CPSC 403 SOFTWARE DEVELOPMENT IN TEAMS AND ORGANIZATIONS

ITERATION 1 - REPORT

HOSPITAL MANAGEMENT SYSTEM

Prepared by James Bertram, Thomas Frampton, Ekaterina Grekova, Matthew Jones, Andrew Midwinter, Mark Mullen

2/24/2012

Contents

Table of Figures

Summary

In this iteration, we began implementing the higher priority core features. As of this iteration, the program supports the ability to login as a system user, add patients to the database, and view patient information. We have also begun development of the user interface. The current iteration does not support the ability to edit patient information after they have been created. This is a known bug and we will develop a fix for it as soon as possible. This report includes a domain model of the HMS, updated prioritized user stories and release plan, design diagrams and work distribution.

Domain Model

Various diagrams were created during initial design to help visualizing and clarifying main requirements of HMS. Our initial design underwent changes during implementation process. During discussions, we came to agreement that a simpler architecture can be developed. That architecture is understood by everyone in the team, and it allows adding new features, testing implemented coded and debugging in an easy manner. The supplementary diagrams were updated and can be found in Appendix A.

Stories

The identified stories remain the same as in Iteration 0. Below are the stories, which are sorted by their priority. A picture of our story board is in Appendix B.

Priority 1:

1.1: Assigning patients to rooms:
This feature allows nurses or administrators to assign patients to specific rooms.
1.2: Tracking Patient Information:
This feature allows administrators to enter, edit, and view information about patients.
1.3: Tracking Bed/Room availability:
This feature keeps track of which beds and rooms are currently available, and which are currently occupied.
1.4: Assigning Nurses to Wards:
This feature allows administrators to assign nurses to wards.
1.5: Encrypting of Patient Information:
Patient information will be encrypted for security.

Priority 2:

2.1: Tracking Nurse Information:
Doctors and head nurses can enter, edit nurses' information.

Priority 3:

3.1: Placing Patients in a Priority Queue:

Patients will be placed in a priority queue to order them by the severity of their illness or injury.

Priority 4:

4.1: Searching for Patients:
Nurses and Doctors can search for information about a specific patient.

Priority 5:

5.1: Scheduling Nurses:
Administrators can create schedules to assign nurses to shifts.

Updated Tentative Release Plan

However, the tentative release plan has changed. The implementation of the base architecture required more time than we estimated initially.

Tentative Release Plan:

Iteration 1:
Build base architecture
Implemented feature 1.2 using TDD

Iteration 2:
Implement features 1.1, 1.3, 1.4, 2.1 and 3.1.

