**EGERTON UNIVERSITY**



**FACULTY OF SCIENCE**

**COMPUTER SCIENCE DEPARTMENT**

**SYSTEM DOCUMENT**

FOR THE PARTIAL FULFILMENT OF DEGREE IN COMPUTER SCIENCE IN EGERTON UNIVERSITY

**OUTPATIENT RECORDS MANAGEMENT SYSTEM**

PROJECT: COMMUNICATION BETWEEN STAFF MEMBERS IN EGERTON

FACULTY: SCIENCE

DEPARTMENT: COMPUTER SCIENCE

UNIT: COMP 390 TEAMWORK PROJECT

PRESENTED TO: MR. BENJAMIN ODIYO

PRESENTED BY: GROUP 13

GROUP MEMBERS:

S13/20757/13 Innocent Omido

S13/09710/15 Kijedi Wesley

S13/09664/15 Christine Peter

S13/09709/15 Michael Makali

S13/09690/15 Erick Ogaro

Part II: Software Requirements Specification

Table of Contents

[System Request 5](#_Toc523455975)

[1. Introduction 6](#_Toc523455976)

[1.1 Purpose 6](#_Toc523455977)

[1.2 Scope 6](#_Toc523455978)

[1.3. Document Overview 6](#_Toc523455979)

[1.4. References 7](#_Toc523455980)

[1.5. Glossary 7](#_Toc523455981)

[2. Overall Description 8](#_Toc523455982)

[2.1 Product Overview 8](#_Toc523455983)

[2.2 Project Functions 9](#_Toc523455984)

[2.3 User Characteristics 10](#_Toc523455985)

[2.4 Constraints 11](#_Toc523455986)

[2.5 Assumptions and Dependencies 11](#_Toc523455987)

[2.6 Apportioning of requirements 11](#_Toc523455988)

[3. Specific Requirements 13](#_Toc523455989)

[3.1 External interface Requirements 13](#_Toc523455990)

[3.1.1 User interfaces 13](#_Toc523455991)

[3.1.2 Hardware interfaces 13](#_Toc523455992)

[3.1.3 Software interfaces 13](#_Toc523455993)

[3.1.4 Communications interfaces 14](#_Toc523455994)

[3.2. Functional Requirements 15](#_Toc523455995)

[3.2.1 User Class 1- Health Care Professional 15](#_Toc523455996)

[3.2.1.1 Functional Requirement 1.1 15](#_Toc523455997)

[3.2.1.2 Functional Requirement 1.2 16](#_Toc523455998)

[3.2.1.3 Functional Requirement 1.3 17](#_Toc523455999)

[3.2.1.4 Functional Requirement 1.4 18](#_Toc523456000)

[3.2.2 User Class 2 – System Administrator 19](#_Toc523456001)

[3.2.2.1 Functional Requirement 2.1 19](#_Toc523456002)

[3.2.2.2 Functional Requirement 2.2 20](#_Toc523456003)

[3.3. Non- Functional Requirements 21](#_Toc523456004)

[3.3.1 Performance Requirements 21](#_Toc523456005)

[3.3.2 Safety Requirements 21](#_Toc523456006)

[3.3.3 Security Requirements 21](#_Toc523456007)

[3.4 System Performance 22](#_Toc523456008)

[3.5 Software Quality Attributes 24](#_Toc523456009)

# System Request

A Records Management System

***Project Sponsor***: Egerton University, Medical Department

***Business need***: This project has been initialized to provide for on demand computerized access to and storage of patients’ medical records.

***Business Requirements***: Using this system, the medical staff should be able to create, update and store patient medical data from their desktop clients. They should be able to store patient records and manage the daily in-flow of patients into the premises. The system should have the following functionality:

* Add a new patient into the hospital records
* Update patient records at the hospital
* Queue patients at the hospital as they come in using the system.
* Record and fetch patient records.
* Archiving and backing up of patient records

***Business Value***: We expect that the cost of acquisition of manual filing material will reduce. Therefore, operational costs are meant to be on the decline after the break-even point. We also expect that customer service will improve due to implementation of the electronic queuing built into the system. The implementation of the system would bring to the department benefits such as:

* Reduced operational costs.
* Space saving due to electronic archiving of patient records.
* Increase in patient satisfaction due to increased system efficiency.
* Easy retrieval of patients’ medical data since the storage is centralized and up-to-date.

