ECE 1778 Final Report:
The Code Blue App

Apper
Ahmed Alterkait

Programers
Simon Chae
Wilson To

Word Count
1999

Apper Context Word Count
419

Table of Contents

1. Introduction

A code blue is a Hospital emergency code that is anounced to notify hospital personnel about a patient related emergency that requires immediate attention. When the code is activated, a team lead by a physician arrives at the patient's bedside. The team relies heavily on the physician for leadership and direction. The environment can often be hectic and loud. Keeping track of all the orders that are given is very difficult. This situation is prone to communication breakdown and may lead to medication errors and/or adverse events[1] [2]

The primary goal of our app is to improve communication and collaboration between team members during a code blue situation or any situation where a patient in a hospital needs immediate resuscitation.
The secondary goals of the app are:

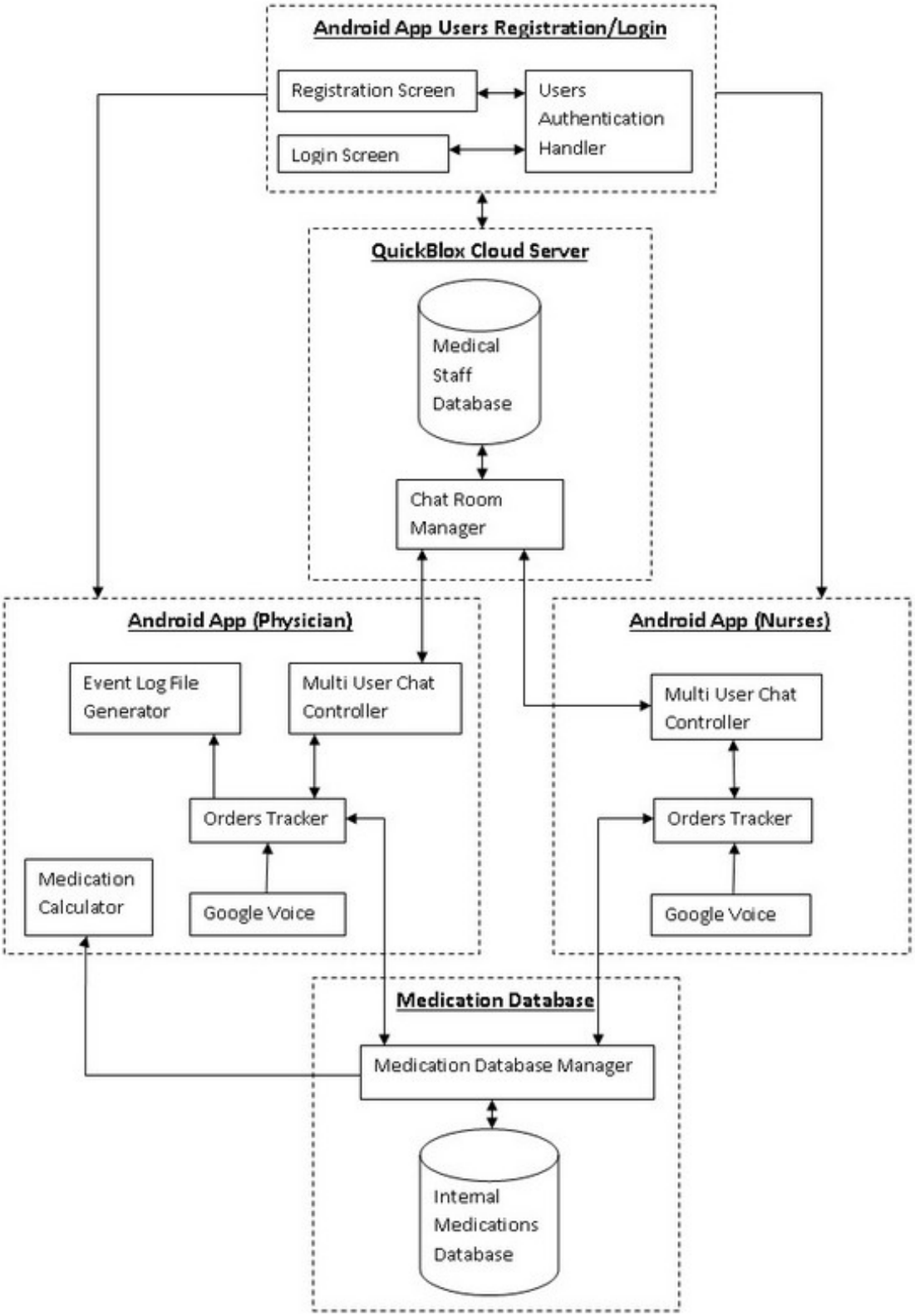
- To reduce the number of errors and adverse events at the patient's bedside
- To provide a recording tool of all the important medications ordered and administered to the patient. This will allow the whole team to track code progress. In addition, it will provide an important record for quality review and assurance
- An educational tool that can be used in simulated codes