Iteration 3:
Implement features 1.5, 4.1 and 5.1

Deliverables of Iteration 1

Figure 1. Sequence Diagram – User can log in to system

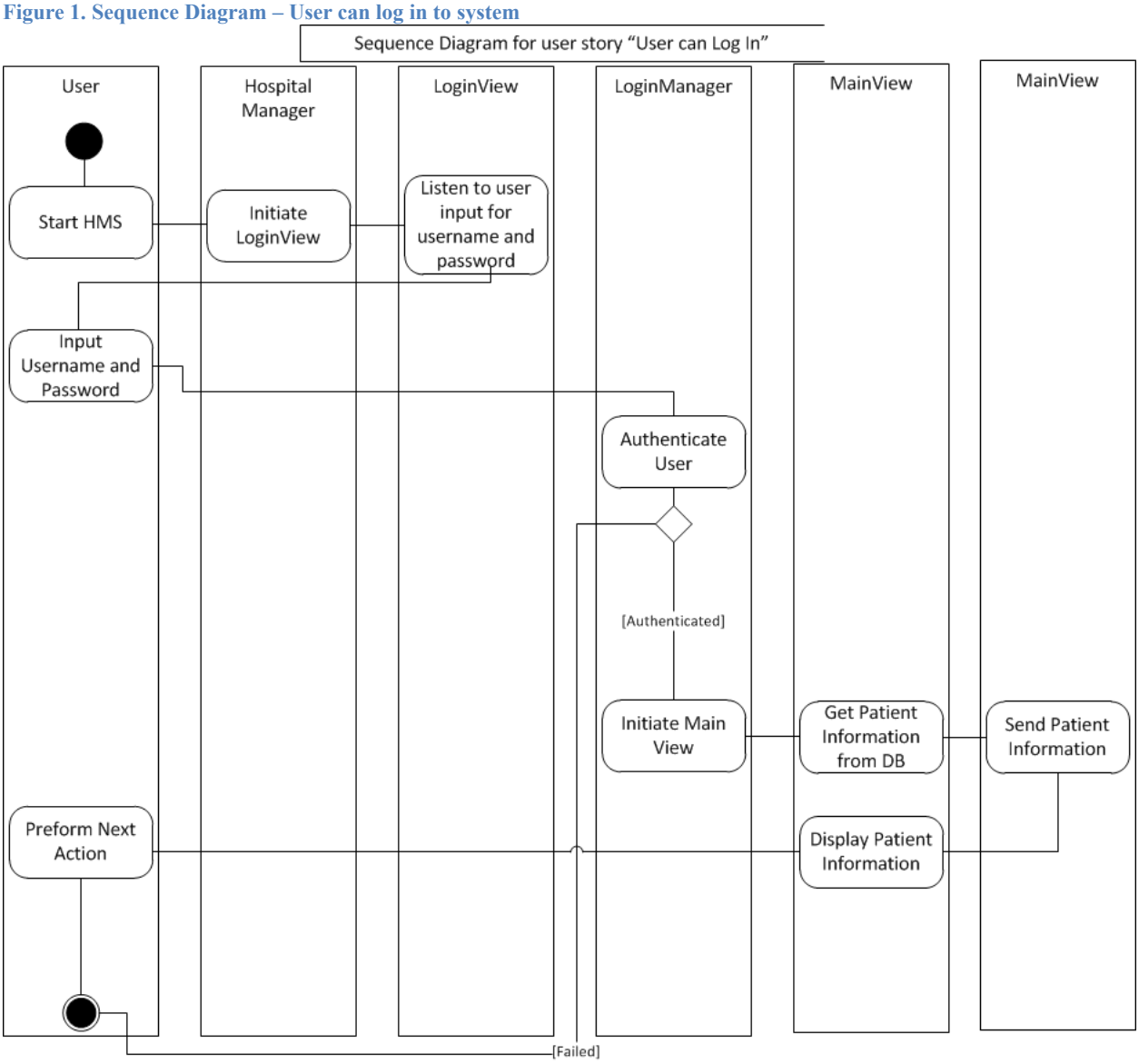


Figure 2. ERD

Entities
========

Patient

* Name
* Ge

Bed

*

Ward

* Name
* Size

Nurse

* Name
* Phone Number
* Head Nurse?

