University of the Witwatersrand

Requirements Analysis Document

Team Name:

Level Seven Crew

Product Name:

HealFolio

Team Members:

Jan Badenhorst
Adam Lerumo
Tumbone Asukile
Daniel da Silva

Senior Lecturer:

Dr. Terence van Zyl

August 14, 2016

1 Introduction

1.1 *Purpose of the System*

The software *(HealFolio)* is aimed at improving service delivery for all types of medical practitioners; specifically the filing system currently used by medical practitioners. This software will provide doctors with easier access to patient records and can provide more security. It will also help them make more accurate diagnoses, leading to better prescriptions and more, when the software is further developed.

1.2 *Scope of the System*

The software will be use by doctors, assistants and even patients themselves. By using a web based application the each of the following users of the software can easily get to and use the information by accessing the user friendly interface. This will make it possible for both the service provider and receiver to give accurate diagnosis and critical feedback

1.3 *Objectives and Success Criteria of the Project*

Objectives:

- To make the precious minutes count in an emergency and avoid any long term complications.
- Take away the unnecessary revisiting or come backs to doctors to a minimal.
- Get accurate diagnosis at the first visit for recurring problems.
- Taking the admin work away for patients, especially for the elderly.

Success:

- Making the software easy to use and understand.
- Open to adaptations and future upgrades.
- Robust and workable on most platforms. *(Mobile friendly, Linux)*
- Having the system to be alert in failure to connect to server, which makes it virtually impossible to lose information.

1.4 *References*

Software Engineering

1.5 *Overview*

Utilizing the resources available and time to create the optimal running application that will incorporate every feature desired by the clients. Keeping in mind future growth in user numbers by keeping efficiency to the maximum.

2 Current System

2.1 *Private Practices*

Most practices keep electronic records of basic patient information, including but not limited to:

- ID number
- Name
- Visitations
- Medical aid number
- Medical aid scheme
- Prescriptions

These systems are largely primitive in nature and have only the bare minimum information required to operate. They don't have a lot of functionality in terms of interactive notifications. For example, it is a common problem that some patients get treatment on another patient's medical aid.

It is also important to note that these systems are practice-specific. A patient's file would have to be formally requested by another practice if the patient goes elsewhere. This is critical because some medications interact poorly with eachother and can be *fatal*.

A large concern for a centralized system is *security*. The idea of having everyone's medical history saved to a server and accessed by any doctor is dangerous ground, security would be paramount to ensure legal compliance, for example-legally-a person has sole discretion on whether to disclose their HIV status or not.

Another concern is *ease of use*. A standard consultation with a GP is ten to fifteen minutes long. The current system has been streamlined to a great proportion due to the time taken to enter the information. The age-old addage *time is money* applies; especially in a profession as highly incentivised as this.

2.2 *Government Practices*

Government practices, at least in the Ekhuruleni Municipality, share a centralized database of patient information. It is currently unknown what information is kept however this is largely irrelevant due to the fact that the system is not commonly used to it's full potential. This is mainly due to the high volume of patients at government hospitals as a result of external factors like poverty in the country.

This re-iterates the importance of *ease of use* in that doctors can not be spending large proportions of their time capturing data.

The concerns in the *Private Practices* sector above generalize to government practices. In summary, the only differences are:

- Government practices already have a central system
- This system is not well used and likely not easy to operate

3 Proposed system

3.1 *Overview*

The proposed system aims to address issues with the current two systems (Private *and* Government practices) while also connecting the two and making patient history ultra portable. Secondary to these tasks is enabling patients to view their own history as well as practice information. This *may* be integrated into the main application later.

3.2 *Functional Requirements*

The functional requirements are generalized as follows:

- All information on patients must be kept in a database accessible to the application at all times
- Doctor able to view patient information including but not limited to:
  - ID number
  - Name
  - Allergies
  - Date of birth
  - Age (calculated)
  - Gender
  - Race
  - Medical aid provider
  - Medical aid scheme
  - Medical aid number
  - Main member ID number
  - Main member name
  - Main member medical aid number
- Doctor able to view patient medical records which consists of the following:
  - Visitations
  - Diagnoses
  - Prescriptions
- Doctor able to view reports such as:
  - Chronic conditions
  - Critical information (eg Allergies, gender, diagnoses etc)
  - Family medical history?
- Doctor able to add information for an existing patient such as a new visitation, diagnoses, prescriptions.
- Doctor able to add new patients.

Table 1: Requirements

Identifier	Priority	Requirement
REQ1	4	Doctor be able to view own added patient information.
REQ2	2	Doctor able to view patient medical history.
REQ3	Discuss	Doctor able to view various reports.
REQ4	4	Doctor able to add new patients.
REQ5	Done	Doctor able to log in to system.
REQ6	Discuss	Doctor able to update patient information.
REQ7	Done	Administrator adding doctor.
REQ8	Done	Doctor able to sign-up.
REQ9	5	Patient able to sign-up.
REQ10	3	Doctor able to view information from referred patient.
REQ11	2	Add verification for doctor to view referred patient information.
REQ12	4	Patient able to update patient information.
REQ13	2	Patient able to remove doctors that has privileges to view the patients information.