This video link illustrates how an actual code blue operates:
http://www.youtube.com/watch?v=4EWlNpKrCuc&list=FLSc7yT4d4PQ4eCTN_D751MA&index=39
This code has been run very well. This is not often the case. Breaks in communication can occur at several points. The app aims to prevent these breaks in communication:

- At minute 2:10 of the video, the physician orders epinephrine. The nurse acknowledges the order by repeating it verbatim. This is one of the crucial points of communication where information can be lost or misinterpreted. An example of that is seen in the video at 3:00. Here the nurse corrects the physician.
- Another crucial moment is when the nurse administers the drug. Ideally the nurse should verbalize exactly how much he/she administered to the whole team. The nurse may forget to let the team know or may not speak loud or clear enough for everybody to hear him/her.

[1] Reader, Tom W, Rhona Flin, and Brian H Cuthbertson. "Communication skills and error in the intensive care unit." *Current opinion in critical care* 13.6 (2007): 732.
[2] Kozer, Eran et al. "Prospective observational study on the incidence of medication errors during simulated resuscitation in a paediatric emergency department." *Bmj* 329.7478 (2004): 1321.

2. Overall Design



Android App Users Registration/Login

Registration/Login Screen

Each staff needs to register/login to the app database before using the app. The registration requires the staff to enter their login ID, password and role.

Authentication Handler

The Authentication handler packages the user information into a request message and sends it to the QuickBlox Server for verification. If the server accepts, it sends the reply back to the authentication handler. The handler will instruct the app to switch to a physician screen or nurse screen depending on the staff's role. If the server rejects the user information, it sends the authentication deny message back to the registration or login screen.

QuickBlox Cloud Server

Medical Staff Database

The Medical Staff Database stores the staff's name, role, registration status, login status and IP address. The Cloud Server uses the information in this database to verify the user information.

Chat Room Manager

A default Chat Room address and name is set up in the Chat Room Manager. The QuickBlox Cloud Server handles the message exchange between the staffs inside the Chat Room.

Android App (Physician Login)

Google Voice

A voice recognition API from Google translates voice input from the phone to a text string. The physician uses voice input to place an order. The result text string is transferred to the order tracker for further analysis.

Orders Tracker

When the Order Tracker receives an order, it looks up the Medical Database for the corresponding medication. The Orders Tracker only accepts the order if the medication is found with proper dosage. The Order Tracker formats the accepted order into a text message and transfers it to the Chat Controller. The text message contains the medication name, dosage, route and order time. When a text message is received from the Chat Controller, the Orders Tracker extracts the given and acknowledge times from the message and update the existing orders list. A timer to track the time for each order.

Multi User Chat Controller (MUCC)

Upon successful login to the app, the Multi User Chat Controller (MUCC) establishes a connection between the app and the QuickBlox Cloud Server. MUCC maintains the server connection and keeps track of server errors. Sending and Receiving of the messages from the Order Tracker is also handled by MUCC. The functionality is the same for the nurse.

Event Log File Generator

The medication name, dosage, route, order time and given time of each order are stored in “csv” text file format in the phone persistent memory.

Medication Calculator

The Medication Calculator uses the patient’s input weight and the dosage information from the Medication Database to calculate the maximum dosage on all of the required medication. The Calculator displays the result in a table format.

