Integration of an Encrypted Relational Database with Android-based Graphical User Interfaces for Supporting Nurses' Tasks at the Point of Care

by

Joseph Marrero Corchado

A project submitted in partial fulfillment of the requirements for the degree of

Master of Engineering
In
Computer Engineering
University of Puerto Rico
Mayagüez Campus
2014

Approved by:

_____	_____
Néstor J. Rodríguez, PhD. President, Graduate Committee	Date
_____	_____
José A. Borges, PhD. Member, Graduate Committee	Date
_____	_____
José Fernando Vega, PhD. Member, Graduate Committee	Date
_____	_____
Omar Colón Reyes, PhD. Representative of Graduate Studies	Date
_____	_____
Pedro I. Rivera Vega, PhD. Department's Director	Date

Abstract

This document describes the integration of a Relational Database with an Android-based graphical user interface for supporting the documentation of nurses' tasks at the Point of Care in a hospital. To accomplish the integration a REST API is developed and a Relational Database is re-designed. The integration is engineered with an approach to secure communication and encrypted data storing/retrieval. The system uses mutual SSL authentication to transmit and encrypt the data between the Android Application and the Server. The data received via the REST API is encrypted into the database. The Database is re-designed to store the encrypted data in tables using the InnoDB engine of MariaDB. The data relationship is enforced using the physician, nurse and patient IDs as keys. An Android application back-end was developed to facilitate the communication between application' graphical user interfaces and the REST API. The back end consists of classes that instantiate objects that the graphical user interfaces use to send data to be inserted in the database. The graphical user interfaces of the nursing documentation application are enhanced versions of a previous version of the nursing documentation system implemented on PDAs (Personal Digital Assistant). These were implemented in Java for the Android platform.

Resumen

Este documento describe la integración de una base de datos relacional con interfaces gráficas de usuario basadas en Android para darle apoyo a enfermeras en la documentación de sus tareas de cuidado a pacientes en hospitales. Para lograr la integración se desarrolló un API REST y una base de datos relacional es rediseñada. La integración es diseñada con un enfoque en comunicación segura y almacenamiento/lectura de datos cifrados. El sistema usa autenticación mutua SSL para transmitir y cifrar la comunicación entre la Aplicación Android y el servidor. Los datos recibidos vía el API REST es cifrada en la base de datos. La base de datos es rediseñada para guardar la información cifrada en tablas usando el motor de MariaDB InnoDB. La relación entre los datos es forzada usando los identificadores de las tablas "physician", "nurse" y "patient" como llaves. Un "back-end" Android (estructura que trabaja detrás de la interfaz gráfica de usuario de la aplicación) fue desarrollado para facilitar la comunicación entre la interfaz de la aplicación y el API REST. El "back-end" consiste de clases cuyo trabajo es instanciar objetos que las interfaces gráficas de usuario utiliza para enviar data para ser insertada en la base de datos. Las interfaces gráficas de usuario del sistema son un rediseño de una versión previa del sistema de documentación de enfermería que corría en PDAs ("Personal Digital Assistant"). Estas fueron implementadas en Java para la plataforma Androide.

To the love of my life: Yanitza Rodríguez.

Acknowledgments

I would like to thank Dr. Néstor J. Rodríguez for his tireless help, advice and direction during my studies and development of the project and for working so hard to help his students succeed. Thank you to Dr. José Borges, Dr. J. Fernando Vega, Dr. Bienvenido Velez and Dr. Manuel Rodríguez for sharing their knowledge and insight during my studies and for their recommendations, help and corrections regarding this project.

Thank you to Miguel A. Aleman, Carlos A.M. Rivera, Fernando M. Mari, Jose Ocasio, Jammy Velez, Brian L. Santiago, Pedro Colón, Javier Colón and Francisco Morales for all their help and excellent work during the development of the Android Application.

Thank you to my wife Yanitza who has stayed by my side and supported my dreams and decisions. Finally I would like to express my biggest gratitude to my grand mother Lusina, my mother Nilda, my parents in law Miguel and Hilda and to mamá Arcilia who supported Yanitza and me; I could have not done this without all their support.

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1 - INTRODUCTION

1.1 Justification

The mobile application for nurses presented in this work proposes a way to handle effectively the nursing documentation that is taken while a patient is under the care of the nurses in a hospital. During the patient's stay in the hospital the nursing staffs intervene with the patients in numerous occasions. Due to hospital and law regulations these interventions have to be documented. Hospitals in Puerto Rico still use traditional paper work to make this documentation. Traditional paper methods may result in documents hard to read and thus, can be misunderstood. This is particularly critical to the health of the patient when the document is a physician order that need to be transcribed by nurses before they can be executed. Papers take considerable space and require that the documents be put inside the patient's records manually. The records can only be accessed by one health care professional at a time.

The application presented in this work is a completely upgraded version of the PDA-based system presented in [Najera07]. That system was developed on MS Pocket PC platform and an MS SQL2000 database. The new version has been engineered from the ground up to run on the Android platform and uses modern web, security and database technology. The Android platform was selected because of its worldwide adoption and versatility. The Android Platform currently has more than 1 Billion activations worldwide. The latest poll shows that 56% of US adults own a smart phone. This data suggests that the majority of the nursing staff would be familiar with the Android graphical user interface, which should result in a shorter learning time in comparison with the PDA-based system because by 2007 nurses did not have experience with PDAs.

1.2 Objectives

The main purpose of this project is the integration of Android, JAVA, PHP and MySQL in the implementation of an Android smartphone-based nursing documentation system. The system is scalable and will allow easy maintenance and expansion in the future. Building the application on top of Open Source technologies allows the future maintainers or academic contributors to access documentation, guides and resources that are not equally accessible on closed source technologies.

The Human Computer Interaction objective of the application is to provide a system that allows easy interaction and high error prevention. The application aims to provide the nurses a tool to correctly document their interventions with the patients. Achieving the HCI objective will allow the nurses to document the patients' progress while providing patient care. It will provide a familiar language that will make the experience intuitive and allow nurses to complete their tasks successfully. The Nielsen Usability Heuristics [Nielsen93] were used to detect and correct usability problems.

The objectives from the technical back-end part are to allow the system to be easily extended or updated in the future and make the application secure. The system will hold private data from patients. Handling medical information creates a requisite that the data must be encrypted. The PHP and JAVA programming languages will be used in a modular way where classes work with a specific interface of the system, creating a closely encapsulated system. The network data and database queries will be passed in a timely (no lag/fast response) manner that allows a responsive system. The passing of the data from the mobile application to the web server to a centralized database will be done securely with major steps taken to prevent data loss during the transmission due to the mission critical nature of a patient health information.

1.3 Outline

The organization of the rest of this report is as follows. Chapter 2 provides a review of the research work preceding this project and other related work. Chapter 3 provides a description of the system developed including the interfaces, backend and database. Chapter 4 describes the findings of a heuristic usability test performed to the graphical users interfaces of the application. Finally, Chapter 5 addresses the conclusions of the project and proposed future work.

CHAPTER 2 – PREVIOUS WORK

2.1 Preceding work

Medical records management is evolving. Applications to handle users medical and insurance data are being developed and deployed in many countries. The need to apply good human-computer interaction principles and security in this field is of great urgency as the medical staff has patients' lives in their hands. Good graphical user interfaces (GUIs) provide a way for staff to better understand a system and to be able to perform tasks more securely and in timely manner. Electronic patient record systems with mobile access allow nurses to document their interventions with the patients at the point of care, thus reducing the mental load that usually takes place when nurses have to remember the patients' condition until they have time to document it [Fraser10]. These issues were the main motivation for the development of a PDA (Personal Digital Assistant) mobile nursing documentation system described in [Najera07]. The feasibility of mobile technology for supporting health care personnel on their intervention with patient in hospitals was demonstrated in a study in 2009 [Rodriguez09]. This study demonstrated that documentation of many nursing tasks with PDAs can be faster than when it is done in a paper record system.

Since the study in 2009 [Rodriguez09], mobile devices and network technology have changed dramatically. Adoption of iOS and Android devices has become as normal as using a laptop or a PC [Ehrler13]. In September 2013 Android was activated in more than one billion devices and those numbers are growing [Yarow13]. These technological changes motivated the development of the system that is presented in this document. It evolves from the system described in [Najera07] and tested with real nurses [Rodriguez07 and Rodriguez09]. Given the proliferation of smart phones the new version of the system should be easier to learn and interact with than with the PDA version since many nurses will be very familiar with the interaction paradigms of the smartphones. In the experiments conducted for the PDA system [Rodriguez07 and

Rodriguez09] none of the nurses had prior experience using PDAs, yet they end up completing over 90% of the tasks and preferred the PDA system over the paper-based or the tablet version.

There are other studies found in the literature like the one presented in this document. These studies are reviewed in the following sections.

2.2 User related vulnerabilities

The authors of [Fraser10] emphasize that in their experience when creating medical record systems we must take in consideration security on the passwords and data validation on the input fields, not only pointing to malicious security breaches but to accidental or negligent actions too. Making of the interfaces intuitive and easy to understand lowers the possibilities of errors committed by the users. In [Harman12] it was found that allowing the nurses to recycle or copy and paste text from reports can make them finish their tasks faster but errors are introduced on the reports. The copy and paste of data can pass information from one patient to another that has nothing to do with their medical data. Problems like this can lead to mistreatment and ultimately to patient death.

Interfaces must provide homogeneity and context awareness. As the author's of [Ehrler13] explain, homogeneity and context awareness provide security and prevent errors. The homogeneity not only means that the system looks similar in all its interfaces but that the system must resemble the previous system used by the nurses. Context awareness will limit what the users options to what they are currently doing thus limiting the input of data on wrong forms.

2.3 Hardware and Software

The research presented in [Sax09, Shiang-Lin11, Kume12 and Ben13] focuses on mobile devices. In [Sax09] a software was developed with senior patient care in mind. It is built on top of the Windows Platform to run on Windows Tablet PC. The software allows care givers in a retirement home to record and retrieve patient information while being next to the patient, specially if the patient is under an emergency. The Software monitors the vital signs information of the patient and provides alerts if the vital signs sensors are triggered. In [Shiang-Lin11] a software is described that focused on nurses information tasks on an Android smartphone. The Android application is designed to help nurses keep track of the notes and procedures they realize. It allows record search, note taking, patient profile updates and retrieval of patient history. In [Ben13] another Android-based software for tablets is described. This software helps manage physician tasks and shares their records with a network of insurance providers, hospitals, physicians and pharmacies. In [Kume12] a software that allows patient access to their own medical records developed for web and the Apple iPad is described. It allows the patient to view detailed information about their medical history, showing reports and charts about progress. All these applications have been developed for medium and small size mobile hardware, which raises the need of stable connections via wireless and user interfaces that can be easily understood on these kinds on screens. Mobile devices with these types of OS allow collecting data in a flexible way that can be specified by the developer, but the selection of the hardware and software stack used will enable how easy a user can interact with the system. The selection of the operating system also will depend of the needs of an open or close platform. The use of PCs, while providing advantages over paper, has the constraint of being at fixed place to take notes or enter data. Tablets and Phones eliminate this constraint. Modern tablets and phones have all the computing power to send and receive data in a timely manner. All these devices are mobile and

with touch screens and allow the medical staff to move and access the patients' clinical data while providing care to the patients.

