecture 6

7. Data Management 6

8. Global Resource Handling 7

9. Software Control Implementation 7

9.1 External control flow (between subsystems) 7

9.2 Concurrent control 7

9.3 Internal control (within a single process) 7

9.4 User Interface 7

10. Boundary Conditions 7

10.1 Initialization 7

10.2 Termination 8

10.3 Failure 8

11. Design Rationale 8

System Design Document

# Introduction

This system design document describes the system requirement, architecture, and database design of an E-health system. This system will provide patient registration, appointment making, and other such related services on an easy-to-use platform.

Patients will be able to login into the system and choose which healthcare provider they would like to consult and set a date of consultation, then the healthcare provider will be notified of the request and upon their acceptance of the appointment will be sent the patients history and any other information that they may need to properly care for that patient.

## Purpose

The purpose of this project is to create a simple to use and access online system that provides both patients and doctors with easier contact procedures that will save them both a lot of time and effort.

## Scope

The proposed software product is the E-Health System. In this project, we are going to design and build a functional web-based E-Health System.

This project aims to create an E-Health System that Patients can able to book doctor’s appointments and follow up with doctors (communicate after appointment). Doctors have access to view appointments and set available times for appointments thereby making it more convenient for them and follow-up patients. The admin also has access to a database to add or delete doctors on the system.

The system will be a web-based online application that is available on all major browsers. should be user-friendly on all usual devices (PC, mobile, tablet).

## Definitions, acronyms, and abbreviations

SDD System design document

UI User interface

User all system users, who can be either patients or doctor

Appointment a certain time for a patient to visit a doctor that is booked

System The E-Health system

Admin The system administrator

Interface between two or more systems describes how the system communicates

Android A Mobile operating system

CSS cascading Style Sheet is used to format the layout of the web page

Database Is an organized collection of data

# References

* Object Oriented Software Engineering -Using UML, Patterns, and Java; Bernd Bruegge & Allen H. Dutoit, Published by Pearson Education Inc., 3rd Edition.
* System Analysis and Design; Alan Dennis & Barbara Haley Wixom &Roberta M. Roth, Published by John Wiley & Sons Inc., 5th Edition.
* Object Oriented Software Engineering; Instructors Course Handout, Unpublished.
* Senior Final Project Document -RAD Template, Unpublished.

# Goals and Trade-offs

## Goals

1. **Functionality**: The basic function of the system is to create a platform that is used to reduce the burden of all the tiring and time-wasting that goes into traditional appointment systems.
2. **Usability:** The system provided will be an effective as well as an efficient system in which it will help provide services that help the user to achieve their goal easily, and also users will spend minimum effort and resources to achieve their goal.
3. **User-friendly interface:** This means that the system is very easy to operate as well as learn and understand by anyone who is either a person familiar with technology or one who is not by giving a well-made and interactive interface that can allow making it very easy to use.
4. **Security:** The system will be fully secured along with any unauthorized access to the system should be restricted. Through our login feature only authorized users can access the system moreover some of the users will have different capabilities they can do in the system and the login information will be stored on the database.
5. **Fault Tolerance:** The system should be able to give a response error message when the user enters incorrect input. This recommends the user enter the correct input.

## Trade-offs

Our design priority for this project was to make a reliable and simple understanding and easy-to-use app and website.

**Rapid prototyping vs. completeness of functionality**

This system was conceptualized to be used both on the web and on phones wherever internet data could be found but due to time shortage, we are forced to use the prototype on one computer.

**Usability vs. functionality**

Since making, many functions in one interface is an annoyance for the user that makes interfaces more difficult to learn and use. Therefore, our system alleviates this problem by allocating specific and similar tasks on one interface.

**Efficiency vs. portability:**

Our application will run in any android operating system and can be transferred from one android mobile environment to another with a good use of processor and memory.

**Cost vs. Reliability:**

For the reliability of the software tests, will write for the main function to make it error free. In addition, all these possible and predetermined errors from the user side will be handled without affecting the cost of the system.

