RICH INTERNET APPLICATION FOR TRACKING THE FILLING AND DISPENSATION OF MEDICATION TRAYS IN HEALTHCARE

Nauman Ahmed Khan

Master of Science Thesis
Supervisor: Prof. Ivan Porres
Advisor: Natalia Díaz Rodríguez
Department of Information Technologies
Åbo Akademi University
April 2014

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Ivan Porres for his patience, motivation, and immense knowledge. I owe a huge debt of thanks to my advisor Natalia Díaz Rodríguez for her invaluable and continuous guidance. I am very grateful to her guidance throughout the period of my research. I would like to express my deepest gratitude to my parents for always been a source of inspiration for me.

Turku, April 2014
Nauman Ahmed Khan

ABSTRACT

Traditionally in hospitals the task of dispensing medicine to the patients can be strenuous for the hospital staff, as many variables have to be considered like searching for a particular medicine in medicine shelves, searching for an alternative medicine, verifying the amount and dosage of the medicine, keeping in mind the timings of the medicine and other details. In addition to that, the dispensation process is the combination of many manual and computerized steps which do not function together. In this work, we have tried to automate the whole process by combining information from different sources and providing it on a single centralized rich internet application. The application helps the nurses in performing the filling and dispensing of the medicine tray efficiently. The application allows the nurses to keep history of the medicine dispensed to the patients. The application makes the whole process smoother and provides more patient safety. A survey was conducted to validate the application usability.

Keywords: Health, Hospital, Medicine, Dispensation, Rich internet application, Automation, Patient Safety,Survey, Usability.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Abbreviations

AJAX	Asynchronous Javascript and XML
API	Application Protocol Interface
DB	Database
GUI	Graphical User Interface
GWT	Google Web Toolkit
HTML	HyperText Markup Language
IDE	Integrated Development Environment
JVM	Java Virtual Machine
MVC	Model-View-Controller
RIA	Rich Internet Application
SQL	Structured Query Language
SSN	Social Security Number
MySQL	An SQL database technology. A subsidiary of Oracle
UI	User Interface
UML	Unified Modeling Language
URL	Uniform Resource Locator

1 INTRODUCTION

The trend of rich internet application is to provide more functionality and minimize the gap between the native web applications and their desktop counterparts [1]. There has been a major paradigm shift in the healthcare sector in recent years [2]. The performance and reliability of internet and mobile applications are well utilized by the medical health care systems [3].

From 700 to 1700 lives are claimed every year due to the hospital medical error in Finland [4]. Medicine dispensing error can be seen as a major cause among these errors. Medicine used in inappropriate way can be harmful for the patient. Such errors can occur because of minor human mistake on dispensing medicine. During last five years, over 340,000 incidents of medical errors and patient safety incidents have been registered[4]. Nurses have to perform many manual and computerized tasks which could lead to potential mix up and hence resulting in an error. In order to enhance the usability of the dispensing system, different tasks involved in dispensation process can be combined in a web application. Such applications are useful for the nurses, where they can access the patient data and monitor the activities accordingly.

Nurses can benefit from such applications running on portable devices such as smart phones and tablet pcs. This thesis addressed the mentioned issues by proposing a smart dispensing application, which can reduce the work load for nursing staff and minimize the chances of mistakes. Data synchronization can become a bottleneck in such situations. To address this issue the data will be saved and accessed from the server. Changes done by one nurse will be immediately visible to any other nurse. This application can make the decision making process convenient for the nursing staff in two phases. In first phase, it assists the nurse when filling the medicines into dispensation trays for multiple patients. In second phase, it helps the process of medicines dispensation to the patients. Generally, the application keeps track record of each task performed by any nurse for a specific patient.

The rest of the thesis is organized as follows. Chapter 2 presents the existing applications and devices for the health care and dispensing process in finnish hospitals. In Chapter 3, the rich internet applications and Vaadin framework are discussed. The smart dosing application architecture is described in Chapter 4. The implementation of the smart dosing application is presented in Chapter 5. Chapter 6 presents the survey, which was conducted to validate the application usability. Chapter 7 provides the possible future works and concludes the thesis.

2 Smart devices and applications

As the population is getting progressively older, the need for higher quality and better efficiency in health and medicine, both at home and in hospital, are becoming more important. It is obviously a requirement for the patient in order to increase comfort, but it is also valuable for the society to increase the efficiency and provide cheaper health delivery. A quality of life is now expected by the patients, even when suffering from various chronic diseases. Patients generally want to be treated at home and with as little pain and discomfort as possible. Medical devices provide such features.

2.1 Smart applications and devices for healthcare

A variety of upcoming applications with different complexity levels exhibit distinct system performance requirements. All these advancements and dynamically changing market requirements have changed the application design approach as well. In addition to traditional design constraints, modern application design considers fault tolerance, flexibility, design effort, usability, design time, and application monitoring techniques.

All these performance requirements with given considerations can be fulfilled by modern smart devices like smart phones and tablets. In a typical health care system, data is collected from different body sensors and finally processed by a smart device to generate useful information, which is mostly understandable by a common person. In modern healthcare applications like a sports tracker, sensors are not used at all but user provides some input parameters manually and then positioning and speed calculating applications like GPS are used to compute the amount of burnt calories.

In recent years, a lot of health care applications have been proposed and developed. Due to the unique requirements from each patient, application customization is the basic requirement. This customization increases the design complexity, prolongs the time-to-market and makes the user interface complicated. Thus each application targets a specific market segment. Some of the available health care applications are discussed here.

In [5] authors have presented a health care system where a standalone computing system collects the real time data from multiple body sensors. In case of any alarming situation the medical advisor receives a notification. Then the medical advisor prescribes the medicine according to the current situation.

Wedjat application [6] uses the medicine properties and prescription to give a reminder to the patient about next intake time of the medicine. For example, if the patient forgets or cannot take the medicine due to some other reason, Wedjat recalculates the next medicine intake time by considering medicine properties and several other parameters such as prescription and dosage. The application interface is made interactive to facilitate the patients.

Healthcare at Home(H@H) is a healthcare system proposed by [7]. The authors have presented three subsystems: Automatic Medicine Dispenser (AMD), Fall Detection and Communication Device (FDCD), and Entry Hall Detection System (EHDS). The AMD subsystem takes care of dosage and intake time. The FDCD subsystem collects the data from multiple sensors and informs the emergency staff in case of fallen accidents by using the available communication channel. The EHDS subsystem informs the registered caretaker about the pending issues when they enter the premises.

The e-pill system [8] is a device which alarms and dispenses the medicine in a cup which is included in it. Medicine and instruction about dispensing are fed to the system by the medical staff. The device works well for patients with different disabilities such as deaf or blind. With the help of this device the patient becomes less reliant on others, avoids medication errors and can improve his quality of life.

2.2 Dispensing process in hospitals

Nowadays, dispensing the medicine to the patient in hospitals is carried out manually [9]. This task involves a set of smaller subtasks. The first subtask is to print the information related to the patient and the medicine which has to be dispensed to that specific patient. Each print is then cut in the form that the information of each patient is separated from each other. The next subtask is to sterilize the work area and hands. The next step is to organize different medicine cups to be filled [9]. Each color of the cup represents a different time for dispensing during the day, such as yellow for 8hrs, red for 14hrs, green for 16hrs and blue for 20hrs [9]. The next job is to fill the medicine cups with medicines. In case they do not find the required medicine in the medicine cabinet, nurses have to look up the medicine in the Pharmaca Fennica [10]. They have to put the medicine cups and the patient record which they have arranged earlier on the tray. These trays are then carried to the ward where the medicine is dispensed to the patients.

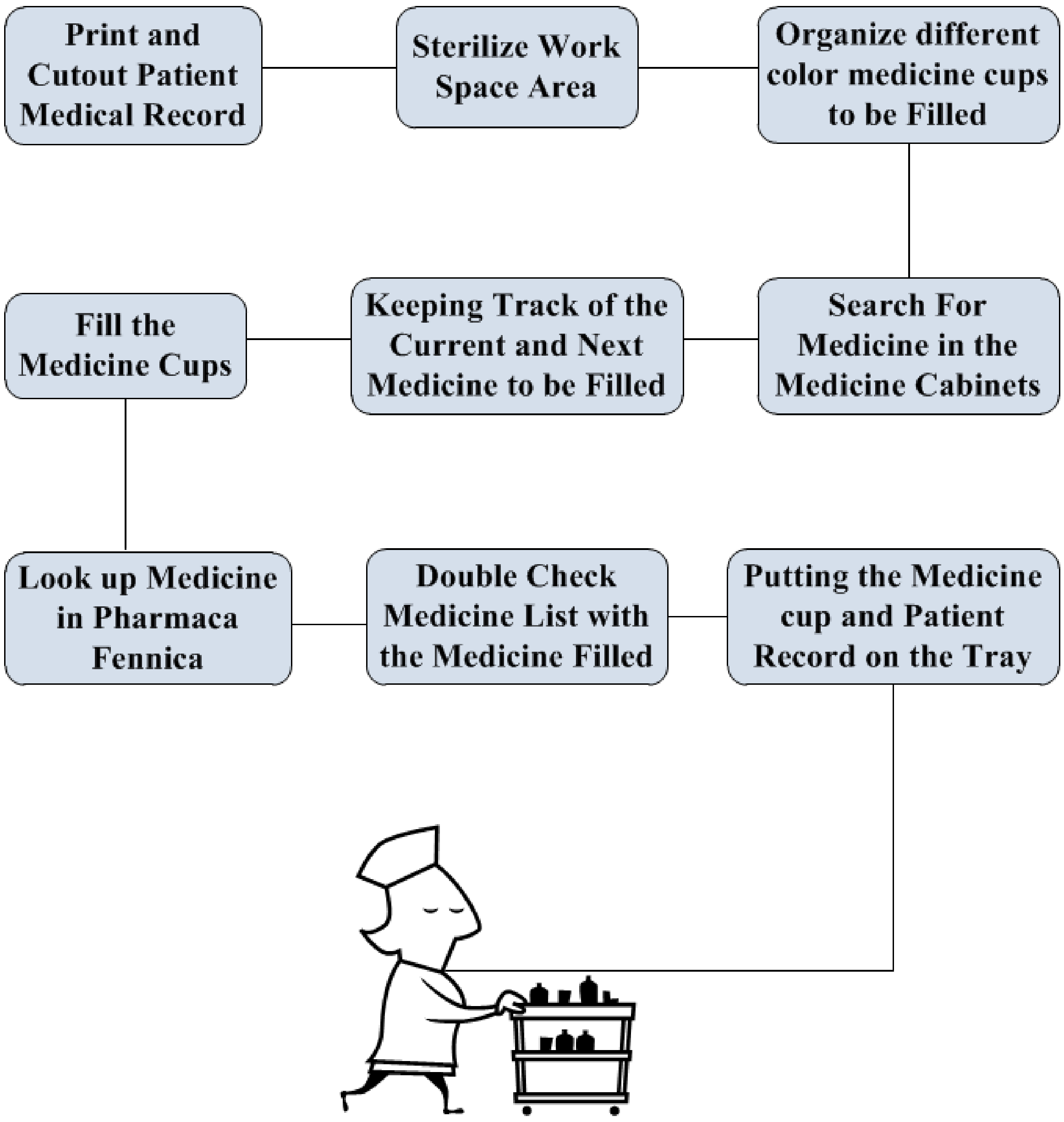

Print and Cutout Patient Medical Record
Sterilize Work Space Area
Organize different color medicine cups to be Filled
Search For Medicine in the Medicine Cabinets
Keeping Track of the Current and Next Medicine to be Filled
Fill the Medicine Cups
Look up Medicine in Pharmaca Fennica
Double Check Medicine List with the Medicine Filled
Putting the Medicine cup and Patient Record on the Tray

Figure 2.1: Current Dispensing Process in Hospital.

3 Rich Internet Applications

3.1 Overview

The traditional webpage consisted of static text and images while data presentation was document oriented with limited user interatcion. Modern web pages have dynamic components which can be a set of small widgets or a complete web application. Traditional web applications are lacking compatibility with modern requirements. Therefore, rich internet applications are gaining popularity due to their powerful interaction capability. The term “Rich Internet Application” was first introduced by Macromedia in March 2002 [11].

Rich Internet Applications are built on the platform of the traditional web applications which have the benefit of added technologies like AJAX. One of the advantages of AJAX is that it carries some of the processing on to the client side which can improve the scalability of the application. In traditional web applications, there are multiple pages which are reloaded according to the request/response procedure of the internet protocol, however the rich internet applications comprises of one web page.