Room

* Number

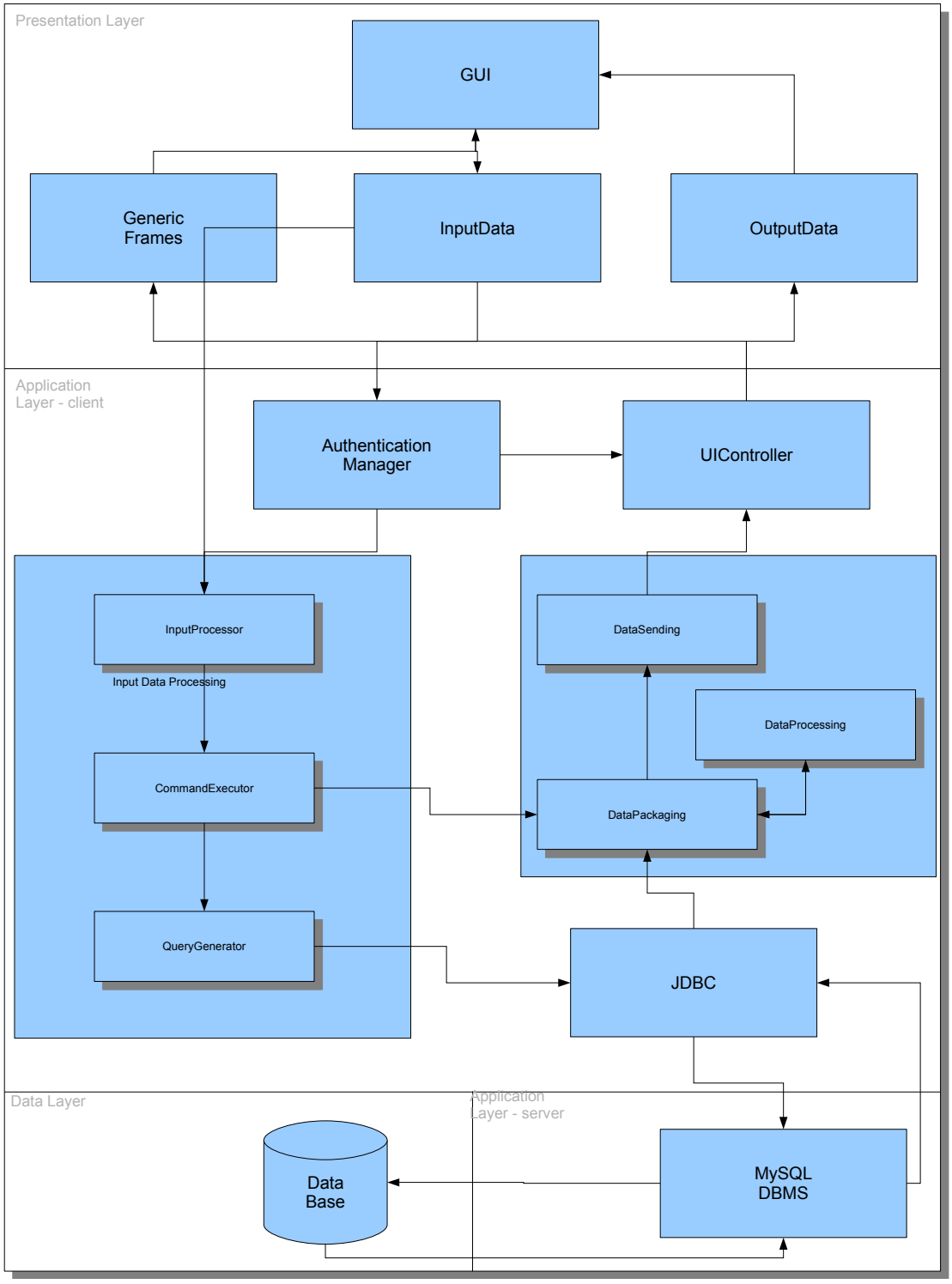
Nurse Type

* Name

Relationships
=============

Patient (1)-< Occupies >-(1) Bed
Bed (n)-< Belongs To >-(1) Room
Room (n)-< Belongs To >-(1) Ward
Nurse (n)-< Assigned To >-(1) Ward
Nurse (n)-< Has A >-(1) Nurse Type
Ward (n)-< Has A >-(1) Nurse Type

Figure 3. Technology Diagram



Work Distribution

Tom

Implemented authentication system (register, login, logout)
Created sequence diagram

Mark

Implemented command methods with Andrew
designed prototypes with Andrew

Andrew

Implemented command methods with Mark
designed prototypes with Mark
Implemented database interface class with Matt

Matt

Implemented database interface class with Andrew
Implemented model classes
Implemented relational schema in SQL
designed ERD