It can be argued that tablet PCs and laptops provide similar mobility to the Nurses. The study by N. J. Rodriguez, et al. [Rodriguez07] proves that a smaller sized device like a PDA provides a similar productivity to that of a tablet PC (see Figure 1). At the time of the study the tablet PC were larger and heavier than current 7-inch tablets. Laptops, on the other hand, have been proven to slow the productivity of the nurses because a nurse cannot easily take the laptop in one hand and write with the other. The phones and smaller tablets allow the nurses to work on the record system without the need of a table or movable desk. Mobile phones with screens ranging from 4 to 5 inches can allow even one hand interactions, leaving the nurse one hand free to touch the patient to see their status if needed. Android applications allow an application to look similar on tablets and phones. Thus, an application can be deployed on devices of different sizes.

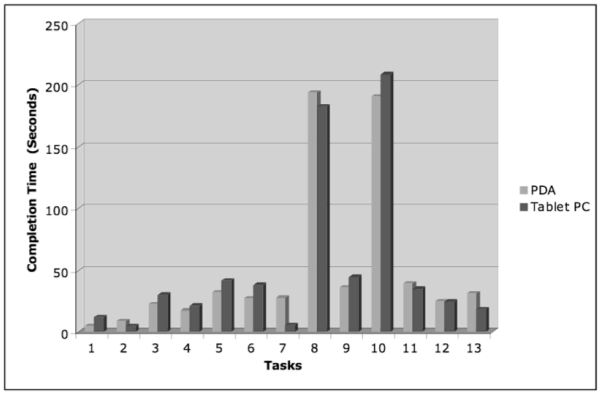


Figure 1: Completion Times for Each Task on the PDA and the Tablet PC Versions. Extracted from [Rodriguez07]

2.4 Data Retrieval

The proposed systems connects using databases via client/server connection using wireless networks. This is important for HCI because of the fast response of those systems, contrary to using paper to access data of the patients. A patient with an emergency that needs to be administrated a medicine will have to wait until the nurses check if the patient has any allergies to the medications. On a system powered via networks a nurse can access the information faster than searching on a paper record. The network systems combined with a good user interface allow the users to retrieve data efficiently and even access reports that would be very time consuming to create manually. The Interfaces shown in Figure 2 show the workflow of selecting and viewing medical records; the left side shows a list of records with a spinner to filter the records, the center screen shows an open record and the right screen shows a graph of one of the items in the opened record [Kume12]. This interface makes data retrieval in the background. All the data available for the next possible tasks is downloaded to the device while the main list (Figure 2 left) is shown. The task of narrowing the scope of the search and showing graphs or reports is done using the already pulled data. This way the user does not get any delay while working on these tasks. A system that follows the approach of retrieving new data from a server on each user request on the interface will be slower and allow user errors by making the nursing staff trying to make the system respond faster by tapping again or switching screens. From the data already retrieved users can view graphical charts created without needing additional network access. This allows better understanding of the patient status and shows the progress or history or patients in a clear and fast way. The paper by [Ilie13] presents a study in two hospitals that showed that intuitive interfaces and automation lead to improved patient care. Automation allows a user to accomplish tasks in smaller steps and less mental effort. Doing something similar with pen and paper would be time consuming. The [Ilie13] study shows how HCI mobile systems provide better solutions for Electronic Record Management.

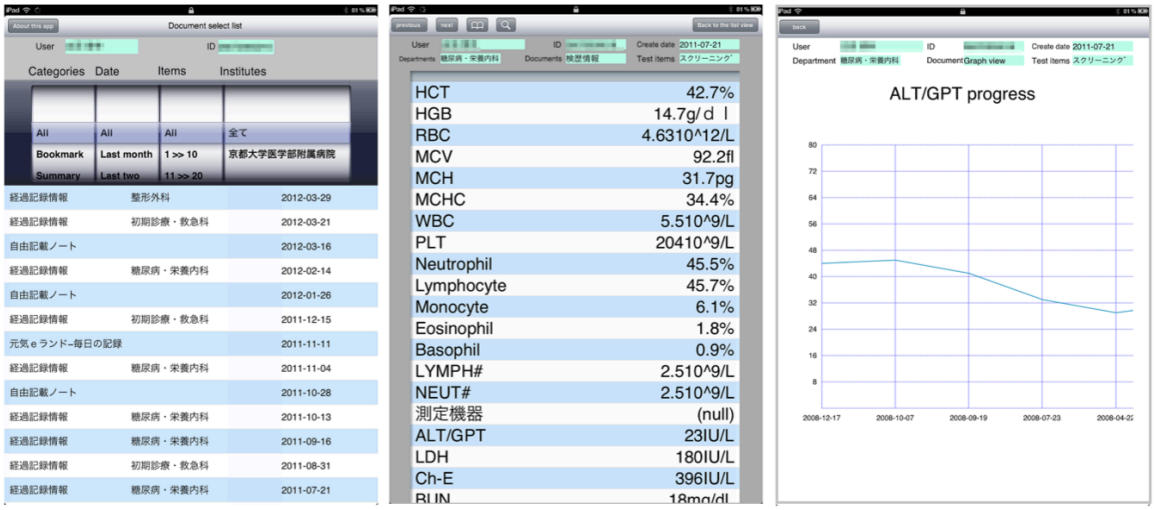


Figure 2: iPad2 data selection interface.
Extracted from [Kume12]

2.5 Automation of Data Collection

The [Sax09] and [Ben13] studies are about systems that collect data using technologies other than standard user input. Some of the technologies like NFC (Near field communication) can be used to acquire patient data from their phones allowing less input from the medical staff and allowing more time for patient care. The system described in [Sax09] uses cameras to monitor a patient. These technologies can be proven valuable in cases where patients are in urgent situations that need 24/7 visual monitoring. NFC technology would work if the patient has an application already on their phones that supports the medical system developed, which makes the NFC application limited for some patients. Other technologies like RFID (Radio Frequency Identification) tags or barcodes can automate processes like registering a patients into a hospital, scan medicine before administration, and monitor medical staff activities. Automation is used to lower the amount of user errors and provide better patient care.

Other technologies like live monitoring of the patients vital signs can be incorporated with the use of third parties equipment. The system described in [Sax09] allows the medical staff to monitor the patients' vital signs in real time. Figure 3 shows an interface that looks very similar to medical equipment monitors. The interface showed graphical charts that allowed the medical staff to identify rapidly if any abnormal situation is taking place with a patient.

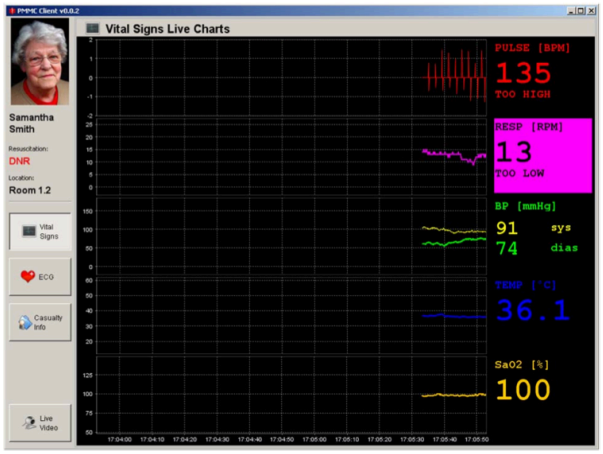


Figure 3: Vital signs live monitoring. Extracted from [Sax09]

2.6 Annotations and Data Error

Doctors and medical staff often use free writing [Axelrod11]. This type of writing can lead to misinterpretations of medical orders or nurses annotations. In Figure 4 we can see free handwriting converted with OCR software. The inscriptions scanned are ambiguous and can lead to errors and malpractice.

Sun25/5 12.30 at home – colapsed in toilet long history of IHD 1aMI bIHD for crem – pt inferior ref for angiogram

Feeling much better ESR27 cut pred to 5mg od re-start FeS04 200mg bd && see 3 wks w FBC && ESR

occasional – but much better. nil exacerbating note negative ETT in May – no evidence of ihd. thinks gastric origin

Figure 4: Example of free text.
Extracted from [Axelrod11]

A user interface that allows the user to enter specific type of texts in specific areas like the interface in Figure 5 can reduce the users input and allows specific data only. This reduces the amount of interpretation on the data [Axelrod11]. Interfaces like the one in Figure 2 allow the users to enter specific numbers using "wheel type" scrolling of menus which is an intuitive design that provides comfort to the user and makes them feel that they already know the system.

Figure 5: Update profile interface.
Extracted from [Shiang-Lin11]

2.7 Adoptions and Reception

Today's doctors and nurses have interacted with computers and mobile devices in some way. The previous experience creates an opportunity to make the user feel they know the application. The research by [Ilie13] shows us that the users are very comfortable using the application that they find similar to things they know. Thus, the feeling of knowing the application is very important when creating a user interface for mobile devices that a user will carry and work with constantly. The [Kume12] study shows tablets that can perform faster than a PC or a phone with a web interface. The [Sax09] study shows that the medical staff was comfortable using the system. The results of the [Ilie13], [Kume12], and [Sax09] suggest that medical staff is willing to use familiar mobile technology similar to the used on their personal computers and smartphones, that will allow them to save time writing and prevent errors.

2.8 Drawbacks of User Interface Restraints

User interface restraints to stop the user from writing freely and set interfaces that constraint input is the normal approach. This approach, as discussed earlier, is believed to lower the amount of errors the users do on the system. However, the [Ilie13] study revealed that these constraints result on new errors when done on not intuitive interfaces. These errors occur when a nurse or medical staff member cannot find an expected answer. This behavior was similar in two hospitals considered in the [Ilie13] study.

This information provides an important discovery because an interface that does not speak the user's language or that does not allow the normal options the medical staff is familiar with will result in errors. The errors created by an un-intuitive interface can be as devastating as the errors created by misinterpretation of free text writing. It is important to emphasize that when creating an interface intuitive means that the user can recognize the system options like they normally do or as stated in [Ilie13], an intuitive interface is "an interface that resembles something the user has already learned or interacted with".

2.9 General Remarks

The design of graphical user interfaces for electronic medical records using Human Computer Interaction (HCI) principles allows medical staff to complete tasks faster and with fewer errors. Data retrieval tasks provide great advantages in terms of amount of data produced by automatic reports and timely response. The data input tasks need to be carefully designed to provide interfaces that do not introduce new errors to the workflow of the medical staff.

Studies performed by [Ilie13], [Sax09], [Shiang-Lin11] and [Kume12] provide information about the interfaces and how well the users react to the new technology. Phones and tablets allow the

users to interact with the patient in a timely manner, resolving situations at bedside. This provides faster treatment to the patient. The users, when presented with graphical user interfaces that are similar to the ones they know in paper, complete the tasks easily and enjoy using the applications. The biggest challenge this technology faces is the free handwriting recognition to eliminate errors. The semantics of understanding free handwriting is not prepared to comprehend medical staff annotations fully, and constraining it can lead to new errors.

The current technology allows the replacement of paper work and provides advantages in speed, redundancy, multitasking and error prevention. The lack of current solutions to fix free handwriting annotations don't challenge the adoption of the HCI applications because paper work suffers of the same problem of hard to read handwriting. HCI designed interfaces and system frameworks (Databases and networks) provide a solution that potentially saves patient lives, hospital costs, and medical staff careers.

CHAPTER 3 - SYSTEM DESCRIPTION

3.1 Introduction

The portability of smart phones and their ability to link to a central database without a hard-wired connection make them an alternative for providing access to Electronic Medical Records (EMR) at the point-of-care. These devices are very well suited for accessing and capturing clinical data at bedside in a hospital. The viability of small mobile devices for accessing medical records has been demonstrated on previous studies [Rodriguez07] and [Rodriguez09]. The system described in this chapter implements security and human computer interaction on the Android platform creating a modern system with the elements required to help nurses tend to patients' needs in a secure, fast and more convenient way than using paper or fixed workstations.