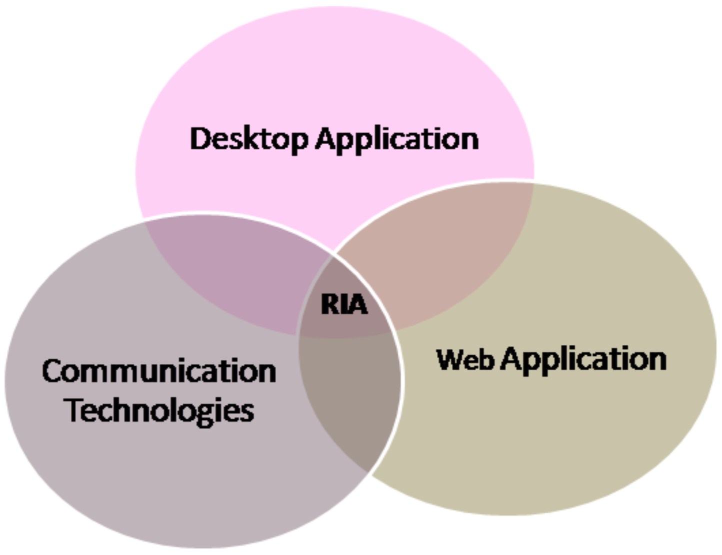


Figure 3.1: Rich Internet Application Architecture

Modern web technologies have given the boost to the developmet of rich internet application. Echo3 [12], JavaFX [13] and Vaadin are the few examples of these frameworks. Figure 3.2 shows an example of rich internet application called Miniusa. It helps the users to build their own mini car according to their choice.

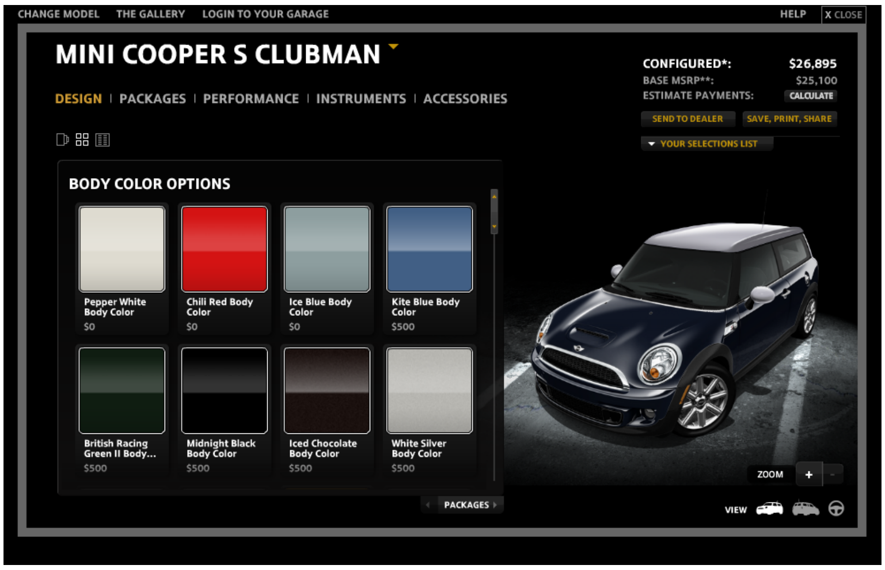

CHANGE MODEL
THE GALLERY
LOGIN TO YOUR GARAGE
HELP
X CLOSE
MINI COOPER S CLUBMAN
DESIGN | PACKAGES | PERFORMANCE | INSTRUMENTS | ACCESSORIES
CONFIGURED*: $26,895
BASE MSRP**: $25,100
ESTIMATE PAYMENTS: CALCULATE
SEND TO DEALER
SAVE, PRINT, SHARE
YOUR SELECTIONS LIST
BODY COLOR OPTIONS
Pepper White Body Color $0
Chili Red Body Color $0
Ice Blue Body Color $0
Kite Blue Body Color $500
British Racing Green II Body... $500
Midnight Black Body Color $500
Iced Chocolate Body Color $500
White Silver Body Color $500
PACKAGES
ZOOM
VIEW

Figure 3.2: Example of a Rich Internet Application

3.2 Vaadin

Vaadin is an open source Java based web application framework for server side and client side development. Vaadin allows to develop the client-server based application in the same way as a traditional desktop application. The development on server side allows developing application logic without any considerations of the client side user interface. Vaadin considers the client side interface as thin client and for its implementation uses HTML, JavaScript and CSS. A web browser can be used as a client side interface. Since modern web browser usage is common these days so this eliminates the extra need for the distribution of the client side code.

The communications between the server and client are handled by the use of AJAX technology. The generation of user interface on client side is handled by GWT [14]. GWT (Google web Toolkit) takes in the interface definition written in Java and generates an equivalent JavaScript code for the web browser. It also provides code optimization so the interface can run on the browser as fast as possible. This enables development to be purely done in Java on both, client and server side [15]. The developer does not need to learn any other languages. Server client communication features are hidden from the user. Vaadin provides an assortment of predeveloped widgets and add-ons.

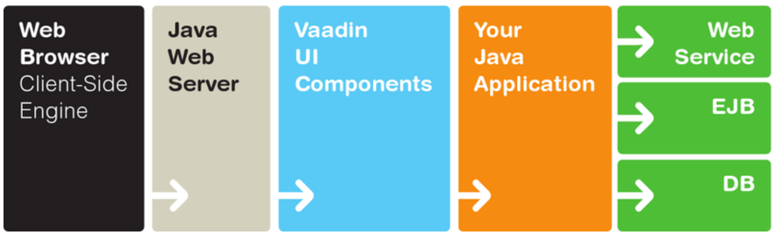


Figure 3.3: Vaadin Architecture [16]

3.3 Vaadin Data Model and Interface

This section describes one of the core concepts in Vaadin framework. In Vaadin, data sources can be bounded with components. The framework provides a set of container interfaces which can be used to represent a data model [17]. The container interface connects the database with the UI components. Vaadin data model is similar to the structure of a traditional database where a property represents a cell, an item represents a row and a container represents a table. The Container interface also provides methods for data manipulation in the database.

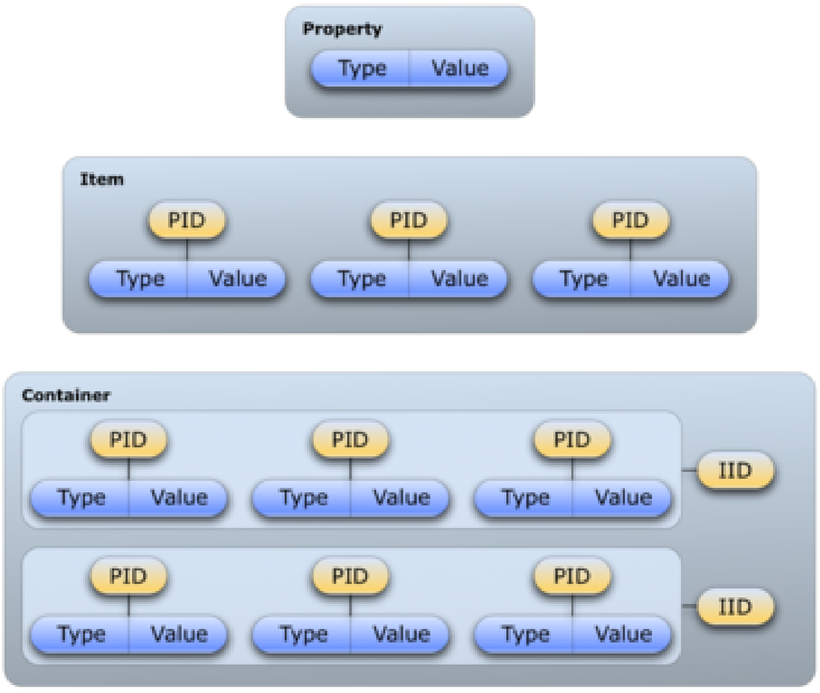


Figure 3.4: Vaadin Data Model [18]

3.4 Vaadin SQL Container

SQLContainer is an implementation of the container interface which is made specifically for the SQL databases. *SQLContainer* class provides two ways for accessing the database. *TableQuery* uses readymade queries for accessing the database, whereas *FreeFormQuery* uses the custom made queries to access the database.

SQLContainer has a simple architecture. The interface Property is implemented as *ColumnProperty* class where as interface Item is implemented by *RowItem class*. *RowID* and *TemporaryRowID* are used for *ItemID*s. In *SQLContainer*, there are two implementations of *JDBCConnectionPool* interface implemented, *SimpleJDBCConnectionPool* and *J2EEConnectionPool*. *SimpleJDBCConnectionPool* helps to connect to JDBC connection in a very simple manner whereas *J2EEConnectionPool* connects to J2EE data sources.

SQLContainer needs to enable its access to and from database is defined in the *QueryDelegate* interface which is enclosed in the query package. *TableQuery* and *FreeFormQuery* are both implementation of the *QueryDelegate* interface. *TableQuery* provides read-write access to the database table. *FreeFormQuery* helps in sorting, filtering and tailored queries. There are two classes, *Filter* and *OrderBy* in the query package which are provided for a more user friendly filtering and sorting.

3.5 Vaadin Touchkit

Vaadin Touchkit is a mobile optimized framework on top of Vaadin which enables to develop mobile application for mobile devices. It provides touch optimized components, geo location, home screen launching, splash screen and many more components which are more suited for mobile devices.

In mobile devices, there are certain aspects which make them different from regular computers. These aspects can be rotation of the display screen which changes the dimensions of the mobile screen, touch screen, and virtual keyboard.

3.6 Model View Controller Architectural Pattern

Rich internet application frameworks commonly utilize the concept of Model View Controller pattern. Web applications in all languages have adopted MVC architecture. In 1979, Trygve Reenskaug, a Norwegian computer scientist defined MVC [19]. This pattern is divided into three parts, namely models, views and controllers. Vaadin framework also uses MVC pattern for the web application development.

3.6.1 Models

Models represents any kind of stored information related to the application. This information is represented with the help of an object or a set of structured objects depending on the application [20]. They are used to represent the persistent data of the application and can even replace controllers for applications with no or less business logic.

3.6.2 Views

Views provide an interface to the user by using models and controllers in tandem to present the user with all possible option with which it can interact with the system. All the data required by the View is predefined in the model in the form of queries and functions.

3.6.3 Controllers

Controllers provide logic in the system. Controller provides relevant functions and processes to manipulate data. They can modify the Model state and View associated to that Model.

4 Smart Dosing Application

4.1 Overview

Smart Dosing Application is a rich internet application which has the characteristics of a native desktop application. The application can be accessed by the user through the use of internet. The targeted user for this application are the nurses working in the hospital ward, often, with no or little technical knowledge of computer usage. The application can be reached by using a thin client such as a web browser on a tablet or a desktop computer. The application provides a user friendly interface while hiding the complexity of the back end from the user. Let us consider Figure 4.1 as an example. Here, the application is running on a remote server and at the moment three nurses are connected to it from three different locations through the internet. This may be different wards in a same hospital or different hospitals altogether.

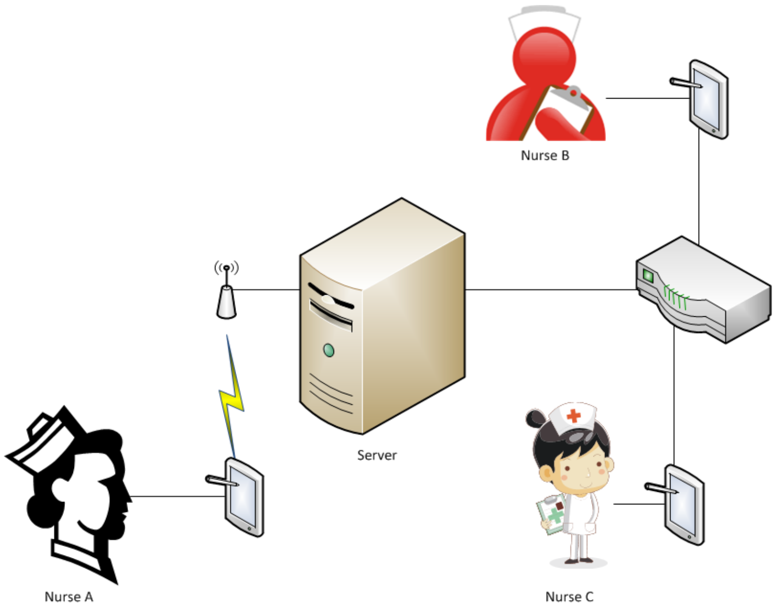


Figure 4.1: Smart Dispensing Application

4.1.1 Application Specification

Following are the platform specifications for the application

- Java 5
- Apache Tomcat v6.0
- Java Servlet 2.4
- MySQL 5.5
- Vaadin 6.8.0
- Vaadin TouchKit 2.0

4.2 Functional and Non Functional Requirements

Functional requirements describe the behavior of the software system whereas non functional requirements are quality attributes of the software system. The functional and non functional requirements of the application are listed below:

4.2.1 Functional Requirements

- The system should help the nurses in filling the medicine tray by displaying the patient information, the current state of the medicine tray they are filling, the picture of the medicine they have to fill and picture of the glasses they have to fill the medicine in.
- When the medicine needs to be dispensed, the system should provide the nurses with the patient information for whom the medicine has to be dispensed, picture of the tray which specifies where the medicine is placed for each patient, physical location of the patient in the ward and picture of the glasses which have to be dispensed to the patient.
- The system should store every action performed by the nurses. It should store the time and date at which the medicine is filled and dispensed to the patient and the name of the nurse who performed these tasks.
- The system should search and recommend an alternative medicine in case the medicine prescribed in not available.
- The system should allow the nurses to report a medicine if a medicine is not available.