**Reusability vs. Cost**

To make the development cost our classes and functions are designed to be reused in the code, we will not write the classes with some functions two times; rather, we follow the design pattern that forces us to use the re-usability of the code. This saves the development time, which leads to decreasing the development cost.

# System Decomposition

In total three sub-systems build up this project. A brief description of these subsystems is:

1. The patient sub-system is responsible for choosing doctors, making appointments, and follow-ups.
2. The doctor sub-system is responsible for accepting or declining a patient's request for an appointment or follow-up.
3. The admin sub-system is responsible for adding or removing doctors from the system.

The sub-systems do interact with each other depending on the circumstance for example for a patient to choose a doctor the system has to retrieve data from a list of doctors from the doctor's database.

## Layers and Partitions

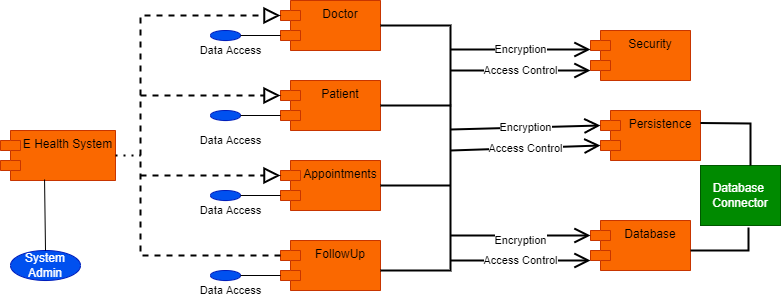
The dependency of the systems on each other is not hierarchy one because the sub-systems are equally dependent on each other.

## System Topology

* + 1. Component Diagram

Component diagrams show certain aspects of relationships that are not apparent in other class diagrams. The relationship between provided and required interfaces helps the reader understand the design better. The ultimate purpose is to help the reader of the system design document to understand the design and the spectrum of choices available for design and implementation.

**Figure 1 Component Diagram**



* + 1. Deployment Diagram

The deployment diagram represents the hardware topology used and the runtime system assigned. The hardware encompasses processing units in the form of nodes as well as communication relationships between the nodes. A runtime system contains artifacts that are deployed to the nodes.

**Figure 2 Deployment Diagram**

# Concurrency Identification

The system can provide access to multiple users from different machines, and since the database server is multi-threaded it can handle several users at a time.

In Dart, we can accomplish concurrency by utilizing the Isolates. The isolates are autonomous

workers that don’t share memory however rather interconnect by ignoring message channels.

.

# Hardware/Software Allocation

* **Hardware requirement for server side:**
* Processor: 5 Core, 2.5GHz
* Secondary Storage: 1TB HDD
* ROM: 52X CD ROM Drive
* Memory: 86GB RAM
* Network Adapter: Ethernet Adapter
* Modem: 1mbps
* Others: 17” Color Monitor, Printer, Keyboard, Mouse.
* The system needs to be operating 24 hours; hence, power supply should be 100% assured.
* **Software Interface for the server side:**
  + - * Platform: Windows, Latest versions are recommended
      * Operating System: Windows 8 and above
      * Back-end tool: JavaScript
      * The system has PHP v8.0 that is used to load and compile server side scripts
* **Software Interface for the client side**
  + - * Mobile android operating system above API 21
      * The system browser should be JavaScript enabled so that it uses JavaScript for client side scripting and compilation of Web Pages.
      * The system browser should be JavaScript enabled so that it uses JavaScript for client side scripting and compilation of Webpages.

## System Performance

### **General system performance**

The system is expected to respond to requests within a few seconds not exceeding 4-8 seconds. For tasks that may take longer it is desired to respond in the 1-minute else request the user to reload.

### **Input/Output Performance**

There is no need for extra hardware to handle the data generation rate of the system. The existing communication bandwidth is sufficient to support all the communication between subsystems.