To accomplish this, the structure of the system is divided in eight components (see figure 6). Five components make up the client application and three the server. The Android XML Interface code (UI Design) is the component with which nurses will interact with. Any graphical element used to interact with the system is built on this component. The second component is the Activity classes that give functionality to the Interfaces. These classes manage the listeners, the activity changes and decide how the data will be shown in the graphical layouts. The third component of the system is the classes that represent an entity or function in the hospital, for example the patients, positions, ulcers, etc. The forth component are the classes that work to create objects of the entity classes. These classes communicate with the Server to retrieve data and create entity objects with that data or send data to be inserted into the database. The last component of the Android application consists of the JSON Parser Class that parse the JSON encoded strings to be sent and received by the REST API and the Secure HTTP Client Class that handles the authentication and data communication with the server. The server part of the system is composed of the Apache web server which is a open source application that we only use and

configure in this system. The Apache web server manages the HTTP and HTTPS (secured) requests. The second component of the server is REST API, developed for the project in PHP. The REST API contains the SQL queries for each of the data requests made from the application. The last component is the Database that contains the data and relations of the data.

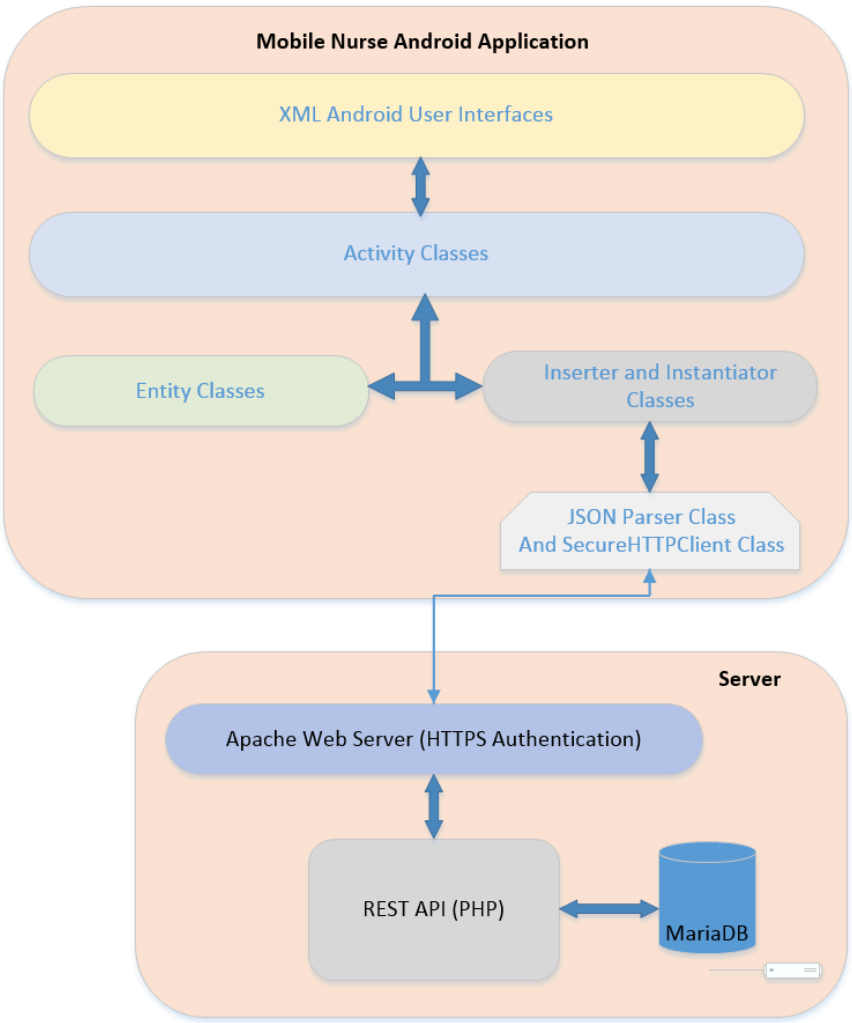


Figure 6: System Architecture.

3.2 System integration

The Database Design, the development of the REST API, and the integration of the graphical user interface are the main contributions of the project. The user interfaces were developed by a group of undergraduate students working on the research project that resulted in the nursing documentation application described in this document. Some of the user interfaces were modified by the author of this document in order to facilitate their integration. The collaboration between the team has been done using the Git version control system. The usage of Git via the hosted provider Github allowed the team to stay informed of what each member of the team is working on, the code is easier to review than other version control systems by providing a web interface and allows each member to have an updated code to work on as soon as a teammate commits it to the repository.

The porting of the system from the PDA-based version to the Android-based version was done in a purely function-based process, since the application had to be re-written completely to accommodate Android features and new network technology. The rewrite of the application made the user interfaces similar to the common applications of the Android platform, while providing the correct functionality to assist nurses with their patient care tasks.

The application was developed using the Android 4.x APIs and support library 4 that allows the use of sidebars, action bars and modern applications look and feel. MariaDB Foundation's MariaDB 10.x was used as the database for the project. The usage of MariaDB allows the application to be encrypted using methods available in the MySQL Database Management System. MariaDB is also fully compatible with other open source technology like Oracle's MySQL. The connection to the Database is made using the PHP language. A REST API is

implemented to do all the calls to the DBMS. The REST API code that works with the notes interface is shown in APENDIX A. This REST API connects with the Android Application to receive and send data. The data received by the application is JSON encoded, parsed, and placed into Array Lists containing objects of each of the interfaces data needs.

The Application and System structure reflects a loosely coupled system that allows each part of the application to be modified or replaced without having to re-write each part. The JAVA programming language is used to program this Android application. Due to the object oriented nature of the project, each data required by the interface is encapsulated. An object is created for each entity represented in the interface. The implementation handles patient information fetched from the Database by creating a patient object and multiple patient objects will be handled as an Array List of patients. Specific classes are in charge of populating the objects with the database-fetched information. By using JAVA and the Android API the application will run at native speed and use the full device resources.

This combination of tools and languages has been selected carefully to allow a multi OS, and secure implementation of the back-end of the mobile application. The combination of these tools prevents vendor lock-in as contrasted by the previous implementation, which locked the implementation to Windows Servers. The current implementation of the back-end with PHP and MariaDB can run on Windows, Mac OS X, FreeBSD, Linux servers and other Unix Like Operating Systems. The system also can be implemented on local or cloud servers allowing the system to be flexible to the future needs of the hospitals. If needed the back-end can be used by Web, iOS, Windows or any mobile, desktop or server application developed to connect to our REST API.

3.3 Security

Security in mobile devices is a very important aspect that must be addressed when working with applications that hold patient data. The development model of Android/Linux is the best fit for the project since it allows minimum-security breaches from the OS stack and provides the ability to encrypt the mobile device. In this way any cached information or connection details to the database are protected. Another decision to protect the system is the usage of MySQL/MariaDB database, which allows encryption of the needed data on the read and write operations. The authors of [Harman12] wrote: *"There are three major ethical priorities for electronic health records: privacy and confidentiality, security, and data integrity and availability."* These priorities were studied and decisions were made to develop the system from the ground up taking the necessary steps to provide the needed security. The REST API enables security, keeping the data of the patients on the server's database. The cached data on the mobile device memory for operation will be in the volatile memory of the device. If a device is stolen or lost, the data will be secured in the server and the device encryption will protect the application. The authentication to the database also occurs on the sever side. In case that a malicious attacker wants to reverse-engineer the executable of the application to gain access to the database it would not be successful in gaining access to the credentials.

The security is handled in each part of the system by encryption and user authentication. The Android application authenticates the user using Username and Password combinations that are encrypted in the database. The sensitive tables in the database are set to the varbinary type and all data of this type is stored encrypted. The data is encrypted and decrypted by the REST API (written in PHP) using the MYSQL function to encrypt using the AES256 standard encryption. To handle the communication and authentication via wireless networks from the Android devices and the server a mutually authenticated SSL (Secure Socket Layer) will be implemented. The

mutually authenticated SSL allows the server to communicate only with the Android application and the Android application only with the server. SSL encrypts all the data communication.

3.4 Main Server requirements

The server used to hold the database and the REST API is a Virtual Private Server running Fedora Linux version 19. The server had to be prepared and configured for the application needs. The first thing done was installing MariaDB 10, Apache and php. To install it on Fedora 19 it was necessary to add the repositories that hold the installation packages for MariaDB 10. We used MariaDB 10 instead of the MariaDB 5 that is included in the standard Fedora repositories because we are using the feature of having two columns in a table that set TIMESTAMP on UPDATE and CREATE of a record. In MariaDB 5 and MySQL 5.5 this feature is not existent. MySQL 5.6 and MariaDB 10 also include Optimizations to the InnoDB engine that makes it faster handling locks, when data collisions can occur [MariaDB14a].

3.4.1 Server and Client SSL configuration

The next step once MariaDB, PHP and Apache were installed and running was to set up the Apache SSL configuration. To accomplish this it was necessary to create a certificate and key pair to setup as server certificates. The keys are created with the openssl application found in most linux and Mac OS X servers. A key with at least 2048 bits is recommended. After creating the certificate and key, the Apache server was configured to specify the details of the newly created key and an https address was set for the application to connect to. Next it was necessary to create a second certificate and key pair for the client (Android application). To set up the application it was necessary to develop a new JAVA class to manage the secure connection. This

class reads the client and server certificates and makes sure that the server is showing the correct certificate before sending the client certificate to be validated. Once the server accepts only the certificate from the Android application and the Android application only accepts the certificate from the server, a mutual SSL authentication is accomplished. A mutual SSL authentication limits any connection of the application and the server to only each other. The Android application holds two certificates. The server certificate is stored in a BKS format and the client certificate with the client key stored in a pkcs12 format. To set up the keys created with openssl to BKS format the Bouncy Castle tool was used: bcprov-jdk15on-150.jar [BouncyCastle14]. The Apache web server configuration can be found in the Apache documentation [Apache14].

3.5 Database Design

The original database used in the PDA version of the system [Najera07] didn't provide a design based on encrypted file types. This allowed any system administrator or hacker to look at the data inside the database if the server password was available to them. The new implementation provides enhanced security by adding an extra layer of encryption to each data inserted in the varbinary fields. Another issue encountered was that the database did not retain the relationships between the tables when it was ported from MSSQL to MySQL.

The designed database concentrates the data into tables related to the graphical user interfaces that need them (see Figure 8). The relationship between the tables is done using the primary keys of each one of the tables. The cascading effect of deleting related entries in other tables is not used to prevent crucial data loss. Since this is sensitive data that holds patient and medical staff information, the database is encrypted using the AESENCRIPT method and VARBINARY fields. An example of the encrypted data can be viewed on APENDIX B. The Data should not be deleted in any moment, the patient that is not currently in the hospital is set to inactive but all

his/her data is preserved. All the reports and transactions made are recorded with the user id and timestamps showing date of creation and update, this data is not removed even after a patient is marked as inactive.

The current database uses row level locking and relationships thanks to the tables using the InnoDB engine that supports such features. The tables that will be updated by the application also have a “time on update” column that uses the new feature from MySQL 5.6/MariaDB 10 to set current time on UPDATE of the row. This allows the database to hold the current timestamp when nurses update patient information. In the previous versions only one column per row could be updated with current timestamp either on create or update, this version allows us to implement a more sensible solution by having both timestamps taken directly form the server.