3.3 *Nonfunctional Requirements*

Nonfunctional requirements identified by the shortfalls of the current system and general knowledge of the field are as follows:

3.3.1 *Usability and Interface*

Usability and Interface are two key elements in the success of the project. They are important in ensuring a doctor's time is used effectively and not wasted on capturing data. They are also key in ensuring the system is actually used, especially in high volume scenarios such as government hospitals.

3.3.2 *Reliability*

The system needs to be reliable to ensure that there are no gaps in the patient's history which could result in medication clashes and other undesirable circumstances. A likely solution would be to cache data entered locally if the server is down and then send it to the server once it is restored. Likewise the data for patients who regularly attend a given practice can be cached at the practice in case the systems go down. This would minimize the impact of downtime on the doctors and patients.

3.3.3 *Performance*

Performance is important in that a doctor cannot wait long periods of time for a patient's file to load. Caching data for patients who attend a particular practice can help with this, however if the patient is new to the practice performance and reliability could be at stake.

3.3.4 *Supportability*

In software engineering and hardware engineering, serviceability (also known as supportability,) is one of the -ilities or aspects (from IBM's RAS(U) (Reliability, Availability, Serviceability, and Usability)). It refers to the ability of technical support personnel to install, configure, and monitor computer products, identify exceptions or faults, debug or isolate faults to root cause analysis, and provide hardware or software maintenance in pursuit of solving a problem and restoring the product into service. Incorporating serviceability facilitating features typically results in more efficient product maintenance and reduces operational costs and maintains business continuity.

3.3.5 *Implementation*

The implementation needs to be easy as a patient might switch from a practice using the system to a practice that isn't yet. In this case the patient may insist on the new practice using the system at least for them. The doctor would need to be able to pull the patient's records without having to download an application or sit through an extended sign up process.

3.3.6 *Legal*

The system would have to adhere to many regulations and laws in place such as the privacy of a patient's HIV status. A decision has to be made as to whether something is declared the doctor's responsibility or the application is made more secure. A good example would be whether or not any doctor can access any patient's information. The responsibility could either be placed on doctors to not use the database for personal reasons or it could be made secure by only allowing doctors access to a patient's information when a patient grants it to them.

3.4 *System Models*

3.4.1 *Scenarios*

- The doctor needs to sign-up for the application to recognize and verify the particular practice number.

3.4.2 *Use Case Model*

Use Case UC1: Name/Identifier: DocSign

Related Requirementsts: REQ5, REQ6

Table 2: Use cases

Actor	Actor's goal	Use case name
Doctor	Doctor able to sign-up and add practice details.	DocSign(UC1)
Doctor	Doctor able to log-in and add a patient.	DocLog(UC2)

Table 3: Detailed Use Case 1

Use Case;: UC1 **Name/Identifier:** DocSign(UC1)
Related Requirements: REQ5, REQ6, REQ8 **Initiating Actor:** Doctor **Actors Goal:** Doctor able to log-in and add a patient. **Participating Actors:** Admin will check to see if the practice is registered for the doctor to sign-up. **Preconditions:** That the application has no practice of the same description. **Post-conditions:** That the application has registered the practice and the doctor is able to add the practice information. **Flow of Events for Main Success Scenario:** → The doctor finds the home page and clicks the sign-up button. ← The client is taken to the sign-up page. → After entering the sign-up page, the practice details are inserted. ← Admin checks if the practice is registered and adds the doctor to the database. → If the practice is registered the doctor will be able to add patients by pressing the add patient button. ← The client is taken to the add patient page where the doctors patients are added. **Flow of Events for Extensions (Alternate Scenarios):** → If the practice is not registered. ← Admin will send an email to the client to of unsuccessful sign-up.

Table 4: Detailed Use Case 2

Use Case;: UC2 **Name/Identifier:** DocLog(UC2)
Related Requirements: REQ5, REQ14 **Initiating Actor:** Doctor **Actors Goal:** Doctor able to log-in and add a patient. **Preconditions:** That the application has to be sign-ed up to be able to log-in and add patient. **Post-conditions:** Patient details can be added to the practice records. **Flow of Events for Main Success Scenario:** → The doctor finds the home page and clicks the log-in button. ← The client is taken to the log-in page. → After entering logging-in, the practice details can be viewed, as well as the add patient button. ← Clicking the add patient button takes the client to corresponding page and the patient details can be added. **Flow of Events for Extensions (Alternate Scenarios):** → If the incorrect log-in details are entered, the client won't be able to proceed.

Figure 1: Use case diagram