### **Processor allocation**

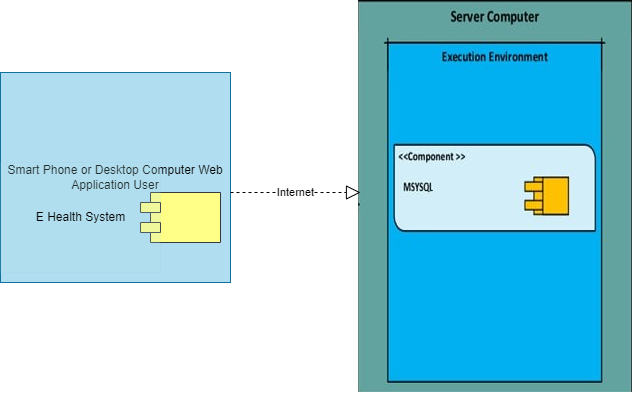
The computation rate is very small compared to other applications that the deployment system’s process can handle normally

### **Memory allocation**

There is enough memory to handle requests. The traffic generated by the application is medium.

## Connectivity

**Figure 3 Connectivity Diagram**



## Network architecture

The general interaction mechanism and protocol for the channels and the communication are synchronous communication so that the server should be available for the client, otherwise the communication will be disconnected. The expected bandwidth for the networking is 25MB/S.

# Data Management

The data generated by the system is saved on a single server’s database using MySQL. and the data is not distributed as a backup on every weak end and the data is accessed very often while the system is being used.

# Global Resource Handling

There is a multitude of resources that are available for different kinds of users but, there needs to be a differentiation between the Patients, the doctors, and Admin. We accomplished this by having different sets of user authentication for these different kinds of users. The authentication scheme used in this program is password-based authentication. The patients for example will log in with their patient's id or email address and password and can access the only needed parts to make appointments and follow-ups with a doctor whereas Doctors will have the choice to respond to patients that request appointments or follow-ups with them and nothing more. The resources are partitioned into different categories depending on who is permitted to access them, and the type of authentication used to log in will determine which sets of resources will be made available to that user.

# Software Control Implementation

## External control flow (between subsystems)

First of all, we have an email or id and password authentication model to authenticate users and detect and differentiate whether the user is a patient, doctor, or admin and a MySQL-based database in which all data management occurs as well as all the different privileges are stored, the admin is capable of adding or removing doctors and accessing appointment history of patients. But a standard user is only capable of accessing only portions of the subsystems. All of the tools provided by the system can be accessed via login with different permissions

## Concurrent control

In this system, there are no subsystems that run concurrently

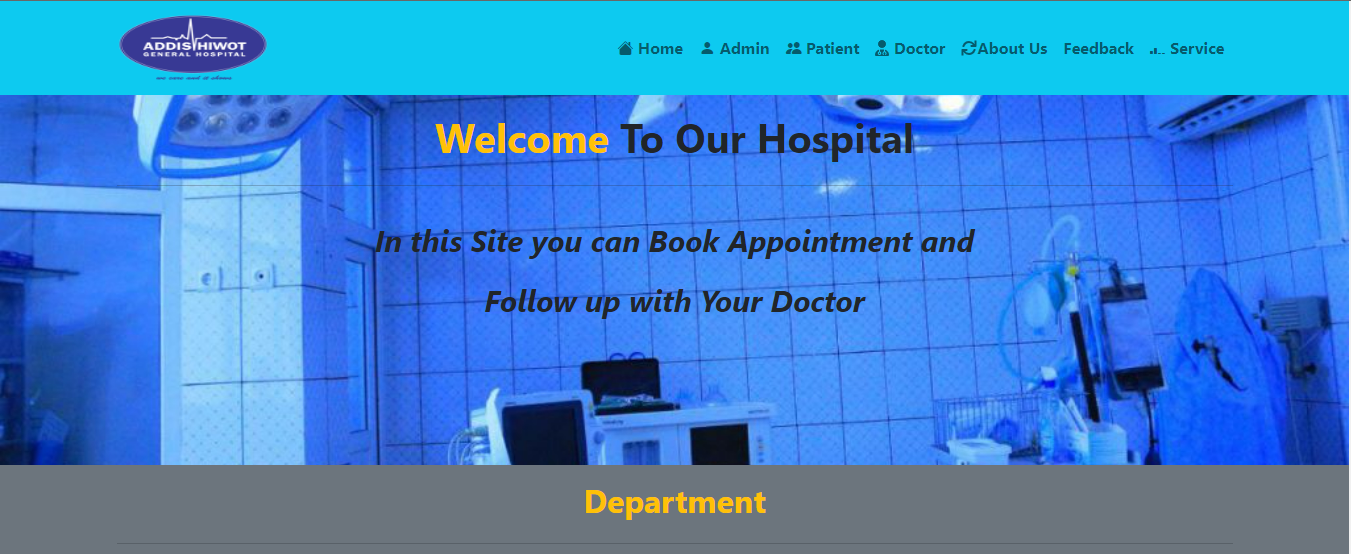
## Internal control (within a single process)