Android App (Nurse Login)

Google Voice

A voice recognition API from Google which translate voice input from the phone to a text string. The nurse uses voice input to acknowledge and complete the order when the medication is given.

Orders Tracker

When the Order Tracker receives an order, it looks up the Medical Database for the corresponding medication. The Orders Tracker only accepts the order found in the physician’s order request list. The Orders Tracker provides the given time and acknowledges the time to a text message and transfers it to the Chat Controller. When a text message is received from the Physician, the Orders Tracker adds the order to the physician order request list. If the message is from another nurse, the Orders Tracker will extract the given time and acknowledge time, and update the orders in the existing request list.

Medication Database

Medication Database Manager

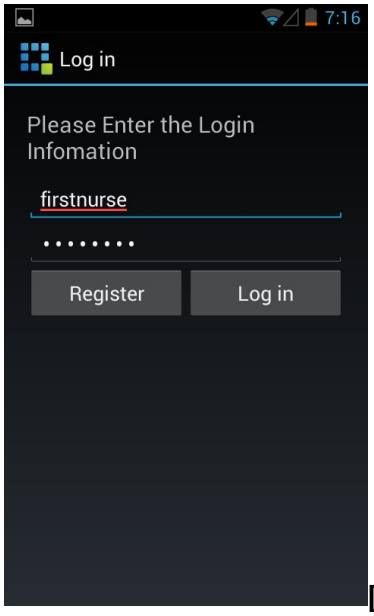
This manager provides functions to insert a medication record to the database and retrieve records from the database

Internal Medications Database

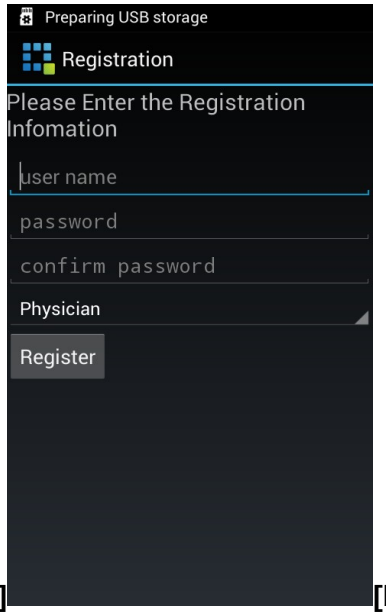
The Medications Database stores all the required medications to operate the code and the dosage formula for each medications.

3. Statement of Functionality and Screen Shots

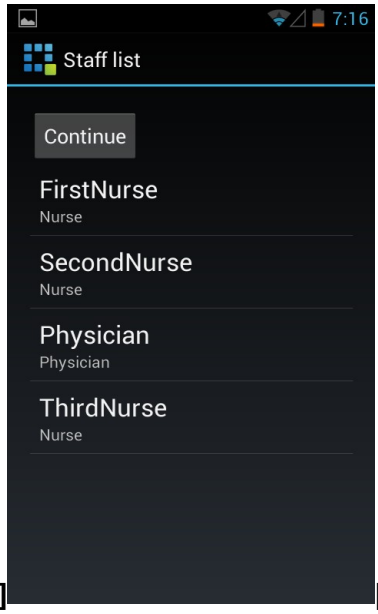
Login & Registration Screens



[a]



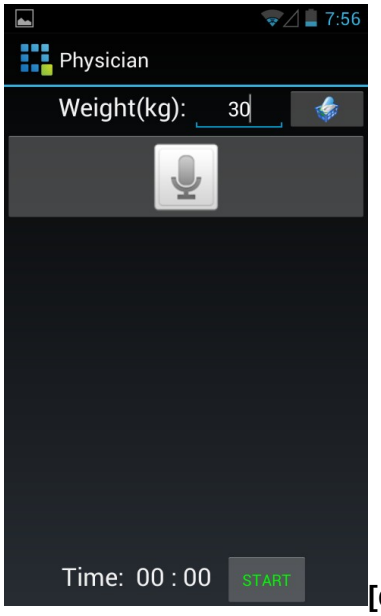
[b]