Katya

Implemented GUI classes (view)
designed use case diagram
storyboarding

Jamie

integration and QA testing
High-level technology diagram
report compilation

Appendix A – Diagrams

Figure 3. Relational Schema

```
-- phpMyAdmin SQL Dump
-- version 3.2.2
-- http://www.phpmyadmin.net
--
-- Host: localhost
-- Generation Time: Dec 22, 2011 at 12:33 PM
-- Server version: 5.0.67
-- PHP Version: 5.2.6



/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- Database: `jhbertra`
--

-- --------------------------------------------------------

--
-- Table structure for table `ASSIGNED_TO`
--

CREATE TABLE IF NOT EXISTS `ASSIGNED_TO` (
  `username` varchar(12) NOT NULL,
  `project_id` int(11) NOT NULL,
  `name` varchar(20) NOT NULL,
  PRIMARY KEY  (`username`,`project_id`,`name`),
  KEY `project_id` (`project_id`),
  KEY `name` (`name`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `COURSE`
--

CREATE TABLE IF NOT EXISTS `COURSE` (
  `Dept_Name` char(4) NOT NULL,
  `c_no` int(11) NOT NULL,
  `course_id` int(11) NOT NULL auto_increment,
  `name` varchar(80) NOT NULL,
  PRIMARY KEY  (`course_id`),
  UNIQUE KEY `Dept_Name` (`Dept_Name`,`c_no`)
) TYPE=InnoDB  AUTO_INCREMENT=11 ;

-- --------------------------------------------------------

--
-- Table structure for table `COURSE_THREAD_OWNER`
--

CREATE TABLE IF NOT EXISTS `COURSE_THREAD_OWNER` (
  `course_id` int(11) NOT NULL,
  `thread_id` int(11) NOT NULL,
  PRIMARY KEY  (`course_id`,`thread_id`),
  KEY `thread_id` (`thread_id`)
) TYPE=InnoDB;

-- --------------------------------------------------------
```

```
--
-- Table structure for table `DISCUSSION_THREAD`
--

CREATE TABLE IF NOT EXISTS `DISCUSSION_THREAD` (
  `thread_id` int(11) NOT NULL auto_increment,
  `topic` varchar(40) NOT NULL,
  PRIMARY KEY  (`thread_id`)
) TYPE=InnoDB  AUTO_INCREMENT=35 ;

-- --------------------------------------------------------

--
-- Table structure for table `EMAIL`
--

CREATE TABLE IF NOT EXISTS `EMAIL` (
  `username` varchar(12) NOT NULL,
  `profile_id` int(11) NOT NULL,
  `email` varchar(40) NOT NULL,
  PRIMARY KEY  (`username`,`profile_id`,`email`),
  KEY `profile_id` (`profile_id`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `FRIENDS`
--

CREATE TABLE IF NOT EXISTS `FRIENDS` (
  `username` varchar(12) NOT NULL,
  `friend_username` varchar(12) NOT NULL,
  PRIMARY KEY  (`username`,`friend_username`),
  KEY `friend_username` (`friend_username`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `INTERESTS`
--

CREATE TABLE IF NOT EXISTS `INTERESTS` (
  `profile_id` int(11) NOT NULL,
  `username` varchar(12) NOT NULL,
  `interest` varchar(40) NOT NULL,
  PRIMARY KEY  (`profile_id`,`username`,`interest`),
  KEY `username` (`username`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `PHONE`
--

CREATE TABLE IF NOT EXISTS `PHONE` (
  `username` varchar(12) NOT NULL,
  `profile_id` int(11) NOT NULL,
  `home_phone` varchar(20) default NULL,
  `cell_phone` varchar(20) default NULL,
  PRIMARY KEY  (`username`,`profile_id`),
  KEY `profile_id` (`profile_id`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `POST`
--
```