- The system should be only accessible to the nurses as it contains the information of the patients.
- Nurses should be allowed to fill the tray according to patient first or medicine first.
- When no medicine is filled the dispensation should be restricted.
- The system should provide the nurses with the option to skip the medicine during filling and dispensing of the medicine and it should ask the nurses for the reason to skip it.
- Nurses should have the option to access all the information about the patients, medicines and the contact information of other nurses.

4.2.2 Non Functional Requirements

- The system should provide information accurately, the information it provides needs to be correct as any misinformation can lead to a dispensation error that can be harmful for the patient.
- The system should be made portable to be used on any device.
- The response time of the system should be kept minimum.
- The usability of the system should be kept simple to provide ease of use to the nurses.

4.3 Application Design

4.3.1 Application Blueprint

Use case diagrams are used to describe the higher level view of an application. A use case diagram defines the actors and lists the available interaction with the systems. It consist of actors, system, functionalities of the system and the relationship between the actors and the functionalities.

Actors

Actors are agents that are external to the system. In smart dosing application the external agents are the nurses who are using the application.

System

A system is an entity which provides functionalities or use cases to the actors to interact with them. In our work, the system is the smart dosing application which is providing the actor (nurse) with the means of interaction.

Functionalities

Functionalities are the use cases which help the system to accomplish the desired output of the task. In smart dosing application there are several use cases helping the system.

Use Cases

Use cases for the application can be seen in Figure 4.2

Table 4.1: Use case 1

Use case ID:	UC 1
Use case name:	Fill the medicine tray
Primary actor:	Nurse
Description:	It provides the application with the function to fill the medicine tray.
Pre conditions:	Nurse should be logged in to the system.
Standard flow:	1. Nurse clicks on the fill tray button. 2. Applicaton displays the information of the first patient. 3. Application shows the information with a tray picture and medicine picture. 4. Nurse fills the medicine tray according to the information provided.
Post conditions:	The filled tray should be dispensed to the patients.

Table 4.2: Use case 2

Use case ID:	UC 2
Use case name:	Dispense Medicine
Primary actor:	Nurse
Description:	It provides the application with the function to dispense the medicine to the patients in the ward whose medicines are filled.
Pre conditions:	UC1 should be performed.
Standard flow:	1. Nurse clicks on the dispense button in the application. 2. The filled tray with the patient information is displayed to the nurse. 3. The information includes the physical location of the patient in the ward. 4. Nurse dispense the medicine to the respective patient.

Table 4.3: Use case 3

Use case ID:	UC 3
Use case name:	View History
Primary actor:	Nurse
Description:	This use case provides the actor to view the medicine history of a patient.
Pre conditions:	UC 1 or UC 2 must be performed.
Standard flow:	1. Nurses select the patient from the patient list. 2. After selecting the patient, nurse clicks on the view history button. 3. The whole history of the patient selected is shown to the patient.
Post conditions:	After looking the history, the history window must be closed.

Table 4.4: Use case 4

Use case ID:	UC 4
Use case name:	Medicine tray view
Primary actor:	UC 1 or UC 2
Description:	This use case provides a graphical view of the current status of a particular medicine tray.
Standard flow:	1. When the filling or dispensing of the tray is being done, the application asks for the current tray view. 2. The graphical view of the tray is then displayed.

Table 4.5: Use case 5

Use case ID:	UC 5
Use case name:	View work shifts
Primary actor:	Nurse
Description:	It informs the nurse about the schedule of their working days for the upcoming week.
Standard flow:	1. First the nurse tab should be clicked. 2. Select the nurse to see the work schedule. 3. Work schedule of the selected nurse is shown to the user.

Table 4.6: Use case 6

Use case ID:	UC 6
Use case name:	View patient treatment
Primary actor:	Nurse
Description:	It informs the nurse about which medicine is prescribed to a particular patient.
Pre conditions:	Patient should be prescribed some medicine from the doctor. Nurses should be in the patient tab or in the fill tray interface to view the treatment of the patient.
Standard flow:	1. Nurses select the patient from the patient list. 2. After selecting the patient ,nurse clicks on the view treatment button. 3. The whole treatment of the patient selected is shown to the patient.

Table 4.7: Use case 7

Use case ID:	UC 7
Use case name:	Search equivalent medicine
Primary actor:	Nurse
Description:	When a medicine gets out of stock, this feature provides possible equivalent medicine with same active ingredients.
Pre conditions:	The medicine prescribed should not be available in the inventory.
Standard flow:	1. Nurse finds out that the medicine is not in the stock which was prescribed to the patient. 2. Nurse clicks the search equivalence button to find out the equivalent medicine. 3. Application finds the equivalent medicine by matching the active ingredients.

Table 4.8: Use case 8

Use case ID:	UC 8
Use case name:	Add List of out of stock items
Description:	When a medicine is out of stock in the hospital, it can be added to a list of medicines which can be ordered from the pharmacy.
Pre conditions:	That medicine should not be available in the inventory.
Standard flow:	1. Nurse tries to find the medicine in the medicine cabinet. 2. Nurse find out that the medicine is not in the stock. 3. Nurse orders the medicine with the quantity to the system.

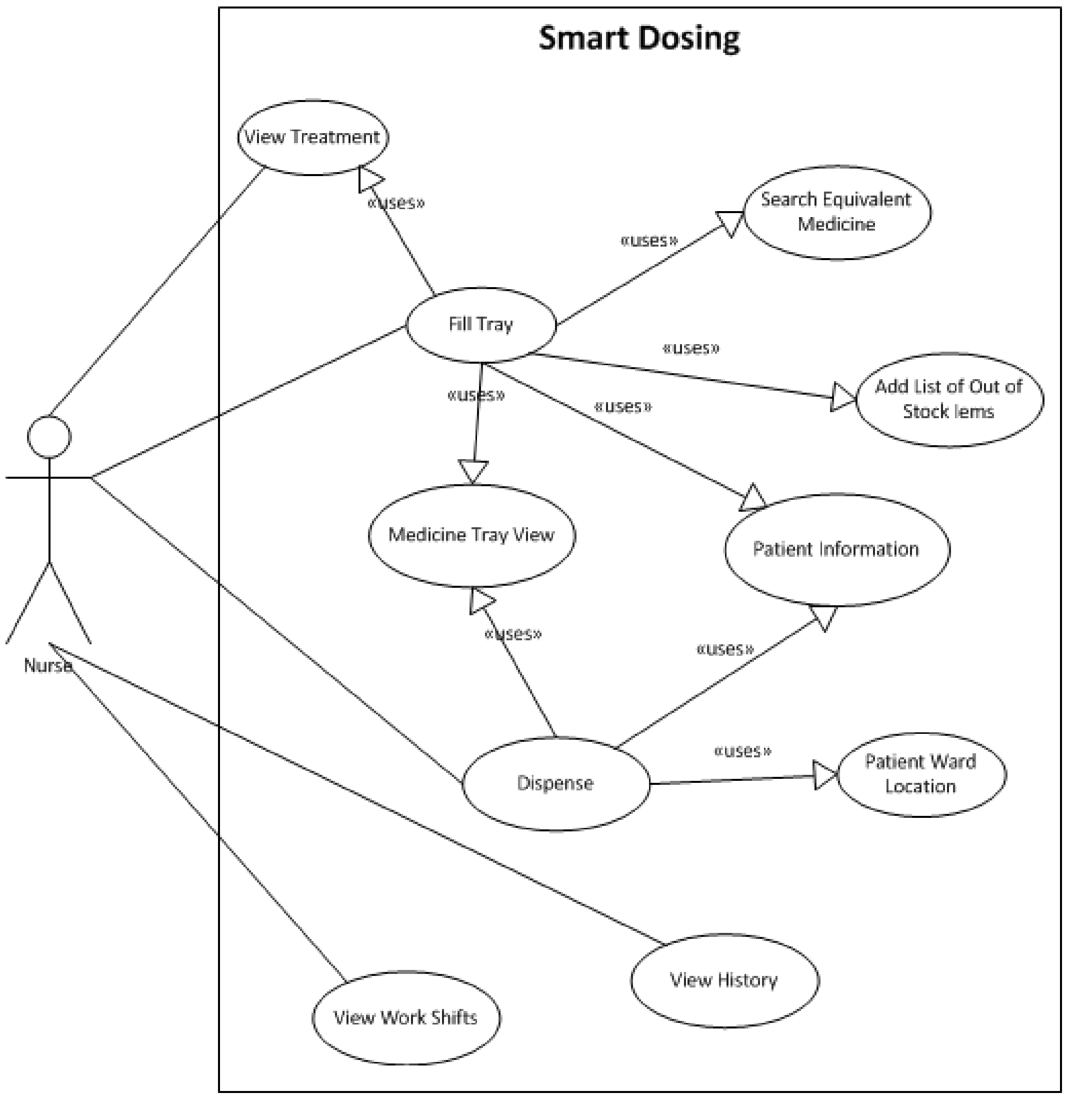


Figure 4.2: Use Case Diagram

.

4.3.2 *Application Event Diagram*

Apart from administrative task like adding patient information, adding medicine information or adding user/nurse to the system, the application has focused mainly on two major tasks which are filling the medicine tray and then dispensing the filled tray. These two tasks can be explained with the help of a sequence diagram. Sequence diagram shows the processes/systems/actors which are involved in a transaction and the order of interaction between them. The transaction details of the filling tray and dispensing the tray are explained with the help of sequence diagrams.

Fill Tray Sequence Diagram

The fill tray transaction is initiated by the user/nurse on the web client. This triggers a request for a patient record on the server. The server then contacts the database and retrieves the required patient information. The information is then relayed back to the user and it is displayed on the user interface. Based on this information the user can initiate a call for filling the tray with medicine when the tray is actually filled with the medicine. On actuation of this call, a notification is sent to the server and this server in turn sends a notification to the database to store the state of tray. In case, a required medicine is out of stock, the client initiates an out of stock request. The initiated call is followed by the required quantity of that medicine and the quantity is then saved by the server on the database. The user can then also search for an equivalent medicine by sending a request to the server with the medicine name. An active communication is established between the server and the database to search for an alternative medicine with the same active ingredients. Filling Tray sequence diagram is shown in Figure 4.3.

Medication Dispense Sequence Diagram

The second main task is to dispense the medicines among patients which have been filled earlier during the first main task of filling the tray. The dispense action begins with the initiation by the user/nurse where they ask the server which particular medicine is to be given to a patient. The server queries the database and informs this back to the user. The user/nurse then physically dispenses the medicine according to the information provided and a notification is sent to the server and also to the database. The sequence Diagram for the dispensing process is shown in Figure 4.5.

4.3.3 Deployment Diagram

Deployment diagram will give a detail account of distribution of the hardware as well as the software components. In this application the user/nurse requires an internet enabled web browser which can be used to connect to our application server. On the application server the following software components are installed.
Apache Tomcat Server. This application is hosted on Apache Tomcat version 6.0.Tomcat is an open source web server which also provides HTTP environment for Java[21].
Vaadin. This application is built on Vaadin 6.8.0[22].
Application. This Application is developed in Java using J2EE and Servlet. The application is hosted on the web server in the form of WAR file format.
Resources. All the images and web content related static asset are stored on to our application server.