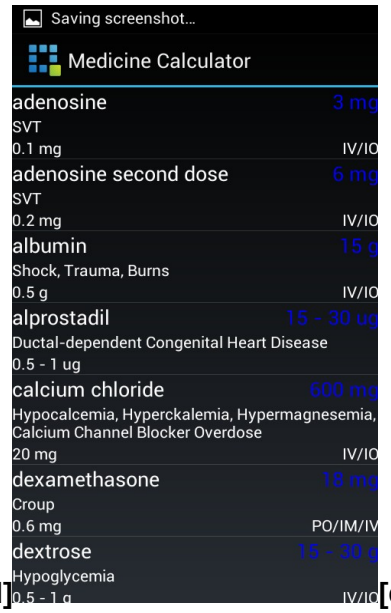
[c]

User can log in or register as a nurse or a physician as shown in **[a]** and **[b]**. Upon successful login, the user will see a list of other users that the user will be communicating with (refer to **[c]**).

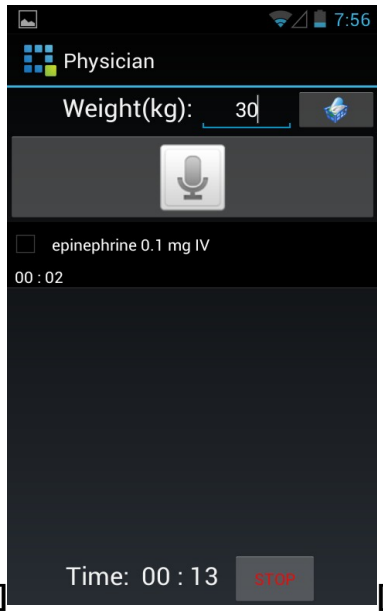
Physician Screens



[d]

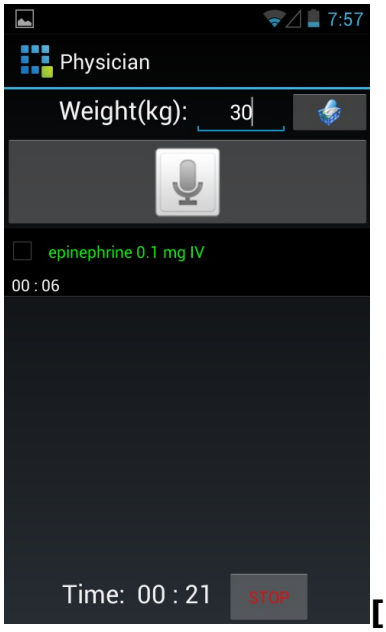


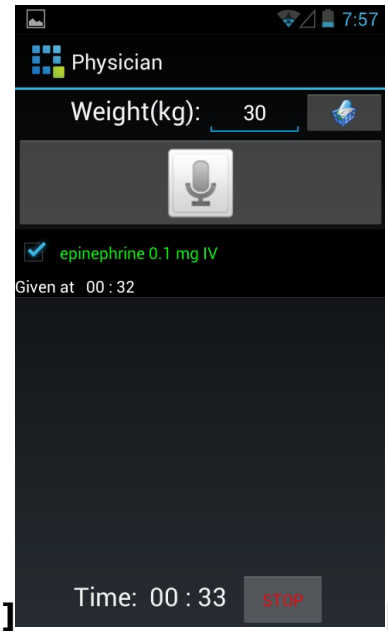
[e]

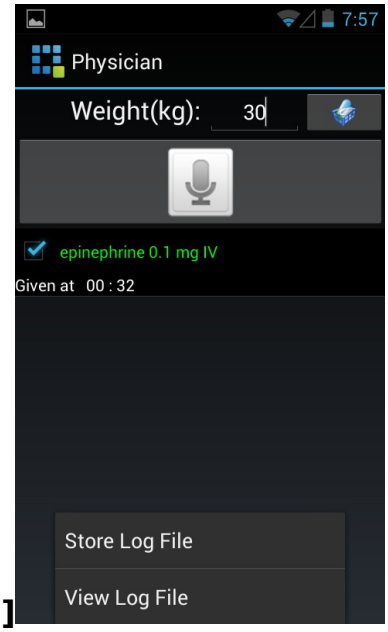


[f]

Upon login, the physician can input the weight of the patient as shown in **[d]**. The physician can access the Medicine Calculator **[e]** by selecting the button next to the weight input in **[d]**. The physician can start the code by selecting the start button and place an order as shown in **[f]**.