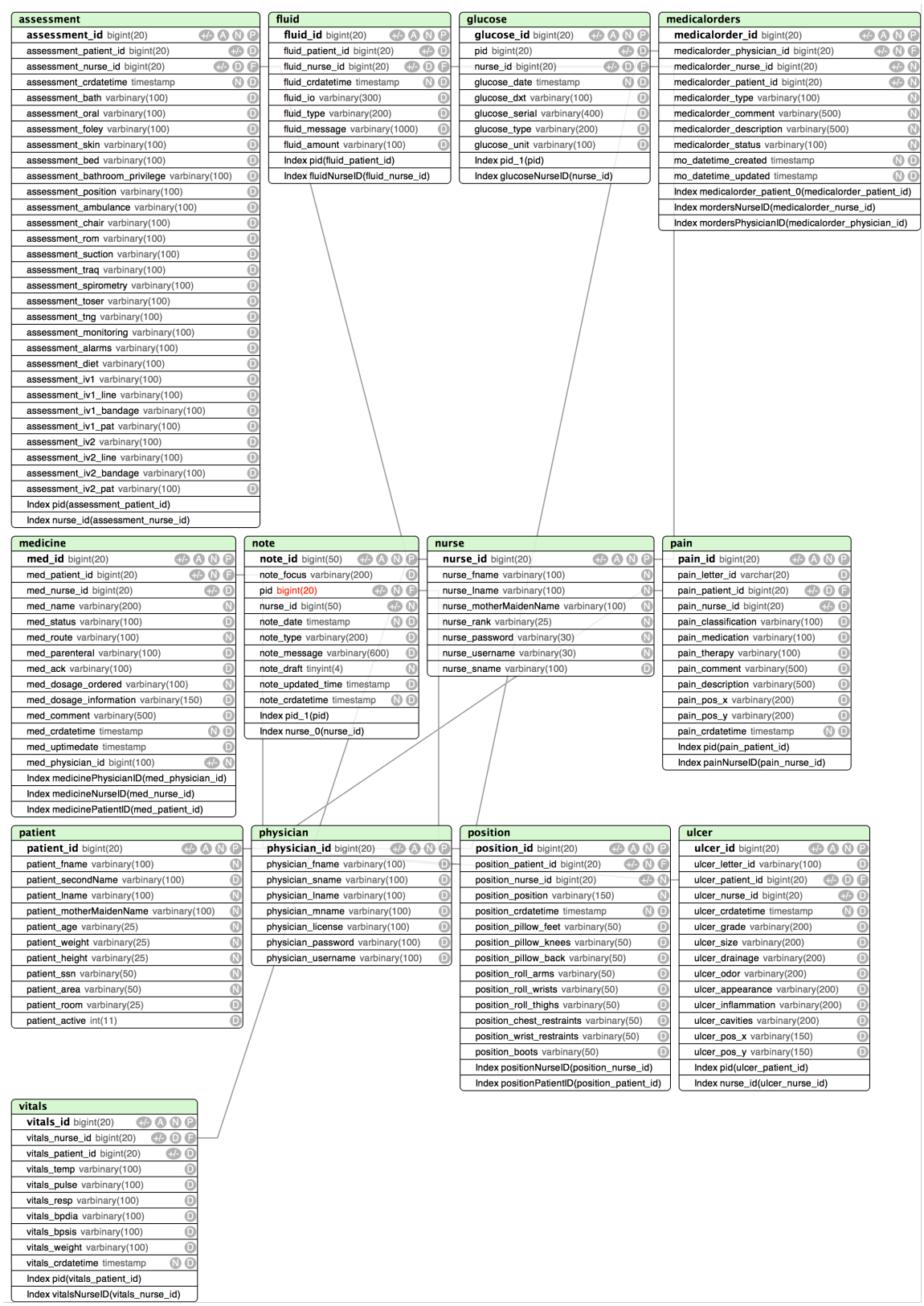


Figure 7: Database Relations Model
Full resolution model available in [Marrero14a]

3.6 – System Interfaces

3.6.1 – General Changes

Most of the functionality of the original PDA-based version of the system [Najera07] has been preserved. However, the look and feel of the graphical user interfaces experienced some changes in order for the application to be consistent with the look and feel of Android applications (see Figure 7). Due to the swipe capability of smart phones, the scroll bars were not implemented in the Android version. The buttons at the bottom of the PDA version were moved to the top bar of the Android version. Some of those buttons were eliminated because their action could be accomplished by tapping an entry of a list.

(a) PDA version [Najera07] (b) Android Version

Figure 8: Nursing notes interfaces

In the PDA version access to other documentation screens was accomplished by tapping the Record button at the lower left corner of the active screen. That action activated a pop-up menu with entries for each of the nursing documentation screens supported. Tapping on an entry opened the corresponding documentation screen. In the Android version the documentation screens are accessed through a side bar (see Figure 8). This mechanism allows the user to move from documentation screen to documentation screen with two taps or one swipe to the right and a tap. By developing this application using the Android SDK we are able to allow the nurses to type on the preferred language with suggestion, correction or auto-correction if needed. Another implication from developing for Android 4.x is that the user has the option to dictate instead of writing.

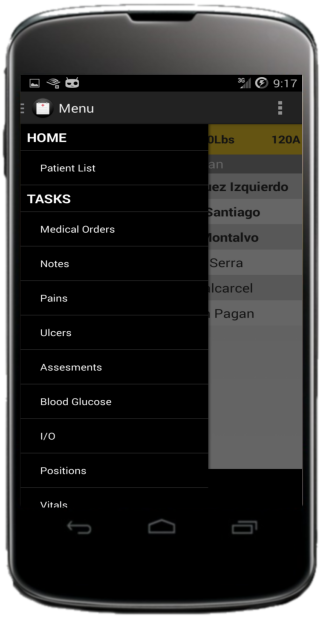


Figure 9: Navigation side bar.

3.6.2 Login Interface

The login interface shown in Figure 9 accepts username and password from the nurse. When the nurse presses the "Sign In" button the system checks the database to determine if the username exists in the dataset. If it does exist then it checks if the password entered corresponds to the password saved in the database. If the username or password is not present or the authentication is not validated the system will show a warning to the user, specifying the error. If the authentication passes the system loads the next activity and displays the patient list. When the system passes to the next activity, the nurse information is passed as an object so that it can be used when she/he makes an entry to the patient's record that requires her/his electronic signature

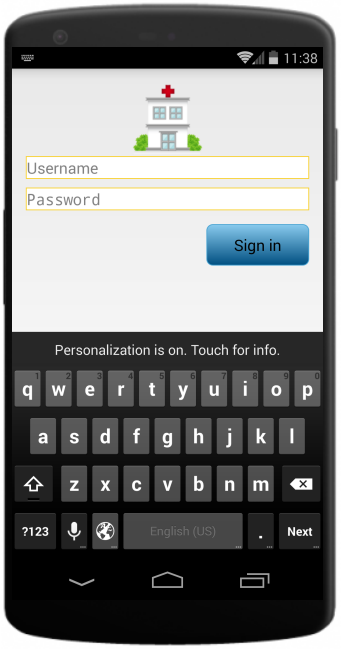


Figure 10: Login Interface

3.6.3 Patient List Interface

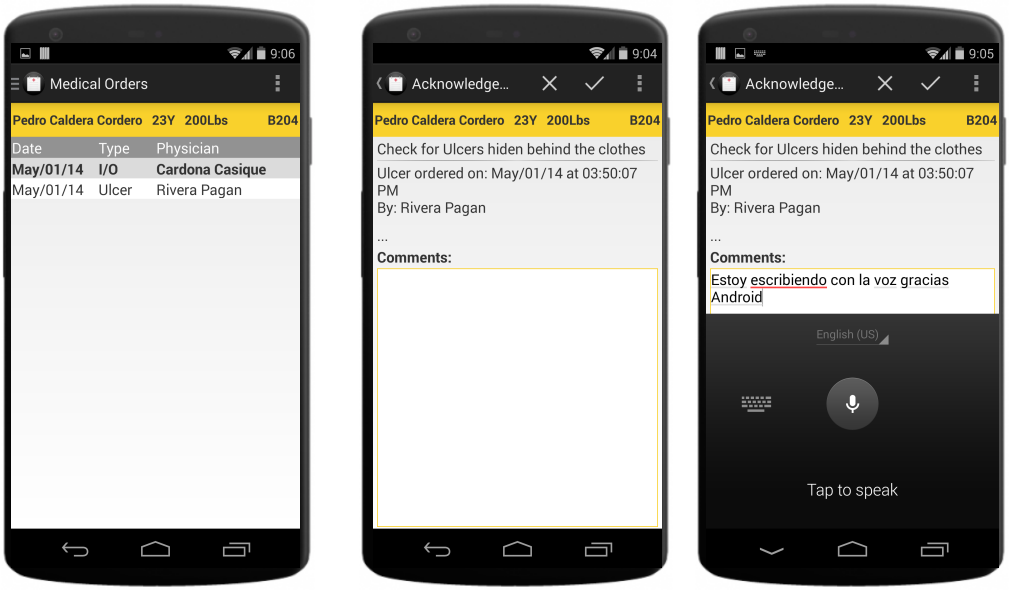
The patient list interface shown in Figure 11.a loads from the database the names of all the patients that are in the hospital. Once all the names are loaded on to the device the system allows the user to filter the patients by clinical area by taping on the Area drop down menu shown in figure 11.b. After a clinical area is selected a list of the patients in that area is displayed (see figure 11.c). When a nurse selects a patient the system takes the patient as an object and passes it to the next activity. This object will remain in the memory of the mobile application as long as the nurse does not select another patient. This prevents a system slow down since no additional network access will be generated to access the record of that patient.

(a) List of all patients **(b)** Listing option **(c)** List of specific hospital area

Figure 11: Patient List Interface

3.6.4 Medical Orders Interface

The selection of a patient from the patient list takes the system to the Medical Orders Interface (Figure 12.a). This interface list all the medical orders generated for a patient with the exception of medication orders. These orders are listed by date and time by default, with the most recent orders at the top. Orders that have not been handled by the nursing staff appear at the top in bold typeface. When a nurse taps on an acknowledged order a new screen appears that shows the details of the order. When a nurse taps on an order that has not been acknowledged a screen with the details of the orders is displayed (Figure 12.b). A nurse can acknowledge the execution of an order by taping the checkmark icon on the top action bar. The order can be left unacknowledged by taping the X icon on the top action bar. The interface provides the option of entering an explanatory comment with the Android keyboard or by using the Android voice recognition feature as shown in Figure 12 c.