***Special issues and Constraints***: The Medical Department views this as a strategic system. This Record Management system will add value to our current business model, and it will also serve as a proof-of-concept for future technical endeavors.

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the “Outpatient Records Management System (ORMS) software. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications. This document is primarily intended to be proposed to the medical department for its approval and a reference for developing the first version of the system for the development team.

## 1.2 Scope

The document will identify the software to the developed and the underlying technology to be used during the implementation of the project. This includes the host database management system, the java supported client module and the inputs and outputs of the system. The SRS will show that the use of a secure socket layer in the protocol stack we hope to address the valid security concerns about the networking and transmission of confidential healthcare information.

By completion of this, this document will make clear the design team’s goals of creating value-adding software which not only correctly captures patient health information, but then efficiently stores it, sorts it, retrieves it, and delivers this critical care information where it is needed by healthcare professionals. The benefit of having accurate, complete, and timely health information is that it will inevitably save human lives.

This software is deliberately focused on medical records and the associated diagnostics. It is important to point out that this system which is life critical will not have cross functionality regarding appointment management, billing, or insurance functions, however diagnostic codes sets will be compliant with present legal addresses.

## 1.3. Document Overview

The Software Requirement Specification will clearly define and illustrate the overall project, its specifications and its requirements- both functional and nonfunctional. In addition the SRS will define the users and their respective characteristics as well as any limitations to development that the team has identified.

The format of the SRS document will address the overall project first- including functions and objectives in an overview. This section will also address how this software interfaces with other legacy systems. Then the subsequent sections will specifically address the components of the larger software system. These sections delineate specifications for every facet of the components design.

## 1.4. References

[IEEE] The applicable IEEE standards are published in “IEEE Standards Collection,”

2001 edition.

[Bruade] The principal source of textbook material is “Software Engineering: An Object-

Oriented Perspective” by Eric J. Bruade (Wiley 2001).

## 1.5. Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| IEEE | Institute of Electrical and Electronic Engineers |
| SRS | Software Requirements Specification |
| ORMS | Outpatient Records Management System |
| AES | Advanced encryption Standard |
|  |  |

# 2. Overall Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

## 2.1 Product Overview

Medical records are an essential part of the medical field. However, these records are not being fully utilized in the case study provided. Often the records are misfiled, inaccurate or duplicated unnecessarily. Recognition of the improvement of data digitization and networking as a constructive force often increases efficiency while lowering costs; it is our view that medical records networking will increase and foster the quality of healthcare offered. The information system, which is a primary link between the healthcare facility, physician and the patient, would be able to capture data, store it and retrieve it efficiently.

The System will be a web portal which will require the users to log in, so as to perform any task using it. There will be six users interacting with the system directly. The first entry part of interacting with the system will be by the receptionist who will be capturing the patient’s data. If the patient is a new patient, the receptionist will open a new record for that patient. If the patient has visited the hospital before and their records have been captured, the receptionist will be required to update the record and add that patient to the queue of that day.

The second person who interacts with the system will be the physician. His/her work will be to record the patient’s vitals; including weight, height, blood pressure, temperature and other related vitals.

The updated data by the physician will be viewable from the Doctor’s endpoint. He/she will take and record the patient’s prognosis and diagnosis into the system. In case the patients’ diagnosis is not direct, the doctor will have to capture tests into the system which will be viewable from the lab technician’s end point.

The fourth part of the system requires the lab technician to log in to the system, view all patients awaiting tests and act on them. He/she will therefore be required to post the test results.