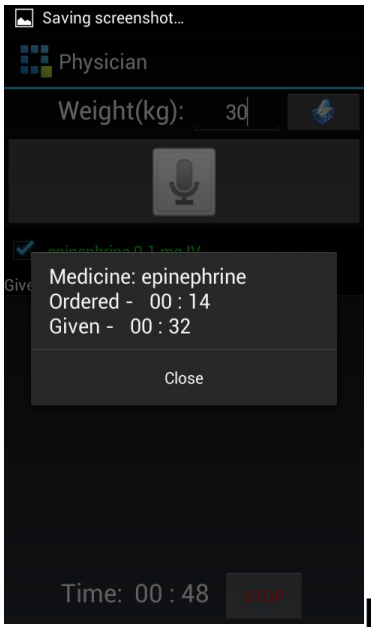


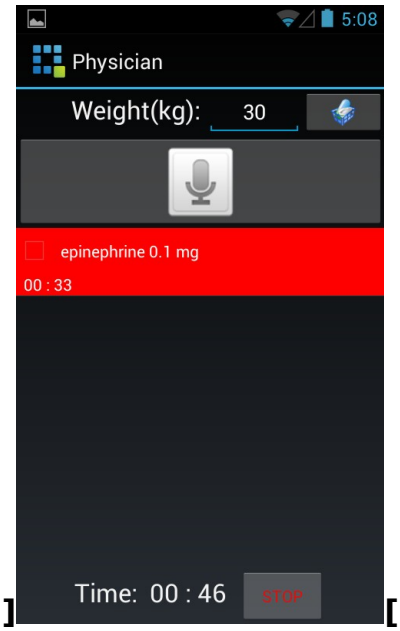
[g]


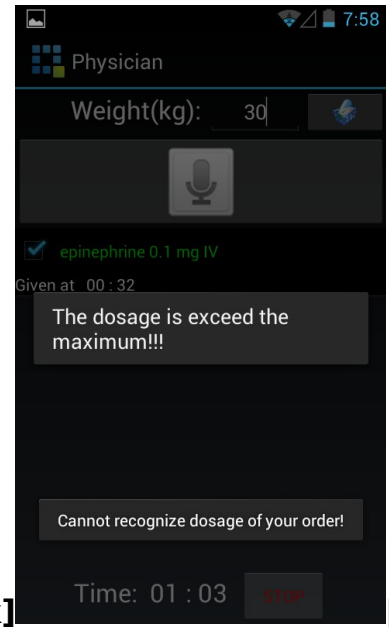
[h]


[i]

When the nurse acknowledges an order, it becomes green in the physician screen **[g]**. When the nurse completes an order, it puts a check mark in the physician screen **[h]**. The user can stop the code by pushing stop button and generate/view a log file by selecting “View Log file” or “Store Log File” from the menu **[i]**.



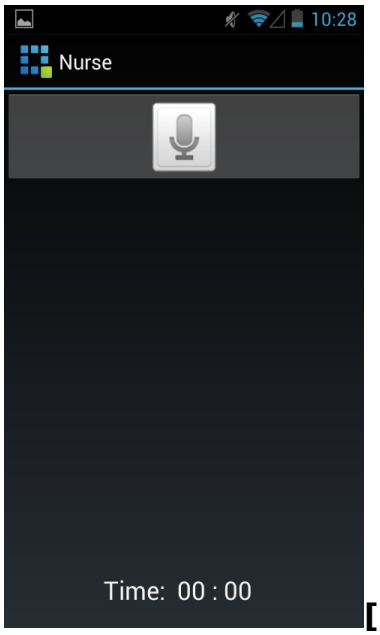
[j]


[k]