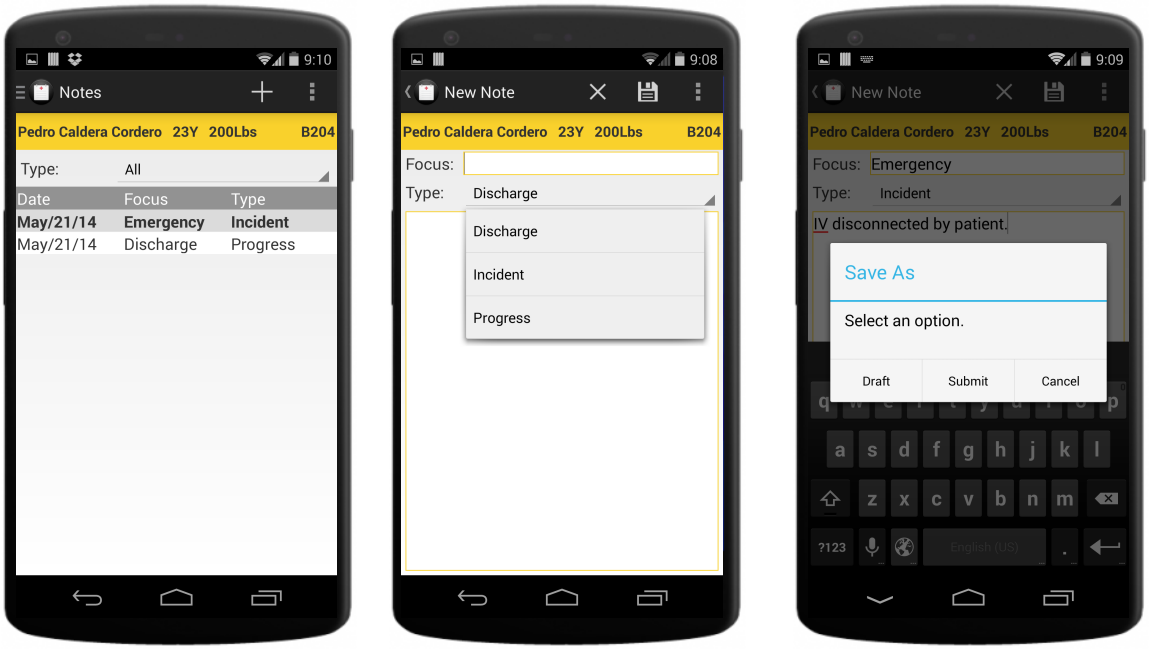


(a) Medical Orders Interface **(b)** Acknowledge Interface **(c)** Voice Recognition

Figure 12: Patient List Interface

3.6.5 Notes Interface

The Notes interface allows the nurses to create new notes and save them as draft. Similar to the medical orders interface, notes are listed by date and time by default, with draft notes at the top in boldface (Figure 13.a). When a note is saved as draft only the nurse that created it can see it or edit it. A note that has been submitted is available to all medical staff and cannot be modified. To create a new note a nurse has to tap the + icon. This will open a screen for entering a new note (Figure 13.b). A note can be saved as final or as draft by selecting the disk icon on the top action bar. That action displays a pop-up window with the two options or cancel (Figure 13.c)

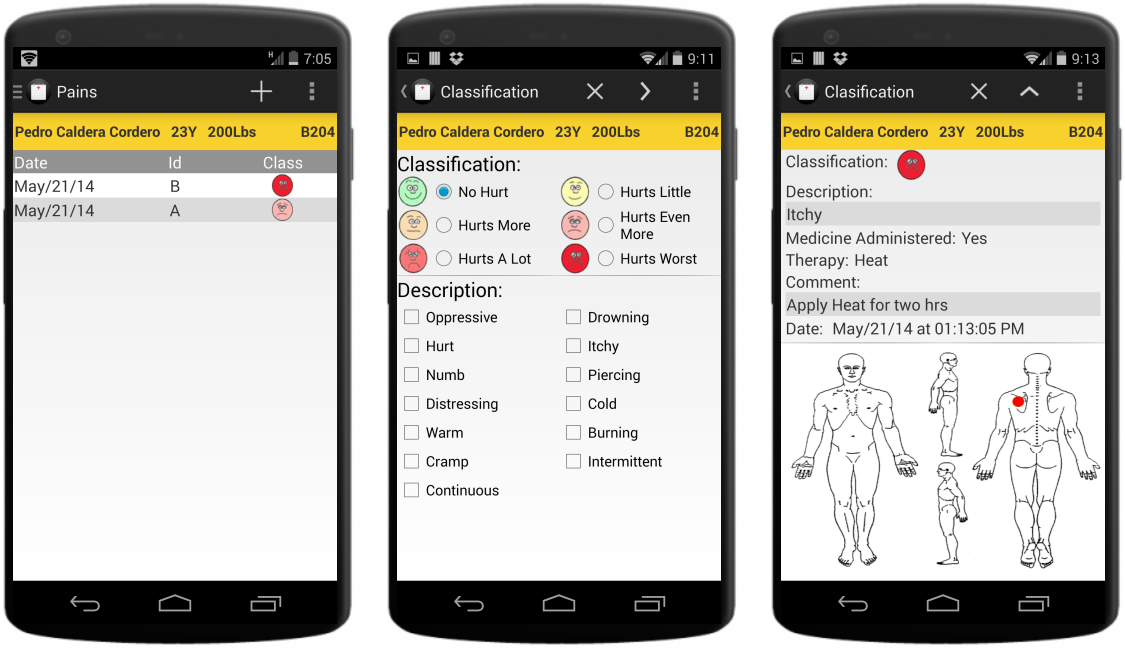


(a) Note List Interface **(b)** New Note **(c)** Draft Dialog

Figure 13: Note Interface

3.6.6 Pain Interface

The pain interface allows the nurses to make an assessment of the patient pain. Since a patient could have different pains during her/his stay in the hospital, each pain is assigned a unique ID. The Pain Interface (Figure 14.a) displays a list of pain assessment. The face icon on the right of the pain entry reflects the level of pain of the most recent assessment. A new pain assessment can be entered by taping the + icon. This action displays the screen shown in figure 14.b. A follow up assessment of an existing pain is accomplished by first taping on the pain entry of the pain list. This action opens a screen that provides the details of the pain assessment (Figure 14.c). From that screen a nurse can enter a follow up assessment by selecting the ^ icon on the top action bar.

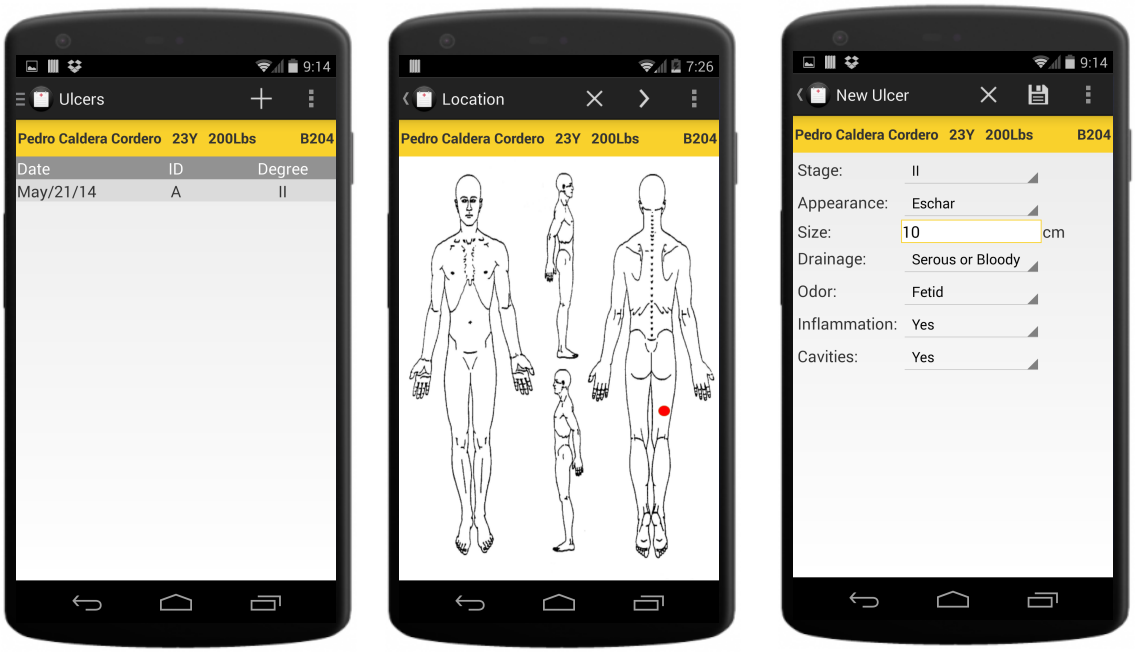


(a) Pain List Interface **(b)** New Pain Classification **(c)** Pain Details

Figure 14: Pain Interface

3.6.7 Ulcers Interface

The ulcer interface functions (Figure 15.a) are very similar to the pains interface. Instead of faces the list shows the degree of the ulcer. A new ulcer can be entered by selecting the + icon. This action opens displays the screen in Figure 15.b that provides a body image to locate the ulcer. Additional details of the ulcer can be specified by selecting the > icon, which opens the screen shown in Figure 15.c. The new ulcer assessment can be saved by taping the disk icon or canceled by selecting the X icon.

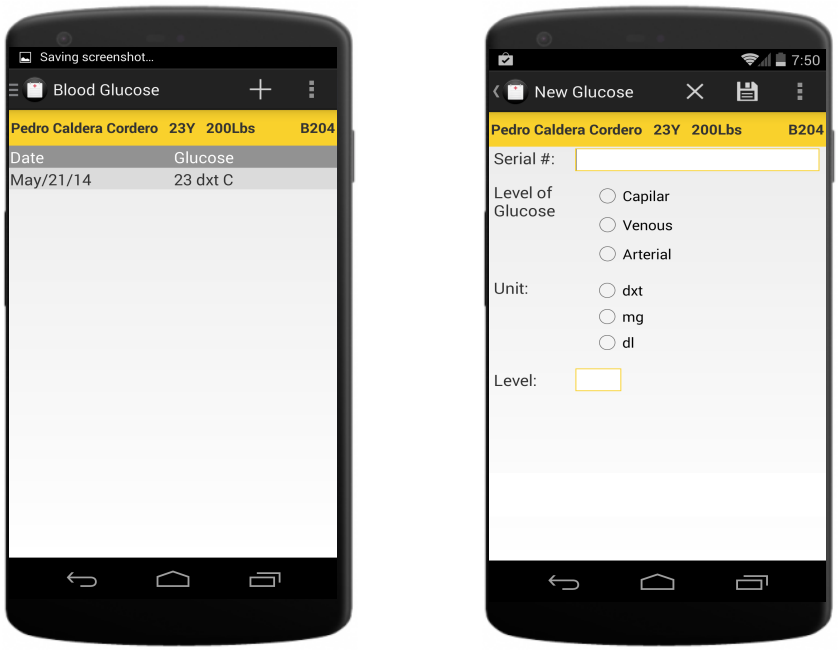


a) Ulcer List Interface **(b)** New Ulcer Location **(c)** New Ulcer Details

Figure 15: Ulcer Interface

3.6.8 Glucose Interface

The Glucose Interface lists the glucose levels recorded for the patient. A new recording can be entered with the screen shown in Figure 16.b, which is activated by selecting the + icon on the Glucose Interface.