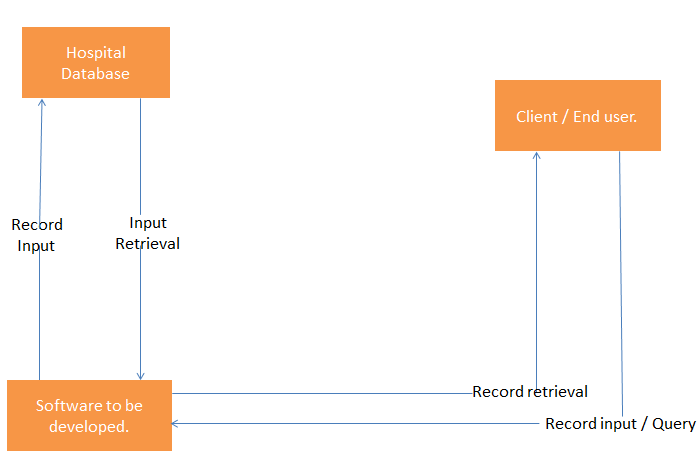
Since this is a data-centric product it will need somewhere to store the data. For that, a database will be used. A no-sql database, Mongo dB was considered for its ease in storage and scalability of data.

This information system is an industry-compliant application, based upon an open architecture, and is designed to function within a standard IEEE compliant Ethernet (10 or 100) Local or Wide Area Network environment. The communications protocol is TCP/IP, and is supported under any routing protocol within an infrastructure (routed or bridged).

The software is based upon standard and emerging web technologies.

## 2.2 Project Functions

* The software code will be portable between different operating systems such as Linux and Windows.
* The software will be easy to use and will require minimum manual operation.
* The software will have a user-familiar interface so that the system will not pose an additional workload to the users.
  + Note. Interface design will follow generally accepted model conventions for placement of dropdown menus and toolbars.
* The software will allow bidirectional synchronous communication between the user and the data source in real time.
* The software will provide security of operation and confidentiality of information (restricting access to non-privileged users), by FAT32 compression of data and Rijndael (AES) encryption algorithms.
* The software should be able to interface and link all components of system.



## 2.3 User Characteristics

* The primary user will be a healthcare professional. These include, a physician, a medical practitioner, a lab technician and a pharmacist.
* Note. This is a Medical Information System therefore to limit access and ensure integrity of the data only licensed medical personnel have access to input, search, and update functions.
* Physician Office Administrators, System Administrators and/or Therapists will have limited access and information capabilities.
* Note: For the reasons clearly stated in 2.3.1 the System Administrator (or Vendor) will only be able to access data with his Admin access code in combination with the Physician’s code while in the physician’s presence.

## 2.4 Constraints

The Health Insurance Portability and Accountability has mandated various standards on security, privacy, transaction and code sets, and unique healthcare identifiers to which this system must adhere.

Legacy systems will be a constraint to the system since they must be considered and modified to interface with the new system design.

The hospital database will need a large storage capability and a process to archive outdated data, which is also a constraint

The Internet connection is also a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function.

## 2.5 Assumptions and Dependencies

The system relies on a healthy care professional relationship with the hospital system with which he/she is a staff member. It is assumed that all professional know the processes of the hospital. Each of them therefore can perform their tasks to the fullest.

The SDLC chosen to implement the system will be model driven and based on subsequent versions to insure data integrity and functionality.

Due to report length constraints imposed by Health regulations will be strictly followed but kept as a stand-alone document.

It is also assumed that every user of the system will be running a machine that has the internet capability and can run the application comfortably.

## 2.6 Apportioning of requirements

In the case that the project is delayed, there are some requirements that could be transferred to the next version of the application. Those requirements are to be developed in the second release.

# 3. Specific Requirements

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

# 3.1 External interface Requirements

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.

## 3.1.1 User interfaces

The system user interface is developed using reactjs, bulma html and (cascading style sheets) css to make it possible to have the screens that the end users will interact with.

A first-time user of the application should see the log-in page when he/she opens the application.

## 3.1.2 Hardware interfaces

The application will require desktop clients for each user of the system with capability of running web applications.