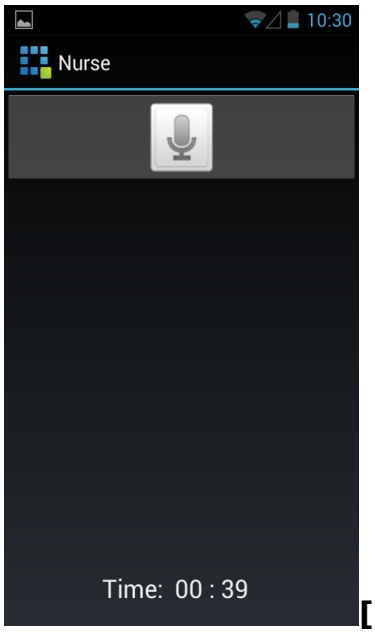
[l]

The physician can track any outstanding order by selecting the order list entry **[j]**. If the order is not given by the nurse within given time limit (30 seconds), then the phone will vibrate and the entry becomes red as shown in **[k]**. If the physician requests an order that exceeds the dosage maximum specified by the database, then it will popup an error message **[l]**.

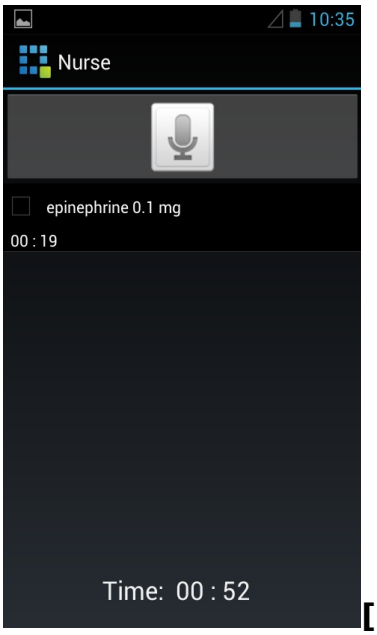
Nurse Screens



[m]

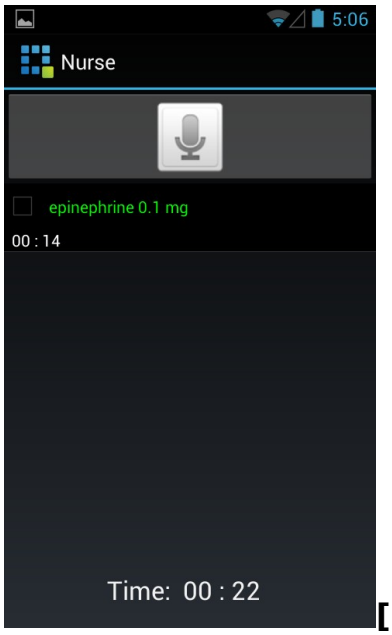


[n]

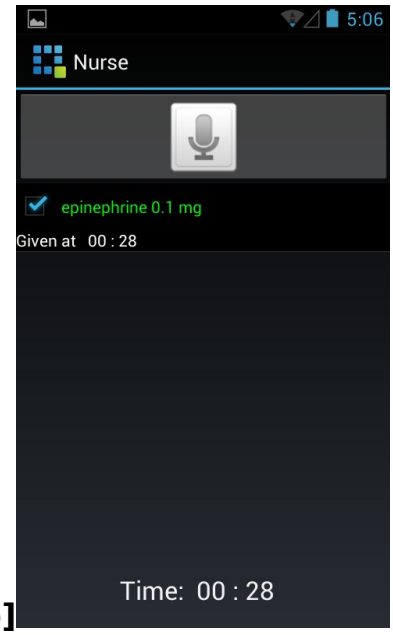


[o]

Upon start of the code **[m]**, nurse's timer will start by synchronizing with the physician's timer **[n]**. When the nurse receives an order, it will be displayed on the screen as shown in **[o]**.