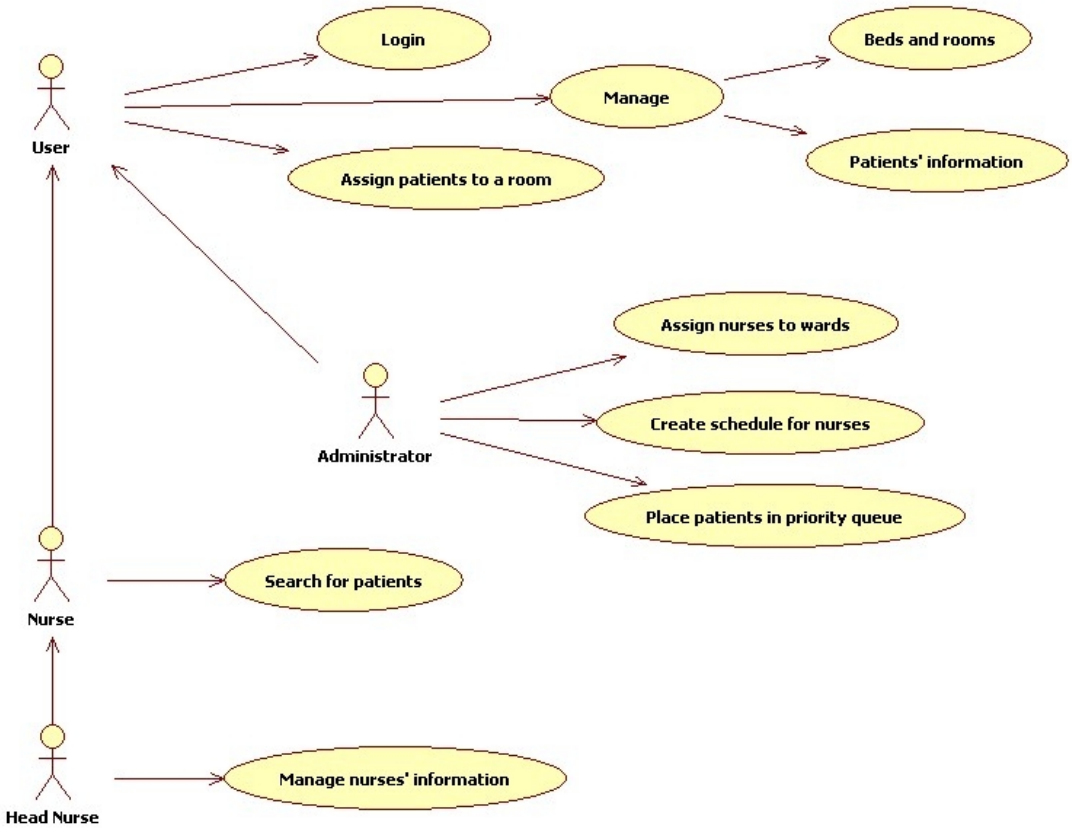
```
CREATE TABLE IF NOT EXISTS `POST` (
  `body_text` mediumtext NOT NULL,
  `post_id` int(11) NOT NULL auto_increment,
  `username` varchar(12) NOT NULL,
  `timestamp` timestamp NOT NULL,
  `thread_id` int(11) NOT NULL,
  `type` enum('Comment','Question','Reply','Reminder') NOT NULL,
  PRIMARY KEY  (`post_id`,`username`,`thread_id`),
  KEY `username` (`username`),
  KEY `thread_id` (`thread_id`)
) TYPE=InnoDB  AUTO_INCREMENT=86 ;

-- --------------------------------------------------------

--
-- Table structure for table `PROFILE`
--

CREATE TABLE IF NOT EXISTS `PROFILE` (
  `username` varchar(12) NOT NULL,
  `picture` varchar(500) default NULL,
  `f_name` varchar(20) NOT NULL,
  `b_day` date NOT NULL,
  `profile_id` int(11) NOT NULL auto_increment,
  `status` mediumtext,
  `l_name` varchar(20) NOT NULL,
  `bio` longtext,
  `website` varchar(100) default NULL,
  PRIMARY KEY  (`profile_id`),
  UNIQUE KEY `username` (`username`)
) TYPE=InnoDB  AUTO_INCREMENT=30 ;

-- --------------------------------------------------------

--
-- Table structure for table `PROFILE_THREAD_OWNER`
--

CREATE TABLE IF NOT EXISTS `PROFILE_THREAD_OWNER` (
  `profile_id` int(11) NOT NULL,
  `thread_id` int(11) NOT NULL,
  PRIMARY KEY  (`profile_id`,`thread_id`),
  KEY `thread_id` (`thread_id`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `PROJECT`
--

CREATE TABLE IF NOT EXISTS `PROJECT` (
  `project_id` int(11) NOT NULL auto_increment,
  `deadline` date NOT NULL,
  `project_name` varchar(20) NOT NULL,
  `project_description` longtext NOT NULL,
  `course_id` int(11) NOT NULL,
  PRIMARY KEY  (`project_id`),
  KEY `course_id_2` (`course_id`)
) TYPE=InnoDB  AUTO_INCREMENT=27 ;

-- --------------------------------------------------------

--
-- Table structure for table `PROJECT_THREAD_OWNER`
--

CREATE TABLE IF NOT EXISTS `PROJECT_THREAD_OWNER` (
  `project_id` int(11) NOT NULL,
  `thread_id` int(11) NOT NULL,
  PRIMARY KEY  (`project_id`,`thread_id`),
  KEY `thread_id` (`thread_id`)
```

```
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `TAKES_COURSE`
--

CREATE TABLE IF NOT EXISTS `TAKES_COURSE` (
  `username` varchar(12) NOT NULL,
  `course_id` int(11) NOT NULL,
  PRIMARY KEY  (`username`,`course_id`),
  KEY `course_id` (`course_id`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `TASK`
--

CREATE TABLE IF NOT EXISTS `TASK` (
  `name` varchar(100) NOT NULL,
  `description` mediumtext NOT NULL,
  `deadline` datetime NOT NULL,
  `project_id` int(11) NOT NULL,
  PRIMARY KEY  (`name`,`project_id`),
  KEY `project_id` (`project_id`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `TASK_THREAD_OWNER`
--