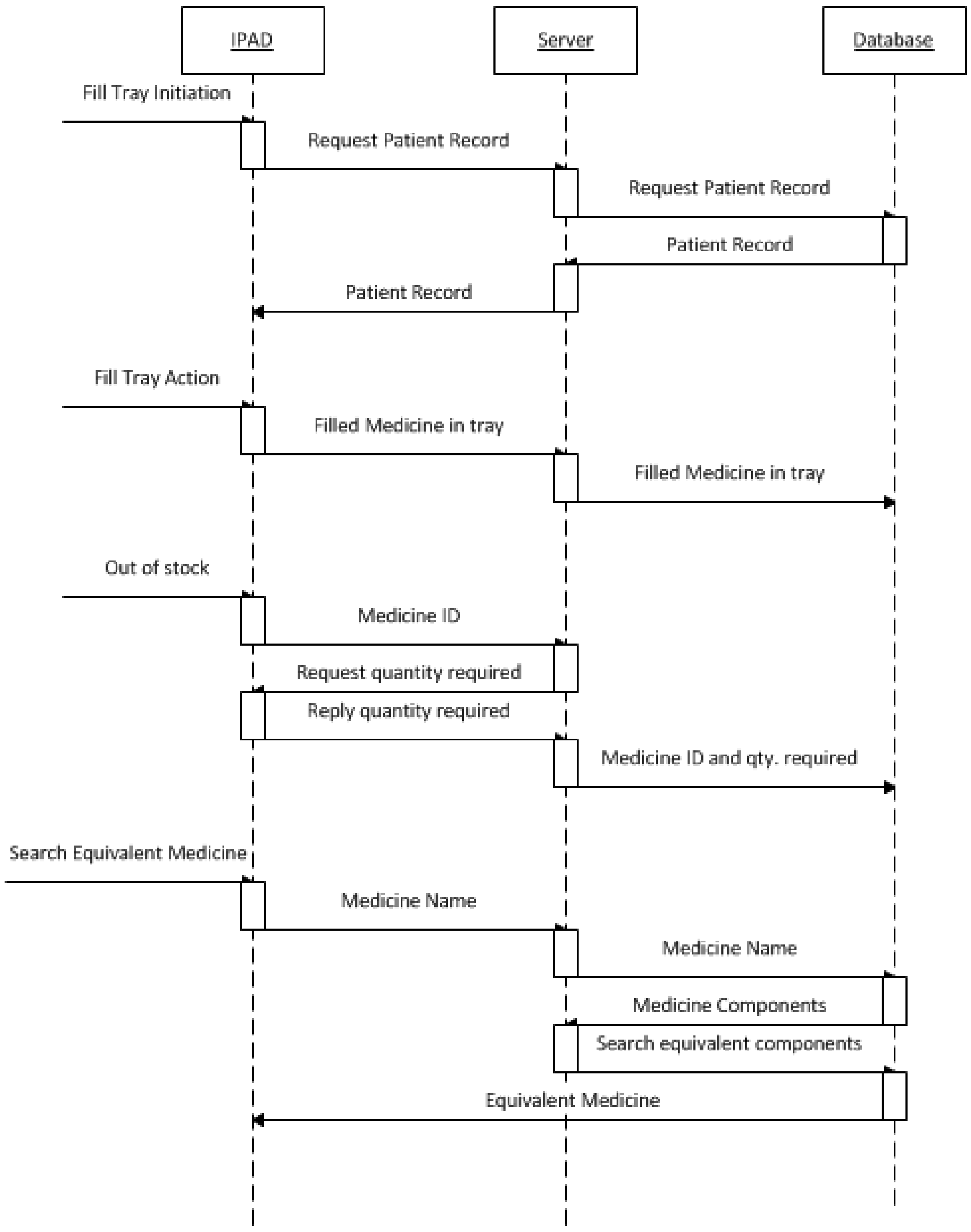


Figure 4.3: Fill Tray Sequence Diagram

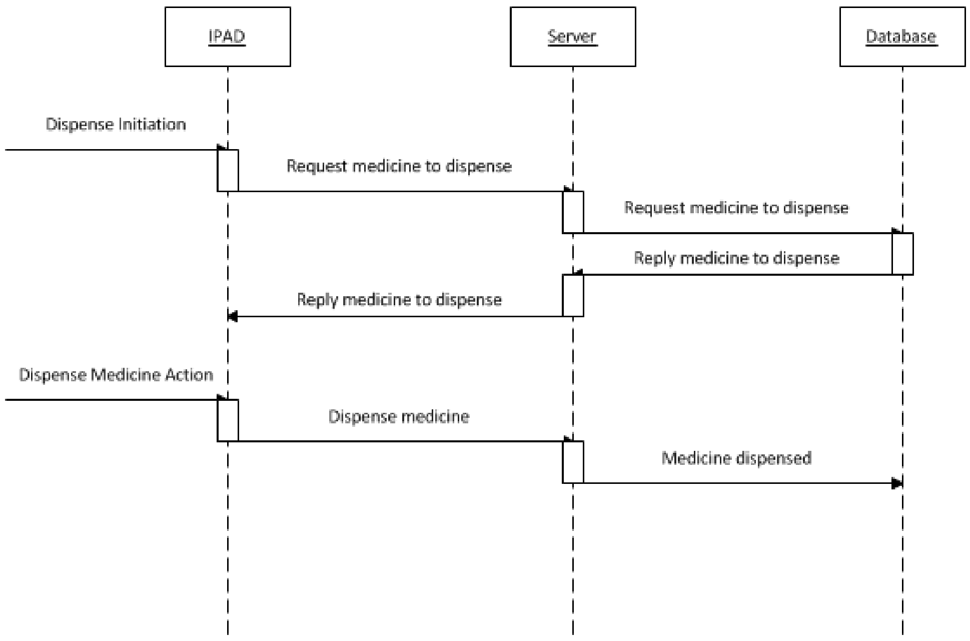

IPAD
Server
Database
Dispense Initiation
Request medicine to dispense
Request medicine to dispense
Reply medicine to dispense
Reply medicine to dispense
Dispense Medicine Action
Dispense medicine
Medicine dispensed

Figure 4.4: Dispense Sequence Diagram

Other than application server the third node is the database server. It consists only of database which is implemented using MySQL server 5.5[23]. The application server is connected to the database server using a JDBC connection[24]. The deployment diagram is represented in Figure 4.5.

4.4 Application Usability

When an application is being designed, the usability of the system has to be given importance. In our design, the end users are nurses, doctors and other staff in hospitals. In such situations the application designer should design the user interface with the following consideration [25] [26]:

- There should be minimum input fields while generating the maximum useful output.
- Navigation complexity makes the user interface complex. Application designer should consider this aspect by making it simple.
- Data entries should be validated. In case of wrong input it should be highlighted to make it visible.
- For easy system monitoring, all ongoing process should be made visible to the end user.

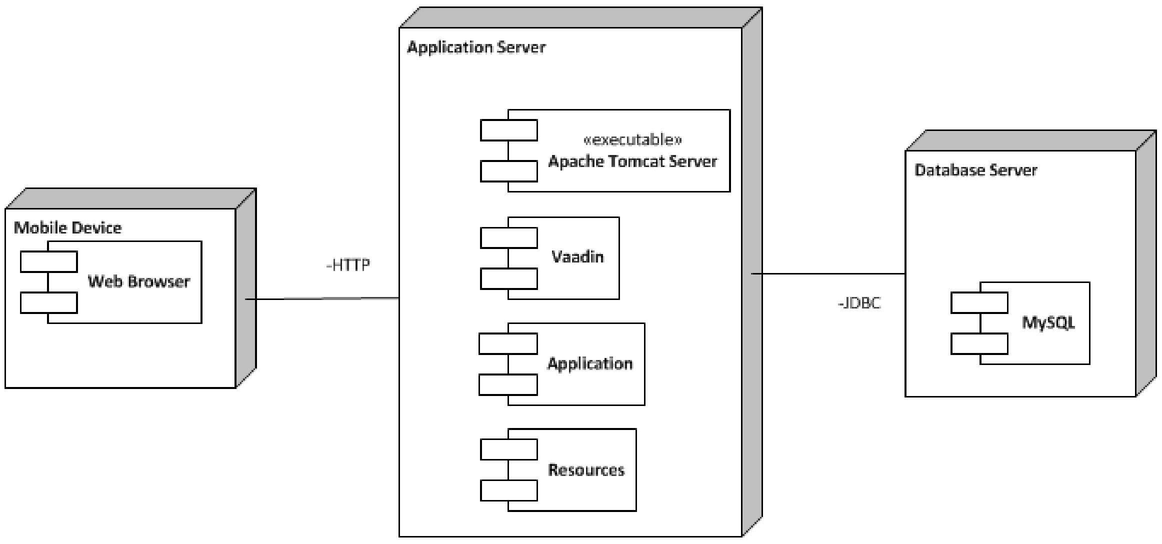


Figure 4.5: Deployment Diagram

4.5 Database Design

A database provides persistence to an application. It can be used to store all kinds of information related to an application. It can be as simple as a text file to being complex as a relational database. Database itself can be managed and manipulated with the help of data definition language (DDL). SQL is one of the most popular Data definition language used in relational database.

Database is the process of defining an efficient model for representing data used in an application. In database design the entities and their attributes are defined. Relationship between the entities is also defined in the designing process. In our database design we have isolated independent data elements and have formulated a relationship between them in the form of database tables. The database design process of smart dosing application is explained below

- In smart dosing application there are multiple entities which are needed to be stored for which a relational database is a suitable tool.
- The next step was to define the entities and their attributes which needed to be saved based on our application.
- The selected entities were formed into tables of a database and their attributes are set as columns in the table created for their respective entity.
- After formation of tables and their columns, primary key for each table is identified as the unique identifier for each record in a table.
- After that the relationship between two tables were set up using primary keys for identification.

- In the end the design was analyzed for errors and normalization rules were applied. Normalization is the process to remove redundancy and dependency.

4.6 Entity Relation Diagram

In smart dosing application entities use tables to represent the database. These entities are briefly explained below:

4.6.1 Patient

Patient entity is used to describe the patient at a hospital. The patient has following attributes

SSN
In Finland every person is assigned a unique number based on their birth date named as Social Security Number. As this number is unique for each person it can also be taken as the primary key of the patient entity.

First Name and Last Name
Name of the patient is stored under these attributes.

Date of Arrival and Time
The attributes are used to save the date of arrival and the exact time of the patient to the hospital.

Ward Room and Bed
Every patient admitted to the hospital is assigned a specific ward with a room and a bed. This information is saved in these attributes to be used by other entities.

Every patient has some medicine prescribed by the doctor. The list of medicine prescribed is saved in the treatment entity. Each patient can have multiple medicines prescribed so the relationship between the patient and treatment entity is one to many.

4.6.2 Treatment

Treatment entity is used to store the information regarding the medicine prescribed to the patients.

Treatment ID
TreatmentID is an automated generated unique number which is used as a primary key for this table.

Patient SSN and Medicine ID

PatientSSN is the social security number of the patient whose medicine is saved in the treatment record. This attribute is a foreign key imported from the patient table. MedicineID is a foreign key imported from the medicine table. It refers to the name of the medicine prescribed to a patient.

Amount and Unit

The prescribed amount of the medicine with its unit for a patient is stored in these attributes.

Posology

This column stores the time and frequency in a day when a medicine is to be dispensed to a patient.

Start Date and End Date

This attribute stores the start and end date of the medicine prescribed by the doctor.

Doctor and Comments

The name of the doctor who prescribed the treatment and their comments are saved under these attributes.

4.6.3 Medicine

Medicine entity is used to store all the information related to the medicines available in the hospital.

Medicine ID and Name

MedicineID is an automated generated primary key to represent a medicine uniquely. The medicine name is stored in the name attribute.

Component

It contains the information about the active components in the medicine.

Indication and Comments

The ailment for which the medicine can be used is stored in Indication and any comment related to the medicine is stored in comments column.

Method of Administration

Each medicine has its own method of administration. There are different types of medicine administration used namely as nasal administration, intravenous, subcutaneous, intramuscular, epicutaneous, intracardiac, intrathecal etc.

4.6.4 *History*

This table contains history information of when a medicine is dispensed to a patient.

ID and Treatment ID
ID is the automated generated unique primary key whereas the treatmentID is the foreign key imported from the treatment table.

Dispensed By and Filled By
Dispensed By and Filled By holds the name of the nurse who has performed the specific task accordingly.

Dispense Time and Fill Time
The exact time of filling the tray and dispensing the tray are stored in DispenseTime and FillTime.

Skipped By, Skipped Time and Skipped Reason
These attributes are used to store the information related to the skip medicine process. It stores the time and name of the nurse who skipped the medicine and reason for skipping the medicine.

4.6.5 *Nurse*

Nurse information is stored in nurse entity.

Nurse SSN
Nurse social security number acts as a primary key for the table.

First Name and Last Name
Name of the nurse is stored in these attributes.

Email and Phone
The contact information like email and phone are stored in their respective attributes.

Password
Every nurse needs to have a password to login to the application. This password is saved in the password column of the table.

4.6.6 *Shift*

Shift table contains the information related to all the nurses work shifts, particularly the date and the time.

4.6.7 Out of stock

This table is used to store list of out of stock medicines. It stores the demanded medicine with quantity and the date it went out of stock.

4.6.8 Fill tray and Dispensation

These tables contain temporary information of the two processes. *FillTray* stores all the information during the fill tray process, similarly *Dispensation* stores the information during the dispensation process. It stores the information of the nurse who filled the tray, at what time and the tray slot. In dispensation table, the information of the nurse who dispensed the medicine with the time of medicine dispensation is stored.