All of the subsystems found in our system will be modeled as to the specifications of the user, for making the whole system more useable and efficient. Also, the way the subsystems can communicate with each other is through complete access to the database. And the whole system and subsystems were designed to have better usability and functionality by allowing page-to-page transference when an object is clicked.

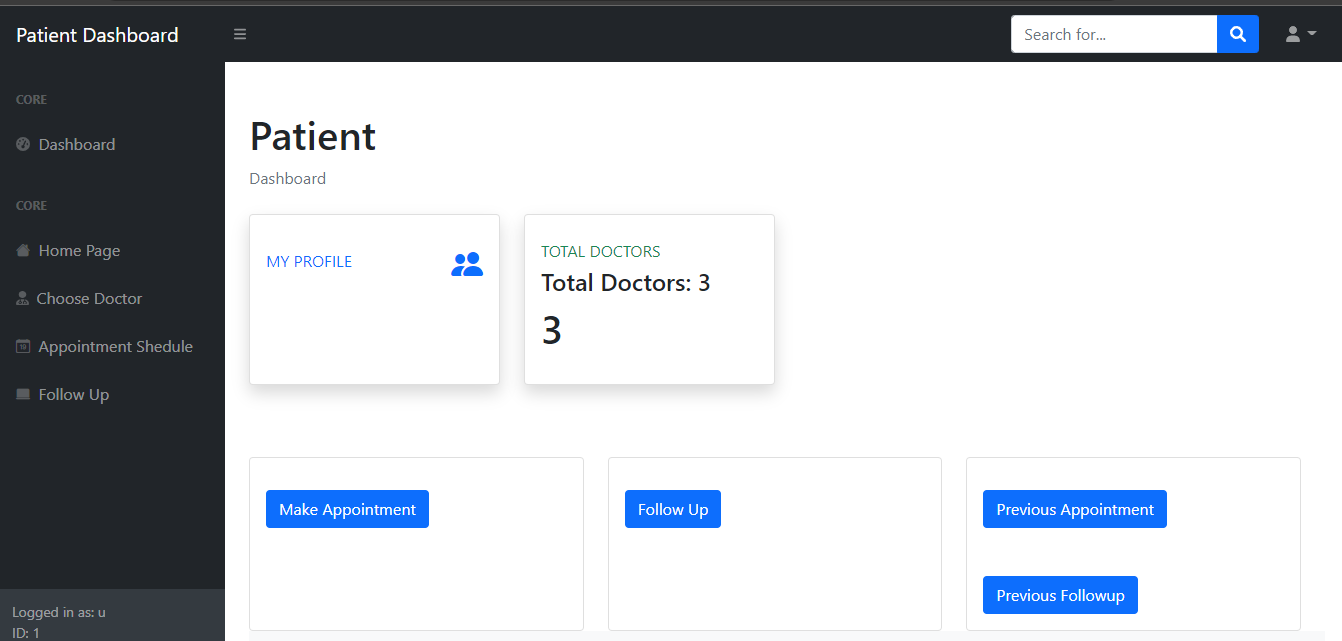
## User Interface

Some of the user interfaces are shown below:

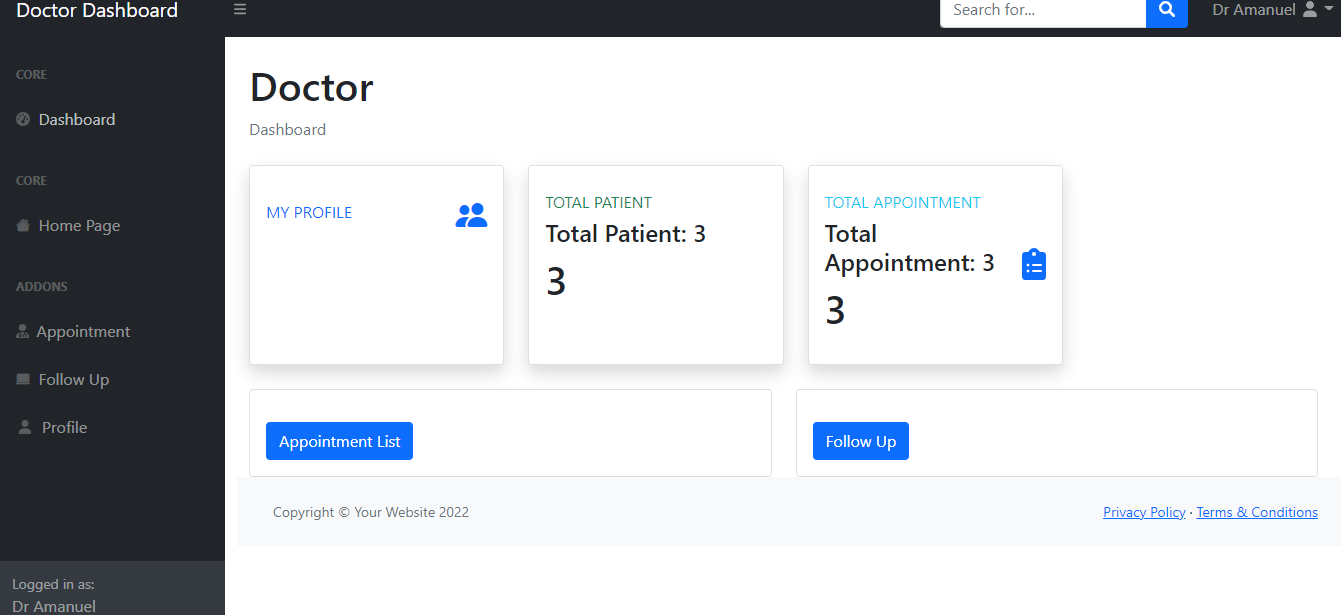
**Figure 4 Home Page**



**Figure 5 Patient Home Screen**



**Figure 6 Doctor Home Screen**



# Boundary Conditions

## Initialization

Before opening the system the user first needs to turn on xampp server so the system can load needed data on start-up after that the system can be opened from an application or website which will first show the front page users can choose not to log in to if they want but they will not be able to access any of the system's functions.

## Termination

The Termination conditions for the system are as soon as the users sign out or destroy the life Cycle of the application. At the time of termination, subsystems will be notified of any action they need to be perform. Database commits will be done by the subsystems when the system terminates. And the database needs to be closed personally by the user.

## Failure

Any incomplete transaction will be rolled back to the point where it was before the system process started.

# Design Rationale

**Maintainability**

* + - * The system will be maintained periodically through effective monitoring and evaluation. This will go a long way to help identify and debug emergency production problems and address them accordingly. A considerable amount of time would be spent to effect changes in data, files, reports, hardware, and software.
      * That means we can remain agile and keep moving out to new releases quickly.

**Testability**

* + 1. **Functionality Testing**

All functions in the application, database connection, and forms used to enter data for submission, editing, and getting or deleting information from users were tested. Developers performed the test of the website. Some functionality requirements were tested during the test security and Database of the system test.

1. **Extensibility**
   1. we will have a better chance of making any of those parts more reusable.