CREATE TABLE IF NOT EXISTS `TASK_THREAD_OWNER` (
  `name` varchar(20) NOT NULL,
  `thread_id` int(11) NOT NULL,
  `project_id` int(11) NOT NULL,
  PRIMARY KEY  (`name`,`thread_id`,`project_id`),
  UNIQUE KEY `project_id` (`project_id`),
  KEY `thread_id` (`thread_id`)
) TYPE=InnoDB;

-- --------------------------------------------------------

--
-- Table structure for table `USER`
--

CREATE TABLE IF NOT EXISTS `USER` (
  `username` varchar(12) NOT NULL,
  `password` varchar(15) NOT NULL,
  `isAdmin` tinyint(1) NOT NULL default '0' COMMENT 'boss boss boss boss boss',
  PRIMARY KEY  (`username`)
) TYPE=InnoDB COMMENT='users, dawg';

-- --------------------------------------------------------

--
-- Table structure for table `WORKS_ON`
--

CREATE TABLE IF NOT EXISTS `WORKS_ON` (
  `username` varchar(12) NOT NULL,
  `project_id` int(11) NOT NULL,
  PRIMARY KEY  (`username`,`project_id`),
  KEY `project_id` (`project_id`)
) TYPE=InnoDB;

--
```

```
-- Constraints for dumped tables
--

--
-- Constraints for table `ASSIGNED_TO`
--
ALTER TABLE `ASSIGNED_TO`
  ADD CONSTRAINT `ASSIGNED_TO_ibfk_7` FOREIGN KEY (`name`) REFERENCES `TASK` (`name`)
ON DELETE CASCADE,
  ADD CONSTRAINT `ASSIGNED_TO_ibfk_5` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE,
  ADD CONSTRAINT `ASSIGNED_TO_ibfk_6` FOREIGN KEY (`project_id`) REFERENCES `PROJECT`
(`project_id`) ON DELETE CASCADE;

--
-- Constraints for table `COURSE_THREAD_OWNER`
--
ALTER TABLE `COURSE_THREAD_OWNER`
  ADD CONSTRAINT `COURSE_THREAD_OWNER_ibfk_4` FOREIGN KEY (`thread_id`) REFERENCES
`DISCUSSION_THREAD` (`thread_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `COURSE_THREAD_OWNER_ibfk_3` FOREIGN KEY (`course_id`) REFERENCES
`COURSE` (`course_id`) ON DELETE CASCADE;

--
-- Constraints for table `EMAIL`
--
ALTER TABLE `EMAIL`
  ADD CONSTRAINT `EMAIL_ibfk_4` FOREIGN KEY (`profile_id`) REFERENCES `PROFILE`
(`profile_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `EMAIL_ibfk_3` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE;

--
-- Constraints for table `FRIENDS`
--
ALTER TABLE `FRIENDS`
  ADD CONSTRAINT `FRIENDS_ibfk_6` FOREIGN KEY (`friend_username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `FRIENDS_ibfk_5` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE;

--
-- Constraints for table `INTERESTS`
--
ALTER TABLE `INTERESTS`
  ADD CONSTRAINT `INTERESTS_ibfk_4` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE,
  ADD CONSTRAINT `INTERESTS_ibfk_3` FOREIGN KEY (`profile_id`) REFERENCES `PROFILE`
(`profile_id`) ON DELETE CASCADE;

--
-- Constraints for table `PHONE`
--
ALTER TABLE `PHONE`
  ADD CONSTRAINT `PHONE_ibfk_3` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE,