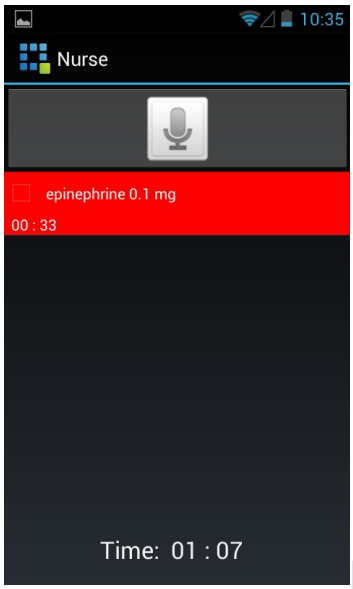


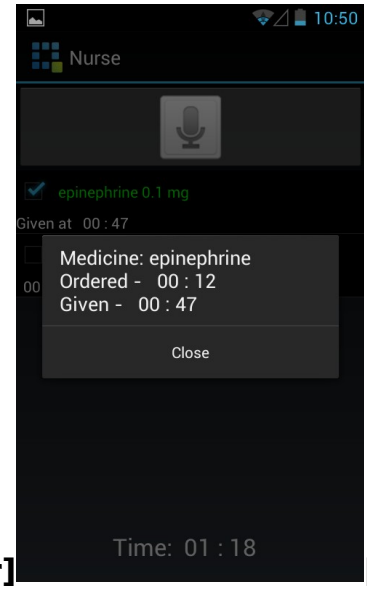
[p]



[q]

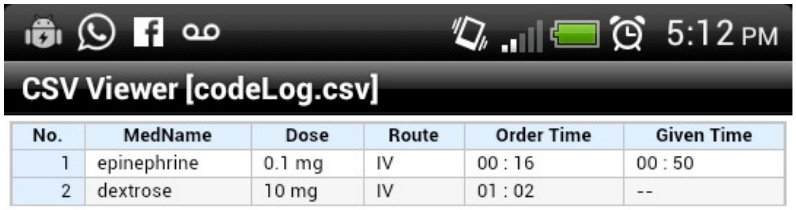
When the nurse acknowledges the order by inputting voice recognition of "... $medication, $dosage ...", the order becomes green and will sync up with the physician side as shown on **[p]** and **[g]**. When the order is given, the nurse inputs "....$medication & given...". This also syncs up the check mark with both nurse and physician screens (Shown in **[q]**).



[r]


[s]

When the nurse fails to respond within 30 seconds, the phone will vibrate and the order becomes red (shown in **[r]**). The nurse can track any order by selecting the order list entry as shown in **[s]**.



No.	MedName	Dose	Route	Order Time	Given Time
1	epinephrine	0.1 mg	IV	00 : 16	00 : 50
2	dextrose	10 mg	IV	01 : 02	--

[t]

[t] shows the Log File Generator output.

4. Learning Experience

Technical aspects

Initially we found that the Google voice recognition works very well with some common drugs. We began our development around Google Voice after the initial testing. However we discovered that not all the drugs can be recognized by Google Voice in the later stages of development. We had to spend some additional time to develop the work around to improve the voice recognition accuracy. If we found the problem earlier, we could have used other voice recognition libraries (eg. Dragon) which may work better and save us some development time.

Additionally, we were learning the new technologies as we developed. This created unexpected discrepancies and unnecessary complications in the later stages. Spending more time on understanding the new technologies and carefully designing the app beforehand would have made the development process a lot smoother.

Communication aspects

It was vital for the group to understand how a code functions. Exposing the programmers to a simulated code would have helped the team understand how a code runs. This would have made it easier for the programmers to improve app functionality.

5. Group Contribution

Ahmed Alterkait (Apper)

- Accommodation of the app context by providing the knowledge around the inner workings of a code blue.
    - How a physician communicates the order
    - How a nurse responds in an ideal code situation
- List of appropriate medications and their respective dosages
- Validation and verification of the app using correct pronunciation of each meds
    - Due to Google Voice API not recognizing many drugs, helped with implementation of basic algorithm to support more medications (e.g. lidocaine)

Simon Chae (Programmer)

- UI/UX Design & Implementation
- Integrating QuickBox Cloud Server and initial Parsing functionality
- Implementation of parsing function and timer functionality on Physician Screen
- Time Synchronization for Order Placement and Order Tracker
- Log File Generator & Reminder popup
- Debugging and enforcing optimization for consistency of the app

Wilson To (Programmer)

- Initial architecture implementation & UI design
- QuickBox Cloud Server setup and Multi User Chat Controller implementation
- Design and implementation of medication database and its manager
- Development and optimization of Google Voice text output parser
- Development of medication calculator
- Improvement of time synchronization for Order Placement and Order Tracker