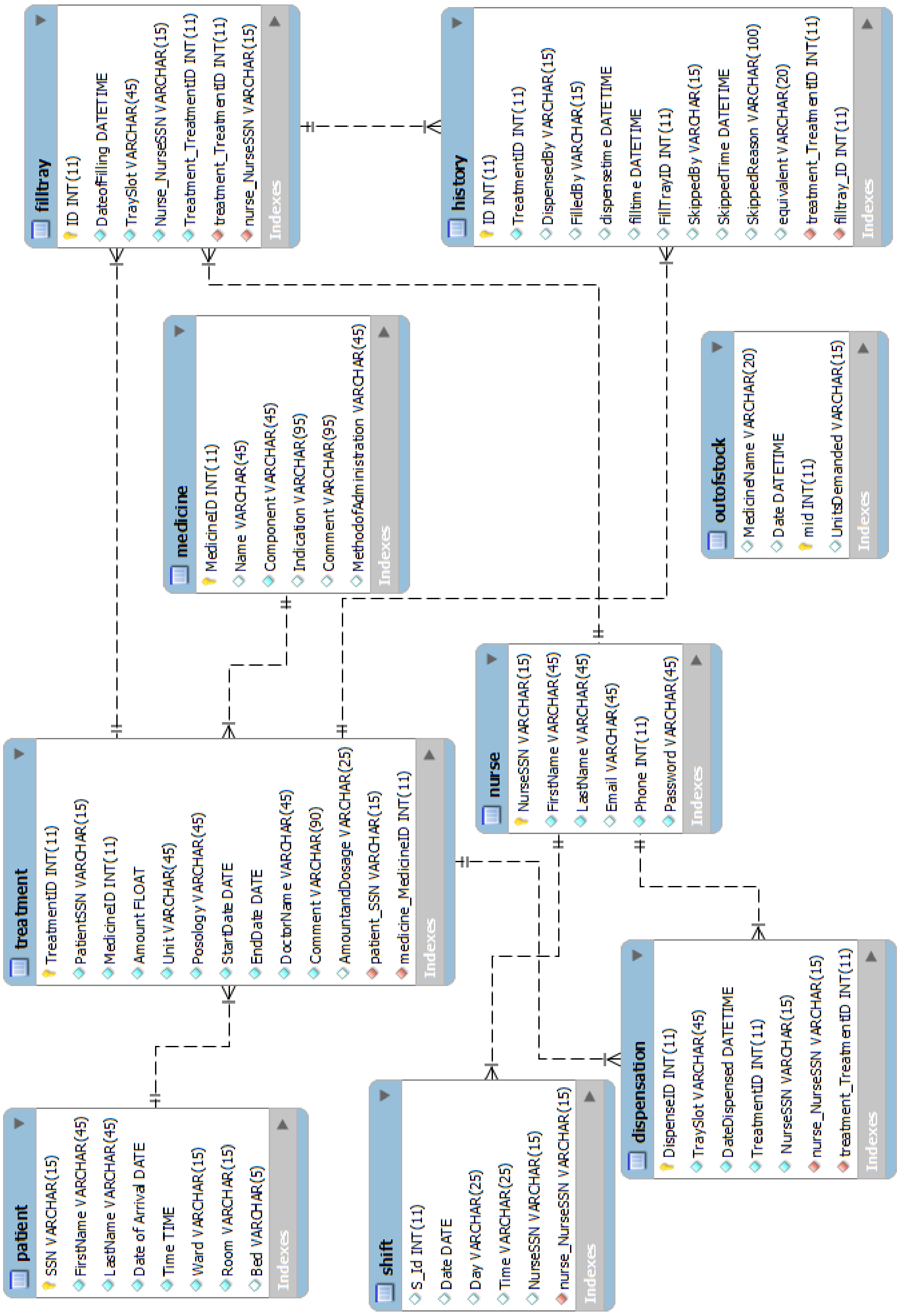


Figure 4.6: Entity Relation Diagram

5 IMPLEMENTATION

5.1 Authentication and Session maintenance

In order to maintain persistence for a user, Vaadin uses cookies as well as Java Servlet. A cookie can store a unique identifier that can be identified by Vaadin and this unique identifier is different for each user. It is also used by Vaadin internally to recognize a user. In Smart Dosing Application after a nurse has logged in, her SSN is stored locally in each instance and this signifies that the user has logged in. Since Java Servlet is multi threaded, there is no possibility of data collision between instances. Authentication for each nurse is required before they can access the system. Authentication is required for two main reasons, which are:

- To restrict unauthorized access.
- To identify a nurse.

The login process is show in Figure 5.1. The authentication of the nurse is handled by the *LoginWindow* class. Firstly, user of the application is presented with a log in screen. The nurse enters her user name and password in the required fields. Nurse user name is basically their social security number. The social security number is validated for its correct format. The SSN and password pair is sent to the LoginWindow Class for authentication which in turn queries the database for the existence of the nurse and correctness of the password. In case there is some error during authentication, the nurse is displayed the error and is taken back to the log in screen. If the credentials are correct the nurse is successfully logged in to the system. The query which is used on the database to validate the existence of user and to authenticate it, is shown in Listing 5.1.

```
FreeformQuery q5 = new FreeformQuery("SELECT * From nurse Where NurseSSN = '"+
    SSN +"' and Password = '"+Password+"'",connectionPool2);
```

Listing 5.1: Query for Authentication

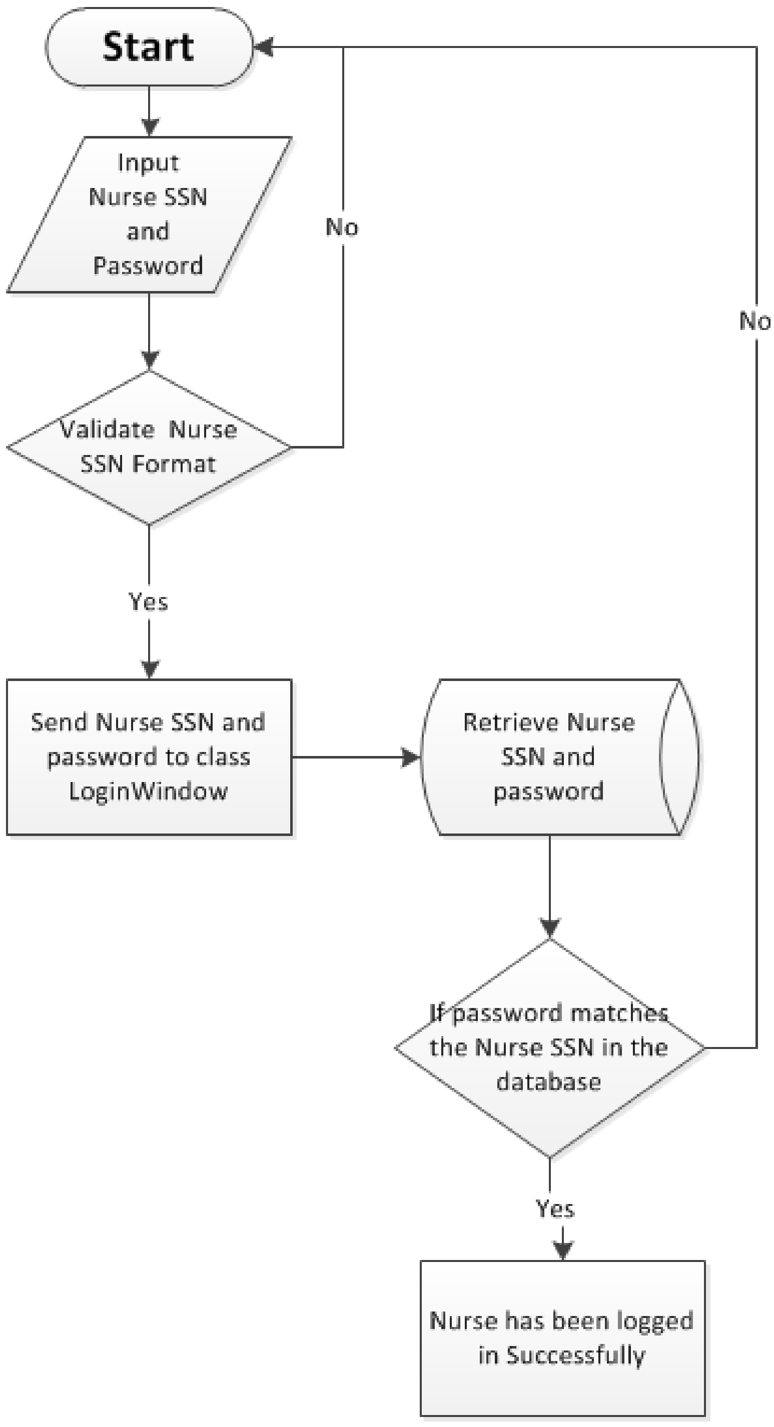

Start
Input Nurse SSN and Password
Validate Nurse SSN Format
No
Yes
Send Nurse SSN and password to class LoginWindow
Retrieve Nurse SSN and password
If password matches the Nurse SSN in the database
No
Yes
Nurse has been logged in Successfully

Figure 5.1: Log In Process Flow Chart

FreeFormQuery query delegate is used to make custom SQL queries to the database instead of using readymade queries made by object mapping classes such as *Table-Query*. *connectPool2* is used to store settings related to database connection. Figure 5.2 shows the user login interface for the smart dosing application.

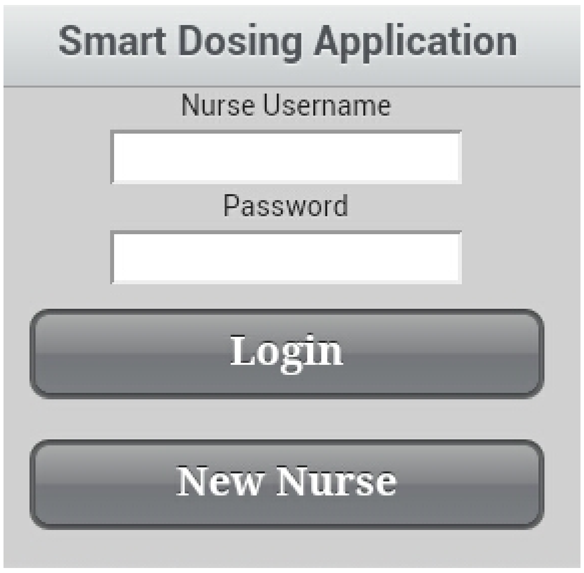


Figure 5.2: User Login Interface

5.2 Medicine Dispensation

Once the nurse has been successfully logged in, she will be presented with the main interface of the application where she can access all the available functionalities of the smart dosing application. The two main processes of the application are filling the tray and dispensing the tray to the patient. The main interface also shows the nurse with different system related information such as patient, medicine, nurse. This information is divided in different tabs at the bottom. The last tab in the tab menu is the settings tab which has different options like adding new information such as nurses, patients, medicines and it also provides options for checking out of stock items and nurses shift. This interface is implemented in the class *tabmenu*, which extends from the Vaadin class *TabBarView*. Each tab in the interface is represented by a different class. The default tab is set to the smart dosing tab. The smart dosing tab is represented by class *smartdosing*. Figure 5.3 shows the main interface of the application after the user succesfully logs in to the application.

In the *smartdosing* class, the two main processes Fill tray and Dispense are represented by use of navigation buttons which are implemented in their respective class.

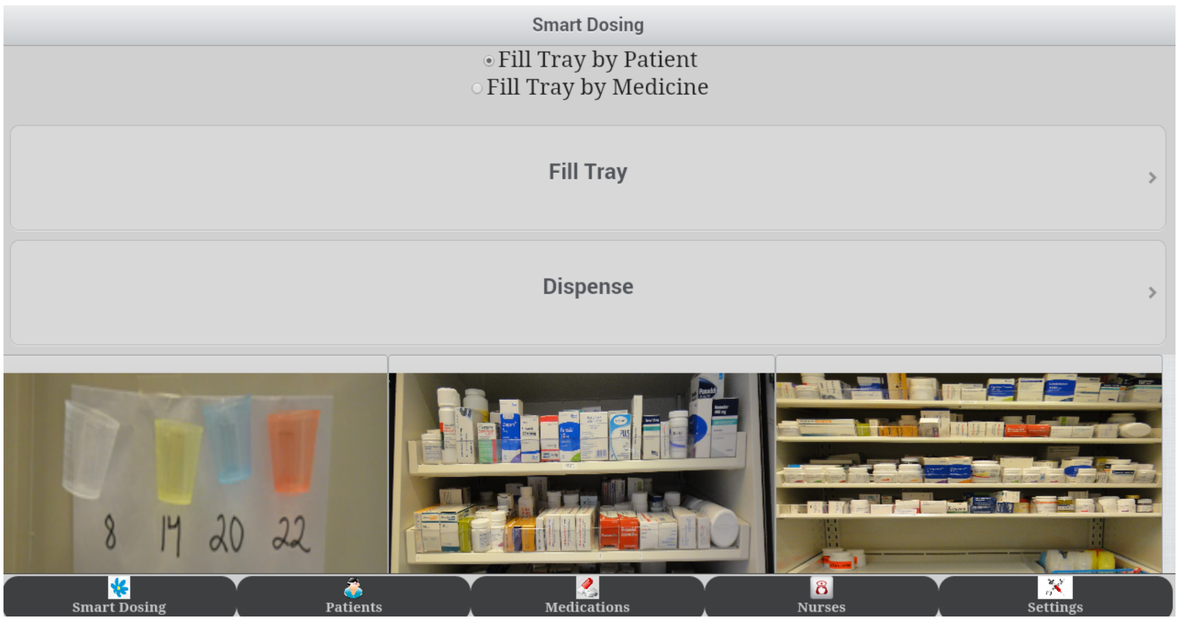


Figure 5.3: Main Interface

The navigation button class is used to implement buttons on the screen which when pressed will perform the actions as specified. Since Vaadin is an event driven framework every action is taken as a reaction of an event. In the navigation button's case the action is the click on the button and this event is handled by a listener. The fill tray and dispense process can be performed in a variation of ways that is by filling the tray according to patient or medicine. The selection between these options is done by use of radio box selection options. These options are presented in the default smart dosing tab.

5.2.1 *Filling Tray*

As previously discussed, the fill tray interface is shown according to the option selected in the main interface. The first available option is to fill the tray according to the patients and the other option is to fill the tray according to the medicine names.