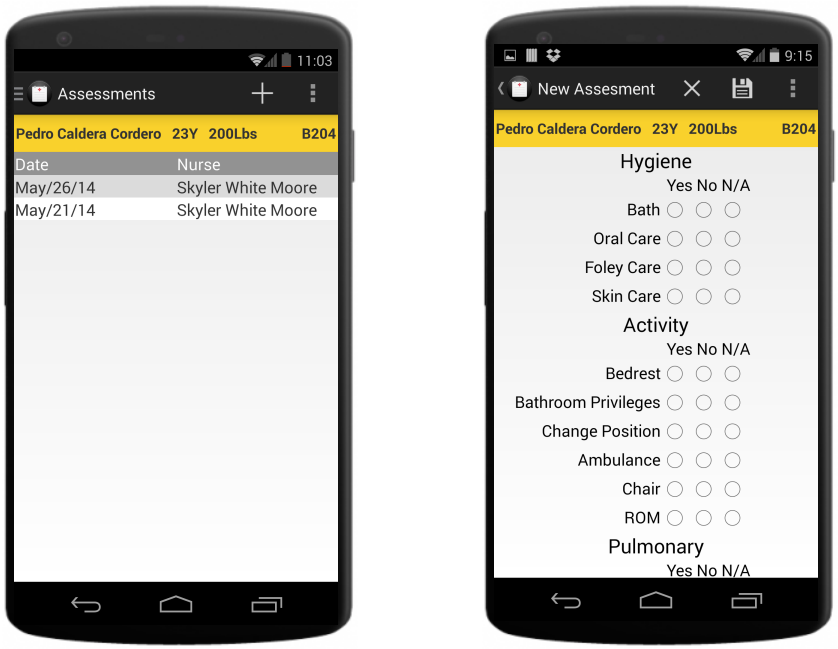


a) Glucose List Interface **b)** New Glucose Level Entry

Figure 16: Glucose Interface

3.6.9 General Assessment Interface

The General Assessment Interface is used to record the general condition of the patient on every nurses' shift. The main screen provides a list of the assessment entries identifying the nurse that made it (Figure 17.a). A new assessment can be entered by taping the + icon. This action displays the screen for entering a new assessment shown on Figure 17.b This interface was developed using an Android vertical scroll view with radio buttons. The interface allows quick scrolling and selection of the items during the assessment of the patient. When a category is not selected it defaults to N/A. In order to be able to implement the user interface in different languages the assessment database table does not hold the entries in text format. Instead, the system passes a true or false to the database.

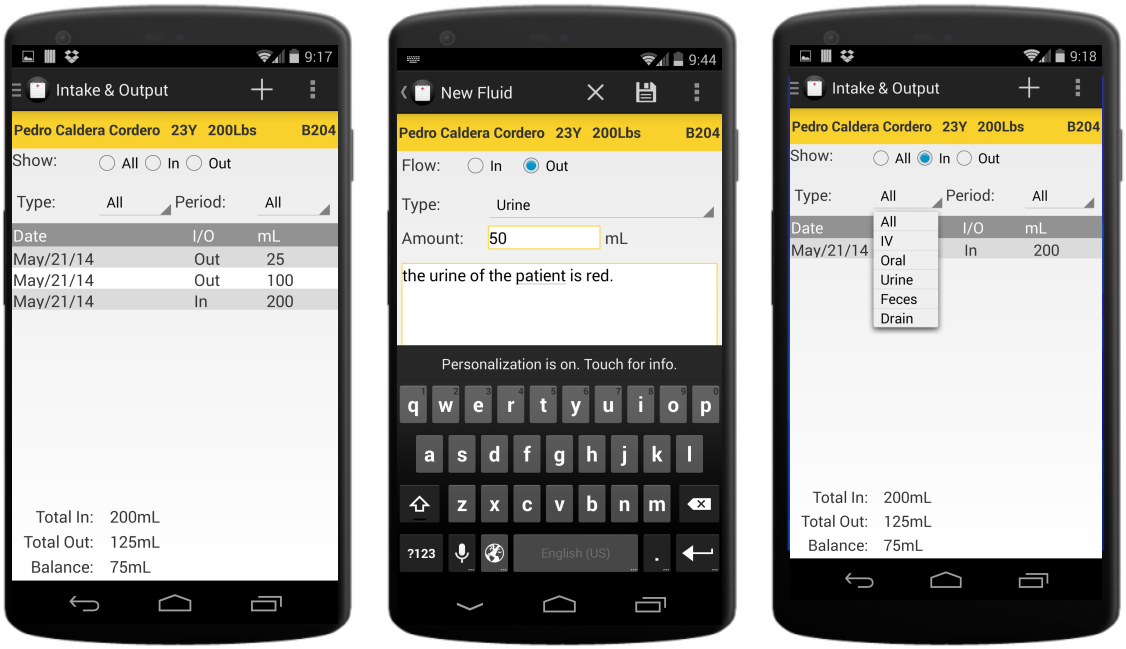


a) Initial Screen **b)** New Assessment Screen

Figure 17: General Assessment Interface

3.6.10 Intake and Output Interface

The main screen of the Intake and Output interface is shown in Figure 18.a. This interface lists the fluids intake/output entries by date and time and provides the total amount of fluids input and output, as well as the fluids balanced during a specified period of time. The interface allows nurses to filter the entries by input or output alone (Figure 18.b). A new intake or output entry can be done with the screen in Figure 18.c.

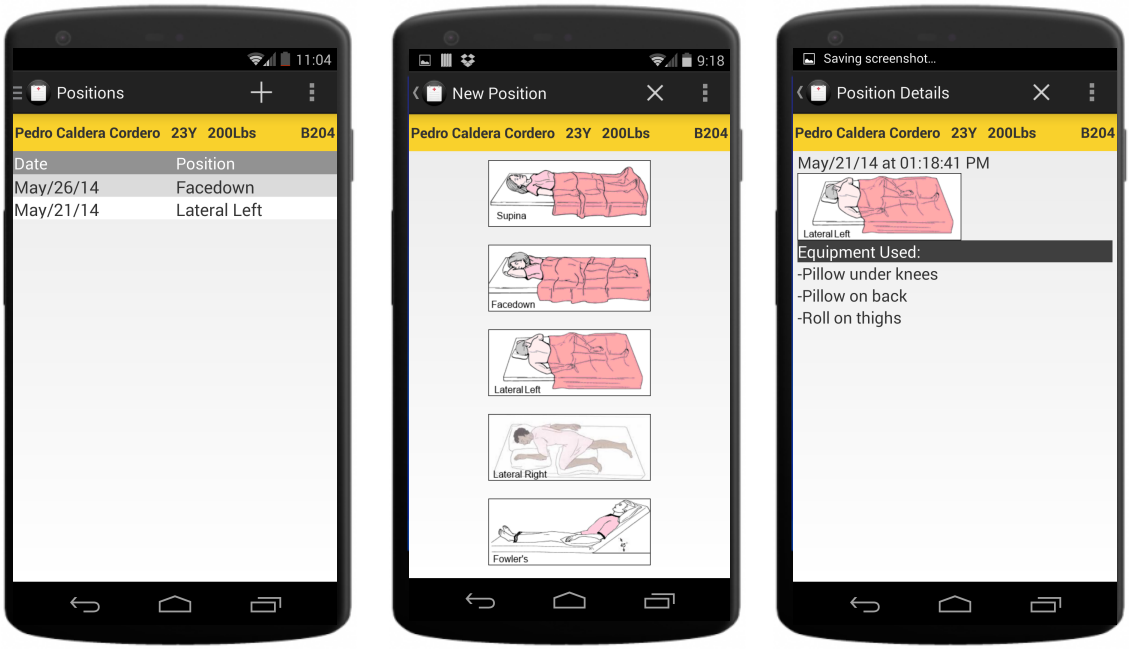


. **a)** I/O List Interface **b)** I/O List Filter **c)** New I/O Entry

Figure 18: Intake and Output Interface

3.6.11 Position Interface

The positions interface allows the nurses to make timely recording of the patient's resting positions. The main screen of the Position Interface is show in Figure 19.a. A new recording can be made with the screen shown in Figure 19.b. Once a position is selected, additional information about the patient's position and equipment used is shown in a secondary screen. Details of a position recording can be viewed by taping on the entry listed in the main screen (see Figure 19.c).

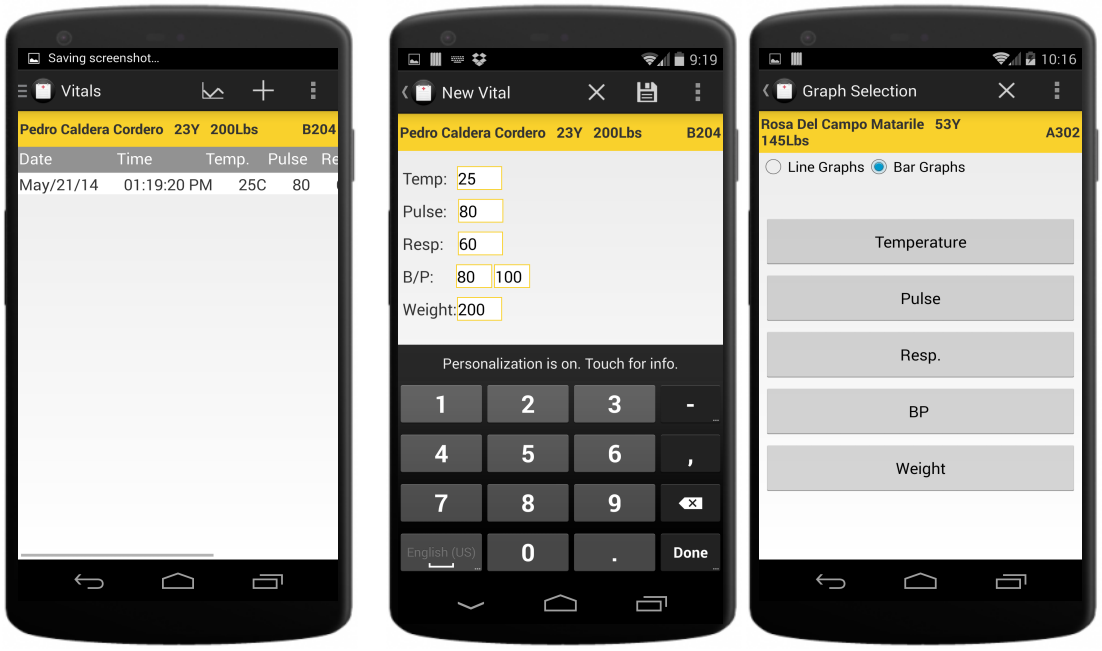


a) Main Screen **b)** Position Selection **c)** Position Details

Figure 19: Positions Interface

3.6.12 Vitals Signs Interface

The main screen of the Vitals Signs Interface displays a list of the main vital signs recorded (Figure 20.a). Due to the space constraint of the landscape view of the interface, the user needs to scroll to the right or place the phone in landscape view in order to see all the vital sign entries. Alternative, the vital signs can be viewed in graphical mode by selecting the graphics icon on the top action bar. This action displays the screen presented in Figure 20.c, which allow the user to select the vital sign to view and the type of graph.