  ADD CONSTRAINT `PHONE_ibfk_4` FOREIGN KEY (`profile_id`) REFERENCES `PROFILE`
(`profile_id`) ON DELETE CASCADE;

--
-- Constraints for table `POST`
--
ALTER TABLE `POST`
  ADD CONSTRAINT `POST_ibfk_4` FOREIGN KEY (`thread_id`) REFERENCES
`DISCUSSION_THREAD` (`thread_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `POST_ibfk_3` FOREIGN KEY (`username`) REFERENCES `USER` (`username`)
ON DELETE CASCADE;

--
-- Constraints for table `PROFILE`
--
ALTER TABLE `PROFILE`
```

```
  ADD CONSTRAINT `PROFILE_ibfk_1` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE;

--
-- Constraints for table `PROFILE_THREAD_OWNER`
--
ALTER TABLE `PROFILE_THREAD_OWNER`
  ADD CONSTRAINT `PROFILE_THREAD_OWNER_ibfk_4` FOREIGN KEY (`thread_id`) REFERENCES
`DISCUSSION_THREAD` (`thread_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `PROFILE_THREAD_OWNER_ibfk_3` FOREIGN KEY (`profile_id`) REFERENCES
`PROFILE` (`profile_id`) ON DELETE CASCADE;

--
-- Constraints for table `PROJECT`
--
ALTER TABLE `PROJECT`
  ADD CONSTRAINT `PROJECT_ibfk_1` FOREIGN KEY (`course_id`) REFERENCES `COURSE`
(`course_id`) ON DELETE CASCADE;

--
-- Constraints for table `PROJECT_THREAD_OWNER`
--
ALTER TABLE `PROJECT_THREAD_OWNER`
  ADD CONSTRAINT `PROJECT_THREAD_OWNER_ibfk_3` FOREIGN KEY (`project_id`) REFERENCES
`PROJECT` (`project_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `PROJECT_THREAD_OWNER_ibfk_2` FOREIGN KEY (`thread_id`) REFERENCES
`DISCUSSION_THREAD` (`thread_id`) ON DELETE CASCADE;

--
-- Constraints for table `TAKES_COURSE`
--
ALTER TABLE `TAKES_COURSE`
  ADD CONSTRAINT `TAKES_COURSE_ibfk_4` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE,
  ADD CONSTRAINT `TAKES_COURSE_ibfk_6` FOREIGN KEY (`course_id`) REFERENCES `COURSE`
(`course_id`) ON DELETE CASCADE;

--
-- Constraints for table `TASK`
--
ALTER TABLE `TASK`
  ADD CONSTRAINT `TASK_ibfk_1` FOREIGN KEY (`project_id`) REFERENCES `PROJECT`
(`project_id`) ON DELETE CASCADE;

--
-- Constraints for table `TASK_THREAD_OWNER`
--
ALTER TABLE `TASK_THREAD_OWNER`
  ADD CONSTRAINT `TASK_THREAD_OWNER_ibfk_4` FOREIGN KEY (`project_id`) REFERENCES
`TASK` (`project_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `TASK_THREAD_OWNER_ibfk_2` FOREIGN KEY (`thread_id`) REFERENCES
`DISCUSSION_THREAD` (`thread_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `TASK_THREAD_OWNER_ibfk_3` FOREIGN KEY (`name`) REFERENCES `TASK`
(`name`) ON DELETE CASCADE;

--a
-- Constraints for table `WORKS_ON`
--
ALTER TABLE `WORKS_ON`
  ADD CONSTRAINT `WORKS_ON_ibfk_4` FOREIGN KEY (`project_id`) REFERENCES `PROJECT`
(`project_id`) ON DELETE CASCADE,
  ADD CONSTRAINT `WORKS_ON_ibfk_3` FOREIGN KEY (`username`) REFERENCES `USER`
(`username`) ON DELETE CASCADE;
```