## 3.1.3 Software interfaces

Following are the software used for the outpatient records management system.

|  |  |
| --- | --- |
| Software used | Description |
| Operating system | We have chosen Windows operating system for its best support and user-friendliness. |
| Database | To save the patient records we have chosen No-SQL database, Mongo dB. |
| Java | To implement the project we have chosen Java language for its more interactive support.  We will be using spring framework for the backend development. |

The communication between the database and the web portal consists of operation concerning both reading and modifying the data.

## 3.1.4 Communications interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems.

## 

# 3.2. Functional Requirements

## 3.2.1 User Class 1- Health Care Professional

## 3.2.1.1 Functional Requirement 1.1

|  |  |
| --- | --- |
| **Use Case Name** | Capture Patient’s Details |
| **Priority** | Essential |
| **Description** | The use case helps validate the patient’s details. |
| **Triggers** | A patient comes into the reception. |
| **Actors** | Receptionist (primary), patient (secondary). |
| **Preconditions** | The receptionist must have an account in the system and log in to verify the patient’s details. |
| **Main flows** | 1. The patient provides the receptionist with their identification documents.  2. The receptionist keys in the identification details of the patient i.e. registration number if student, id number if a community member, or staff number if the patient is a staff.  3. After verification the receptionist sends the patient to the physician.  4. exits the system. |
| **Alternate flows** | 1. Fails to identify the patient if not a registered member.  2. Creates a new patient. |
| **Post condition** | The receptionist is on the ORMS home page. |

## 3.2.1.2 Functional Requirement 1.2

|  |  |
| --- | --- |
| **Use Case Name** | Capture patient’s vitals |
| **Priority** | Essential |
| **Description** | The use case helps capture the patient’s vitals. This include temperature, weight, height and BMI. |
| **Triggers** | Patients on queue list. |
| **Actors** | Physician (primary), patient (secondary). |
| **Preconditions** | The physician must have an account in the system and log in to receive all the patients’ entries from the receptionist. |
| **Main flows** | 1. The physician receives the patient with the entries from the receptionist..  2. He/she takes the patient’s vitals that include temperature, weight, height and blood pressure.  3. The physician then records the patient’s vitals.  4. exits the system. |
| **Alternate flows** | None. |
| **Postcondition** | The physician is on ORMS home page |

## 3.2.1.3 Functional Requirement 1.3

|  |  |
| --- | --- |
| **Use Case Name** | Record patients’ prognosis and diagnosis |
| **Priority** | Essential |
| **Description** | The use case helps capture the patient’s diagnosis and prognosis. |
| **Triggers** | Patients on queue list. |
| **Actors** | Doctor (primary), patient (secondary). |
| **Preconditions** | The doctor is on the ORMS home page. |
| **Main flows** | 1. The doctor looks up the patient at the top of queue and calls them.  2. He/she conducts a prognosis for the patient. They also conduct a diagnosis thereafter.  3. From the diagnosis, the doctor prescribes the patient’s medicine and directs them to the pharmacy to collect the drugs.  4. exits the system. |
| **Alternate flows** | 1. In the case that the patient’s problem cannot be identified from the diagnosis, the patient is send to the lab to take some related tests to help the doctor prescribe the medicine for the patient.  2. The doctor exits the system until he/she receives test results queue. |
| **Postcondition** | The doctor is on the his/her home page |

## 3.2.1.4 Functional Requirement 1.4

|  |  |
| --- | --- |
| **Use Case Name** | Record test results |
| **Priority** | Essential |
| **Description** | The use case helps capture the result of the patient’s tests done by the lab technician. |
| **Triggers** | Patients on the test queue. |
| **Actors** | Lab technician (primary), patient (secondary). |
| **Preconditions** | The lab technician is in ORMS home page. |
| **Main flows** | 1. The lab technician views the patient test queue and takes the tests requested by the doctor.  2. He/she then updates the records.  3.exits the system |
| **Alternate flows** | None. |
| **Postcondition** | The lab technician is on his/her home page. |