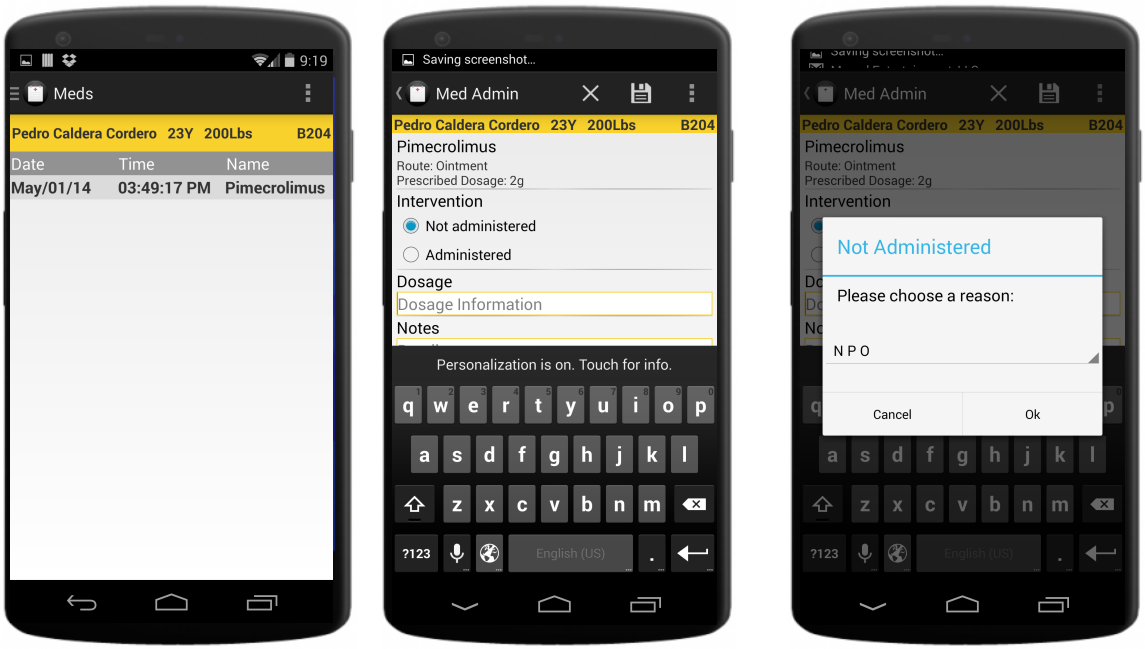


a) Vitals List **b)** New Vital Entry **c)** Graphs Selection

Figure 20: Vitals Interface

3.6.13 Medicine Interface

The Medicine Interface is used for acknowledging medication orders. The main screen of this interface is very similar to the main screen of the Orders Interface. Pending medication orders are listed at the top in boldface (Figure 21.a). The screen shown in Figure 21.b allows nurses to acknowledge the administration of a medicine. If a medicine is not administered the nurse is required to specify a reason on a pop up window (Figure 21.c).



a) Meds List Interface **b)** Meds Administration **c)** Not Administered

Figure 21: Medicine Interface

CHAPTER 4 – HEURISTIC USABILITY EVALUATION

In order to identify potential usability problems a heuristic usability evaluation was performed on the graphical user interfaces of the application. A group of students with knowledge of usability heuristic testing evaluated the application based on Jakob Nielsen's Usability Heuristics [Nielsen93]. In this evaluation method each evaluator runs the application to identify potential usability problems. A description of each problem found was recorded. A summary of the problems found is provided in the tables below. Consistency errors that were part of various interfaces, are mentioned only once in this report. A copy of the full usability evaluation can be found in [Marrero14c]. The problems found were corrected, producing a cleaner, consistent and more intuitive graphical user interface.

Log In Screen				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
Sign In and Register Button are the same.	Prevent users errors and Consistency	3	Separate Sign In and Register into different buttons.	Removed the Register word. The System scope does not need user registration.
No application Name	Feedback, Consistency, Aesthetic/Design	1	Put name of application on the log in screen.	None, Final Name for the Application has not been set.
No way to recover password	Prevent user errors and Minimize user memory load.	4	Add a link that says "Forgot Password?"	None, if a nurse forget the password he/she has to contact a system admin to get a new password. The system is designed that way to enforce security.

Patient List				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
Side bar is not available.	Consistency, Cleared Marked Exits	4	Add a menu with settings.	Added a menu with settings, about and logout.
The format of the screen distorts the information depending on device screen.	Consistency and Aesthetic/design.	4	Make a standard format for all systems no matter system version.	Locked Android app to portrait mode except for Vitals Interface, fixed sizing issues for devices 4.5 inches using stock android and up. The system is to be used with pure unmodified Android.
When pressing the back button the application closes without warning.	Prevent User Errors	3	Add a message asking the user if really wants to get out of the application.	This is normal android behavior on the first activity of an app. But we changed the implementation to reload the list when user pressed back one time and close when pressed back twice.

Medical Orders				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
Titles in Side Menu and Interfaces are Different.	Consistency	1	The titles of the sidebar and the interfaces they open should be the same.	Implemented on current version.
Pending/Acknowledge states are hard to establish with bold letters only.	Aesthetic/ Design	3	Add a column stating the status of the record. Or any other recognizable way.	We left the current implementation, since we understand the bold letters plus the top placement in the list are good indicators.
The header of the patient information is confusing	Aesthetic/ Design	2	Eliminate unneeded information and add it to another interface or separate the information with “|”.	We augmented the spacing between data. The system uses this format since it was proved successful in Rodriguez09 and Rodriguez07 Studies.
The sidebar does not show where the user is currently located in the application.	Feedback	2	The stars on the side of the menu should change color to indicate the interface the user is currently working on.	We eliminated the stars and added a blue hover on top of the interface name where the user is located in.

Order Details				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
When user presses the top back button a message shows up but when the android back button is pressed no message.	Consistency	3	Make a standard way for the app to behave when back buttons are pressed.	Fixed in current version. All back buttons show same message when getting out of a new entry interface.

Pains and Ulcers				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
The ID column shows the ID in Capital Letters on the Ulcer Interface and but not in the Pain Insterface.	Consistency	2	Select a standard format and apply it to both interfaces.	Selected Capital letters and it is implemented on both interfaces in the current version.

Pain->Classification				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
The Red Dot does not appear in the place it was created.	Consistency and Feedback	4	Make the system synchronize the entered data with the details.	Bug Fixed.
The Follow up button does not work in the interface	Design and Feedback	4	Make follow up button work	Bug Fixed.
If the system does not get a value it displays “null”	Speak the user’s language.	3	Change the null for N/A or similar simple language.	Implemented in current version.
The system shows Spanish and English text.	Consistency and Speak the users language.	4	Use a single Language.	Implemented, now only using English.

Assessment ->New Assessment.				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
The headers of IV are inconsistent. One interface shows IV and IV and other interface shows IV1 and IV2.	Consistency	2	Pick a format and apply it to all.	Implemented in current version.

Input Output				
Problem	Heuristic	Severity	Recommendations	Implemented Solution
When filtering the data it only takes in consideration the last filter.	Consistency	4	The problem should consider all filters present in the interface. Redesign the screen it is hard to understand.	Filters Fixed in current version. Spacing changed to make interface components more visible.
When Application is in Landscape the data shows up distorted.	Aesthetic/ Design	4	Make the screen rotation look similar to the portrait mode or lock screen rotation.	Implemented: Locked screen rotation.

CHAPTER 5 – CONCLUSION AND FUTURE WORK

5.1 Recapitulation

The main objective of this project was to integrate the Android interfaces with a Relational SQL database. This document shows how the integration was accomplished and that the interfaces are completely operational with full functionality. The system connects securely via wireless network to a REST API, which access the database. The implementation of the required structures to accomplish the integration were made using modern and easily expandable technologies, which allows the system to be used as a base for future projects, or modified as needed with minor hardships. The challenges presented during this project were mainly the different tools used to build the final system. Knowledge of JAVA, PHP, mariaDB, Linux Server Configurations and encryption tools was used during the course of the project. A great deal of this knowledge was acquired in the process of the system integration.

The integration of the project was made possible with the contribution of the undergraduate research assistants that worked on the development of the user interfaces of the nursing documentation application. To prove that the back-end can be easily used on other platforms a web interface was developed to insert data into the database, which is automatically seen by the Android application. Furthermore, the code for the SSL/HTTPS connection was released using Android publicly on GITHUB [Marrero14b]. In this way other developers can build upon the secure connection classes used in the Android Application. This code can be easily ported and used in Android platforms and JAVA Desktop Applications.

5.2 Future Work

One of the main tasks for the future is to develop the graphical user interfaces for handling physicians tasks. This task has been initiated and it is currently at an early stage of development. Another task for the future is to extend the PHP REST API to handle physicians' tasks. The development of an Administrator Interface to add and remove users, and modify or expand key elements of the database is an important component that needs to be implemented. This interface can be implemented on a Desktop computer extending the REST API and Database.

It would be an interesting task to replicate the [Rodriguez07] and [Rodriguez09] studies with the new Android Application. It could reveal the effect of interacting with a device and user interface paradigm that is very familiar, and consequently, the system should have an easier path to adoption.

While the developed system uses United States government approved encryption technology for security, the U.S. Health and Human Services requires that systems managing patient health information are compliant with Health Level Seven International (HL7) standards. To be able to use this system on production, modifications have to be made to bring this system to HL7 standards. This future task has to be completed to comply with U.S. regulations before putting real patient data in the system.

References

[Apache14] Apache 2.4 SSL How To, http://httpd.apache.org/docs/current/ssl/ssl_howto.html

[Axelrod11] Axelrod, L.,Fitzpatrick, G., Henwood, F., Cassell, J., Smith, H., Nicholson, A. and Rait, G.," Data recording in primary care field studies: Patient records enhancement project" in Proceedings of The 5th International Conference on Pervasive Computing Technologies for Healthcare (Dublin, Ireland, May. 2011)

[BouncyCastle14] The Legion of the Bouncy Castle, http://www.bouncycastle.org/latest_releases.html

[Ben13] Ben Yahmed, M.A., Bounenni, M.A., Chelly, Z., Jlassi, A., "A new mobile health application for an ubiquitous information system" in Proceedings of the 6th Joint IFIP Wireless and Mobile Networking Conference (Dubai, United Arab Emirates, Apr. 2013)

[Ehrler13]Ehrler F, Wipfli R, Teodoro D, Sarrey E, Walesa M, Lovis C, "Challenges in the Implementation of a Mobile Application in Clinical Practice: Case Study in the Context of an Application that Manages the Daily Interventions of Nurses"JMIR Mhealth Uhealth 2013;1(1):e7, URL: http://mhealth.jmir.org/2013/1/e7/

[Fraser10] Fraser Hamish SF, Blaya Joaquin, "Implementing medical information systems in developing countries, what works and what doesn't", in Proceeding of the 2010 American Medical Informatics Association Symposium(Washington, DC, USA. Nov, 2010)

[Ilie13] Ilie, Virginia, Turel, Ofir, Witman, Paul D., "Towards a New Design Paradigm for Complex Electronic Medical Record Systems: Intuitive User Interfaces" in Proceedings of the 2013 46th Hawaii International Conference on System Sciences (Wailea, HI, USA, Jan. 2013)

[Kume12] Kume, N., Hirayama, Y., Ohboshi, N., Okamoto, K., Takemura, T., Araki, K., Yoshihara, H, "The mobile environment of EHR browsing verified on tablet terminal" in Proceedings of the Joint 6th International Conference on Soft Computing and Intelligent Systems and 13th International Symposium on Advanced Intelligent Systems (Kobe, Japan, Nov. 2012)

[Harman12] Harman Laurinda B., Flite Cathy A., Bond Kesa, "Electronic Health Records: Privacy, Confidentiality, and Security", American Medical Association Journal of Ethics September 2012, Volume 14, Number 9: 712-719.

[MariaDB14a] MariaDB 10 Features, https://mariadb.com/kb/en/what-is-mariadb-100/

[Marrero14a] High Resolution Relationship Model of the project MariaDB Database, http://ece.uprm.edu/jmarrero/project/db/DBModel.png

[Marrero14b] Joseph Marrero's Github Repository Containing HTTPS Android Code, https://github.com/jmarrero/sslAndroidConnect

[Marrero14c] Usability Evaluation by Dr. Borges group, http://ece.uprm.edu/jmarrero/project/docs/UsabilityEval.pdf

[Marrero14d] Step-by-Step Instruction of HTTPS Android Configuration and Programming, http://stackoverflow.com/questions/23114049/ssl-mutual-authentication-fail-on-android-client-accepts-servers-certificate-but

[Najera07] Najera Isabel, "Development of a PDA-Based Nursing Documentation Application for Hospitals", in Department of Electrical and Computer Engineering of The University of Puerto Rico Mayaguez Campus,(Mayaguez, Puerto Rico, July 2007)

[Nielsen93] Nielsen, J. "Usability Engineering". Academic Press A Harcourt Science and Technology Company. 1993.