First if we have a look at the SQL queries, all necessary information about the patient spanning over multiple tables is collected by the query and is filled into a single query result container. The container used to collect the result from the query is SQL container which is extended from Vaadin container interface; it allows fetching of data from the database. The query fetches data from three tables in the database namely, patient, treatment and medicine. From the patient table, *SSn*, *FirstName*, *LastName*, *Ward*, *Room*, *Bed* are selected. From the medicine table, *Name* is selected. From the treatment table, *TreatmentID*, *AmountandDosage*, *unit*, *posology* are selected. First of all, *SSn* from the patient table is used as a search term in the treatment table in the form of a foreign key. In addition to that, *medicineID* from the medicine table is used as a secondary search term in the treatment table in the form of a foreign key. In case of filling the tray according to the patient, the result of the query is ordered with respect to the patient SSN and incase of medicine, it is ordered with respect to medicine name.

```
FreeformQuery q3 = new FreeformQuery("SELECT t.TreatmentID,p.SSn,p.FirstName,p.
      LastName,p.Ward,p.Room,p.Bed,m.Name,
t.AmountandDosage,t.Unit,t.posology FROM treatment t INNER JOIN patient p ON t.PatientSSN = p.
      SSN
  INNER JOIN medicine m ON t.MedicineID= m.MedicineID ORDER BY p.SSN ",
      connectionPool);
```

Listing 5.2: Query for patient data collection when tray is filled by patient

```
FreeformQuery q3 = new FreeformQuery("SELECT t.TreatmentID,p.SSn,p.FirstName,p.
      LastName,p.Ward,p.Room,p.Bed,m.Name,
t.AmountandDosage,t.Unit,t.posology FROM treatment t INNER JOIN patient p ON t.PatientSSN = p.
      SSN
 INNER JOIN medicine m ON t.MedicineID= m.MedicineID ORDER BY m.Name ",
      connectionPool);
```

Listing 5.3: Query for patient data collection when tray is filled by medicine

For navigation, there are two buttons to iterate through all the items. An *ListIterator* instance is created which allows traversal through the items. The *ListIterator* traverses the items on the basis of their ids which it collects from the SQL container where the items were created as a result of the query. The *ListIterator* class uses two methods for traversing through the items. When the click listener for the next button is invoked, the *ListIterator* uses the method *next()* to move to the next item in the container. When the click listener for the back button is invoked, the *ListIterator* uses the method *previous()* to move back to the previous item in the container.

In the middle of the interface as shown in Figure 5.4, there is an overview of the current status of the tray which is to be filled. The state of the tray is dependent on the number of patients that have been or going to be given their medicine. For each patient whose medicine is filled, the medicine is shown on a particular slot on the tray. On the top right corner of the interface, there is a figure of medicine glasses which specify timing for intake of a medicine. Different colors of glasses represent a different time of the day. Depending upon the posology of the medicine of each patient, the glasses are filled with medicine. Usually there are five times per day when a patient can take a medicine. Below the glasses picture there is a picture of the medicine which is to be filled in the tray.

Fill tray interface provides three more functionalities to the nurses. These functionality buttons are under the medicine picture. First of these three is "Search equivalence". Search equivalence provides the nurses with the option to search for an equivalent medicine if a given medicine is not available for any reason. The equivalent of the medicine is searched using the active component of the medicine. Medicine with the same active component is termed as an equivalent of the medicine. First of all the components of the medicine are found and then compared with the components of the medicines in the database. The medicines with the same component names are then shown as the equivalent of the medicine.

```
FreeformQuery q7 = new FreeformQuery("SELECT * FROM medicine WHERE Component
    =component_value,connectionpool);
```

Listing 5.4: Query to search medicine equivalence

The second option provides the nurses with overview of the treatment prescribed by the doctor to the patient whose medicine they are filling at that time.

The third functionality provides the nurse with the report of the medicine that is out of stock. In case the nurse is filling a medicine in the tray and it runs out from the store, the nurse can report that medicine as out of stock and she also has the option to order the required amount.

```
statement.executeUpdate("INSERT INTO outofstock (MedicineName,Date,mid,
    UnitsDemanded) VALUES (' name ',NOW(),DEFAULT,'quantity")");
```

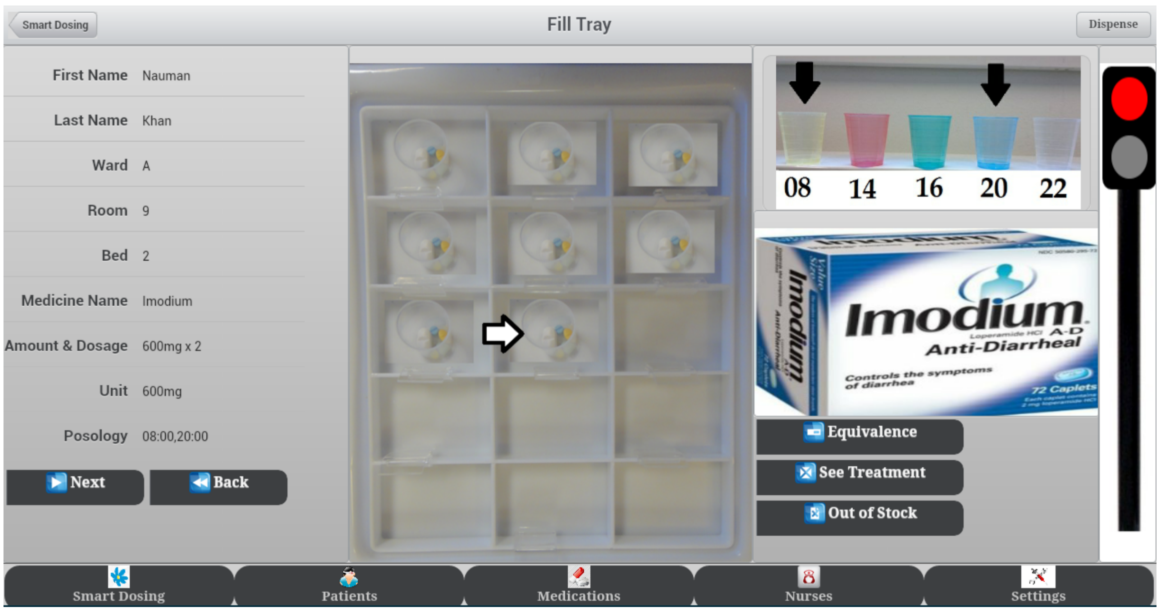
Listing 5.5: Query to enter out of stock medicine

The method *NOW()* is a SQL method which gets the current date and time of the system. The fill tray interface allows the nurse to navigate between different interfaces. Nurses are provided with the option to navigate back to the main smart dosing interface or they can also navigate to medicine dispensing interface through the navigation buttons on the top corner of the interface.

5.2.2 Dispensing Tray

After filling the medicine tray, nurse has to dispense the medicine to the patients in the ward. The dispense interface helps the nurses in performing these tasks. Dispense interface has almost the same layout of that fill tray interface so the nurses have the ease of using this application. The interface is divided into three main columns. The first part of the column shows the information about the patient like, name of the patient, medicine to be dispensed, location of the patient in the ward and time of medication. When the medicine in the tray is filled with the medicine, the information is stored in the database in the table *FillTray* with the place of the medicine in the tray. When the tray is ready to be dispensed, the information from the fill tray table is collected in the SQL container.

The SQL query used to fetch data in the container is showed in the Listing 5.6. The query fetches the data from the combination of different columns from patient,



First Name	Nauman
Last Name	Khan
Ward	A
Room	9
Bed	2
Medicine Name	Imodium
Amount & Dosage	600mg x 2
Unit	600mg
Posology	08:00,20:00

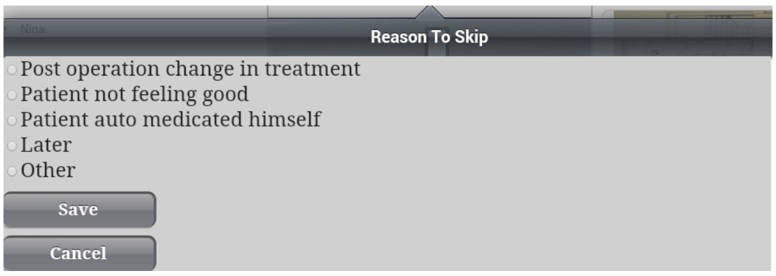
Figure 5.4: Interface for filling tray process

medicine, treatment and filltray tables. It collects the patient name, ward, room and bed from patient table and joins this data with the medicine name from medicine table and amount and posology from treatment table as previously done in the *filltray* class. The only difference in the *dispense* class in that all this data is joined with the place of the medicine in the tray which is stored in the *filltray* table in the *trayslot* column.

```
FreeformQuery q4 = new FreeformQuery("SELECT t.TreatmentID,p.FirstName,p.LastName,
    p.Ward,p.Room,p.Bed,m.Name,t.AmountandDosage,t.Unit,t.Posology,f.ID,f.TraySlot FROM
    filltray f INNER JOIN treatment t ON f.Treatment_TreatmentID = t.TreatmentID INNER JOIN
    patient p ON t.PatientSSN = p.SSN INNER JOIN medicine m ON t.MedicineID= m.MedicineID
    ORDER BY f.TraySlot ",connectionPool1);
```

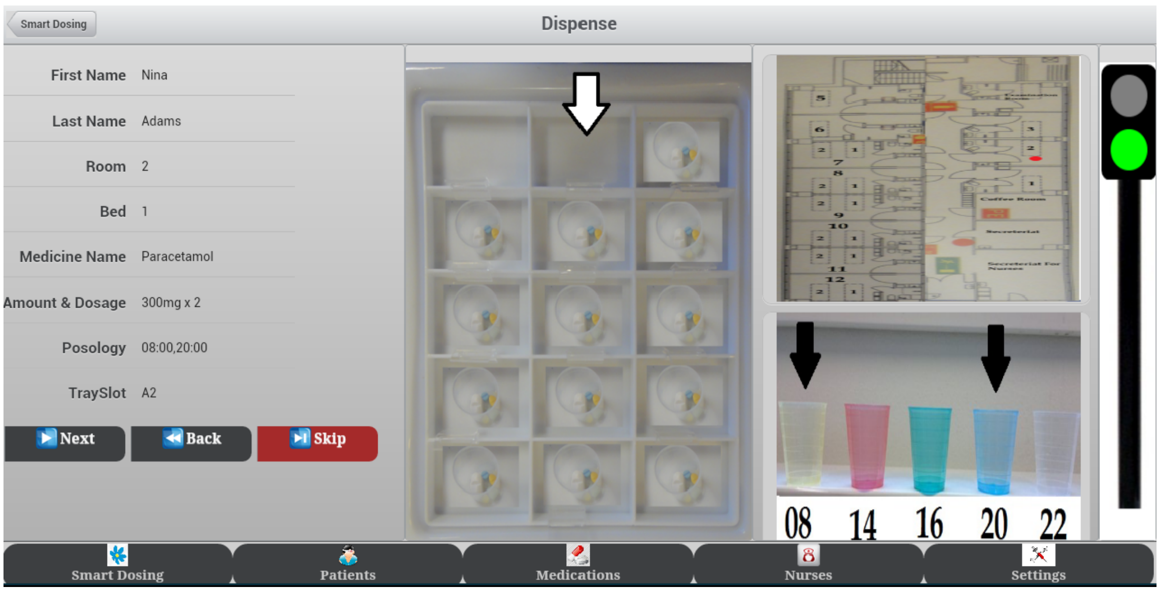
Listing 5.6: Query for patient data collection for dispensing

Like the fill tray interface, nurses have the same buttons to navigate through the items in the container. The class *ListIterator* allows traversal through the items. The navigation buttons, next and back uses the method provide by the *ListIterator* class. In the dispense interface as shown in Figure 5.6, there is one other option for the nurses to skip the medicine. When the nurse clicks the skip button to skip a medicine, she is provided with a pop up window which asks her to enter the reason for skipping a particular medicine. A nurse is already provided with the most common reasons to skip the medicine but if there is any other reason she can also write it on the window. The reasons provided to the nurse are shown in Figure 5.5,


Reason To Skip

Post operation change in treatment
Patient not feeling good
Patient auto medicated himself
Later
Other

Figure 5.5: Reasoning for skipping the medicine


Dispense

First Name	Nina
Last Name	Adams
Room	2
Bed	1
Medicine Name	Paracetamol
Amount & Dosage	300mg x 2
Posology	08:00,20:00
TraySlot	A2

Figure 5.6: Interface for Dispensing process

5.3 History Maintenance

One of the core features of this application is that it provides all the medicine intake record of the patient. All the information related to the patient like medicine given to the patient, time of the medicine, name of the nurse giving the medicine are stored in the database. The history table is populated as the fill tray and dispensing process starts. On each transaction both of these operations, an entry related to the transaction is saved into the history table. For the fill tray process, the query for updating the history table is given in Listing 5.7.

```
statement.executeUpdate("INSERT INTO History (ID,filltime,FillBy,TreatmentID,FillTrayID
    ) VALUES (DEFAULT,NOW(), '"NurseSSn"',TreatmentID,Filltray_id)");
```

Listing 5.7: Query to update history table after fill tray process

The columns which are updated after the fill tray process are *ID*, *filltime*, *FillBy*, *TreatmentID*, *FillTrayID*. The *filltime* is filled with the current date and time of the system by using the method NOW(). *FillBy* stores the information of the nurse performing the task and *TreatmentID* saves the primary key of the table treatment as a foreign key. In the dispensing process when a transaction is successfully performed the history record of the item is updated with the missing values which were earlier missed when it was filled in the tray.

```
statement.executeUpdate("UPDATE History SET DispenseBy='"+nurseSSn+"',
    DispenseTime = NOW() Where FillTrayID =fillid AND TreatmentID ="+treatmentid);
```