## 3.2.2 User Class 2 – System Administrator

## 3.2.2.1 Functional Requirement 2.1

|  |  |
| --- | --- |
| **Use Case Name** | Add New user |
| **Priority** | Essential |
| **Description** | The use case helps create new health care professionals’ accounts in the system. |
| **Triggers** | Employment. |
| **Actors** | Administrator. |
| **Preconditions** | The admin must be connected to the internet and on ORMS home page. |
| **Main flows** | 1. The Admin Logs into his portal.  2. He/she creates a new user, giving them the right credentials depending on their position.  3. exits the system. |
| **Alternate flows** | None |
| **Post condition** | The admin must be connected to the internet and on ORMS home page. |

## 3.2.2.2 Functional Requirement 2.2

|  |  |
| --- | --- |
| **Use Case Name** | Delete user |
| **Priority** | Essential |
| **Description** | The use case helps delete health care professionals’ accounts in the system. |
| **Triggers** | Staff ceases to be an employee at the medical department. |
| **Actors** | Administrator. |
| **Preconditions** | The admin must be connected to the internet and on ORMS home page. |
| **Main flows** | 1. The Admin Logs into his portal.  2. He/she deletes user, giving them no right to access the portal again.  3. exits the system. |
| **Alternate flows** | None |
| **Post condition** | The admin must be connected to the internet and on ORMS home page. |

# 3.3. Non- Functional Requirements

These are requirements that are not functional in nature. Specifically, these are the constraints the system must work within.

## 3.3.1 Performance Requirements

***3.3.1.1 Normalization***

The basic objective of normalization is to reduce redundancy which means that information is to be stored only once. Storing information several times leads to wastage of storage space and increase in the total size of the data stored.

If a database is not properly designed it can give rise to modification anomalies. Modification anomalies arise when data is added to, changed or deleted from a database table. Similarly, in traditional databases as well as improperly designed relational databases, data redundancy can be a problem. These can be eliminated by normalizing a database.

Normalization is the process of breaking down a table into smaller tables. So that each table deals with a single theme. There are three different kinds of modifications of anomalies and formulated the first, second and third normal forms (3NF) is considered sufficient for most practical purposes. It should be considered only after a thorough analysis and complete understanding of its implications.

## 3.3.2 Safety Requirements

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery method restores a past copy of the database that was backed up to archival storage (typically tape) and reconstructs a more current state by reapplying or redoing the operations of committed transactions from the backed up log, up to the time of failure.

## 3.3.3 Security Requirements

Security systems need database storage just like many other applications. However, the special requirements of the security market mean that vendors must choose their database partner carefully.

* The software interface must follow design conventions which allow for familiar location of drop down menus, help etc.
* Input errors will be returned in red with appropriate message box.
* More than three attempts at login and failure will produce a red flag to system administrator.

## 3.4 System Performance

* The Hospital software should be able to support up to 5 simultaneous users.
* The Hospital software should support a no-SQL server database.
* 95% of the transactions shall be processed in less than one second.
* Data should be secured and backed up every quarter hour.
* Power supply should have a backup and a disaster recovery plan.
* System should be operable 24 hours a day and accessible in real-time.

In order to use the system

The User:

*Should be logged in to the desktop client*

*Scenario: Successful log-in*

*Given the user wants to log in*

*When the user logs in with his/her account*

*Then the user should be logged in health care professional.*

*Scenario: Retrieve password*

*Given the user wants to log in*

*And has lost the password*

*When the Admin enters the user’s identification number in the “Retrieve password” form*

*And submits the form.*

*The password is then reset*

Feature: Administrator log in

*In order to administer the system*

*An administrator*

*Should be logged in to the web-portal*

*Scenario: Successful log-in*

*Given the administrator wants to log in*

*When the administrator logs in with an administrator account*

*Then the administrator should be logged in as an administrator*

## 3.5 Software Quality Attributes

***Availability***: The system should be available for use at all times.

***Correctness***: The system should contain correct patient data as keyed in the system.

***Maintainability***: The system records should maintain correct patient records as keyed in the system.

***Usability***: The patient records should satisfy a maximum number of customers’ needs.