[Rodriguez07] Rodríguez Néstor J., Borges José A., Crespo Gilberto, Pérez Carlos, Martinez Carlos, Colón-Rivera Celia R., Ardín Aixa,"A usability study of nurses' interaction with tablet PC and PDA nursing documentation applications", in Proceedings of the Second IASTED International Conference on Human Computer Interaction, Pages 232-237 (Anaheim, CA, USA 2007)

[Rodriguez09] Rodríguez Néstor J., Borges José A., Crespo Gilberto, Pérez Carlos, Martinez Carlos, Colón-Rivera Celia R., Ardín Aixa, "Nurses can do Better with PDA-Based than with Paper-Based Nursing Documentation Systems", in Proceedings of the 5th International Conference on Universal Access in Human-Computer Interaction. Part III: Applications and Services, Pages 395 - 403 (Springer-Verlag Berlin, Heidelberg 2009)

[Sax09] Sax, C., Lawrence, E., "Point-of-Treatment: Touchable E-nursing User Interface for Medical Emergencies" in Proceedings of the Third International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (Sliema, Malta, Oct. 2009)

[Shiang-Lin11] Shiang-Lin Lin, Chen-Shu Wang, "The interactive interface implementation for the smartphones intelligence NIS" in Proceedings of the 15th North-East Asia Symposium on Nano, Information Technology and Reliability (Macao, China, Oct. 2011)

[Yarow13] Yarow Jay, Business Insider, "Chart of Android Device Activations" http://www.businessinsider.com/chart-of-the-day-android-activations-hit-1-billion-2013-9

[Six11] Six Jeff, 2011,"Application Security for the Android Platform", 1st Edition, O'Reilly Media

APENDIX A – PHP Code Queries for Note Interface

```php
<?php

/*
 * Following code will list all the notes
 */

// array for JSON response
$response = array();


// include db connect class
require_once __DIR__ . '/db_connect.php';

// connecting to db
$db = new DB_CONNECT();
$key = ENKEY;

if (isset($_GET["patient_id"]) && isset($_GET['nurse_id'])) {
    $patient_id = $_GET['patient_id'];
    $nurse_id = $_GET['nurse_id'];
// get all notes from note table
$result = mysql_query("SELECT note_id, pid, note_date, note_draft,
    AES_DECRYPT(note_focus, '$key'),
    AES_DECRYPT(note_type, '$key'),
    AES_DECRYPT(note_message, '$key'),

    AES_DECRYPT(nurse.nurse_fname, '$key'),
    AES_DECRYPT(nurse.nurse_lname, '$key'),
    AES_DECRYPT(nurse.nurse_motherMaidenName, '$key')
    FROM note,nurse

    WHERE note.nurse_id = nurse.nurse_id and note.note_draft = '0' and pid = '$patient_id'


    UNION

    SELECT note_id, pid, note_date, note_draft,
        AES_DECRYPT(note_focus, '$key'),
        AES_DECRYPT(note_type, '$key'),
        AES_DECRYPT(note_message, '$key'),

        AES_DECRYPT(nurse.nurse_fname, '$key'),
        AES_DECRYPT(nurse.nurse_lname, '$key'),
        AES_DECRYPT(nurse.nurse_motherMaidenName, '$key')
        FROM note,nurse

    WHERE note.nurse_id = nurse.nurse_id and note.nurse_id = '$nurse_id' and note.note_draft = '1' and pid = '$patient_id' ");

// check for empty result
if (mysql_num_rows($result) > 0) {
    // looping through all results
    $response["note"] = array();

    while ($row = mysql_fetch_array($result)) {
        // temp user array
        $note = array();
        $note["note_id"] = $row["note_id"];
        $note["patient_id"] = $row["pid"];
        $note["note_datetime"] = $row["note_date"];
        $note["note_draft"] = $row["note_draft"];

        $note["note_focus"] = $row["AES_DECRYPT(note_focus, '$key')"];
        $note["note_type"] = $row["AES_DECRYPT(note_type, '$key')"];
        $note["note_message"] = $row["AES_DECRYPT(note_message, '$key')"];

        $note["nurse_fname"] = $row["AES_DECRYPT(nurse.nurse_fname, '$key')"];
        $note["nurse_lname"] = $row["AES_DECRYPT(nurse.nurse_lname, '$key')"];
        $note["nurse_mname"] = $row["AES_DECRYPT(nurse.nurse_motherMaidenName, '$key')"];


        // push single note into final response array
        array_push($response["note"], $note);
    }
    // success
    $response["success"] = 1;

    // echoing JSON response
    echo json_encode($response);
} else {
    // no notes found
    $response["success"] = 0;
    $response["message"] = "No notes found";

    // echo no users JSON
    echo json_encode($response);
   }
}else {
        // required field is missing
    $response["success"] = 0;
    $response["message"] = "Required field is missing";

    // echoing JSON response
    echo json_encode($response);
}
```

Figure A-1: PHP Code doing SELECT SQL Query to get the Note List.

```
<?php
// array for JSON response
$response = array();

// check for required fields
if (isset($_POST['note_id']) && isset($_POST['nurse_id']) && isset($_POST['note_draft']) && isset($_POST['note_focus']) && isset($_POST['note_type']) &&
isset($_POST['note_message'])) {

    $note_id = $_POST['note_id'];
    $nurse_id = $_POST['nurse_id'];
    $note_draft = $_POST['note_draft'];
    $note_focus = $_POST['note_focus'];
    $note_type = $_POST['note_type'];
    $note_message = $_POST['note_message'];

    // include db connect class
    require_once __DIR__ . '/db_connect.php';

    // connecting to db
    $db = new DB_CONNECT();
    // get encryption key
    $key = ENKEY;

    // mysql updating a row
    $result = mysql_query("UPDATE note
SET nurse_id = '$nurse_id', note_draft = '$note_draft',
note_focus = AES_ENCRYPT('$note_focus', '$key'),
note_type = AES_ENCRYPT('$note_type', '$key'),
note_message = AES_ENCRYPT('$note_message', '$key')
WHERE note_id =  '$note_id'");


    // check if row updated or not
    if ($result) {
        // successfully inserted into database
        $response["success"] = 1;
        $response["message"] = "Note successfully Updated.";

        // echoing JSON response
        echo json_encode($response);
    } else {
        // failed to update row
        $response["success"] = 0;
        $response["message"] = "Oops! An error occurred.";

        // echoing JSON response
        echo json_encode($response);
    }
} else {
    // required field is missing
    $response["success"] = 0;
    $response["message"] = "Required Note ID field is missing";

    // echoing JSON response
    echo json_encode($response);
}
```

Figure A-2: PHP Code doing UPDATE SQL Query to Update Note

```
<?php

// array for JSON response
$response = array();

// check for required fields
if (isset($_POST['pid']) && isset($_POST['nurse_id']) && isset($_POST['note_focus'])&& isset($_POST['note_draft'])  && isset($_POST['note_type']) &&
isset($_POST['note_message'])) {

    $pid = $_POST['pid'];
    $nurse_id = $_POST['nurse_id'];
    $note_focus = $_POST['note_focus'];
    $note_type = $_POST['note_type'];
    $note_message = $_POST['note_message'];
    $note_draft = $_POST['note_draft'];

    // include db connect class
    require_once __DIR__ . '/db_connect.php';

    // connecting to db
    $db = new DB_CONNECT();
    // get encryption key
    $key = ENKEY;

    // mysql updating a row
    $result = mysql_query("INSERT INTO note (nurse_id, pid, note_draft, note_focus, note_type, note_message)
VALUES ('$nurse_id', '$pid','$note_draft',
AES_ENCRYPT('$note_focus', '$key'),
AES_ENCRYPT('$note_type', '$key'),
AES_ENCRYPT('$note_message', '$key')) ");


    // check if row updated or not
    if ($result) {
        // successfully inserted into database
        $response["success"] = 1;
        $response["message"] = "Note successfully Created.";

        // echoing JSON response
        echo json_encode($response);
    } else {
        // failed to update row
        $response["success"] = 0;
        $response["message"] = "Oops! An error occurred.";

        // echoing JSON response
        echo json_encode($response);
    }
} else {
    // required field is missing
    $response["success"] = 0;
    $response["message"] = "Required fields are missing";

    // echoing JSON response
    echo json_encode($response);
}
```

Figure A-3: Code doing INSERT SQL Query to insert a new Note

APENDIX B – Encrypted Database Tables

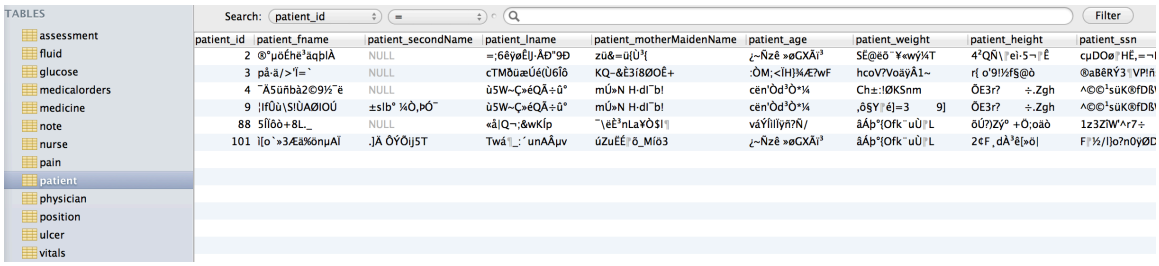


Figure B-1: Nurse table Encrypted Data.

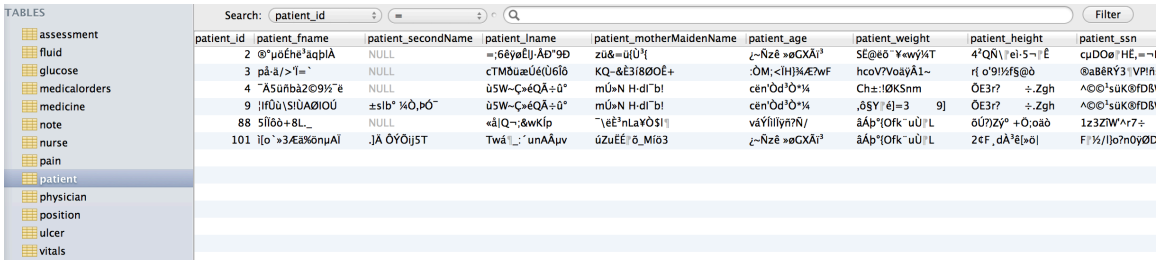


Figure B-2: Patient table Encrypted Data.

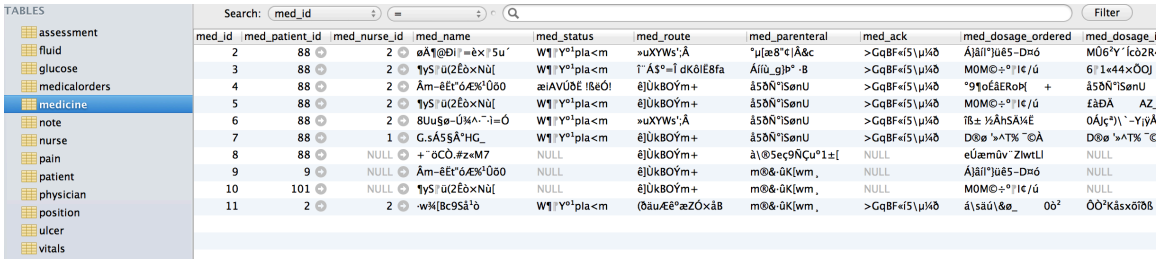


Figure B-3: Medicine table Encrypted Data.