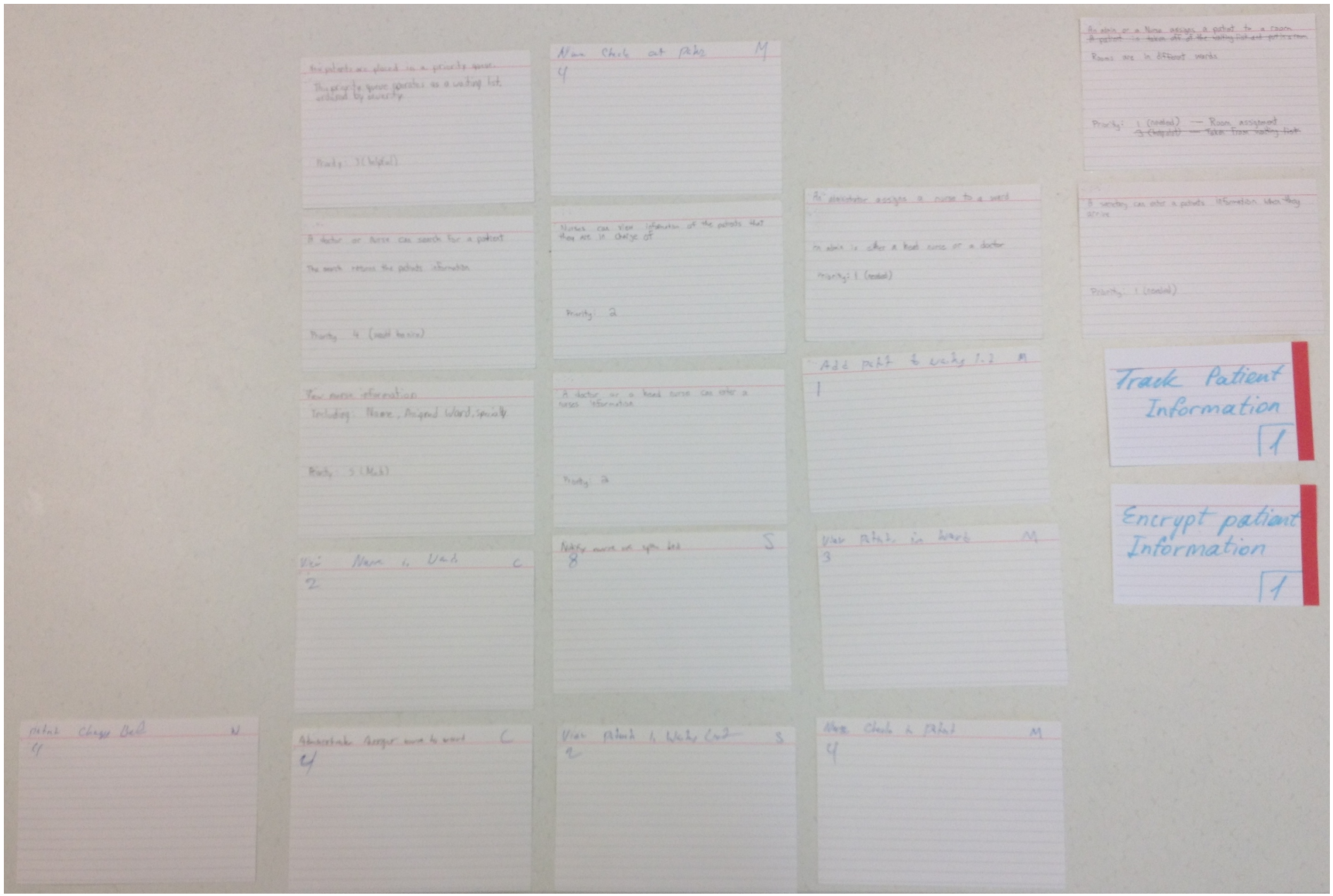
Figure 4. Use Case Diagram



Appendix B – Story Board

Additional details about the feature and to whom it was assigned are written on the back of the card.

Figure 6. Our story board as of February 24, 2012


Track Patient Information
1
Encrypt patient Information
1