6. Future Work

Voice recognition

The app currently recognizes 5 medications. These medications are recognized by google voice. We are planning to implement more complex algorithms to improve voice recognition. One strategy is to integrate an autocomplete algorithm. NexJ Systems Inc. has been working on many cloud based services on health and wellness programs; one of which is 'MEDforYOU' that is a "web-based Personal Health Assistant with an embedded terminology engine."[1] The other strategy to improve our voice recognition system is to implement a string comparison algorithm. One string comparison algorithm we looked at was Jaro-Winkler distance[2], which measures "similarities" between two strings by using the area of "record linkage"[3]. The other string comparison algorithm is to integrate Levenshtein distance[4], which compares characters by characters; hence it is harder to implement correctly.

Continuous voice recognition

Currently, all orders have to be recorded individually. Allowing the app to recognize orders without pressing a button will add extra functionality that will eliminate extra steps in the code and increase the efficiency of the code.

Incorporating the app into real practice

This would be the next big step. After ensuring proper voice recognition of all medications we hope to test the app in simulated code blue environments. The step after that would be to use it in real time codes in the Hospital. In addition, The app will be used as a research tool to measure code team performances.

[1] http://www.newswire.ca/en/story/1117041/nexj-systems-adds-terminology-services-to-nexj-connected-wellness
[2] http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance
[3] http://en.wikipedia.org/wiki/Record_linkage
[4] http://en.wikipedia.org/wiki/Levenshtein_distance

7. Apper Context

This is the first app of it’s kind in the medical field. The app can contribute to the field of emergency medicine and acute medical care on many fronts.

Quality Improvement

The code blue app can provide vital information that can measure team performances in code blue situations. The main reason behind developing this app is to improve patient safety by preventing adverse events and medical errors. In current practice in an emergency situation or a code blue, a nurse records all the orders and the medications given in written form. Using the app provides a more consistent and accurate record of the events taking place in a code while minimizing errors. This will also provide important feedback during and after the code. In case an adverse event does occur, the code is reviewed in detail in Morbidity and Mortality (M&M) meetings. All records of the case are reviewed and presented to a group of physicians . The records include patient history, vital signs throughout the code, medications administered and the notes on the code blue sheets. This review process attempts to identify the source of the error and implement strategies to prevent future adverse events. The log file in the app is a great tool that can be saved and used as a reference for review in M&M meetings.

Medical Education and Simulation

Emergency medicine teams and code blue teams are trained through simulation to manage and resuscitate sick patients in an artificial environment. Using the app will enhance the learning aspect of the training. It will teach physicians how to correctly order medications and train nurses how to acknowledge the orders. The track log function allows the team to reflect on the code during a debriefing session that usually takes place after the simulation.

Research

The app has to be validated first as a tool that can be used in real codes compared to manual recording. The primary objective is to examine if the app can actually decrease the number of adverse events in a code. Then, it can be used to measure different aspects of a code both qualitatively and quantitatively. These include:

- Medication use patterns
- Response time to administer drugs
- Adverse events in codes in various institutions or specialties
- Quality of team communication

One important issue that will need to be addressed prior to use of the app is patient information privacy. This will likely slow the progress of implementing this app as a standard of practice. However, it is a necessary step in the process.

8. Businesses School Interest

We are not interested in having a Business School marketing/entrepreneurship student(s) to take up Code Blue App. This is because our app is not intended for business use.

9. App Resources

Quickblox Cloud Service

We used the Quickblox Cloud Service for our Medical Staff Database. We also used the Chat Room service to handle the message exchanges. Here is the link of the Quickblox website http://quickblox.com/. Unfortunately this website does not provide a Multi User Chat example. The Multi User Chat information can be found in this website: http://www.igniterealtime.org/builds/smack/docs/3.2.2/javadoc/ The QuickBlox library includes the Smack Library already.

Google Voice API

Google Voice is available for free to use at http://code.google.com/p/google-voice-java/.
Installation and integration of the Google Voice API is very straightforward, and Google Voice Wiki is very well documented[1].

10. Open Source Availability

We would like to have our source code not available for open source. We may reconsider upon validation from simulated code situation.

[1] http://code.google.com/p/google-voice-java/w/list