Listing 5.8: Query to update history table after dispense process

The time of dispensing, the medicine and the nurse who has dispensed the medicine is updated in their respective columns. In case the medicine is skipped for any reason, then the reason for skipping the medicine is stored in the history table and *DispenseTime* and *DispenseBy* columns are left blank in the table for the particular item. The query delegate used for the medicine, which is skipped, is listed in the Listing 5.9.

```
statement.executeUpdate("UPDATE History SET SkipBy='"+nurseSSn+"',SkipTime =
    NOW(),Reason='"+reason+"' Where FillTrayID ="+fillid+" AND TreatmentID ="+
    treatmentid);
```

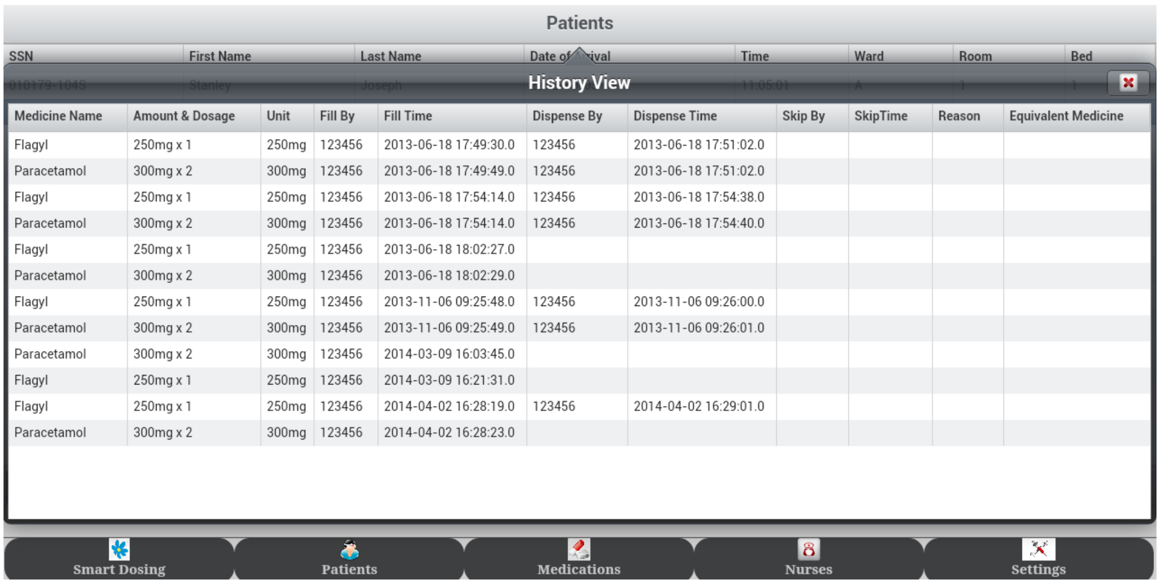
Listing 5.9: Query to update history table incase medicine is skipped

A nurse has the option to view the history of any selected patient from the patient tab. She can select the patient from the patient table in the patient tab. History of the selected patient is shown in a new window. The query used for fetching the history of the selected patient is listed in the Listing 5.10.

```
FreeformQuery hist = new FreeformQuery("SELECT m.Name,t.AmountandDosage,t.Unit,h
    .FillBy,h.filltime,h.DispenseBy,h.dispensetime,h.SkipBy,h.SkipTime,h.Reason,h.equivalent
    FROM history h INNER JOIN treatment t ON h.treatmentID = t.treatmentID INNER JOIN
    medicine m ON t.MedicineID= m.MedicineID WHERE t.PatientSSN = ssn",connectionPool
    );
```

Listing 5.10: Query to show history of the patient

The history table contains the primary key of the table treatment as a foreign key. The query gathers the information from the tables by using the keyword *INNER JOIN*. The SSN of the person whose history is to be shown is used to collect all the relevant information from the database tables. When the nurse selects a patient whose history is required to be displayed, the SSN of the selected patient is passed to the method which executes the query delegate to combine all the information from different columns in different tables in the database.



Medicine Name	Amount & Dosage	Unit	Fill By	Fill Time	Dispense By	Dispense Time	Skip By	SkipTime	Reason	Equivalent Medicine
Flagyl	250mg x 1	250mg	123456	2013-06-18 17:49:30.0	123456	2013-06-18 17:51:02.0				
Paracetamol	300mg x 2	300mg	123456	2013-06-18 17:49:49.0	123456	2013-06-18 17:51:02.0				
Flagyl	250mg x 1	250mg	123456	2013-06-18 17:54:14.0	123456	2013-06-18 17:54:38.0				
Paracetamol	300mg x 2	300mg	123456	2013-06-18 17:54:14.0	123456	2013-06-18 17:54:40.0				
Flagyl	250mg x 1	250mg	123456	2013-06-18 18:02:27.0						
Paracetamol	300mg x 2	300mg	123456	2013-06-18 18:02:29.0						
Flagyl	250mg x 1	250mg	123456	2013-11-06 09:25:48.0	123456	2013-11-06 09:26:00.0				
Paracetamol	300mg x 2	300mg	123456	2013-11-06 09:25:49.0	123456	2013-11-06 09:26:01.0				
Paracetamol	300mg x 2	300mg	123456	2014-03-09 16:03:45.0						
Flagyl	250mg x 1	250mg	123456	2014-03-09 16:21:31.0						
Flagyl	250mg x 1	250mg	123456	2014-04-02 16:28:19.0	123456	2014-04-02 16:29:01.0				
Paracetamol	300mg x 2	300mg	123456	2014-04-02 16:28:23.0						

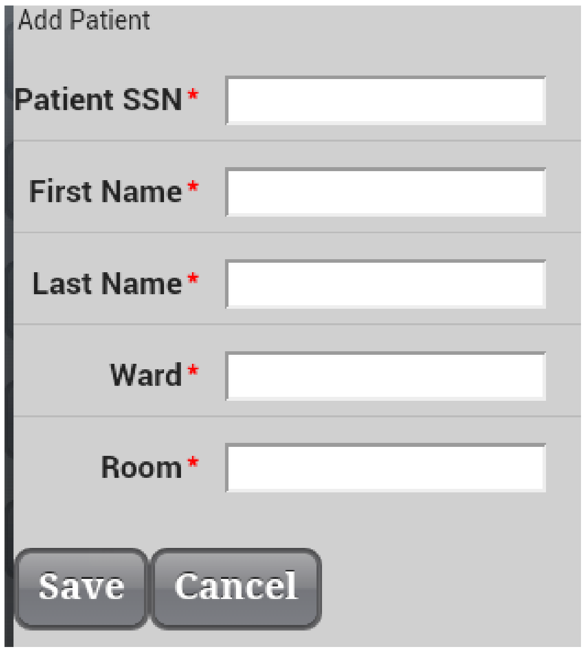
Figure 5.7: History view of a patient

5.4 Miscellaneous

The application provides many added functionalities to the nurse. These functionalities help the nurses to perform the medicine dispensation efficiently. All these functionalities are provided in the *Settings* tab.

5.4.1 *Addition of new record*

The application provides the nurse with the function to add a new patient to the database. The nurse has the option to add a patient with all their attributes like name, social security number and location in the ward. These attributes are saved in the database with the help of a query delegate which makes use of the *Insert* query. All the attributes of the medicine like name, component, indication and comment about the medicine are passed to the method which executes the query for storing these attributes in the medicine table. All existing nurses have the access to add a new nurse to the database and the new nurse can perform the process of dispensation with their own account. Figures 5.8, 5.9, 5.10 show the interfaces for adding a new record to the database.

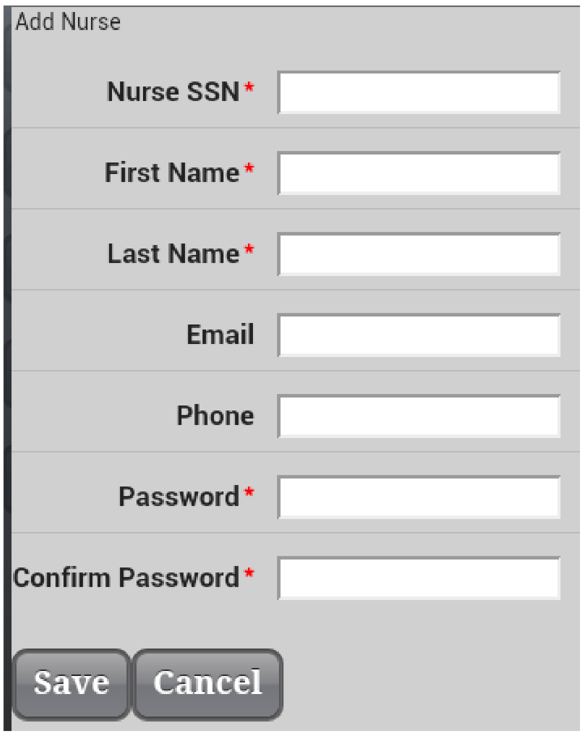

Add Patient

Patient SSN*	
First Name*	
Last Name*	
Ward*	
Room*	

Figure 5.8: Application interface to add new patient

5.4.2 *Nurse shift*

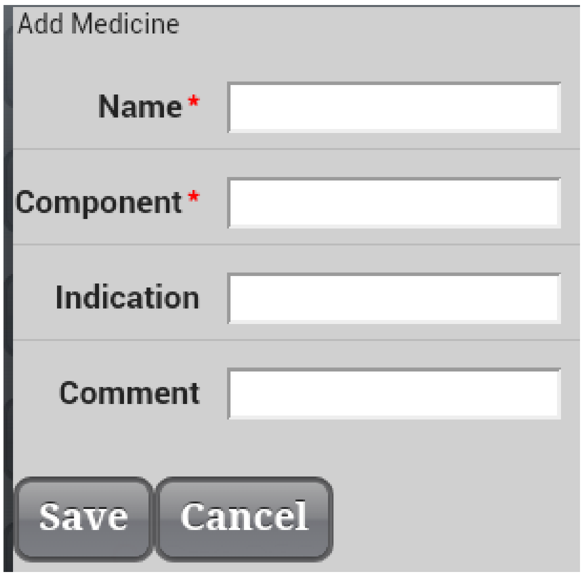
The nurses in the hospital work in shifts. Every nurse has her own work schedule. This functionality provides the nurses with the option to see their shift for the entire week. Nurse shift shows the schedule in the form of the days and the time they have the shift. Figure 5.11 shows the overview of the shift schedule of the nurses.



Add Nurse

Nurse SSN*	
First Name*	
Last Name*	
Email	
Phone	
Password*	
Confirm Password*	

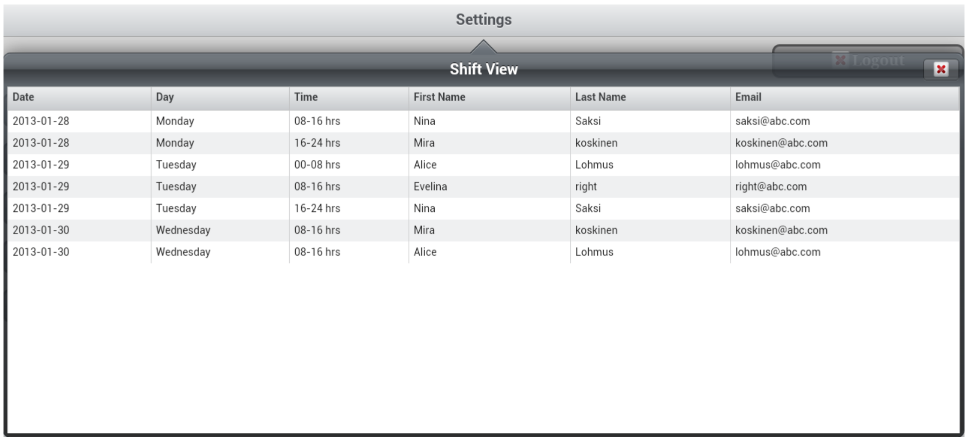
Figure 5.9: Application interface to add new nurse



Add Medicine

Name*	
Component*	
Indication	
Comment	

Figure 5.10: Application interface to add new medicine

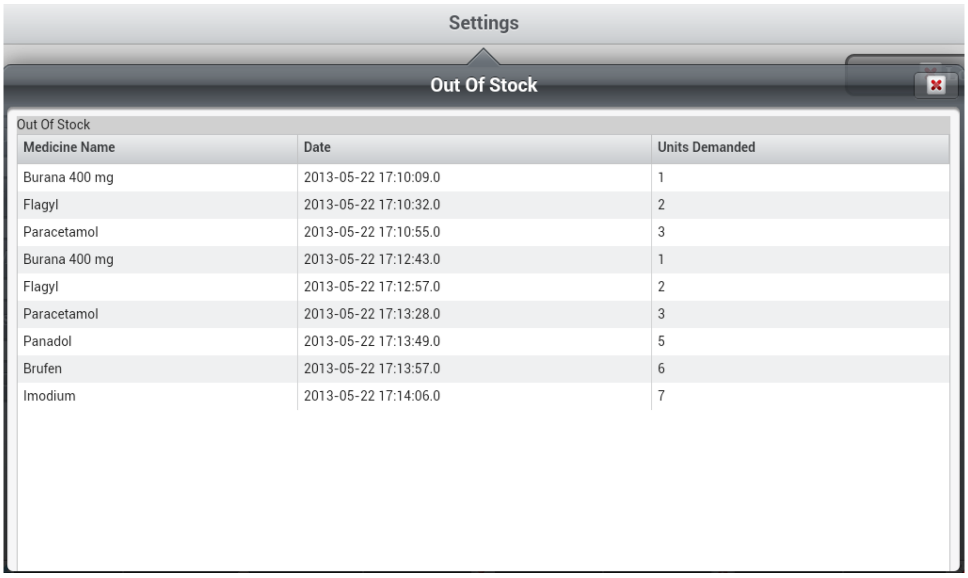


Date	Day	Time	First Name	Last Name	Email
2013-01-28	Monday	08-16 hrs	Nina	Saksi	saksi@abc.com
2013-01-28	Monday	16-24 hrs	Mira	koskinen	koskinen@abc.com
2013-01-29	Tuesday	00-08 hrs	Alice	Lohmus	lohmus@abc.com
2013-01-29	Tuesday	08-16 hrs	Evelina	right	right@abc.com
2013-01-29	Tuesday	16-24 hrs	Nina	Saksi	saksi@abc.com
2013-01-30	Wednesday	08-16 hrs	Mira	koskinen	koskinen@abc.com
2013-01-30	Wednesday	08-16 hrs	Alice	Lohmus	lohmus@abc.com

Figure 5.11: Shift schedule of a nurse

5.4.3 *Out of stock*

All the medicines which were reported to be out of stock during the fill tray process are shown in this window. The out of stock medicines with their quantity demanded and the date and time on which they were reported to be out of stock are listed in this window.



Medicine Name	Date	Units Demanded
Burana 400 mg	2013-05-22 17:10:09.0	1
Flagyl	2013-05-22 17:10:32.0	2
Paracetamol	2013-05-22 17:10:55.0	3
Burana 400 mg	2013-05-22 17:12:43.0	1
Flagyl	2013-05-22 17:12:57.0	2
Paracetamol	2013-05-22 17:13:28.0	3
Panadol	2013-05-22 17:13:49.0	5
Brufen	2013-05-22 17:13:57.0	6
Imodium	2013-05-22 17:14:06.0	7

Figure 5.12: List of out of stock medicines

5.4.4 Log out

When the nurse has performed the entire required task, the nurse can close the session with the help of log out button. Log out button helps the nurse to close down all the session. The button click invokes the command to close down the window which removes its state from the session and the application is restarted.

6 Validation and Usability

To validate the application and understand the behavior of nurses regarding its usability, a survey was conducted. There were in total 13 participants in the survey, belonging to the age group of 24 to 42 years old, with a working experience ranging from 2 to 14 years. All participants are working in different hospitals in Turku, Finland. The main motivation for conducting this survey was to verify the application design and its usability with its end users, which are the nurses, working at different hospitals. The following eight questions were asked in the survey questionnaire.

1. Overall experience of the application?
2. Using the application would improve the job performance?
3. Using the application would make it easier to do the job?
4. Do you find the application would be useful in your job?
5. Learning to operate the application would be easy?
6. It would be easy to become skillful at using the application?
7. Ease of use of the interface?
8. Are there elements in the application interface that you believe could be improved?

The participants were asked to score their response between 1 and 5 (1 - Unlikely, 5 - Likely). The answers to the series of questions asked are summarized in Table 6.1. Most of the participants reported a high level of satisfaction in terms of usability and how it helps to minimize their workload.

6.1 Analysis and Discussion

Based on the survey results, shown in Table 6.1, the average score for each aspect is shown in Figure 6.1. Table 6.1 shows the count of each answer made by the participants for each question.The average scores ranged from 3 to 5 out of 5 for different

Table 6.1: Statistics for user experience regarding application usability.

Questions	Score 1	Score 2	Score 3	Score 4	Score 5
Overall Experience	0	1	4	7	1
Using the application would improve the job performance	0	2	6	2	3
Using the application would make it easier to do the job	0	3	4	6	0
Do you find the application would be useful in your job	0	0	5	5	3
Learning to operate the application would be easy	0	1	4	7	1
It would be easy to become skillful at using the application	0	0	4	6	3
Ease of use of the interface	0	0	4	7	2
Are there elements in the application interface that you believe could be improved	1	2	8	2	0

aspects. On average, 78% of the participants think that it is easier to become skilful at using the application. This high percentage reflects the simplicity of the application. Around 64% of the participants think that the application will make the job of the nurse easier. This percentage depicts that there are still some tasks which the nurses have to perform manually. Approximately 76% think that it can be useful to use the application in the hospitals. Nearly 70% participants have the opinion that using the application will improve the job performance of the nurses. The previous percentages show that adoption of the application in hospitals will bring about a positive change in the dispensation process of medicine. According to these statistical values, the application is easy to use and adaptable for the nursing staff.

The participants were asked about their feedback for improvements in the application. Firstly, the participants indicated that the nurses do not have the option to change the patient information, such as the location of the patient in a ward. It was also noted that a participant had requested inclusion of sound alarms in the application. Lastly, a feature in the application was requested for the storage of medical measurements, such as blood pressure, heart rate, sugar level etc. These measurements should be saved by the nurse along with the other patient information. In addition, the participants mentioned in feedback comments that such applications would address the workload issues and increase the comfort of nursing staff and minimize the error probability. In their opinion, this application can be easily adopted for regular use in Finnish hospitals

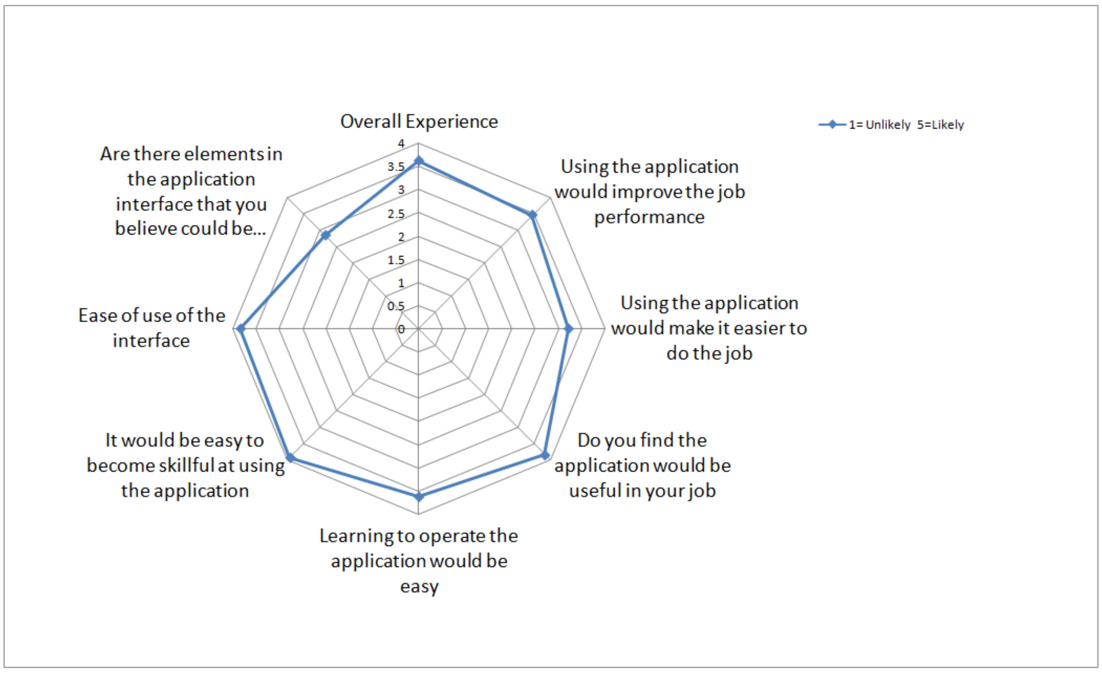


Figure 6.1: Spider chart for the average response to the questionnaire

7 Future Work and Conclusion

7.1 Conclusion

During the dispensation of medicine there is a chance of human error which could result in patient getting a wrong medicine. This probability of mistake is mainly due to the fact that information about the dispensation of medicine to a patient at a hospital is not centralized. The information is collected from various computerized and manual sources. In this application all the relevant information has been put in one place and special care has been taken to make it usable for a nurse without requiring much technical knowledge.

7.2 Future Work

One of the most important aspects which should be taken into consideration when handling the medical data, is that the security of the data is crucial. The privacy and the confidentiality of the medical information of the patient is always encouraged. Therefore, the application can be made more secure with the help of data encryption. Currently the application database is limited to some medicines only. It does not contain the information of all the existing medicines. The application will be more useful when it will have all the information of the medicinse available in the pharmacy. The efficiency of finding equivalent medicines will be increased. Pharmaca Fennica [10] is the most used drug information database in Finland; it has the information of almost all the medicines in Finland. The application will be integrated with the already existing database of Pharmaca Fennica [10].

A pilot test was conducted in two university hospital wards, where eye tracking glasses and stress response was used to observe the nurses' performance while dispensing the medicines to the patient [9]. It was noted that there is a space for improvement and a mobile application can be used to optimize the process [9]. In the next pilot test, the application will be implemented in the hospitals and hopefully it will reduce the time of the nurses and will make the dispensing process less prone to errors [9].

BIBLIOGRAPHY

[1] J. Farrell and G.S. Nezlek. Rich internet applications the next stage of application development. In *Information Technology Interfaces, 2007. ITI 2007. 29th International Conference on*, pages 413–418, June.

[2] W. Liu and E.K. Park. e-Healthcare cloud computing application solutions: Cloud-enabling characteristices, challenges and adaptations. In *Computing, Networking and Communications (ICNC), 2013 International Conference on*, pages 437–443, Jan 2013.

[3] Xuemin (Sherman) Shen. Emerging technologies for e-healthcare [editor's note]. *IEEE Network*, 26(5):2–3, 2012.

[4] Yle News. http://yle.fi/uutiset/medical_errors_becoming_epidemic/6438319.

[5] W. Shangguan F. Miao, X. Miao and Y. Li. Mobihealthcare system: Body sensor network based m-health system for healthcare application. *E-Health Telecommunication Systems and Networks*, 1(1):12–18, 2012.

[6] Mei-Ying Wang, P.H. Tsai, J.W.-S. Liu, and J.K. Zao. Wedjat: A mobile phone based medicine in-take reminder and monitor. In *Bioinformatics and BioEngineering, 2009. BIBE '09. Ninth IEEE International Conference on*, pages 423–430, June 2009.

[7] S. Wagner. Towards an open and easily extendible home care system infrastructure. In *Pervasive Computing Technologies for Healthcare, 2008. PervasiveHealth 2008. Second International Conference on*, pages 42–45, Jan 2008.

[8] e-pill. http://www.epill.com/epillstation.html.

[9] Natalia Díaz Rodríguez, Johan Lilius, Riitta Danielsson-Ojala, Hanna Pirinen, Lotta Kauhanen, Sanna Salanterä, Joachim Majors, Sebu Björlund, Kimmo Rautanen, Tapio Salakoski and Ilona Tuominen. Can IT health-care applications improve the medication tray-filling process at hospital wards? An exploratory study using eye-tracking and stress response. In *Submission: 2014 IEEE 16th International Conference on e-Health Networking, Applications and Services*, Natal, Brazil.

[10] Laaketietokeskus.
http://www.laaketietokeskus.fi/en/pharmaceutical-data/drug-databases.

[11] Rich Internet Application.
https://en.wikipedia.org/wiki/Rich_Internet_application.

[12] EchoWeb Framework.
http://echo.nextapp.com/site/echo3.

[13] JavaFX - The Rich Client Platform.
http://www.oracle.com/technetwork/java/javase/overview/
javafx-overview-2158620.html.

[14] Google Web Toolkit.
http://www.gwtproject.org/.

[15] Thomas-Åke Mattsson. Java persistence layers in stateful rich internet application. MSc Thesis, Åbo Akademi, 2011.

[16] Vaadin.
https://vaadin.com/book/vaadin6/-/page/intro.html#figure.intro.architecture.

[17] Johan Selänniemi. An efficient NoSQL database adapter for a rich internet application framework. MSc Thesis, Åbo Akademi, 2011.

[18] Vaadin data model. https://vaadin.com/book/-/page/datamodel.html.

[19] Model View Controller History.
http://c2.com/cgi/wiki?ModelViewControllerHistory.

[20] T. Iulia-Maria and H. Ciocarlie. Best practices in iPhone programming: Model-view-controller architecture #x2014; Carousel component development. In *EUROCON - International Conference on Computer as a Tool (EUROCON), 2011 IEEE*, pages 1–4, April 2011.

[21] Apache Tomcat. http://tomcat.apache.org/download-60.cgi.

[22] Vaadin.
https://www.vaadin.com.

[23] MySQL. http://www.mysql.com/.

[24] Oracle. http://www.oracle.com/technetwork/java/overview-141217.html.

[25] Deborah J. Mayhew. The usability engineering lifecycle. In *CHI '99 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '99, pages 147–148, New York, NY, USA, 1999. ACM.

[26] C. Ardito, M.F. Costabile, M.De Marsico, R. Lanzilotti, S. Levialdi, T. Roselli, and V. Rossano. An approach to usability evaluation of e-learning applications. *Universal Access in the Information Society*, 4(3):270–283, 2006.